Artificial Neural Networks and Deep Architectures, DD2437

Short Report on Lab Assignment 4

Restricted Boltzmann Machines and Deep Belief Nets

Emil Hed, Ivar Karlsson and Elias Kollberg

February 19, 2025

1 Main Objectives and Scope of the Assignment

The primary scope of this lab assignment is to gain practical experience with key concepts regarding deep neural network architectures. Specifically, we are looking into Restricted Boltzmann Machines (RBMs) and Deep Belief Nets (DBNs). The objectives include:

- Understanding the learning process of RBMs using contrastive divergence (CD).
- Developing an RBM for recognising MNIST images.
- Investigating the effect of varying the number of hidden units (from 500 down to 200) on the reconstruction loss.
- Extending the single-layer RBM into a deeper network using greedy layer-wise pretraining to construct a DBN.
- Evaluating the DBN in terms of reconstruction, recognition performance, and generative capabilities.

2 Methods

For the implementation we used Python and the provided code framework. The following libraries were central to our work:

- **Numpy** for numerical operations.
- **Matplotlib** for plotting and data visualization.
- Additional Python libraries for handling data (e.g., `struct`) and animations (`matplotlib.animation`) as needed.

The experiments were performed on the MNIST dataset, where images (28×28 pixels) are flattened into 784-dimensional vectors. We adapted the CD1 algorithm for training the RBM, monitoring reconstruction loss as a measure of convergence and stability. For extending the model into a DBN, we employed a greedy layer-wise pretraining strategy followed by clamping label units and running Gibbs sampling in the top RBM.

3 Results and Discussion

3.1 Task 4.1: RBM for Recognising MNIST Images

In this section, we describe the implementation and results of the RBM trained with CD1. We also compare the reconstruction loss for the two set ups with different number of hidden units (200, and 500). In the assignment instructions the hyperparameters and setup was given as shown in table 1. After performing a grid search on 10% on the dataset, see fig 1, we then ran a grid search for with the best configurations using all the data and confirmed the proposed hyperparameters from the assignment instructions.

Epoch	Mini-batch	Learning Rate	Momentum	Init Bias H	Init Bias V	Init Weight
20	20	0.01	0.7	$\mathcal{N}(0, 0.01)$	$\mathcal{N}(0, 0.01)$	$\mathcal{N}(0, 0.01)$

Table 1: Given training parameters for the neural network

- **Initialization:** The weight matrix (including visible and hidden biases) was initialised with small random values sampled from a normal distribution $\mathcal{N}(0, 0.01)$.
- **Training:** Training was carried out for 20 epochs using mini-batches of around 20 samples each. The convergence was monitored by observing the decrease in the average reconstruction loss (mean error between the original and reconstructed input).
- **Evaluation:**
  - **Reconstruction Loss:** Plots of the reconstruction loss per epoch indicate how the network gradually stabilises, see fig 2 where we can see the difference in convergence for the two different set ups using 500 and 200 hidden units respectively.
  - **Receptive Fields:** Selected hidden units' weights were reshaped into 28×28 images to visualise the receptive fields, offering insights into the feature detectors learned from the MNIST images, see fig 3.
  - **Weight distribution:** The weights from our model were plotted in a histogram showing the distribution of weights, confirming that the weights have values within a reasonable range, see fig 4.

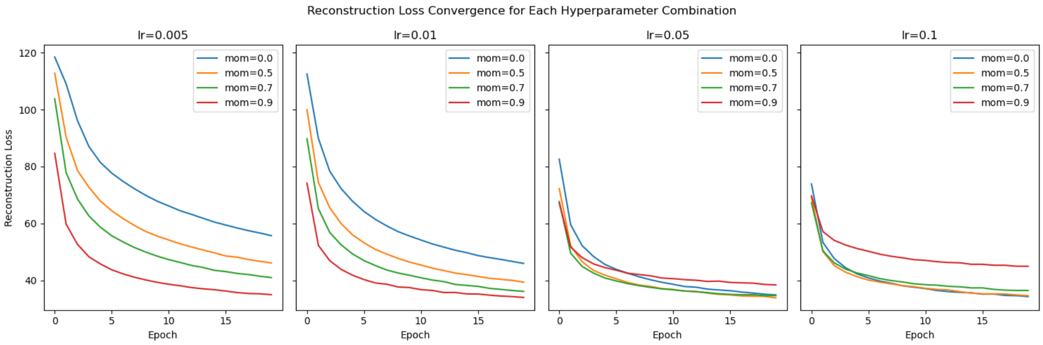


Figure 1: Reconstruction loss for different hyperparameters

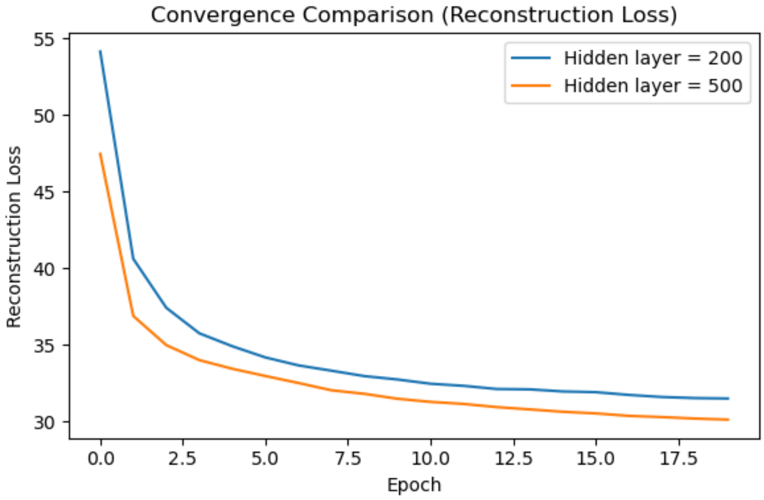
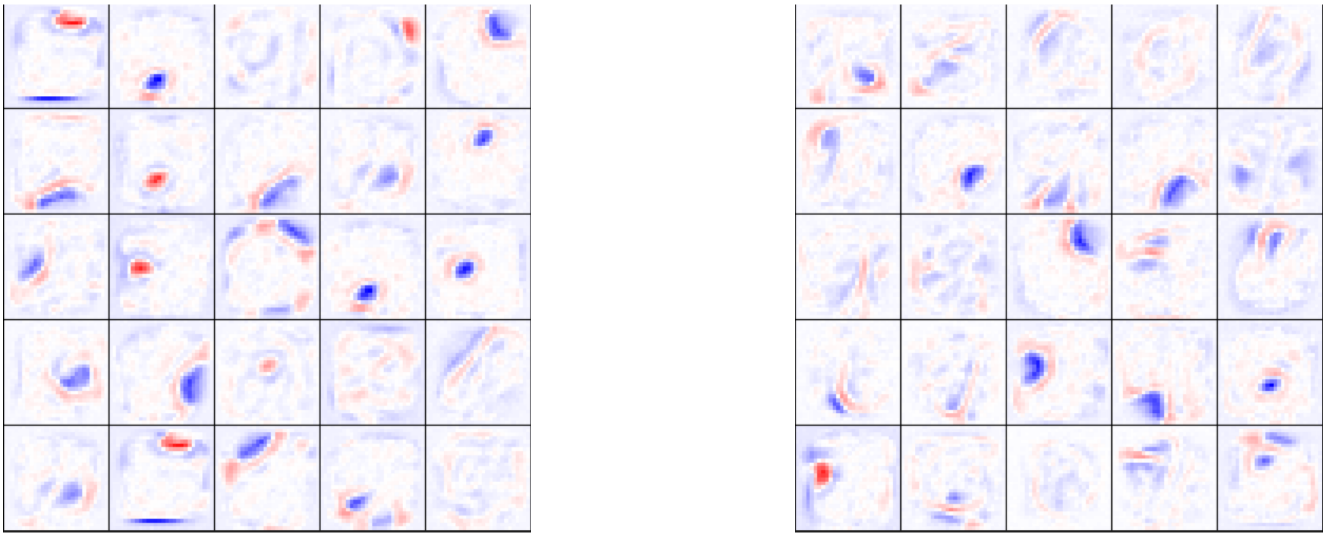


Figure 2: Reconstruction loss over epochs for the RBM.



(a) Receptive field visualisation for 200 hidden layers.

(b) Receptive field visualisation for 500 hidden layers.

Figure 3: Receptive fields visualisation for different numbers of hidden layers.

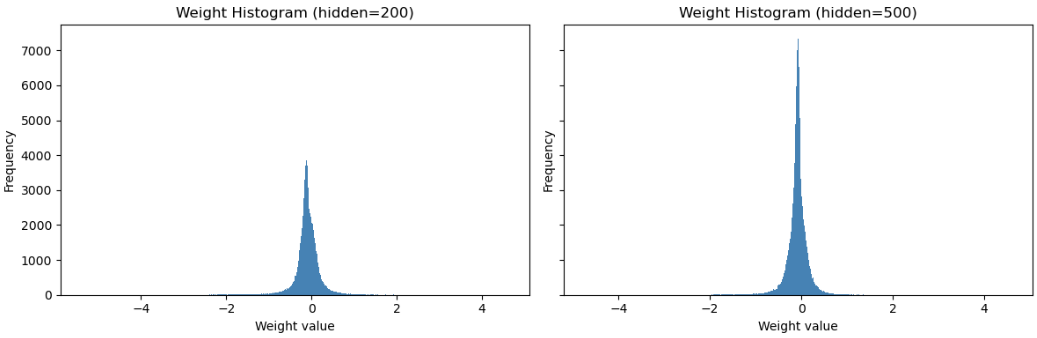


Figure 4: Weight distribution.

As we can see by inspecting the figures, the model is converging and we can see that it is learning some underlying representations of the inputs, however we would expect the receptive fields to better describe 'strokes' or 'circles', representing edges of positive and negative values of weights, from the digits in the MNIST dataset.

Task 4.2: Towards Deep Networks - Greedy Layer-Wise Pretraining

In this section we describe extending the RBM into a deeper architecture (DBN):

- **Network Architecture:** The initial architecture was based on 784 input units and two stacked RBMs with 500 hidden units each. For the top RBM, label units were concatenated with the hidden layer to form a (500+10)-unit layer, which was then connected to a top layer with 2000 units.
- **Layer-Wise Pretraining:**
  - The first RBM was trained on the MNIST images.
  - The hidden activations from the first RBM were used as input for training the second RBM.
  - Reconstruction losses for all layers were recorded, see fig 5.
- **Recognition and Generative Evaluation:**
  - **Recognition:** After pretraining, the network was evaluated for classification. MNIST images were propagated through the network with label units initialised to 0.1, followed by multiple iterations of Gibbs sampling. A softmax operation was applied to the label units to obtain classification probabilities and overall recognition accuracy was recorded, see table 2.

- **Generative Mode:** By clamping the label units to a one-hot vector (for a chosen digit), Gibbs sampling was used to generate images. Several samples were generated for qualitative comparison with the original MNIST digits (The assignment suggested to use 200 iterations of Gibbs sampling, but for the results in fig 7 we used 300 iterations).

Iterations of Gibbs Sampling	Training Accuracy	Test Accuracy
15	79.74%	80.59%

Table 2: Label accuracy for training and test set

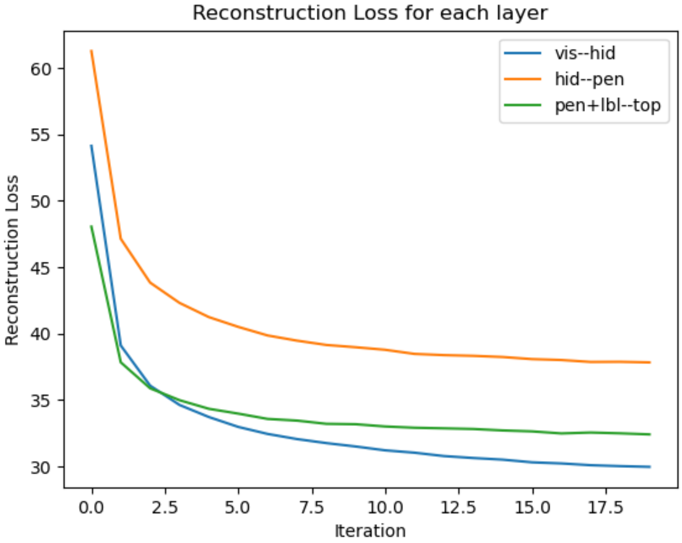


Figure 5: Reconstruction losses for the RBMs during greedy layer-wise pretraining.

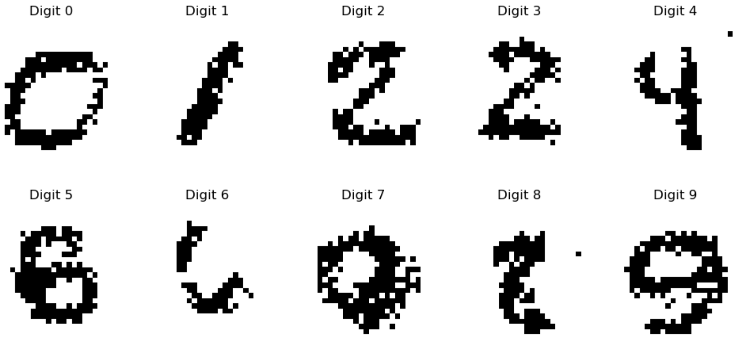


Figure 6: Samples generated from the pretrained DBN running Gibbs sampling for 200 iterations (with clamped label units).

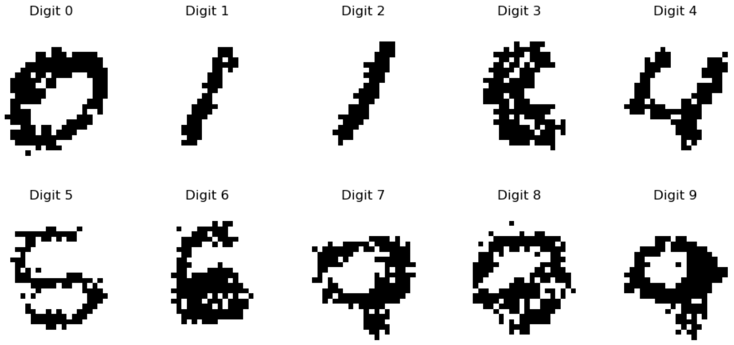


Figure 7: Samples generated from the pretrained DBN running Gibbs sampling for 300 iterations (with clamped label units).

We can notice that the generation of digits is highly dependent on the number of iterations of Gibbs sampling, running it too short results in few recognizable digits where most digits consists of a blob of black pixels in the middle, and running it for too long leads to mostly white pixels.

4 Final Remarks

This lab assignment offered valuable insights into both unsupervised and semi-supervised deep learning paradigms. In Task 4.1, training an RBM with CD1 on MNIST allowed us to observe convergence through decreasing reconstruction loss and to visualise the emergence of meaningful receptive fields. In Task 4.2, extending the architecture to a DBN via greedy layer-wise pretraining demonstrated how additional layers can improve feature representation and enable both recognition and generative capabilities.

Key observations include:

- The reconstruction loss can be a useful metric to monitor the stability and convergence of the network during training, but should be used in combination with other metrics to track the behavior of the network, such as the sizes of the weights or its receptive fields, as per [1].
- Visualisations of receptive fields reveal the network's ability to capture salient features from the input data.
- For the DBN, pretraining both layers independently allowed us to achieve good recognition performance when label units were introduced, and the generative samples were qualitatively similar to the original MNIST images.
- Hyperparameters such as the number of hidden units, learning rate, and mini-batch size have a significant impact on both the reconstruction fidelity and recognition accuracy.

Overall, the assignment deepened our understanding of the mechanisms behind RBMs and DBNs, particularly the interplay between unsupervised pretraining and subsequent supervised fine-tuning (where applicable).

References

[1] Geoffrey E. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*, pages 599–619. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.