Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 4

Restricted Boltzmann Machines and Deep Belief Nets

Franco Ruggeri, Fredrik Danielsson

June 13, 2020

1 Main objectives and scope of the assignment

Our major goals in the assignment were:

- to implement a Restricted Boltzmann Machine trained with Contrastive Divergence.
- to understand the key aspects of Restricted Boltzmann Machines.
- to implement a Deep Belief Net pre-trained with a greedy layer-wise approach and fine-tuned with the wake-sleep algorithm.
- to understand the functionality of Deep Belief Nets, including both the generative and recognition modes.
- to study the advantages related to the size of the network, including both the size of the layer (i.e. number of units) and the depth of the network (i.e. number of layers).

The main limitation was the available time. Because of that and considering that the focus was not on achieving the best possible accuracy, we decided to use the suggested hyper-parameters (learning rate, batch size, etc.) without choosing them with cross validation.

2 Methods

We used the provided code in *Python* for all the tasks with the support of just the *numpy* and *matplotlib* libraries. We followed the indications included in [1] for RBMs and [2] for DBNs (in particular the pseudo-code of appendix B for the wake-sleep algorithm).

Dataset	Number of hidden units	Reconstruction loss
Training	200	0.01180 ± 0.00002
Training	300	0.01118 ± 0.00002
Training	500	0.01095 ± 0.00001
Test	200	0.02888 ± 0.00005
Test	300	0.02740 ± 0.00003
Test	500	0.02681 ± 0.00002

Table 1: Reconstruction loss of RBMs with varying number of hidden units. Means and standard deviations computed on 5 runs.

The hyper-parameter setting we used is: $batch\_size = 20$, $n\_epochs = 10$, $\eta = 0.01$. For contrastive divergence (RBM and DBN pre-training) we added also momentum ($momentum = 0.7$). We did not include it when fine-tuning to avoid picking up too much speed and going too far away from the pre-trained solution. This choice was made empirically, because with momentum the fine-tuning step greatly worsened the accuracy.

3 Results and discussion

3.1 Restricted Boltzmann Machines

One (though not perfect, see [1]) metric to analyze the convergence and stability of the RBM is the reconstruction loss. In fact, if a training image has low energy (or equivalently, high probability), there is high probability that the state does not change much in one alternating Gibbs sampling and so the reconstruction is similar to the original image, despite the bottleneck introduced by sampling in the hidden layer.

The reconstruction losses for RBMs with varying number of hidden units trained with *contrastive divergence* are shown in figure 1 and table 1. As expected, the network fits better the training data with more hidden units, since the model has more degrees of freedom.

Considering the architecture with 500 hidden units, some randomly selected receptive fields and reconstructions of test images are shown in figure 2. The receptive fields show some templates of several digits mixed together. Regarding the reconstructions, instead, they are clearly very good and confirm that the network learnt a representation of the hand-written digits.

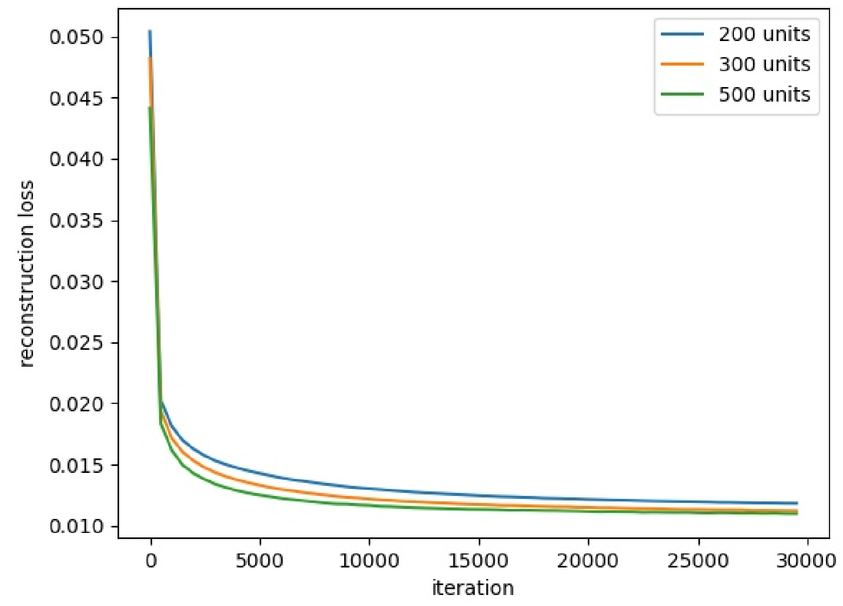

Figure 1: Reconstruction loss on the training set for RBMs with varying number of hidden units (learning curve).

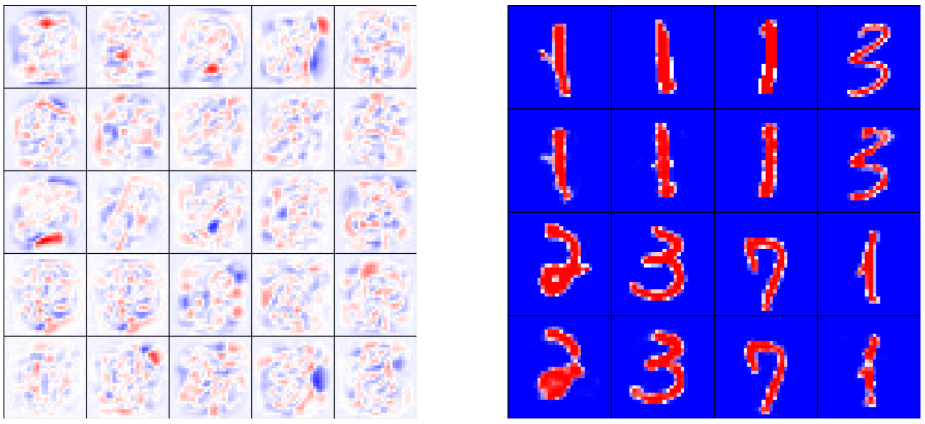

Figure 2: Receiptive fields (a) and reconstructions of some test images (b) for an RBM with 500 hidden units trained for 10 epochs. In (b), the even lines (lines 0 and 2) contain the original images, while the odd lines (lines 1 and 3) contain the reconstructions.

Dataset	Architecture	Learning step	Accuracy
Training	784-500-500+10-2000	Pre-training	0.888 ± 0.006
Training	784-500+10-2000	Pre-training	0.916 ± 0.001
Training	784-500-500+10-2000	Fine-tuning	0.924 ± 0.001
Training	784-500+10-2000	Fine-tuning	0.942 ± 0.001
Test	784-500-500+10-2000	Pre-training	0.891 ± 0.006
Test	784-500+10-2000	Pre-training	0.919 ± 0.003
Test	784-500-500+10-2000	Fine-tuning	0.925 ± 0.002
Test	784-500+10-2000	Fine-tuning	0.945 ± 0.002

Table 2: Test accuracy of the studied DBNs. Means and standard deviations computed on 5 runs.

3.2 Deep Belief Nets - Greedy layer-wise pre-training

In this and in the following sections we are going to consider the architecture proposed in [2] (784-500-500+10-2000).

First, we pre-trained the stack of RBMs with a greedy layer-wise approach. The reconstruction losses are shown in figure 3, confirming that each layer was pre-trained well until convergence.

Then, we evaluated the pre-trained DBN in both recognition and generative modes. The classification accuracy is very good considering the greedy approach and is reported in table 2. On the other hand, most of the generated images, shown in figure 4, are hardly interpretable.

3.3 Deep Belief Nets - Supervised fine-tuning

After pre-training, we fine-tuned the DBN with the *wake-sleep algorithm*. The training accuracy (learning curve) is shown in figure 5.

In table 2 and figure 6 is shown the evaluation for recognition and generative modes, respectively. Both are clearly improved after fine-tuning. The generated images, in particular, are much more convincing.

3.4 Deep Belief Nets - Simplified architecture

We trained and evaluated also a simplified DBN with architecture 784-500+10-2000 (one layer less). The accuracy is reported in table 2. We found that, with the given hyper-parameters and learning process, the simplified architecture even outperforms the deeper one. We think that this may be due to the more complex landscape of the loss function (likelihood), which has much more local minima, or to small gradients caused by the depth.

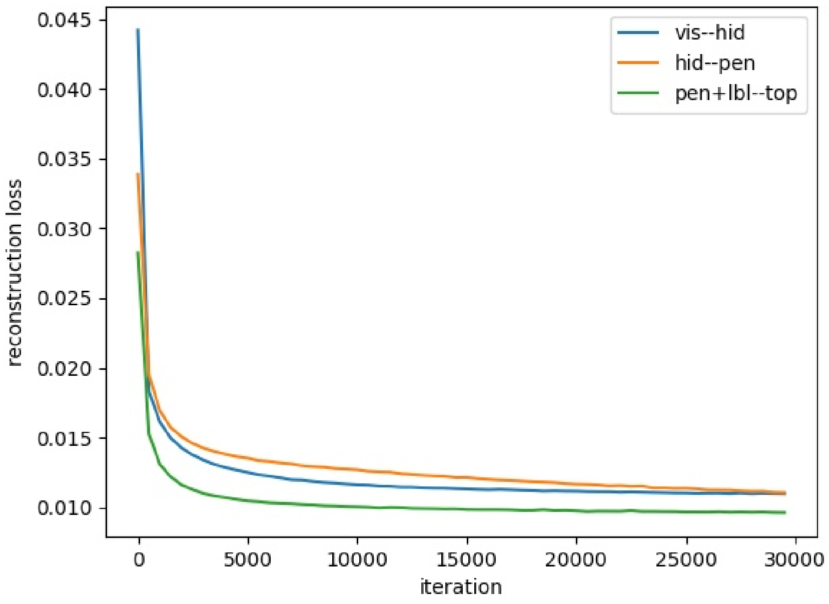

Figure 3: Reconstruction loss while pre-training the stack of a DBN with architecture 784-500-500+10-2000.

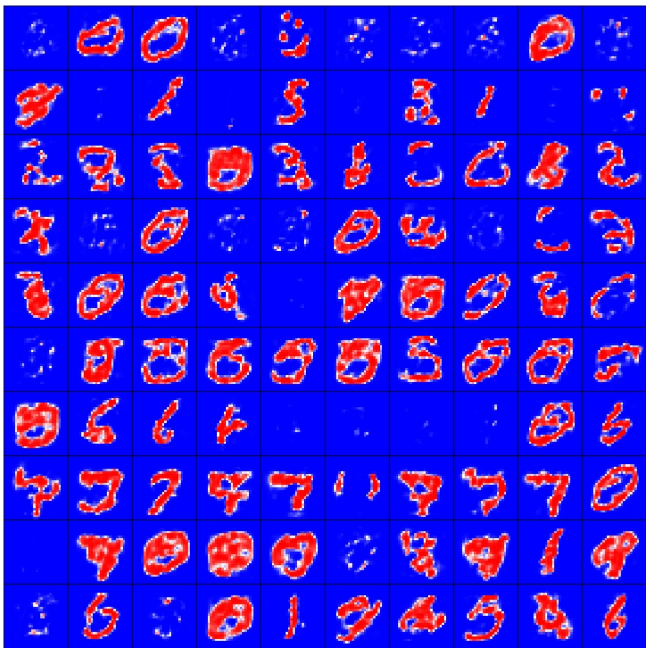

Figure 4: Images generated by a pre-trained DBN with architecture 784-500-500+10-2000. Each line corresponds to a digit (line 0 → digit 0, line 1 → digit 1, and so on).

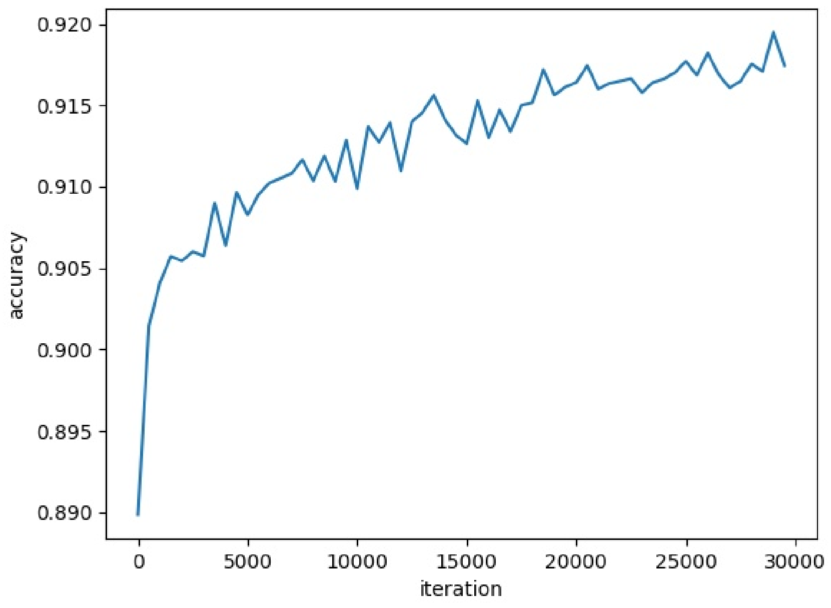

Figure 5: Training accuracy while fine-tuning the pre-trained DBN with architecture 784-500-500+10-2000.

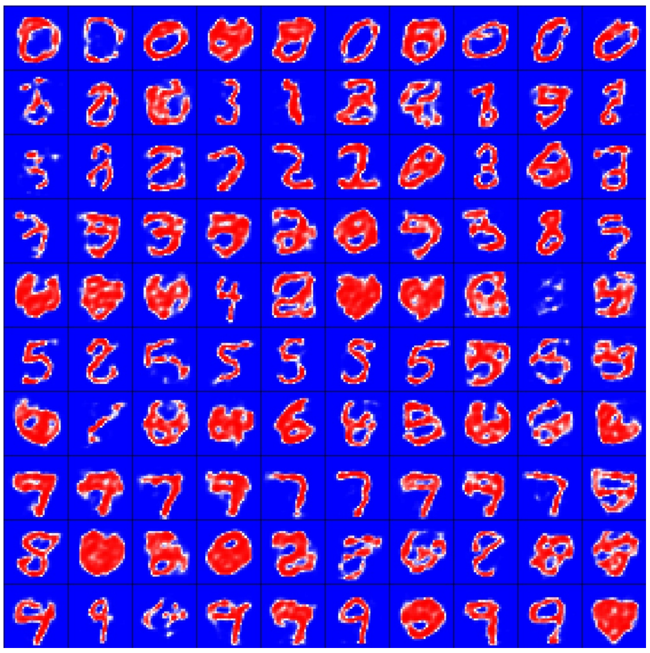

Figure 6: Images generated by a fine-tuned DBN with architecture 784-500-500+10-2000. Each line corresponds to a digit (line 0 $\rightarrow$ digit 0, line 1 $\rightarrow$ digit 1, and so on).

4 Final remarks

The assignment was very useful to understand the key concepts of Restricted Boltzmann Machines as well as Deep Belief Nets. Moreover, in this assignment we got a practical experience with the historically first way of training deep architectures, that is using greedy layer-wise pre-training as initialization in order to overcome the difficulties related to vanishing and exploding gradients.

References

[1] Geoffrey Hinton. “A Practical Guide to Training Restricted Boltzmann Machines (Version 1)”. In: *Technical Report UTML TR 2010-003, University of Toronto* 9 (Aug. 2010). DOI: `10.1007/978-3-642-35289-8_32`.

[2] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. “A fast learning algorithm for deep belief nets”. In: *Neural computation* 18.7 (2006), pp. 1527–1554.