Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 4

Restricted Boltzmann Machines and Deep Belief Nets

Nouhaila AGOUZAL, Abdessamad BADAOUI and
Nasr Allah AGHELIAS

October 16, 2023

1 Main objectives and scope of the assignment

Our major goals in the assignment were :

- to Understand the fundamental concepts in RBM learning processes.
- to Apply unsupervised greedy pretraining and supervised fine-tuning in RBM layer setups.
- to Design multi-layer neural networks using RBM layers for classification tasks and explore the generative capabilities of Deep Belief Networks (DBNs).

2 Methods

The programming was done in python and we used the numpy library for mathematical calculations and matplotlib.pyplot to generate graphs. We didn't use any existing library for developing RBM layers of a DBN. And we relied on the attached code framework.

3 Results and discussion

3.1 RBM for recognising MNIST images

In our first task of developing an RBM with binary stochastic units, we initialize our weights using a zero-mean normal distribution. To train it, we use

contrastive divergence over minibatches of size 20. The **reconstruction loss** is a good metric for assessing the model's performance in terms of image reconstruction. However, it is important to note that the reconstruction loss alone is not indicative of convergence or stability. This is because RBMs seek to minimize energy, not the reconstruction loss. Furthermore, contrastive divergence, even if it works very well, does not minimize the energy, as it does not maximize the likelihood of the data distribution. Therefore, the reconstruction loss is not a reliable metric for monitoring stability. Instead, we should focus on monitoring the convergence of weights and biases themselves, as this provides a safer and more accurate way to assess the model's progress.

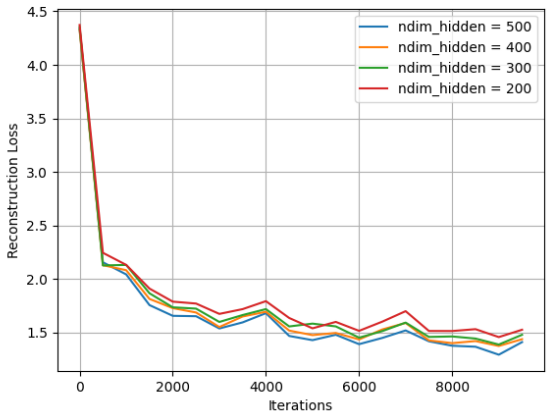
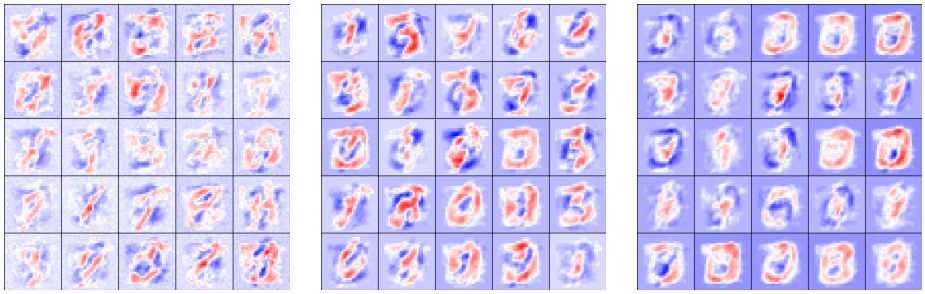


Figure 1: Effect of the number of hidden units on RBM Reconstruction Loss

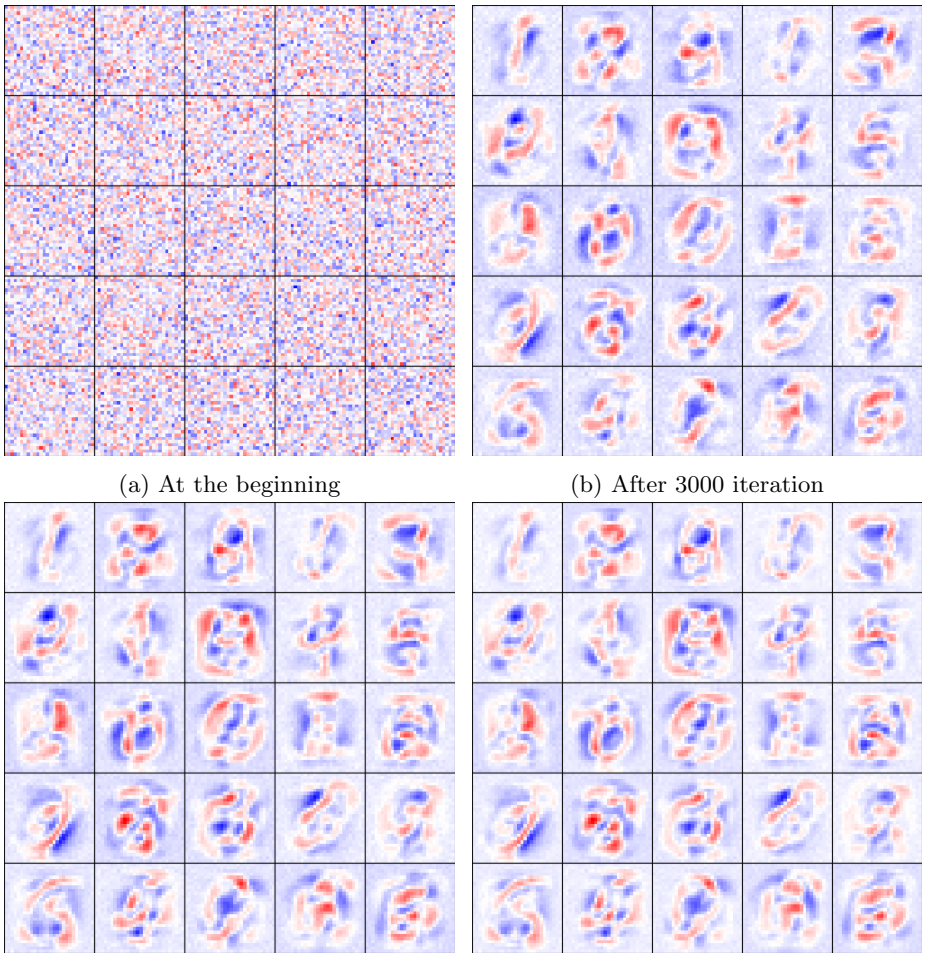
As we can see in Figure 1, increasing the number of hidden units in an RBM typically results in better reconstruction performance, as it allows the model to capture more complex patterns in the data. Striking the right balance between improved reconstruction and computational efficiency is crucial and often necessitates finding an optimal number of hidden units tailored to the dataset and task. It's important to note that while reconstruction loss is a useful metric, it may not be the most suitable criterion for model selection in all cases, prompting the need to consider additional evaluation measures.



(a) $\lambda = 0.1$ (b) $\lambda = 0.5$ (c) $\lambda = 1$

Figure 2: Weights for 25 different units

We have examined the impact of weight decay regularization on the weights at the Figure 2. It's evident that as we increase the regularization term, the weights become more blob-like and less grainy. Their distribution also becomes somewhat continuous, with a significant portion of the weights converging to very small values, as represented by the blue zones.



(a) At the beginning

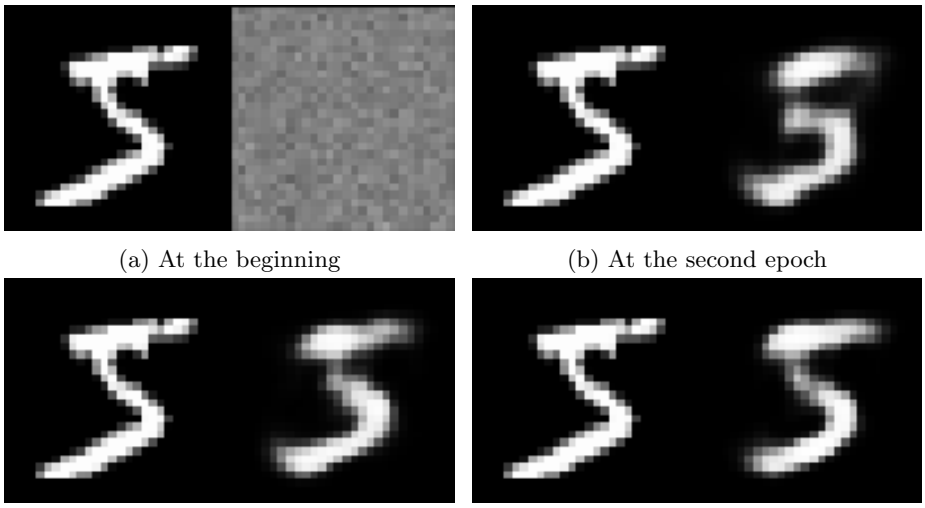
(b) After 3000 iteration

(c) After 6000 iteration

(d) After 9950 iteration

Figure 3: Weights for 25 different units

As shown in Figure 3, the weights commence as uncertain, blurry images that gradually transform into representations resembling digits. Initially, these weights are randomized or initialized to exhibit a semblance of noise or blurriness. As the training progresses, they undergo a phase of progressive refinement, gradually capturing fundamental features such as edges and simple patterns, albeit still appearing somewhat blurry. As the model further refines its features through learning, the weights start resembling digit-like patterns, since the data being processed is closely associated with digits. With continued training, the weights become sharper and more defined, taking on recognizable digit shapes with clearer edges and patterns. Ultimately, as the model converges, the weights



(a) At the beginning (b) At the second epoch

(c) At the third epoch (d) At the fourth epoch

Figure 4: Input image and weights corresponding to its reconstruction

distinctly capture features crucial for discriminating between different digits or patterns. This evolutionary progression showcases the model's capacity to transform vague initializations into distinct and discriminative representations through the learning process.

4 Towards deep networks - greedy layer-wise pre-training

In the figure 5, we observe that the reconstruction loss decreases for both RBMs as training progresses. However, the reconstruction loss for the top RBM is slightly higher as it learns to reconstruct a more abstract representation from the hidden layer of the bottom RBM. Ultimately, both RBMs achieve lower reconstruction loss as they learn to encode and reconstruct features from their respective input layers.

Increasing the number of Gibbs sampling iterations in the recognition process of the top layer of a deep belief net (DBN) gives rise to two hypotheses regarding its impact on accuracy. Firstly, as the iterations progress, there is a tendency for the hidden units to converge towards a mixed representation of possible labels. This suggests that the model explores a wide array of configurations for the hidden units based on observed data, potentially causing a loss of specificity towards any individual class or label. Secondly, the iterative sampling procedure inherently introduces noise, and with an increased number of iterations, this noise accumulates and may interfere with the accurate representation of class-related information. These hypotheses propose potential reasons behind the observed degradation in recognition accuracy as the Gibbs sampling iterations

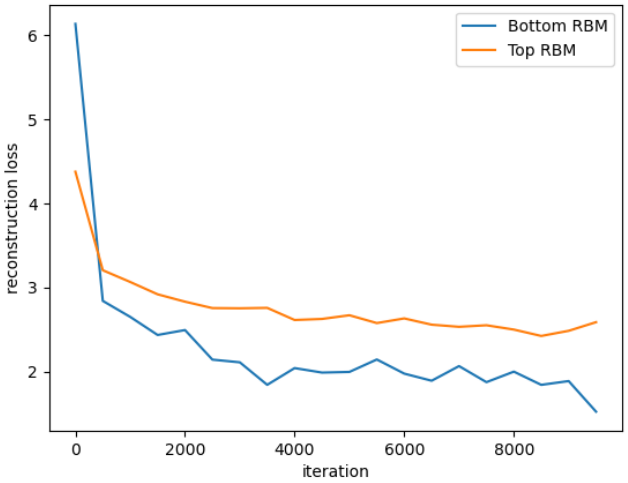


Figure 5: Reconstruction loss for bottom and top RBMs

are extended.

nbr gibbs recognition	1	3	5	15
Train accuracy	82.45%	78.01%	73.61%	62.02%
Test accuracy	83.02%	78.23%	75.03%	62.67%

Figure 6: Train and Test accuracies with different Gibbs sampling iterations

Many key factors determine the quality of generated images, and one of them is training. This means that if the stacked RBMs were not trained sufficiently, we may experience poor performance. It's important to keep in mind that even if we had good performance in recognition, image generation remains difficult because it has to predict 748 pixels, unlike recognition, which only has to predict 10 nodes. Another crucial point to mention is to keep the label clamped during contrastive divergence in the top layer of the DBN.
Here is one example of generating the number zero using our DBN :

However, it's not very robust and sometimes generates random images. Additionally, for certain numbers, it always fails to generate them.

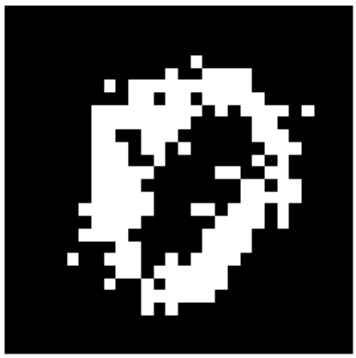


Figure 7: Image of 0 generated by our DBN

5 Final remarks

One of the interesting aspects of this lab is that it is computationally prohibitive. The training of the DBN on the whole data set takes a significant duration hence debugging any potential missteps is time and resource-consuming.
Among the most important challenges of the lab is the ability to know when to use probability distributions or samples drawn from them amidst the training.