Deep Learning with Stacked AEs & RBMs

DD2437 - Artificial Neural Networks & Deep Architectures - Lab 4

Niels Agerskov agerskov@kth.se
Lukas Bjarre lbjarre@kth.se
Gabriel Carrizo gabcar@kth.se

This lab will examine two different artificial neural network structures, Auto Encoders (AE) and Restricted Boltzmann Machines (RBM). Their effectiveness in a learning task and the effect of the layer depth of the models will be tested and evaluated.

1 Feature learning

In this first task shallow versions of both models are trained as benchmarks for the later deeper versions. The dataset used is a subset of the MNIST dataset containing 28×28 images of handwritten digits from 0 to 9 together with correct labels of the written digit. All the pixel values has for simplicity's sake been converted to binary values via simple thresholding. A total of 10000 images are used from the dataset, which has been further subdivided into a training set of size 8000 and a validation set of size 2000.

1.1 Hidden unit size

The input size hyperparameter for both of the models is decided by the shape of the data. In our case we require $28 \times 28 = 784$ input nodes, one for each image pixel. We do however have a choice in the number of hidden units, n_{h}.

Both models were trained with $n_{\mathrm{h}} = 50, 75, 100, 150$ hidden units. The error curves on the validation set during the training are displayed in fig. 1 for the RBM, and in fig. 2 for the AE using Stochastic Gradient Descent (SGD). However, the AE clearly converges to the same values no matter the number of hidden units. Complementary to using SGD to train the AE ADADELTA was also used, which error curve can be seen in fig. 3. ADADELTA shows a similar improvement given more hidden units as the error curves for the RBM. The errors are also on one order of magnitude smaller compared to the errors using SGD, which is why the ADADELTA trained AE is only used furthermore.

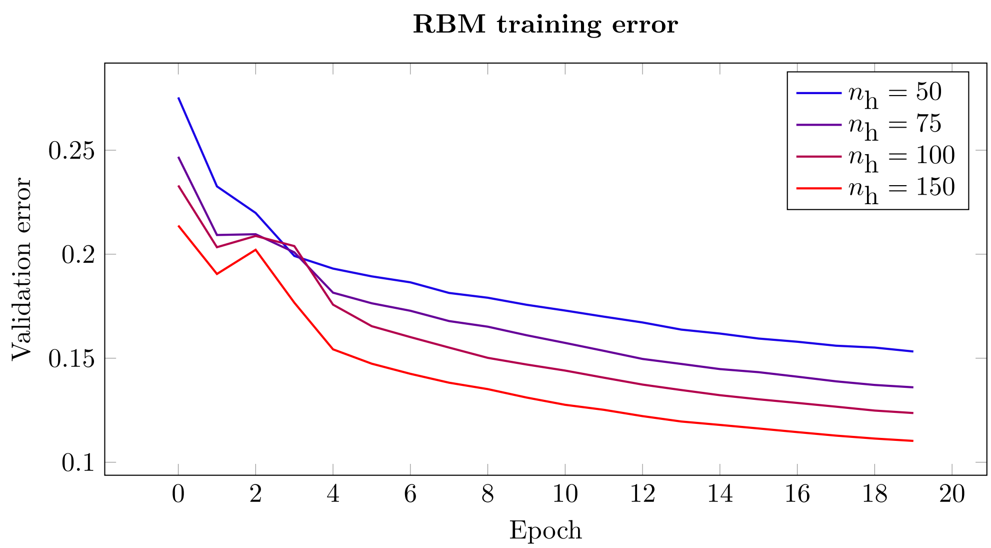


Figure 1: Error curves on the validation set for the RBM.

The quality of the models can be seen in fig. 4, where one image of each class have been used to get both models recalled versions.

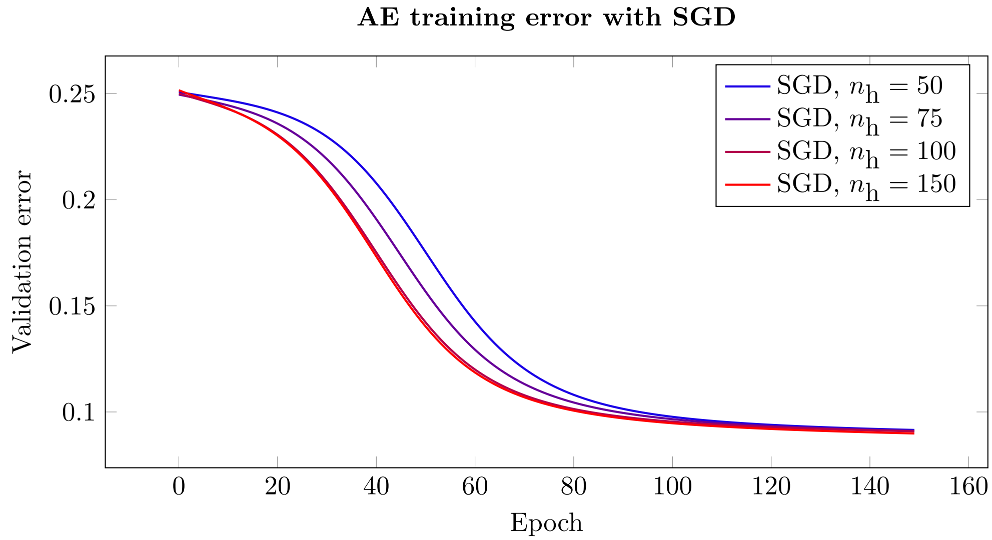


Figure 2: Error curves on the validation set for the AE using SGD.

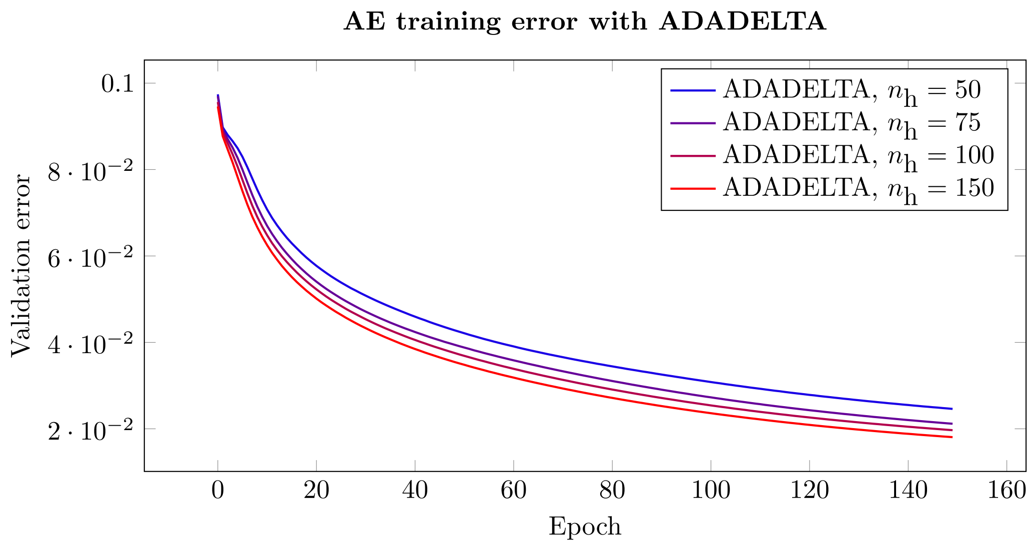


Figure 3: Error curves on the validation set for the AE using ADADELTA.

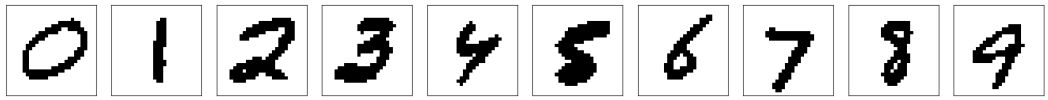
1.2 Learned features

The learned weights of each model can be examined to get an idea what and how the models are learning. By reshaping the weight vectors back into 28×28 grids each hidden units weights can be represented as images where each pixel value corresponds to the strength of that weight from the given pixel to the hidden unit. Plots for these are found in fig. 6. The AE models seems to in general have better trained hidden nodes as more of the components seems to have learned a specific shape instead of something that resembles white noise.

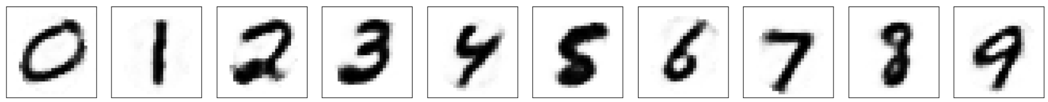
2 Deep architectures

The next step is to make the architectures deep by introducing more hidden layers. For the Auto Encoders this corresponds to adding more encoding layers after the first encoder layer, resulting in a structure called Stacked Auto Encoders (SAE). The deeper version of the RBM is also constructed by adding more hidden layers after the first, resulting in a Deep Belief Network (DBN).

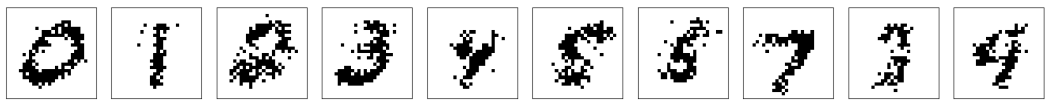
The task for the deep architectures was to classify a given image to the corresponding digit. To



(a) Input patterns

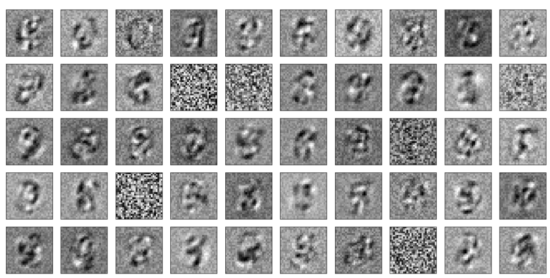


(b) Recalled patterns of the AE, 150 hidden units

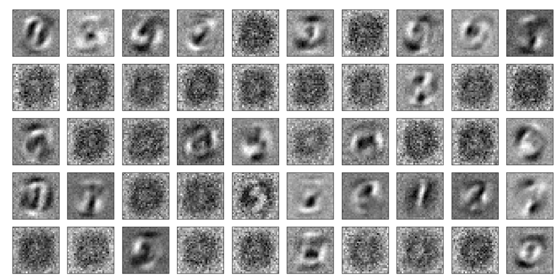


(c) Recalled patterns of the RBM, 150 hidden units

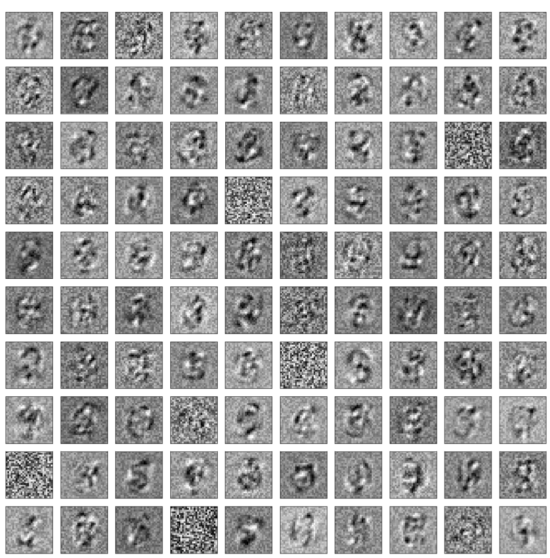
Figure 4: Recalled images from the models.



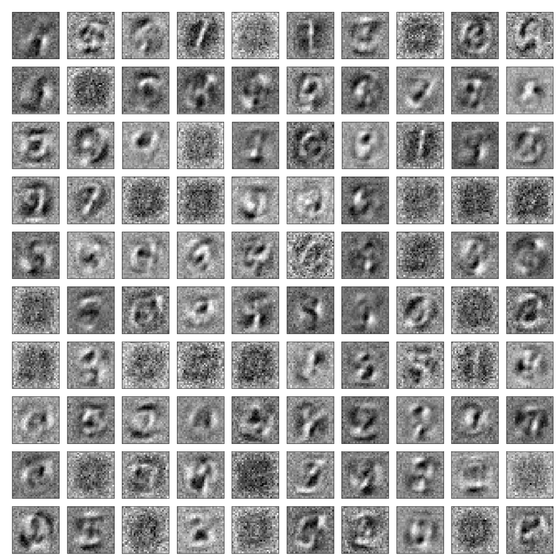
(a) AE weights, 50 hidden units.



(c) RBM weights, 50 hidden units.



(b) AE weights, 100 hidden units.



(d) RBM weights, 100 hidden units.

Figure 5: Image representations of the hidden weights in the AE and RBM models for 50 and 100 hidden units.

achieve this one additional perceptron layer was added to the end of all the deep models. The idea of the structure then becomes that the deep layers are supposed to find and model hierachical representations of the images in lower and lower dimensions, which the final perceptron layer can use to classify the digit.

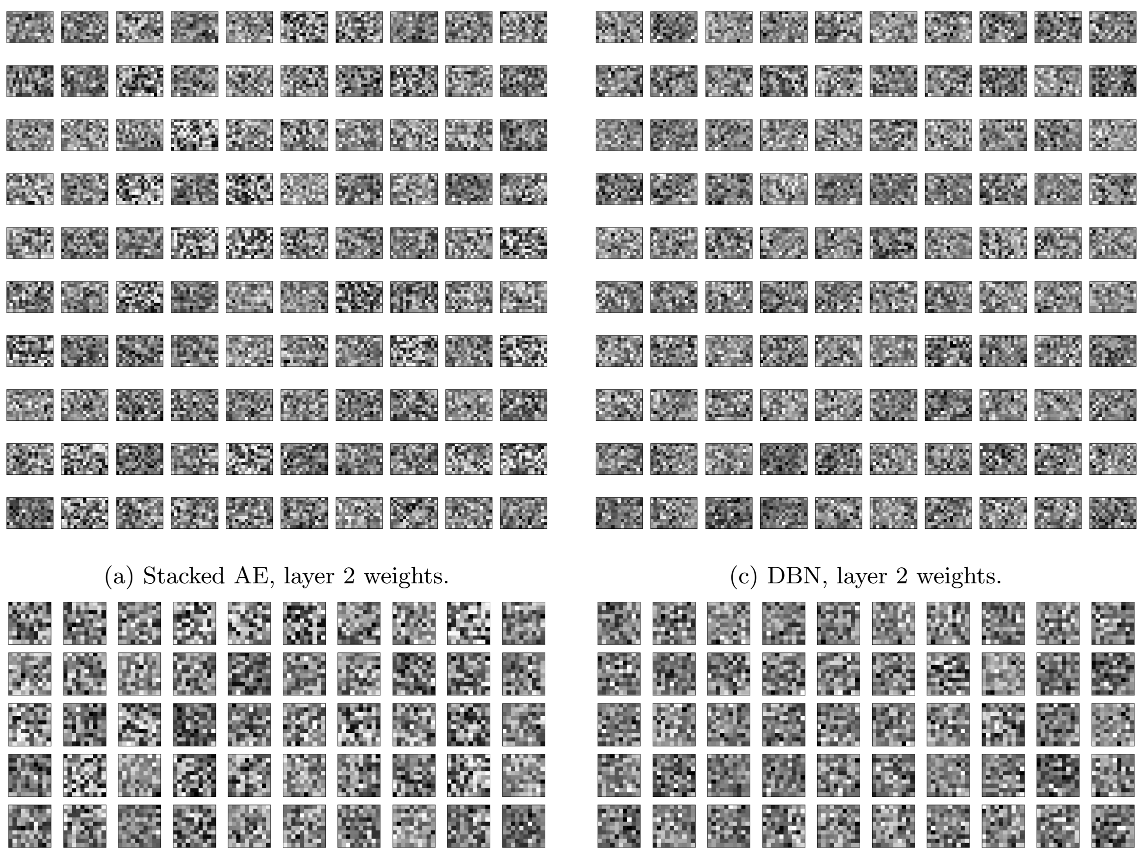
The method used to train the networks was to pre-train the hidden layers one-by-one using the unsupervised methods used previously. After one layer was trained the dataset was transformed using the trained weights and then the next layer used this transformed set to train its weights. One final supervised training run on the entire architecture was also used at the end to fine-tune the weights.

Models with different layer depths ranging from one to three were trained and evaluated. The size of each hidden layer was determined first from the best performing versions of the shallow models then decreasing this size for the consecutive layers. The best hidden unit size can be seen from fig. 1 and fig. 3 which in both models is $n_{\mathrm{h}} = 150$. This was set to be the first hidden layer size. To decrease the consecutive layer sizes the two other sizes was set to 100 and 50.

Table 1: Accuracy of different deep architectures

Layers	DBN	SAE
0	0.902	
1	0.937	0.937
2	0.907	0.943
3	0.859	0.866

In table 1 the accuracy of the trained models on the validation set is presented. The 0 layer size represents a simple perceptron layer trained directly on the dataset. We can see that the performance in general is similar for the different setups, with the difference of seeing a slight decrease in performance for the deepest setups. Overall the best performance was seen with the Stacked Auto Encoder with two hidden layers.



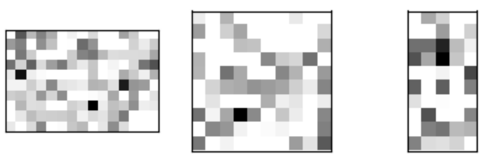
(a) Stacked AE, layer 2 weights.

(c) DBN, layer 2 weights.

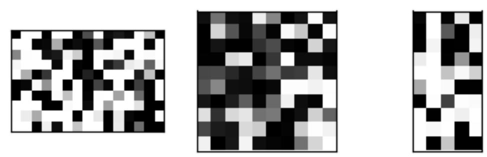
(b) Stacked AE, layer 3 weights.

(d) DBN, layer 3 weights.

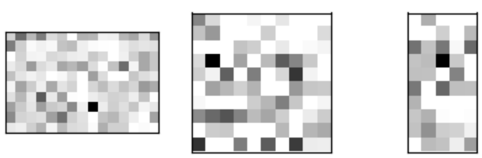
Figure 6: Image representations of the hidden weights in the AE and RBM models for 50 and 100 hidden units.



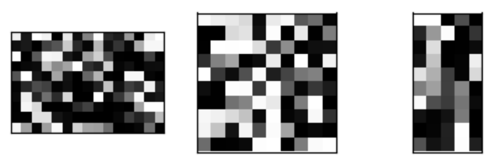
(a) Activations of hidden units in SAE for a 0 digit.



(b) Activations of hidden units in DBN for a 0 digit.



(c) Activations of hidden units in SAE for a 8 digit.



(d) Activations of hidden units in DBN for a 8 digit.