Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 4

Restricted Boltzmann Machines and Deep Belief Nets

Jakub Růžička, Hauke Wernecke, Peng Zhang
(Group 14)

October 25, 2023

1 Main objectives and scope of the assignment

Our major goals in the assignment were

- Train Restricted Boltzmann Machines (RBM) on image data and read out weights as receptive fields.
- Use Deep Belief Networks (DBN) to classify images, but also generate new images on the basis of the mnist-data set.
- Understanding the relation between training individual RBMs and teh overall structure of DBNs.

2 Methods

Custom python scripts.

3 Results and discussion

3.1 RBM for representing MNIST images

Parameter:

- input dimension: 28 x 28 pixels
- Batch size: 20

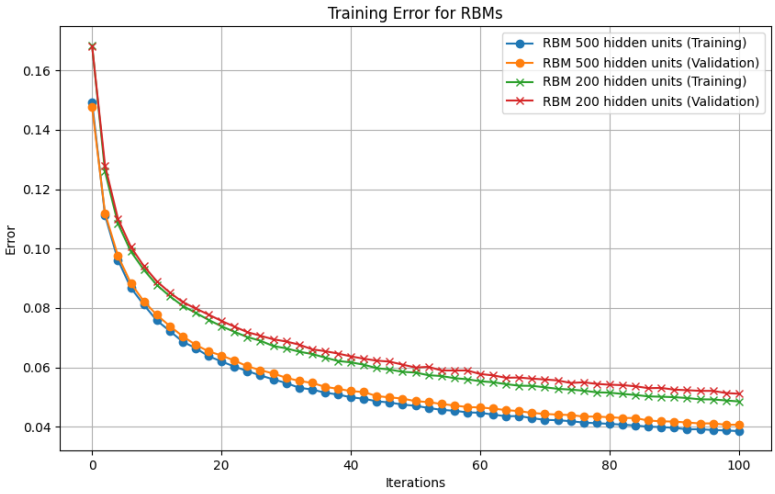


Figure 1: Mean-squared error of the reconstruction error in RBMs with 200 and 500 nodes in teh hidden layer, respectively (See legend).

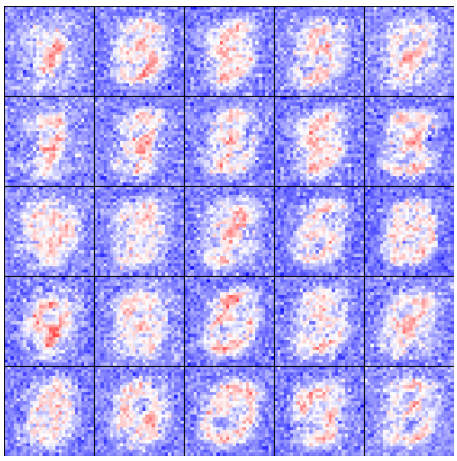
- Training epochs: 100
- number of hidden units: 500 / 200

In the first part, the RBM is trained to learn useful representations of the input data (visible layer) in the hidden layer. Although the reconstruction error is not used as a learning signal, it is a common and decent measure to monitor the learning progress. As shown in Fig. 1 , a network with 500 nodes in the hidden layer has an overall lower reconstruction error than the RBM with 200 nodes. The validation error is calculated in a different set of training samples, which the network has not seen for training.

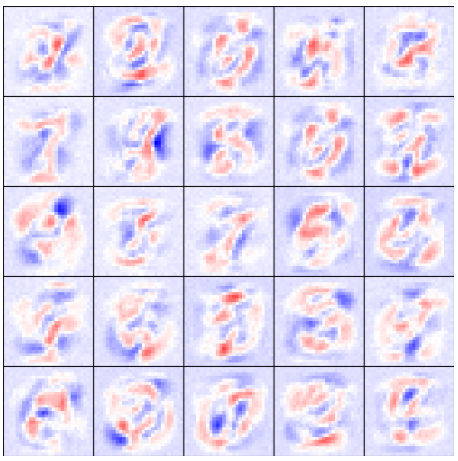
In order to study the receptive fields, we plot the weights of 25 random nodes of the hidden layer. They represent which parts of the input image the specific nodes are most sensible to. One can see in Fig. 2a, that an interpretation of the receptive fields is yet difficult. However, a few features of written numbers can be seen, e.g. threes, zeros, maybe even nines. In contrast, with 500 nodes in the hidden layer, a useful representation is still missing. In both cases, after 30 epochs two major changes become visible: (1) the outer parts of the image are represented with very low weights compared to the central part, and (2) the represented features seem more complex as mixture of multiple numbers.

3.2 DBN for classification

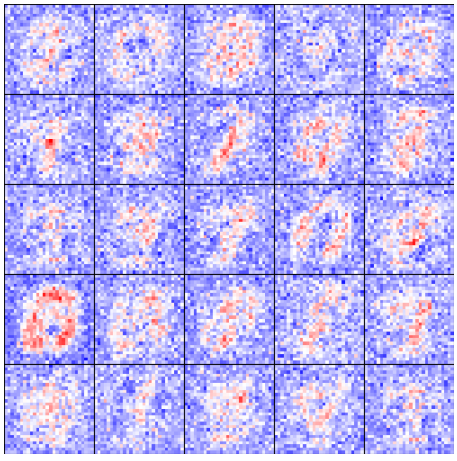
After the greedy layer-wise training of our DBN model, we tested the training accuracy using the train_imgs and train_lbls, then we get an accuracy of 76.79%,



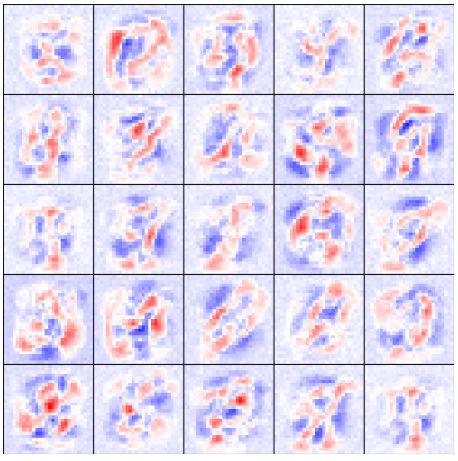
(a) 200 hidden nodes: Receptive fields of 25 random nodes after 1 epoch.



(b) 200 hidden nodes: Receptive fields of 25 random nodes after 30 epoch.



(c) 500 hidden nodes: Receptive fields of 25 random nodes after 1 epoch.



(d) 500 hidden nodes: Receptive fields of 25 random nodes after 30 epoch.

```
dbn.recognize(train_imgs,  train_lbls)
dbn.recognize(test_imgs,  test_lbls)
```

```
loaded rbm[vis--hid] from trained_rbm
loaded rbm[hid--pen] from trained_rbm
loaded rbm[pen+lbl--top] from trained_rbm
accuracy = 76.79%
accuracy = 75.49%
```

Figure 3: RBN for classification.

then we test it on the test_imgs and test_lbls, and get an accuracy of 75.49%. As show in 3

3.3 DBN for generation

After passing the 75% accuracy baseline with our Deep Belief Network on the classification task, the task of generation was also examined. As the training time can be very high, when utilizing all training data - about 50.000 examples, when 10,000 are left out for testing and another 10.000 for validation - only a portion of the available data were used.

The architecture consisted of three Restricted Boltzmann machines, with the size 784-500-500+10-2000. We trained using the greedy pretraining on 10.000 samples, with the learning rate 0.01 for $2,000$ epochs. The receptive fields for the randomly sampled 25 hidden units are shown on Figure 4. Even though the receptive field seems reasonable, the results obtained by the generation itself were far from ideal. Most digits were illegible, see Figure 5. We can see that the network learned something, but it was not enough to generate images.

What we observed was, that using more training examples resulted in weights being more blurry and all starts looking similarly - close to some bad zero.

4 Final remarks

Even though we knew that learning the RBMs and from them build DBNs may be time demanding, the length of training times surprised us. The expected gap in difficulty between the classification and generation could be observed from the results.

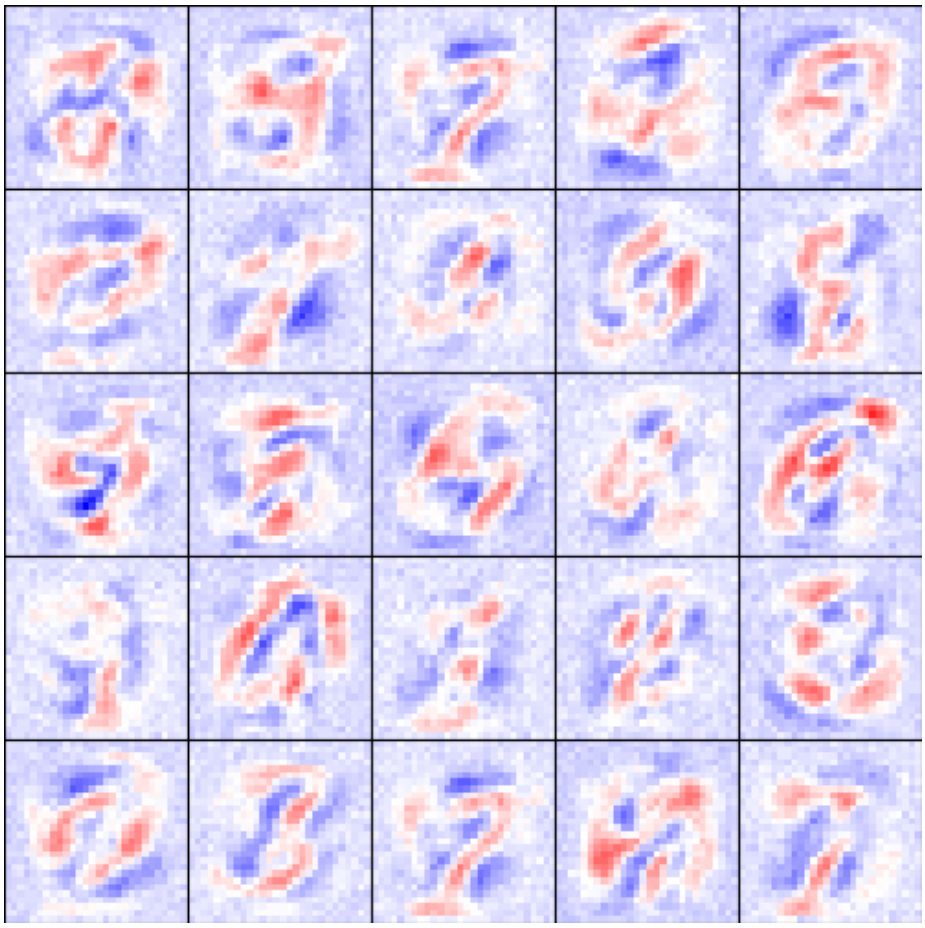


Figure 4: Receptive fields for 25 hidden units of the bottom RBM in the DBN stack.

Figure 5: Numbers from 0 to 9 as generated by our DBN.