Non-Linear Hyperspectral Subspace Mapping using Stacked Autoencoder

Niclas Wadströmer and David Gustafsson
Swedish Defence Research Agency (FOI), Sweden
email: {niclas.wadstromer, david.gustafsson}@foi.se

Abstract

Stacked Auto-Encoder (SAE) is a rather new machine learning approach which utilize unlabelled training data to learn a deep hierarchical representation of features. SAE:s can be used to learn a feature representation that preserve key information of the features, but has a lower dimensionality than the original feature space. The learnt representation is a non-linear transformation that maps the original features to a space of lower dimensionality. Hyperspectral data are high dimensional while the information conveyed by the data about the scene can be represented in a space of considerably lower dimensionality. Transformation of the hyperspectral data into a representation in a space of lower dimensionality which preserve the most important information is crucial in many applications. We show how unlabelled hyperspectral signatures can be used to train a SAE. The focus for analysis is what type of spectral information is preserved in the hierarchical SAE representation. Results from hyperspectral images of natural scenes with man-made objects placed in the scene is presented. Example of how SAE:s can be used for anomaly detection, detection of anomalous spectral signatures, is also presented.

1 Introduction

Hyperspectral imaging reveal information about the scene that can not be perceived with a visual camera. The spectrum radiated from each point of the surfaces in the scene is captured in many separate narrow wavelength bands. Usually a visual camera have three wavelength bands capturing red, green and blue colours. A hyperspectral sensor can capture tenths, hundreds and even thousands of wavelength bands. The spectrum for single pixels can be viewed and compared with each other. But an image could have millions of pixels each with hundreds of wavelength bands. Thus an image could be composed of 10^9 values. The information in a hyperspectral image can not easily be comprehended by a human observer. On the other hand is not unlikely that a scene only contains a handful of materials suggesting that maybe the information in the image can be found in a low dimensional space. So how can we find a suitable space which will fit the information without discarding any information? What dimension of the space can we expect to find? What class of mappings between the spaces should be expected to be efficient?

Here we will assume that the hyperspectral data can be found in a subspace of the measurement space. It is also possible that the scene consists of a countable set of different spectrums. In this case can the data be represented by an index number referring to the spectrum found in each pixel.

But for now let us assume that the data is composed of spectrums from a subspace of lower dimensionality.

Many analysis algorithms have a computational complexity which is proportional to the number of dimensions, some have exponential complexity, and are thus costly to work with. If it is possible to map the data to a space of lower dimensionality than less computational resources will be needed.

Deep learning using stacked autoencoders have shown in many examples to give a compressed representation that is useful for many different applications. For example a representation trained for a specific classifier has shown to be useful for classification of classes not at all considered in the original

training.

There is also a question of how easy it is to find the important information. It may be possible that the information in the encoded space is more difficult to find than the same information in the uncoded data.

2 Hyperspectral imaging

A hyperspectral image can be viewed as set of spectral measurements $x_i : i = 1, 2, \ldots, N$ where i a spatial index. Each measurement x_i is represented as a M-dimensional vector

$$x_i = [L_i(\lambda_1), \ldots, L_i(\lambda_M)] \quad (1)$$

where $L_i(\lambda_k)$ is the measurement of spectral band λ_k and M is the number of spectral bands. This representation stresses the spectral dimension of the data, while the spatial relations are ignored. As the focus of the paper is spectral dimensionality reduction the spectral vector representation is used.

2.1 Dimensionality Reduction

Hyperspectral images (HSI) contain both spatial and spectral information about the scene at hand. How much and what kind of information the spatial and the spectral dimensions carry depends on the situation. Often it is of interest to determine how much and what information that is carried by the spectral dimensions solely. The strong focus on the spectral dimension in hyperspectral imaging can partly be explained by the strong connection between hyperspectral signatures and material properties in a scene. So, henceforth in this paper we will only consider the spectral dimensions. Thus, a hyperspectral image will be thought of as a large number of pixels, each with a spectrum, and with no spatial relation to any other pixels.

Hyperspectral data represented as vectors, Equation (1), is often of high dimensionality (i.e. the number of sampled spectral bands are many). The spectral bands are typically highly correlated which indicate that the data resides in a space of lower dimensionality than M. The dimensionality of the hyperspectral space is also limited by the number of materials with different spectral signatures. If there is a countable number of materials in the scene and each material has its own unique spectrum then the scene can be represented by a number for each material. In this case the information in the scene is digital. If each material in the scene has its unique spectrum and the radiation varies with the lighting then it is possible to reduce the number of dimensions needed to represent the data, and represent the data in a common low dimensional space.

One way of considering dimensionality reduction which is related to the sensor is to reduced the number of wavelength bands and possible adjust their widths to find a smaller set of spectral bands containing the interesting information about the scene. In dimensionality reduction by sub-band selection the problem is to determine how many and which spectral bands that are required to solve the problem at hand. This kind of dimensionality reduction may influence the construction of the sensor. Fewer spectral bands may mean less complexity in the sensor.

Other kinds of dimensionality reduction requires that all bands are captured and then transformed into a space of lower dimensionality. It may still be of importance to reduce the number of dimensions to reduce complexity of signal analysis which then can be done in a subspace of lower dimensionality. It is not given that analysis of the data in the low dimensional space is less complex. However, many results show that the inner representation actually is meaningful and that the representation is useful and makes for example classifiers are more easily trained on the reduced representation. Linear transformations of the hyperspectral data, such as Principal Component Anlysis (PCA) [4] and Independent Component Analysis (ICA) [8], for dimensionality reduction are frequently used pre-processing methods. Kernel-PCA [5], which is a non-linear extension of the (linear) PCA-transformation, is also a frequently used method.

Dimensionality reduction is an encoding problem meaning that we seek an efficient representation of the data and in this context efficient means few dimensions. Efficient representation could also mean few bits if the information is discrete. Encoding problems compared to classification and regression means that no annotated data is needed for the training. We assume that the data contain the information of interest and nothing else. If there is some kind of noise that can be discarded this can

be captured by requiring that the data is reconstructed at least as well as a limit on the given error measure is obtained.

Neural network (NN), and recently deep neural network, has been used for dimensionality reduction of hyperspectral images. Zeng an Trussel [10] used NN to implicitly reduce the dimensionality of hyperspectral images in a classification setting. A data dependent error function - sum of square error (SSE) - was combined with a sparseness criteria on the weight in the NN which penalizes non-zero weights. The NN was trained for classifying hyperspectral signatures and was trained using classified spectral signatures of different materials. The dimensionality reduction was implicit in the training procedure through the sparseness criterion and no explicit low dimensional representation of the spectral signatures was generated.

Chen et al. [3] propose a DL based approach for classification of hyperspectral signatures. Chen et al. uses Stacked Auto Encoder (SAE), as described in Section 3.1, to pre-train the NN using unlabelled training data. The input features to the AE:s are spectral signatures, spatial representation and a joint spectral-spatial representation. The spatial information is represented with a flattened neighbouring region of a PCA transformation in the spectral domain. The SAE is fine tuned using a labelled training set. Chen et al. [2] use a similar approach but instead of using SAE in the unsupervised pre-train step, they use layered Restricted Boltzmann Machines (RBM). The dimensionality reduction, using SAE:s and layered RBM, was implicit and as an pre-processing step for supervised learning of a classifier.

The paper addresses how SAE:s can be used for dimensionality reduction of hyperspectral signatures without explicitly connecting the learnt representation to a specific classification task. What spectral information is preserved?

3 Deep learning and Autoencoder

Deep Learning (DL) is a relatively new approach in pattern recognition that has achieved remarkable results in many applications ([9, 1, 6]). The key concept in DL is to represent the features in a way

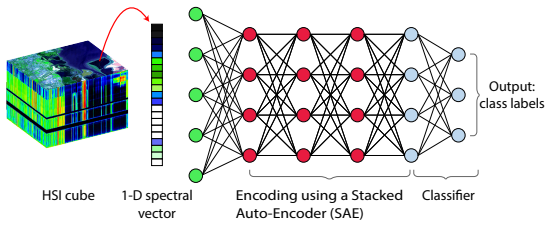

Figure 1: A 1-D spectral feature vector is hierarchically represented - the red nodes - using a Stacked Autoencoder (SAE). The output of the encoding part of the SAE generate a compressed representation of the vector used as features for the classifier. The SAE - red nodes - is optimized using an unlabelled training set (see figure 2), while the classifier require labelled data.The ability represent hyperspectral signatures in a low dimensional space using SAE is investigated.

that improves further analysis. The features is, in DL, represented hierarchically where the different levels in the hierarchy represent different levels of abstraction.

A hierarchical feature representation can, in principal, be learnt from training data using a multilayer neural network (also called Multi Layer Perceptron (MLP)). The weights in a MLP are adjusted by minimizing an error function, commonly the mean-square error (MSE) - equation 9, over a training set commonly using the Back-Propagation (BP) algorithm. BP calculate the gradient, i.e. the derivative of the error function with respect to the weights, of the MLP and the weights are updated by moving in the negative gradient direction.

It is hard, or even impossible, to train a deep MLP, i.e. a MLP with 2 or more hidden layer, due to the *the vanishing gradient problem*. BP calculate the gradient by propagate the error from the output layer toward the input layer and the derivatives decreases as the error is propagated through the layers in the MLP. The gradient is small in the bottom layers of the MLP and the weights are almost unchanged after updating, which make learning very slow (or impossible).

Instead of training a deep neural network directly it can be trained in two phases:

1. **Pre-training** In this phase unlabelled training data is used to learn a hierarchical rep-

resentation of the data. A layer-wise greedy strategy to learn a latent representation of the unlabelled data is used. Autoencoders (AE) and Restricted Boltzmann Machines (RBM) are two frequently used techniques.

2. **Fine-tuning** The weights obtained from the pre-training phase are used for initialising the weights in the full deep neural network. The hierarchical representation learnt in the pre-training phase is a good starting point for fine tuning the deep neural network using labelled data and the back-propagation algorithm.

3.1 Autoencoder

An AutoEncoder (AE) is neural network trained to reconstruct or reproduce its inputs as its outputs. An AE is composed of two parts; an *encoder* and a *decoder*. The encoder takes an input X and maps it to a hidden (or latent) representation U. The latent representation is often of lower dimensionality which imply that the encoder compress the information in the features. The decoder reconstruct the input X from the latent representation U. An AE is an neural network composed of one or more hidden layers which maps the input features onto itself. An AE can be learnt using the BP algorithm and unlabelled training examples. No labels are required because an AE maps the input onto it self independent of any class labels. In the training an optimal encoder and decoder of the input features though the latent representation is learnt. The latent representation can be viewed as a compression of the features containing the most important information.

Multilayer AEs, called Stacked AutoEncoders (SAEs), are constructed using a greedy layer-wise strategy. An AE is trained using an unlabelled training set and some features X. The trained AE maps the features X of the training set to the learnt latent representation, called, U. U is a representation of lower dimensionality than the original features. An AE is trained using U as features resulting in a latent representation V of even lower dimensionality and so on (See Figure 2).

A deep structure, SAE, is constructed by stacking the greedy layer-wise learnt AEs. The encoding part of the SAE maps the original features through a hierarchical representation to a low dimensional compressed representation.

Let $\{\mathbf{x}_i\}_{i=1}^{N_t}$ be a set of training vectors where each vector is the spectral information from one pixel, $\mathbf{x}_i = (x_1, x_2, ..., x_M)$, each vector describes the spectral intensity of M spectral bands. Let $\{v\}_{i_{i=1}}^{N_v}$ be a set of validation vectors. Let $\Phi(x)$ be a non-linear activation function in this case

$$\Phi(x) = 1.7159 \tanh(\frac{2}{3}x) \tag{2}$$

A node in the neural network performs the function

$$f_n(\mathbf{x}) = \Phi(w_0 + \sum_{i=1}^{M} w_i x_i) \tag{3}$$

which can also be written

$$f_n(\mathbf{x}) = \Phi(\mathbf{w}\mathbf{x}^\top). \tag{4}$$

A layer in the network is composed of n nodes

$$f(\mathbf{x}) = \Phi(\mathbf{W}\mathbf{x}^\top) \tag{5}$$

where $\mathbf{W}$ is a matrix with one row for each node or output signal and one column for each input signal.

An autoencoder (one layer in a stacked autoencoder) consists of two layers of nodes, a hidden layer and a output layer. The output layer have a linear activation function, thus

$$f(\mathbf{x}) = \mathbf{W_d}\Phi(\mathbf{W}_e\mathbf{x}) \tag{6}$$

where $\mathbf{W}_e$ is the parameters of the encoding layer and $\mathbf{W}_d$ is the parameters of the decoding layer.

Training is done by iteratively adjusting the parameters of the network using backprogagation.

Let e_{SAE} and d_{SAE} be the encoder respective decoder parts of the SAE then reconstruction of a sample x_n is defined as

$$\tilde{x}_n = d_{SAE} \circ e_{SAE}(x_n) \tag{7}$$

where $\circ$ is the function composition operator. The reconstruction residual r_n of a sample x_n is defined as

$$r_n = x_n - \tilde{x}_n \tag{8}$$

and reconstruction error is defined as $e_i = \|r_i\|$. Norms of interest include the $\|x\|_{L_1}$, $\|x\|_{L_2}$ and $\|x\|_{L_\infty} = max(|x_1|, |x_2|, \cdots, |x_m|)$ norms. The reconstruction error e_i of a hyper spectral signature

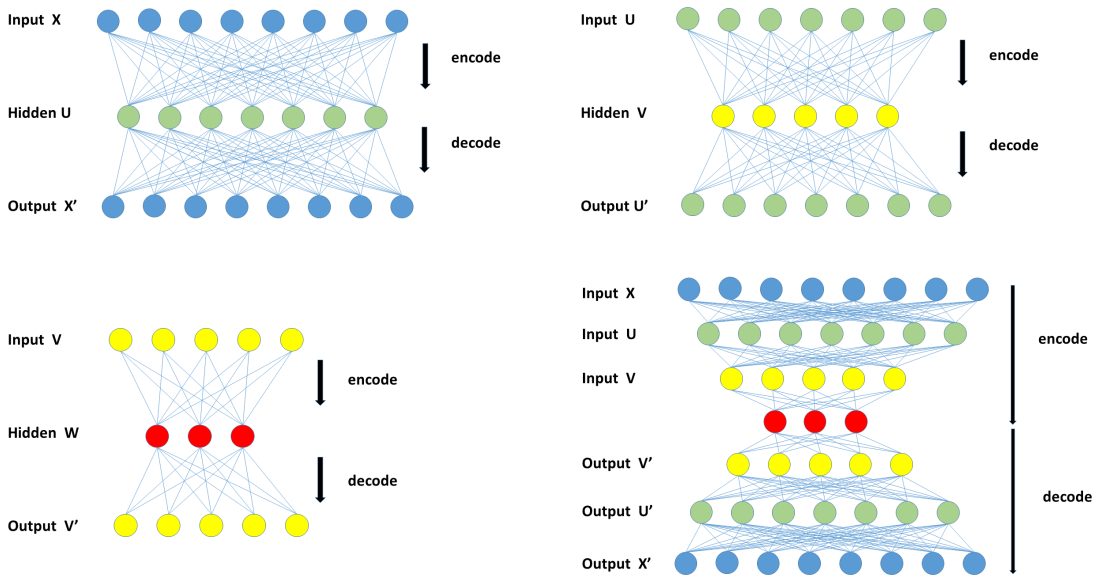

Figure 2: Stacked Autoencoder (SAE) is a greedy strategy to learn a hierarchical feature representation of the data. An AE which maps the features X onto it self though a hidden layer U is learnt. The hidden layer U is a latent representation of lower dimensionality of the features. The latent representation U is used as input to a another AE which learn a latent representation V and so on. A deep structure is composed by stacking the layer-wise trained AEs and is hence called a SAE. A SAE can be used to encode (and decode) original features to any of the dimension the latent representations.

indicate how well it can be approximated in the subspace by the SAE.

The mean-square (reconstruction) error is defined as

$$\mathcal{E} = \frac{1}{N}\sum_{n=1}^{N} \|e_i\|_2^2. \tag{9}$$

where N is the number of samples in training set.

The mean-square reconstruction error of an SAE indicate the performance of the hierarchical representation of the features. The reconstruction error also gives information about the dimensionality of the original data i.e. the number of dimensions which are required for a good reconstruction.

3.2 Dimensionality Reduction using stacked autoencoder

There is a choice of the topology of the neural net, that is the number of hidden layers and the number of neurons in each layer. The topology will determine the number of parameters in the encoder.

There is a relationship between the number of parameters and what the net can be expected to represent, how complex data that can be encoded with the mapping. There is also a relationship between the number of parameters and the amount of training data that is required to be able to determine the parameters. If there are too many parameters then there is a risk that the system will be over trained meaning the the system will learn too much detail of the data. If the system is overtrained the system will not generalise well to coming data from the same source.

There are two important variables, the number of parameters and the reconstruction error.

This is a source coding problem. A small increase in the reconstruction usually means a large improvement in the data rate.

We assume or hope that data can be reduced with respect to the number of dimensions while re-

taining the information about the scene. Not much is known and thus can be assumed about the space in which the relevant aspects of the scene is represented. We hope that a trained multilevel NN will be able to reconstruct the original data with little distortion.

The output from one layer is the input to the next layer so the output of the first layer should be adapted to be suitable as input to the next layer.

A proper choice of the learning rate is critical for the convergence of the learning algorithm in reasonable time. However, there is no guarantee that the algorithm will converge, converge at all or converge to the global minimum point. The learning rate depends on the number of inputs to the neurons. In a SAE the learning rate should be adapted to the depth of the layer, or to the number of inputs that the neurons have in the particular layer.

First experiment. A hyperspectral image from a natural scene was chosen. The scene contains a gravel road and vegetaion close to the road. There are man-made object placed in the scene. In the context where the data were captured the detection of the objects is of interest as could also a classification of the area in terms of different types of terrain. If the data can be represented in a lower dimensional space then the information might be easier to access.

We could also try if the reduced representation could be used for anomaly detection. If data does not fit the trained implicit model then the reconstruction should be worse than for common data from which the model was trained.

The representation obtained with a stacked autoencoder could be compared to for example a PCA, choosing spectral bands and representation with a gaussian mixture model. What should be compared is the number of parameters in the encoder, the number of dimensions in the obtained representation and the reconstruction error. Also the computational complexity of the methods is of interest, both the using the encoder and determining the encoder.

4 Experiments

Let $\{\mathbf{x}_i\}_{i=1}^{N_t}$ be a set of training vectors where each vector is the spectral information from one pixel, $\mathbf{x}_i = (x_1, x_2, ..., x_M)$, each vector describes the spectral intensity of M spectral bands. Let $\{v\}_{i}{}_{i=1}^{N_v}$ be a set of validation vectors.

In our experiments we have used Rasmus Berg Palm's [7] Matlab implementation of a stacked autoencoder.

We have used hyperspectral images (e.g. see Fig. 3) from a natural scene with man made objects placed on the ground. The scene is 5-10 x 5-10 meters and the objects are a few decimeters in size. The images are mostly of undergrowth and some tree trunks. There are also parts of a gravel road. There are some calibration boards in the images but those are disregarded in the experiments. Figure 3 shows a visual image of one of the scenes and a mask showing where in the image to find background, objects and calibration boards.

These hyperspectral images have been collected for a project investigating among other things methods for anomaly detection of surface laid objects. There is a number of different objects.

The available vectors were divided into a training set and a validation set.

First we want to find out if it is possible to find an efficient representation of the hyperspectral data using a stacked autoencoder. With efficient we mean a representation in a space with few dimensions from which it is possible to reconstruct the original data with a small reconstruction error using mean square error to measure the error. The mean square error is a general error measure and not very specific which seems reasonable since we do not have a specific application in mind.

In our data (Fig. 4) the distribution of the intensity varies between the wavelength bands which means that the relative error of some bands contribute much more then other bands to the total mean square error. We hope that information in the original hyperspectral signal is retained in the encoded data.

We will use a stacked autoencoder to learn an efficient representation of the background data excluding the man-made objects in the scene. The compact representation could be used for anomaly detection. Here we consider detection of anomalous pixels since each spectrum is encoded separately without any regard to any other pixels. In many cases the objects have spatial properties that could be considered. But in this case we only want to explore the spectral properties.

If the man-made objects are different from the

typical natural scene then they should be detectable in the sense that the stacked autoencoder fitted for the background will not be able to reconstruct the spectrum of the objects as well as spectrums from the background.

The data was normalised to improve convergence of the training algorithm by translation of each spectral band to give zero mean. The total set of spectral values was then normlised to unit variance. Figure 5 shows a few normalised spectrums.

In classification problems there is a choice of target values. In our case with a stacked autoencoder the last activation function was chosen to be a linear function. The training example data was shuffled for every epoch of the training. There is a choice of batch or stochastic training. In stochastic training the net is adapted to one data vector at a time. In batch learning the gradient is computed as an average for a set of training vectors. This will give a better approximation of the gradient but the stochastic training can get advantage from the random walk. In the implementation we used there was a significant difference in speed of going through an epoch in the learning algorithm. We choose a solution between the extremes by using small batches. The learning rate was determined by trial and error so that the learning algorithm converged reasonable fast. If it learning rate was chosen to high then the algorithm did not converge but kept jumping around a minimum value.

The stacked autoencoder is trained layer by layer. When the first layer is trained then the data is mapped to the latent representation and then these data are used to train the next layer. When all layers are trained they are put together into a stacked autoencoder with several hidden layers. The full stack is trained to fine tune parameters. In the case of a classifier the inner most hidden layer is attached to a few layers used for classification of the data and then the classifier is trained which include updates to the autoencoder layers apart from training the layers doing the classification.

Figure 7 shows an example of the training. 7(a) shows ten data vectors (spectrums) from the validation set and 7(b) shows the reconstruction of the spectrums given by the autoencoder and 7(c) shows the reconstruction residual (Eq 8). The data need to be normalised to suit the autoencoders learning algorithm, 7(d) is the ten vectors of 7a normalized.7(e) is the reconstructed vectors which are reconstructed in 7(b). 7(f) is the reconstruction residual (Eq 8) for the normalized vectors.

7(g) shows an image of the encoding matrix $\mathbf{W}_e$ of the first layer in the stacked autoencoder. The matrix has 60 rows one for each node and 240 columns, one for each insignal. Each point in the image represents the coefficient of one insignal to one node. 7(h) is the endcoded vectors and 7(i) is the corresponding reconstructed vectors. An image of the reconstruction matrix $\mathbf{W}_d$ is shown in 7(j). This matrix has 60 rows, one for each insignal and 240 rows, one for each output signal.

This SAE is composed of two layers. 7(k) shows an image of the parameters of the encoder in the second layer which has 60 input signals (columns) and 30 output signals (rows). 7(l) shows the encoded signals and finally 7(m) is an image of the reconstruction matrix wich 60 rows (insignals) and 30 columns (outputs signals).

The data in the innermost layer supposedly contain most of the information and this is shown by a small reconstruction error. For a specific classifier it is possible that it would suffice with fewer dimensions. From our experiments it seems that the original data has low dimensionality. Only a few nodes is needed in the hidden layer, e. g. 240 - 8 - 240 nodes. This is consistent with using PCA to reduce the dimensions. Only a few dimension contain most of the energy of the signal.

The representation obtained by the autoencoder may be used to detect anomalies. Figure 3 (bottom) shows the reconstruction error. Most of the man-made objects have considerably higher reconstruction error than most background pixels. It seems that the reconstruction error can be used as a measure of anomaly.

We also made an experiment with constructed data consisting of a handful of spectral signatures or functions. In this case the problem could be seen as a discrete coding problem. The innermost layer need only represent the index of the spectral signal. Thus only one dimension would suffice with the level indicating which spectral signal is current. We did not find a stacked autoencoder with only one node in the innermost layer but using 5 nodes in the innermost layer would give perfect reconstruction. In this case it means up to the noise that was added to the training and validation data.

There is a significant difference with regard to computation time when considering batch versus

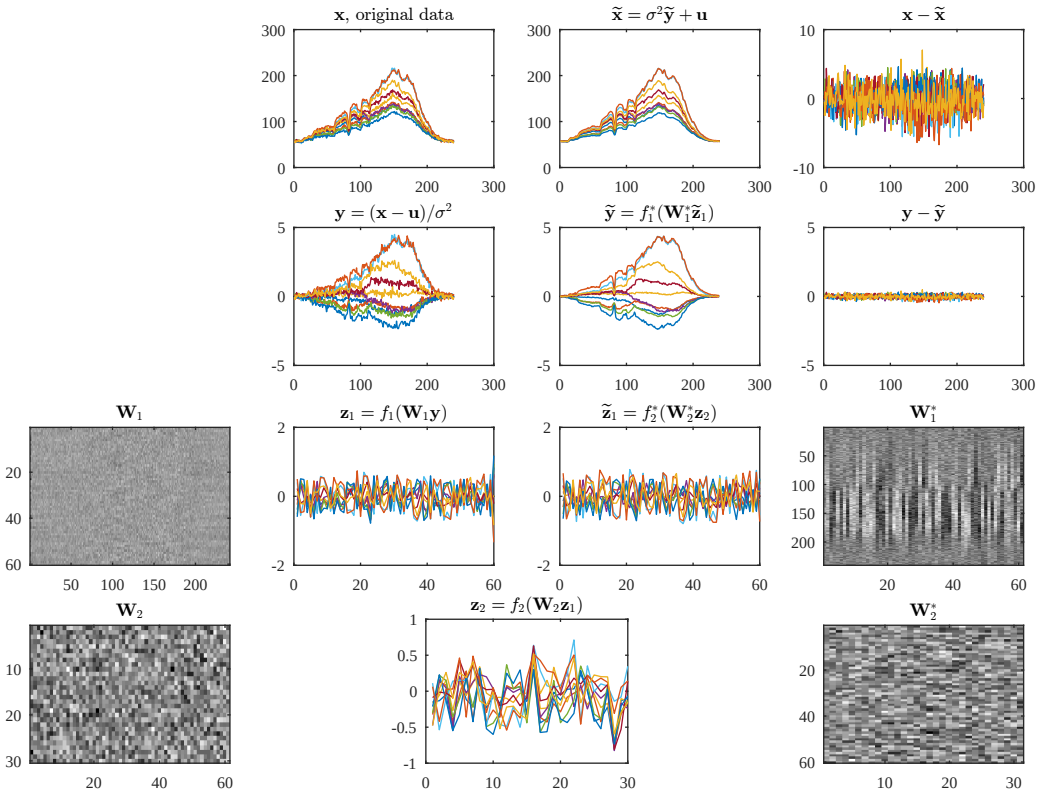

Figure 7: Data in the steps of a stacked autoencoder. The graphs are refeered to row by row as, a b c, d e f, g h i j, k l m. The first row shows (a) ten original spectral data vectors, (b) the ten corresponding reconstructed spectral data vectors from the SAE and (c) the reconstruction residual. The second row shows (d) the corresponding normalized data vectors, (e) the corresponding reconstructed data vectors and (f) the reconstruction residual. The fourth row shows (g) an image of the matrix with the encoder parameters, (h) the latent representation of the data vectors, (i) the reconstructed data vectors and (j) an image of the matrix with the decoder parameters. The fifth row shows (k) an image of the matrix with the encoder parameters of the second layer, (l) the latent representation of the data vectors in the second layer and (m) an image of the matrix with the decoder parameters of the second layer.

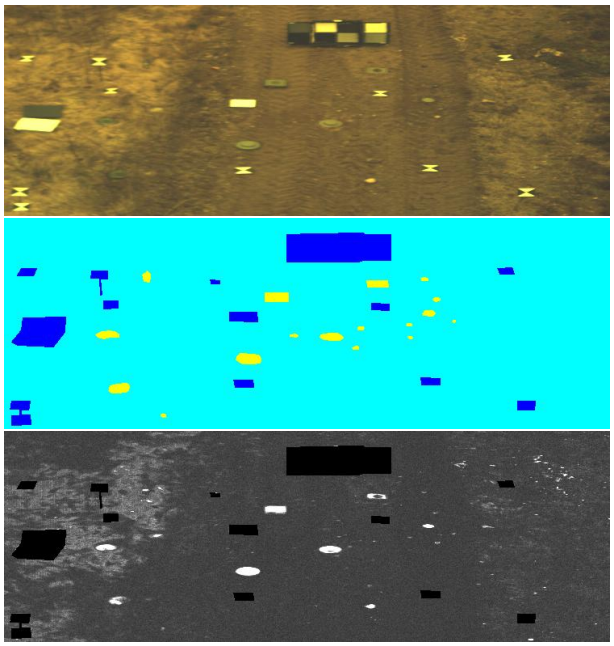

Figure 3: (Top) A visual image of the scene from which the test data is taken. (Middle) The different regions in the image, background (turquoise) objects (yellow) and reference boards (blue). (Bottom) The reconstruction error when spectrums are represented by a stacked autoencoder with 240,60,30,60,240 nodes in the layers.

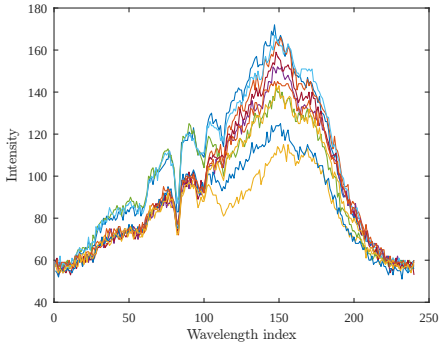

Figure 4: Examples of spectrums from some pixels in the test data.

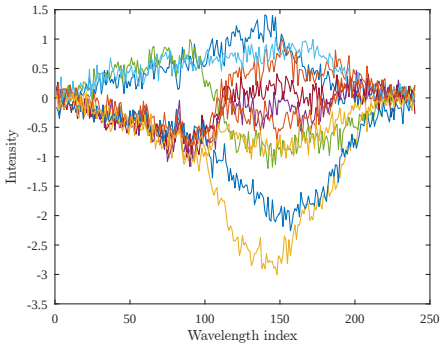

Figure 5: Examples of normalised spectrums used in the training.

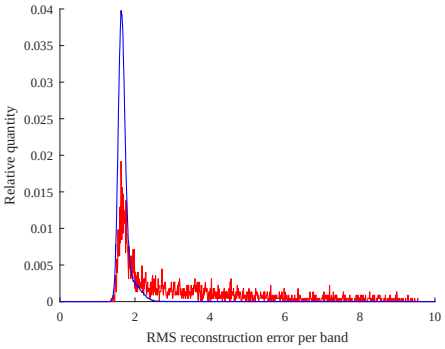

Figure 6: Histogram of the RMS error per spectral band for back- and forground pixels separately.

random choice of a single training vector. There is a significant gain in time to compute all vectors once.

5 Conclusion

In this paper we describe how stacked autoencoders (SAE) can be used to reduce the spectral dimensionality of hyper spectral data. We show that the hierarchical representation learned by a SAE can encode the spectral information with small mean square error. The results are shown using hyperspectral signatures from images of a natural scene containing man-made objects. We also show how the SAE can be used for anomaly detection, with promising results, on the same dataset.

Even if SAEs is a promising tool for dimensionality reduction and learning compact spectral representations it is a rather steep learning curve before one can apply SAEs.

Acknowledgement

This work is funded by the Swedish Defence Materiel Administration, FMV and the Swedish Armed Forces.

References

[1] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[2] Y. Chen, X. Zhao, and X. Jia. Spectral - spatial classification of hyperspectral data based on deep belief network. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, PP(99):1–12, 2015.

[3] Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu. Deep learning-based classification of hyperspectral data. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 7(6):2094–2107, June 2014.

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

[5] M. Fauvel, J. Chanussot, and J. A. Benediktsson. Kernel principal component analysis for feature reduction in hyperspectrale images analysis. In *Proceedings of the 7th Nordic Signal Processing Symposium - NORSIG 2006*, pages 238–241, June 2006.

[6] David Gustafsson, Henrik Petersson, and Mattias Enstedt. Deep learning: Concepts and selected applications. Technical Report FOI-D-0701-SE, Swedish Defence Research Agency (FOI), Sweden, December 2015.

[7] R. B. Palm. Prediction as a candidate for learning deep hierarchical models of data. Master's thesis, Technical University of Denmark, DTU Informatics, 2012.

[8] A. Villa, J. Chanussot, C. Jutten, J. A. Benediktsson, and S. Moussaoui. On the use of ica for hyperspectral image analysis. In *2009 IEEE International Geoscience and Remote Sensing Symposium*, volume 4, pages IV–97–IV–100, July 2009.

[9] Xue wen Chen and Xiaotong Lin. Big data deep learning: Challenges and perspectives. *Access, IEEE*, 2:514–525, 2014.

[10] Huiwen Zeng and H.J. Trussell. Dimensionality reduction in hyperspectral image classification. In *Image Processing, 2004. ICIP '04. 2004 International Conference on*, volume 2, pages 913–916 Vol.2, Oct 2004.