Deep Learning II Project
Institut Polytechnique de Paris

Pierre Loviton

Table des matières

1 Introduction

The goal of the project is to create a deep neural network, pre-trained or not, for the classification of handwritten digits. We also compared the performance, in terms of good classification rate, of a pre-trained network and a randomly initialized network, depending on the number of training data, the number of layers of the network and finally the number of neurons per layer.

We have broken down the code into different chapters in a notebook :
1. Import and Data
2. Restricted Boltzmann Machine (RBM) on Binary AlphaDigits
3. Deep Belief Network (DBN) on Binary AlphaDigits
4. Deep Neural Network (DNN) on MNIST
5. Analysis
6. Variational Auto Encoder (VAE)

To run the code, you have to run all the cells in the good order.

We trained the RBM and the DBN to generate the letter 'F' from the Binary AlphaDigits dataset. We get pretty good results.

2 Analysis

The goal of the analysis was to compare performances of two networks, one pretrained, not the other one.
Firstly, we specified parameters related to networks and their training (size, epochs, batch size, learning rate, number of data).
We initialized them randomly.
We trained one DNN in an unsupervised way, considering it as a stack of RBMs.
Then we trained it in an supervised way with the back-propagation algorithm.
We trained the second one only in a supervised way too.
Then we calculate misclassification rates of this two networks, and we varied some parameters and generated 3 different graphs to present these results.

The orange curve represents the pre-trained network and the blue one the other network.

2.1 Number of layers

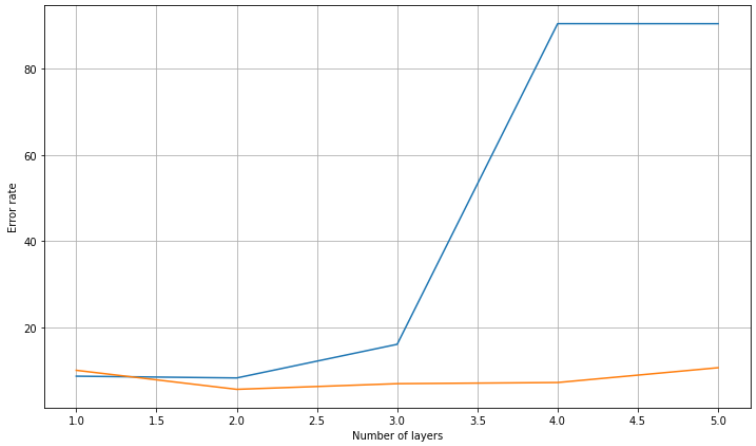

FIGURE 1 – Evolution of the error rate according to the number of layers

This figure shows us that with 1 or 2 layers, we get fairly close results. The best result is with the pre-trained network and 2 layers. We can see that the not pre-trained one has really bad results hen we increased his layers.

2.2 Number of neurons by layer

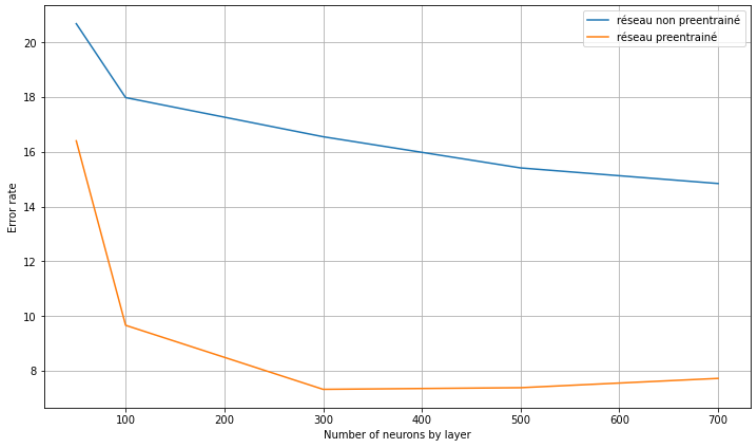

FIGURE 2 – Evolution of the error rate according to the number of neurons per layer

This time we trained our networks with 2 layers and we varied the number of neurons by layer. This second figure shows us that the pre-trained network get the best results at any time. The best result is with 300 neurons by layer. And we know that when you add more neurons in a layer, the time is increased too.

2.3 Number of data trained

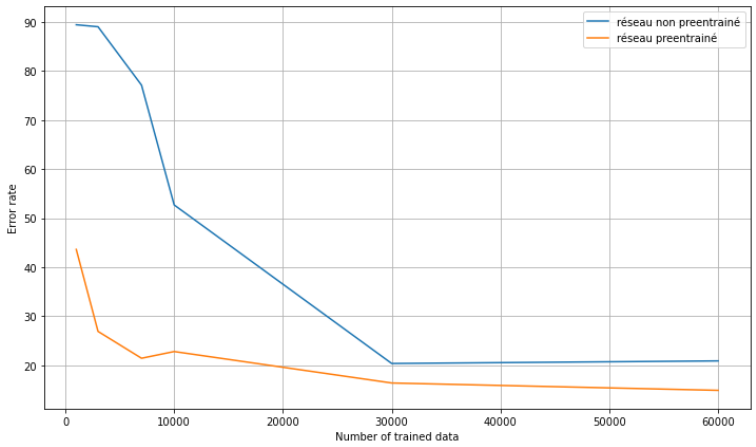

FIGURE 3 – Evolution of the error rate according to the number of training data

For this last analysis, we varied the number of data used from 1,000 to 60,000. We trained our networks with 2 layers of 50 neurons each. We can see that the pre-trained network get also the best results, but when the number of data is high (>30,000), the results are closer. The best result is obtained with 30,000 data, even if it decreases after, the time complexity is too high for the capital gain.

3 Bonus : Variational Auto Encoder

The goal of this analysis was to compare the images generated by a VAE, a RBM and a DBN trained with approximately the same number of parameters, when these networks were trained on the MNIST dataset.

3.1 Variational Auto Encoder

We created a VAE with this architecture :

VAE(
(fc1) : Linear(in-features=784, out-features=512, bias=True)
(fc2) : Linear(in-features=512, out-features=256, bias=True)
(fc31) : Linear(in-features=256, out-features=10, bias=True)
(fc32) : Linear(in-features=256, out-features=10, bias=True)
(fc4) : Linear(in-features=10, out-features=256, bias=True)
(fc5) : Linear(in-features=256, out-features=512, bias=True)
(fc6) : Linear(in-features=512, out-features=784, bias=True)
)

We can see that our VAE can pretty well generated different recognizable numbers (6, 1, 0, 9).

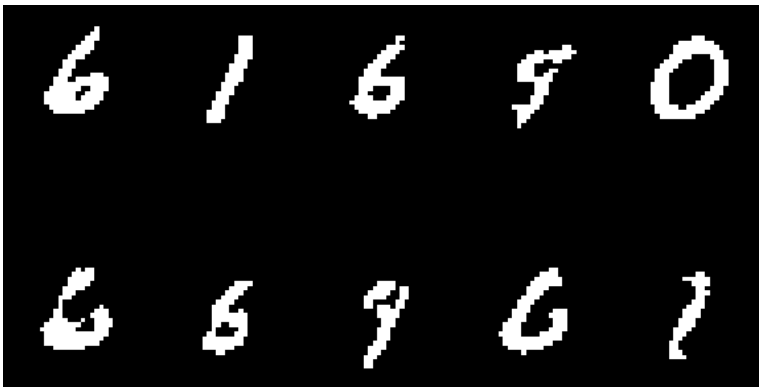

FIGURE 4 – Images generated by a VAE trained on the MNIST dataset

3.2 RBM

We created a RBM with a size of 784 * 600 = 470 400 parameters and get this loss and these results :

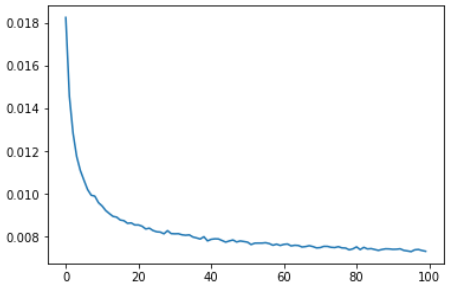

FIGURE 5 – RBM loss evolution during training

FIGURE 6 – Images generated by a RBM trained on the MNIST dataset

We can see that our RBM don't work well in generating numbers while is big size...

3.3 DBN

We created a DBN with a size of 784 * 400 + 400 * 300 + 300 * 100 = 463 600 parameters :

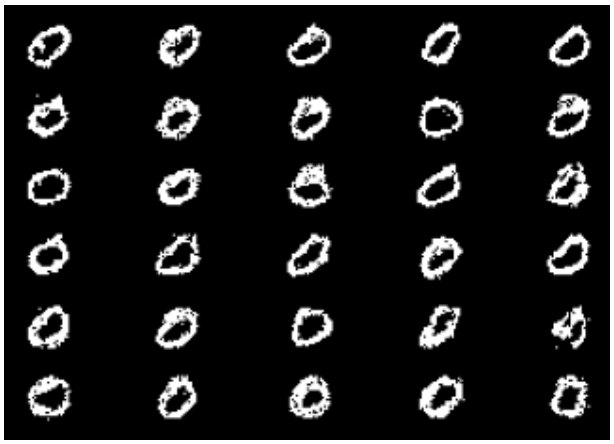

FIGURE 7 – RBM loss evolution during training

We can see that our DBN generate only 0. It did it well but even if we reload the code, it generated only 0...

4 Conclusion

As we have seen on the 3 plots of the second part, the best DNN is maybe a pre trained network trained with 30 000 data, 300 neurons by layer and 2 layers.

Concerning the generative part with the VAE, we saw that VAE worked well contrary to RBM. DBN worked well but generated only zeros...

Bibliography

Références

[1] Dennis Michael Fisher Joe Collenette, Louise A. Advising autonomous cars about the rules of the road. *EPTCS*, 371 :62–76, 2022. doi : https://doi.org/10.4204/EPTCS.371.5.

[2] National Transportation Safety Board. Collision between vehicle controlled by developmental automated driving system and pedestrian, 2019. URL `https://www.ntsb.gov/investigations/AccidentReports/Reports/HAR1903.pdf`.