Università degli Studi di Padova

Dipartimento di Fisica e Astronomia “Galileo Galilei”

Corso di Laurea in Physics of Data

Exercise 4 Deep Learning and Neural Networks

Author

Marco Agnolon

Abstract

The aim of this exercise is to implement an autoencoder that is able to reconstruct images taken from MNIST dataset and corrupted with some source of noise

Anno accademico 2019-20120

Data prepocessing

Data preprocessing plays a very important role. MNIST dataset contains samples of handwritten digits (Figure 1).

Corruption transformations

It is needed to create the transformations to the data in order to check the performance of the network.
Two possible sources of noise have been taken into account: Gaussian noise and occlusion noise. The first is a gaussian noise, with mean 0 and standard deviation 1, added to the original images that after this need to be renormalized. The importance of this noise effect could be tuned with a parameter (multiplied in front of the noise term), Figure 2 shows the results for a level of 0.5.
The second is an occlusion of the image, this means setting to 0 all the pixel inside a square of dimensions decided a priori using a parameter that represents the fraction of pixels occluded in each dimension. Figure 3 is an example of this noise application.

Train-Validation-Test

The data are divided in train set and test set. The latter contains 10000 samples, then they are further divided in validation and test, composed of 8000 and 2000 samples respectively.
If the network implemented is a simple autoencoder, the noise transformations described above are applied only to the test set. The train and the validation sets are filled with original samples.
If the network is a denoising autoencoder then also the train and validation set are transformed: two new dataset are created, one with gaussian noise and one with occlusion. Then the used train and validation sets are sampled from the three types of dataset with different probabilities. Sampling from original images (not corrupted) is ten times more probable than sampling from gaussian noise or occluded dataset. So the data are a mixture of original images and corrupted images, see Figure 4.
This is very important to implement a denoising autoencoder because it needs to be filled with both normal and corrupted images in order to be able to tell the differences between them and reconstruct the wrong ones.

The neural network

The network is a composed of two sub-network, the encoder and the decoder. Each of them contains a CNN and linear layers. What really matters is the number of neurons that linked the two parts because these neurons create the encoded representations of the inputs and they are the starting point for the reconstructions. They represent the dimension of the encoded space.
It was chosen to implement 5 different networks and compare their results. They have 2,4,6,8 or 10 neurons in the encoded representation.

Autoencoder

The autoencoder is training over 60000 original samples divided in batches of 512 each. The best effective learning rate is $1 \cdot 10^{-3}$ (chosen after trying different numbers).
At every epoch that corresponds to a training over all samples, the validation error is computed. This allows to create an early stopping procedure as a way to avoid overfitting. The selected algorithm (Lutz Prechelt, *Early Stopping | but when?*, Fakultat fur Informatik; Universitat Karlsruhe D-76128 Karlsruhe; Germany, 1998) stops the algorithm when the validation error at the last epoch is α times greater then the minimum error found till that epoch. In formulas:

$$GL_\alpha(t) = \frac{E(t)}{E_{min}} - 1 > \alpha$$

The parameter α represents the minimum percentage of the minimum error that is not allow to exceed. It is chosen accordingly to the necessity, the greater the value, the greater the goodness of the found minimum but the stop condition will arise later. In this case $\alpha = 0.01$. Figure 5 shows a comparison among the

networks.
Increasing the number of dimension of the encoded space the validation error reaches lower plateaus.

Denoising autoencoder

The denoising autoencoder needs different samples, it has to learn both to recover the original samples and to reconstruct the corrupted ones. Therefore it is fed with the mixed dataset (the one that contains also the corrupted images). But the error of the network is computed comparing the reconstructions of the network with the original images that correspond to the corrupted ones. The network is trained over the strongest corrupted images, so with maximum noise level 0.5 and maximum occlusion size of 1/2.
The network should learn how to take a bad images and reconstruct the original one, hence it has to minimize the mean square error between the output of the network and the original not corrupted sample.
Since, in this case, the training procedure is way longer than before, another early stopping procedure is added to the previous one: this time if the ten but one last epochs have a mean error smaller than the last one then the algorithm is stopped. The result are shown in Figure 5, the errors reaches a plateau each time lower increasing the number on dimension in the encoded space.
Each time for both types of network a plateau is reaches: the early stopping procedure works as expected.

Results

Numerical results

Each network (it means both autoencoder and denoising for each number of encoded space dimensions) is tested on different test sets:

- The original test set non-corrupted
- The test set corrupted with increasing level of gaussian noise: 0.1, 0.2, 0.3, 0.4, 0.5
- The test set corrupted with occlusions of different sizes: 1/14, 1/7, 1/4, 5/14, 1/2 of the side

The x-axis represents the number of dimension of the encoded space and in the y-axis the test error in all the graphs showed in the appendix.
The first case, where the comparison is done in the non-corrupted test set (Figure 6), the two types of networks behave quite similarly, the error goes down as the number of dimension of the space increases; this happens since also the complexity of the network indeed increases and it becomes able to catch finer representation of the input. However from N=6 the denoising network starts performing better than the rival. This could be attributed to the fact that the former is fed with corrupted images and this contribution prevents overfitting.
Figures 7 and 8 show the behavior of the two types of network with gaussian noise and occlusion increasing in strength. For what concern the autoencoder the error shows an increasing or stationary trend (Figure 7) while for the denoising one the error tend to decreasing increasing the number of space encoded dimensions. However the main purpose of this graph is to confirm the expected behavior: both the types of network present an increasing error as the corruption becomes more important (different lines tend to be separated), so the network behaves reasonably.
Figure 9 compares the error for the same type of network in different datasets. When considering the corrupted dataset the error plotted is the mean error among the different level of corruptions. The autoencoder, on the left, shows a slightly decreasing behavior for the test and the occluded set, whereas the denoising autoencoder presents a decreasing behavior in all the three cases. These graphs show that both networks perform better in the original test set, but then they react differently when considering different corrupted dataset: in fact the normal autoencoder seems to fail in the gaussian noise case.
Finally, the most significant plots (Figure 10) show direct comparison between the two types of networks in the corrupted datasets. It is clear that the denoising autoencoder outperforms the standard one in both cases. Moreover the normal autoencoder starts to perform worse when increasing the number of dimension in the gaussian noise corruption data, whereas this does not happen in the occlusion case. It could be due to the fact that a larger number of dimension in the encoded space leads to a better representation of the non-corrupted inputs (since it is fed only with those). If the image is corrupted with occlusion a better representation allows a better retrieving of the original sample from a limited part of the image, however a

random noise corruption fools the network and it give rise a sort of overfitting caused by the larger number of dimension and the too good representation of the input images.
For what concern the denoising autoencoder, it improves drastically the performance of the normal one in both the corruptions, since its error is always smaller. Furthermore, since its error decreases as the number of dimension increases (see right part of Figure 9), larger number of neurons should allow better reconstruction capabilities.
However the performance of the denoising should decreases as long as the number of dimension increases and the trad-off between these two quantities will weigh the more the dimensions in order to obtain a small representation of the input; that remains the main scope of this kind of networks. Indeed, looking at Figure 11 it is clear that the performances of the denoising autoencoder improve less. Moreover in the case of gaussian noise the 10 dimension net, in average, performs better with respect to the 12 dimension one.
Therefore, denoising autoencoder with 10 dimension encoded space is chosen as best network balancing performances and smaller encoded space dimensions.

Graphical results

Figures 12 and 13 show the reconstruction procedure for a normal autoencoder with space dimensions 2 and 10. The left image represent the corrupted input fed to the network, the central how the input should look like if not corrupted and the right one the reconstruction done by the network. The networks both fails when gaussian noise is used, while the occlusion give better (not optimal results).
Figures 14 and 15 are the correspondent of the previous ones for the denoising autoencoder. They both are better with respect to the normal case and also visually the 10 dimension net outperforms the 2 dimension one. So the numerical results are corroborated by the visual ones.

Generation

It is possible to generate new digits sampling from the encoded space. This is easy to show in the two-dimensional case since also the space could be mapped. Figure 16 shows a map of the encoded two-dimensional space for the denoising autoencoder. Different colors represent pictures of different number contained in the test set encoded in this space. One may notice that the cluster are more or less visible in the map, depending on the number considered. This map obviously is meaningless for higher dimension since it is only a projection of the bigger space into a two-dimensional one.
Looking at this map, one could choose two coordinates that are supposed to belong to a certain cluster and see if the result is coherent. Figure 17 shows some generated samples respectively using: (-8,-12), (20,20) and (20,-5). Making a simple check with the map, it is clear that the generated digits are coherent with the clusters, also for the last one that might have been more problematic.

Appendix

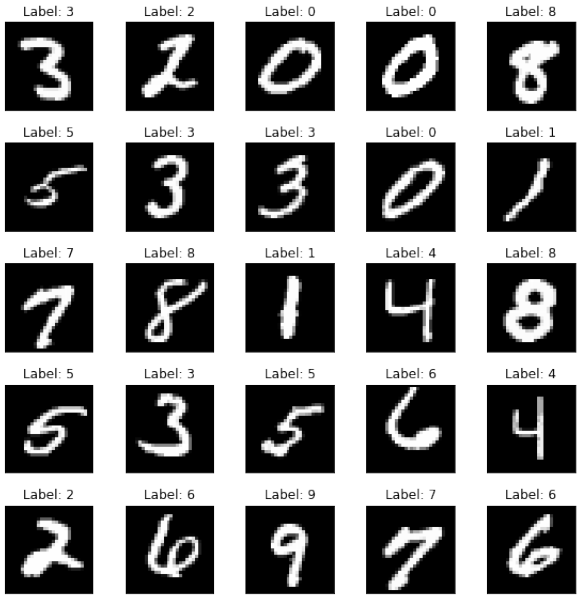

Figure 1: Original dataset

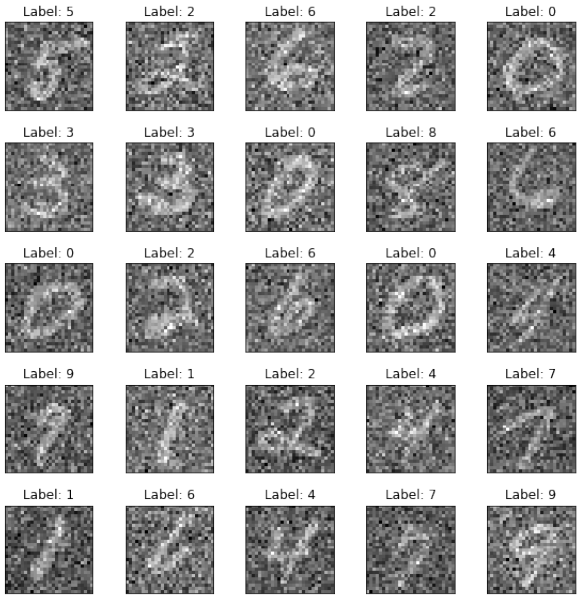

Figure 2: Images with gaussian noise (level=0.5)

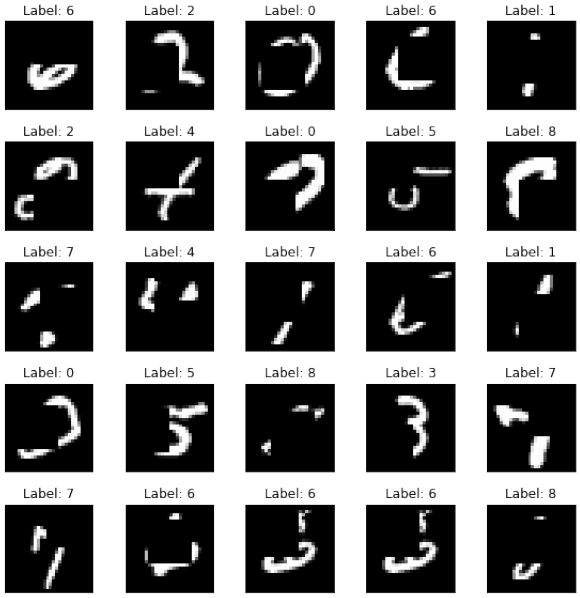

Figure 3: Occluded images of 1/2 of the side

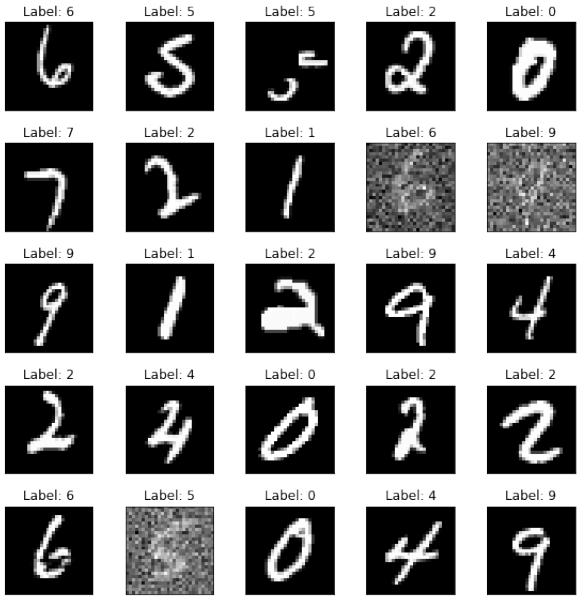

Figure 4: Original, with noise and with occlusion images

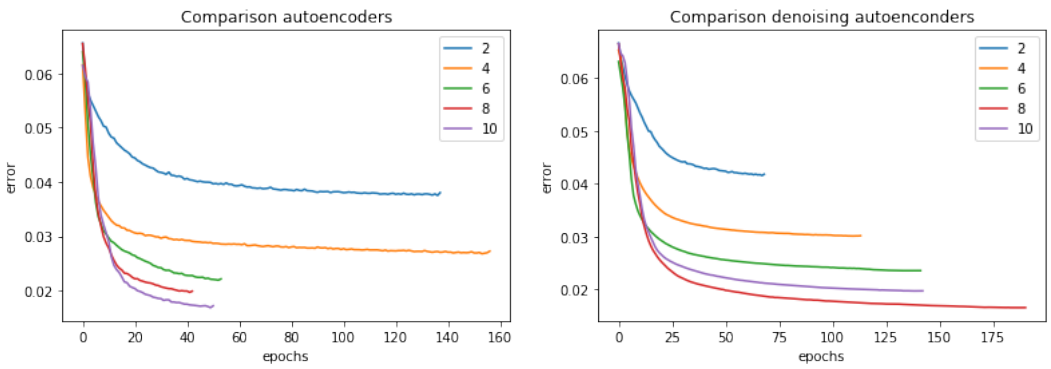

Figure 5: Validation error for different networks

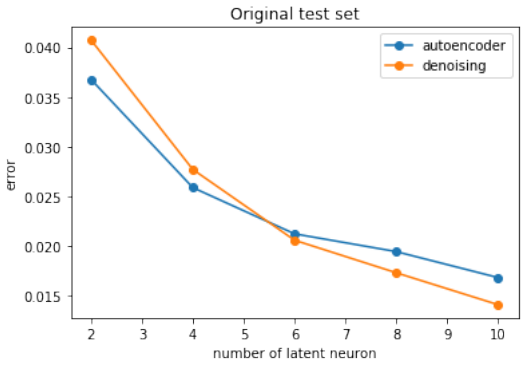

Figure 6: Comparison in the original test set

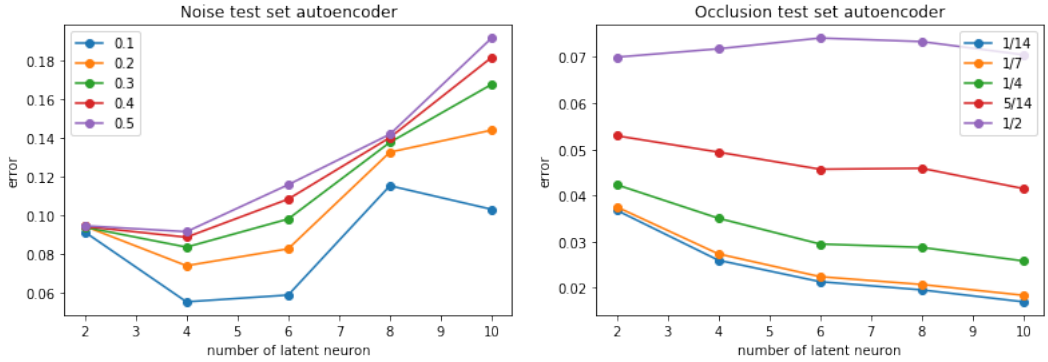

Figure 7: Autoencoder performances in corrupted datasets

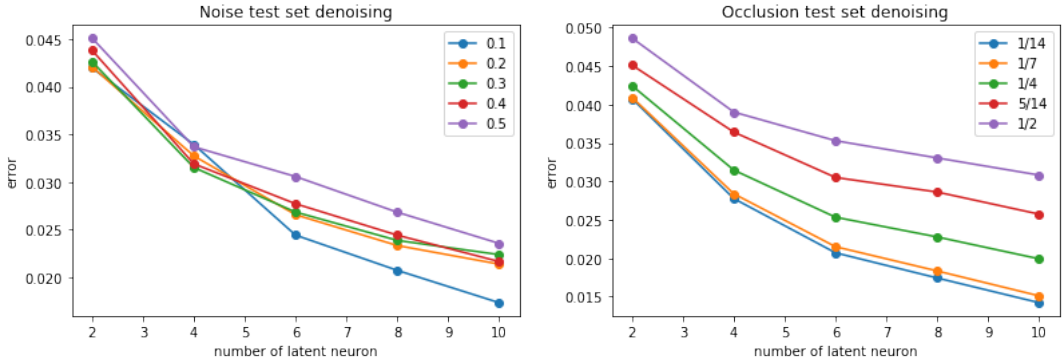

Figure 8: Denoising performances in corrupted datasets

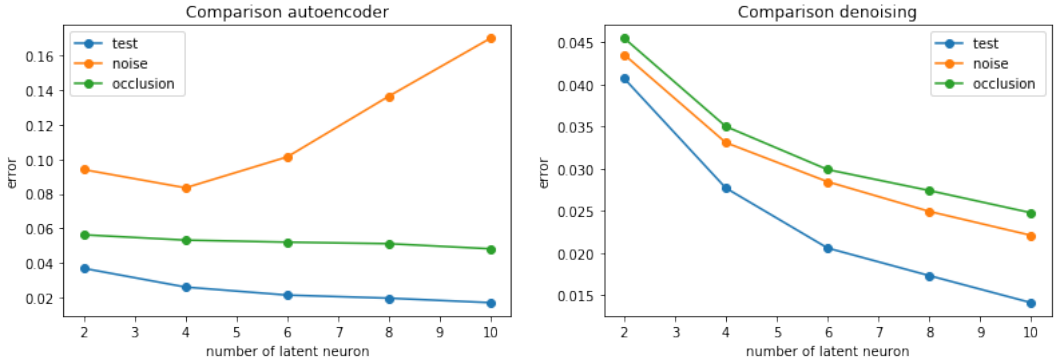

Figure 9: Performances of the two type of network in the three different datasets

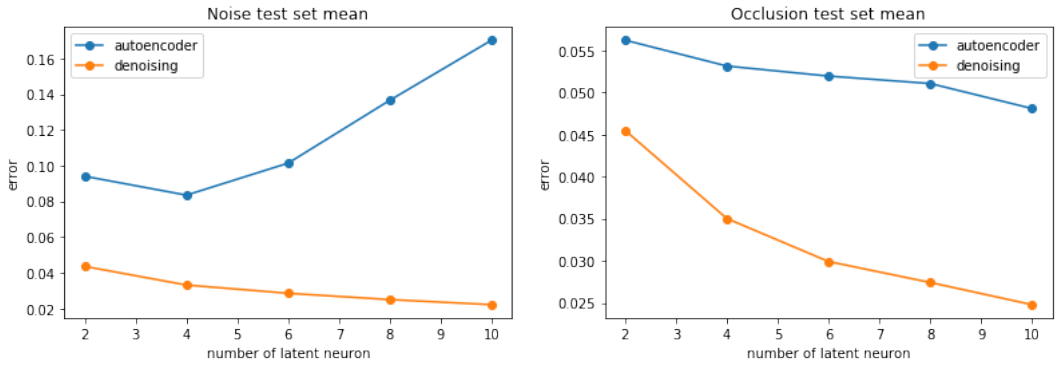

Figure 10: Comparison in the corrupted datasets

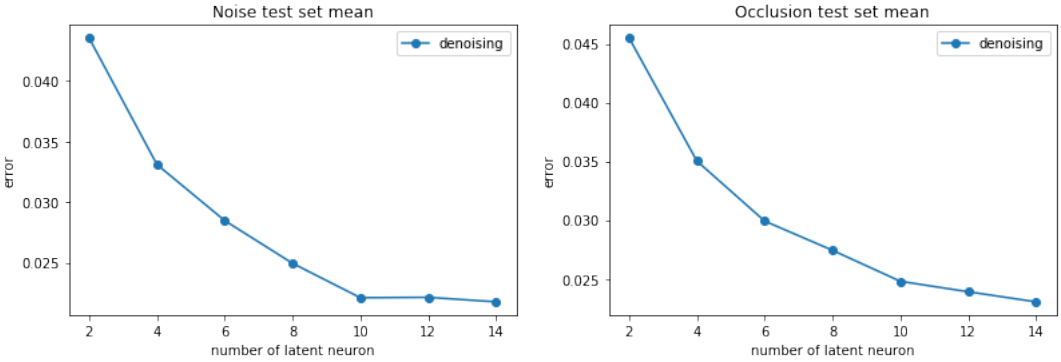

Figure 11: Larger dimension for corrupted datasets

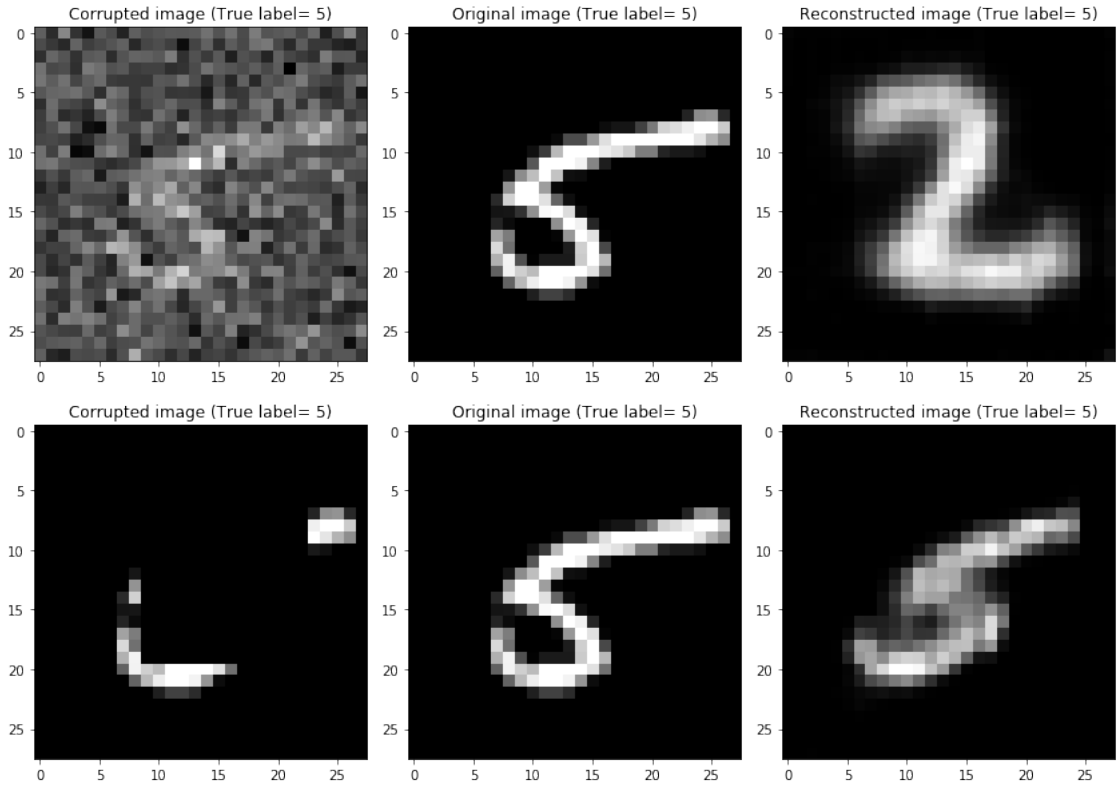

Figure 12: Reconstruction for autoencoder in dimension 2

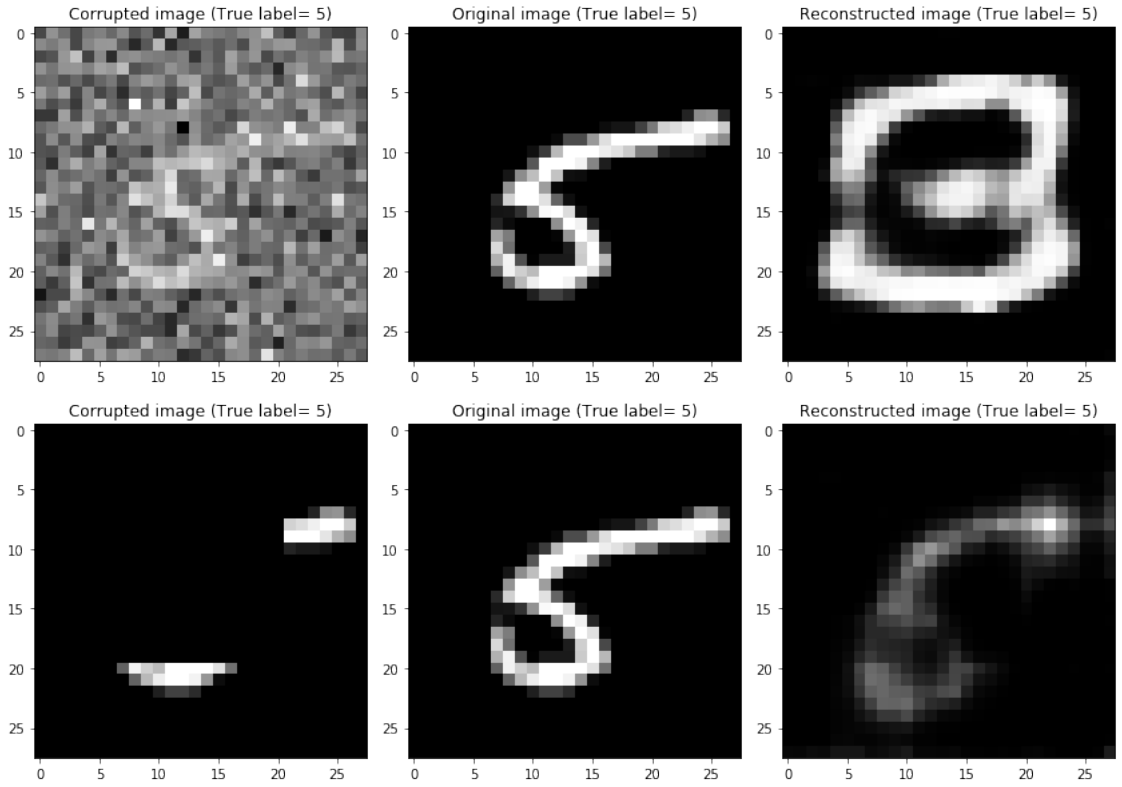

Figure 13: Reconstruction for autoencoder in dimension 10

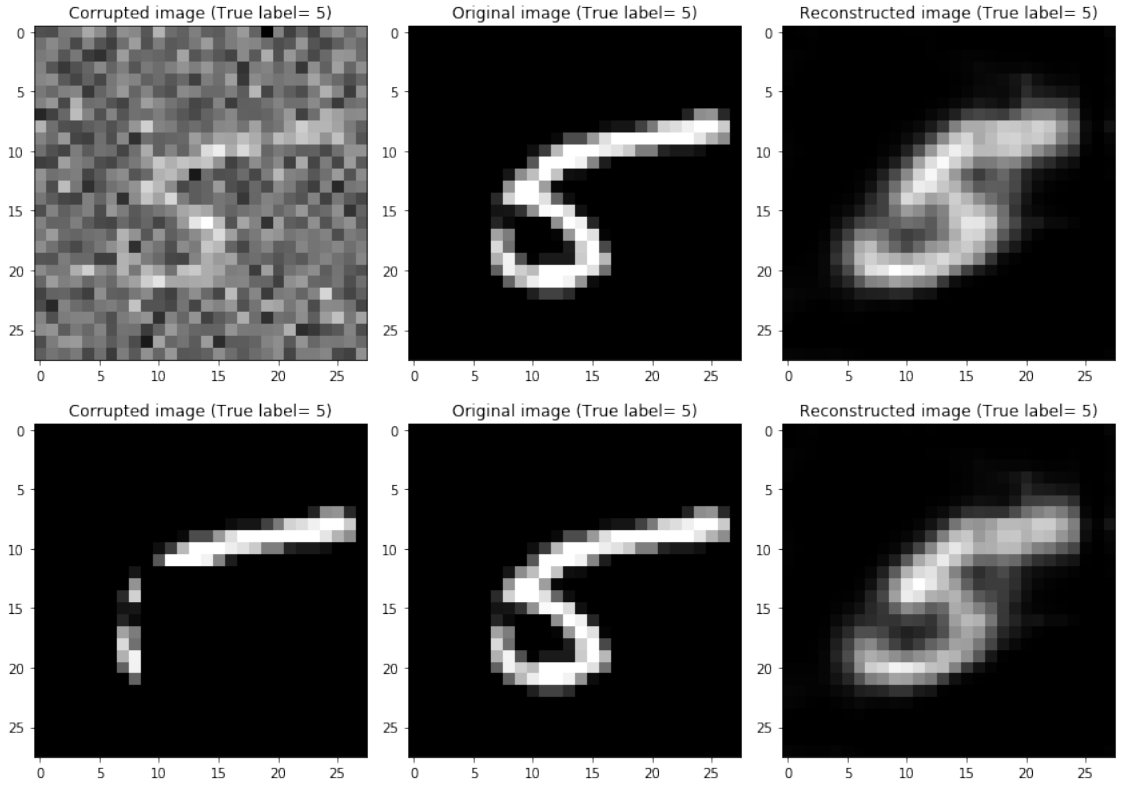

Figure 14: Reconstruction for denoising autoencoder in dimension 2

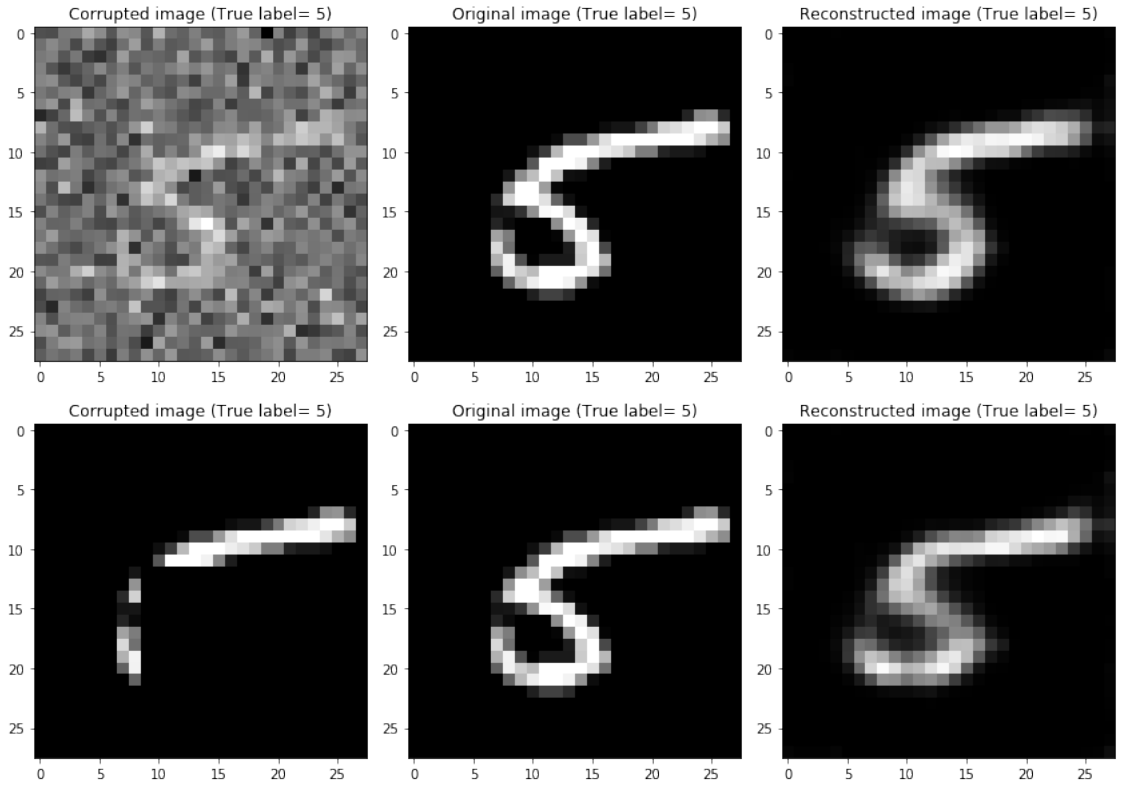

Figure 15: Reconstruction for denoising autoencoder in dimension 10

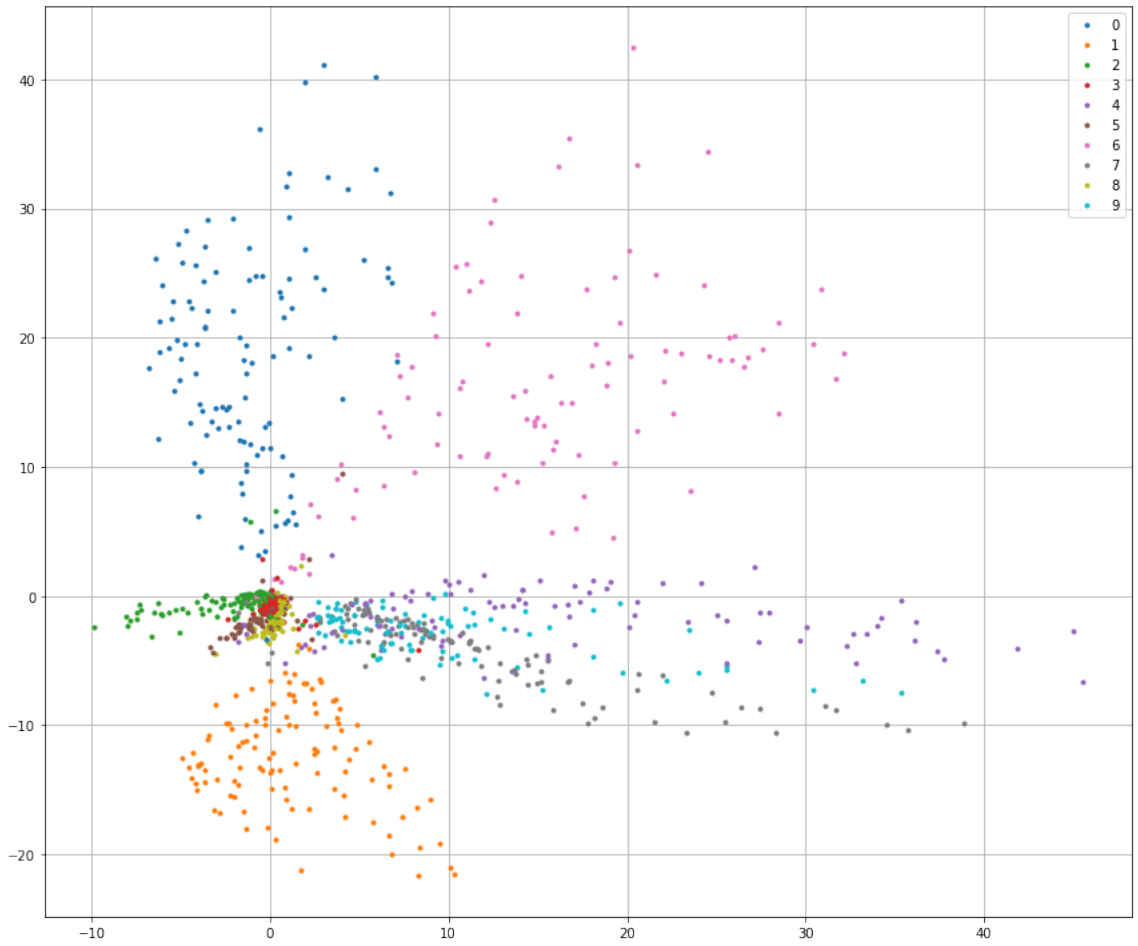

Figure 16: Map of encoded space

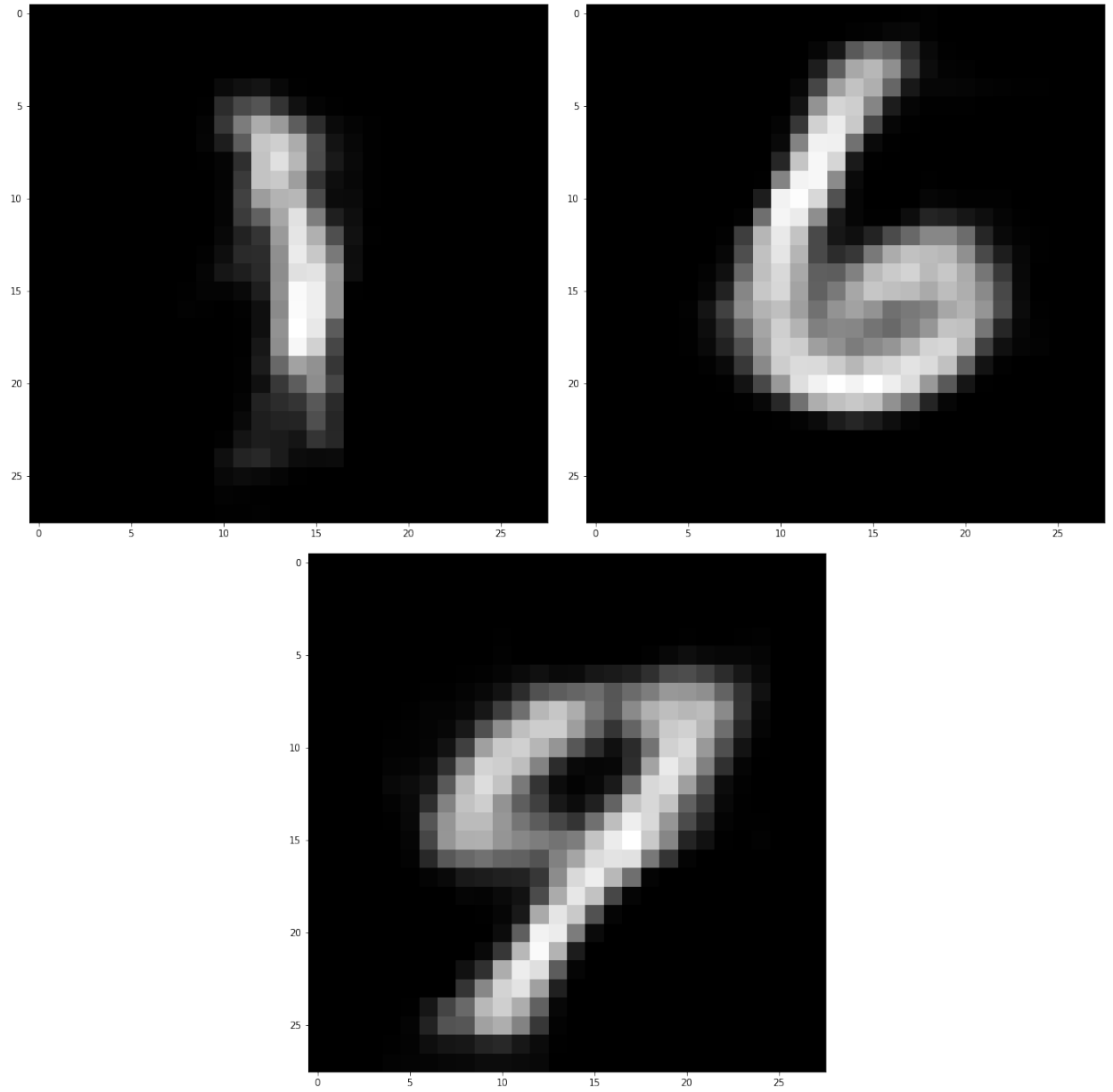

Figure 17: Generated samples