Neural networks - Project 1

Olga Bonachera del Pozo - Daniel Kwapien - Alejandro Sánchez

1. Initial network architecture

First off, we started by defining the general functions in PyTorch for our autoencoder. These functions are generalized so we could try a network with any dimension of hidden layers, any number of layers and any projected dimensions. They are called **Autoencoder_basic** and **Autoencoder_extended**.

2. Validate networks and study of the influence of the projected dimension (15,30,50,100)

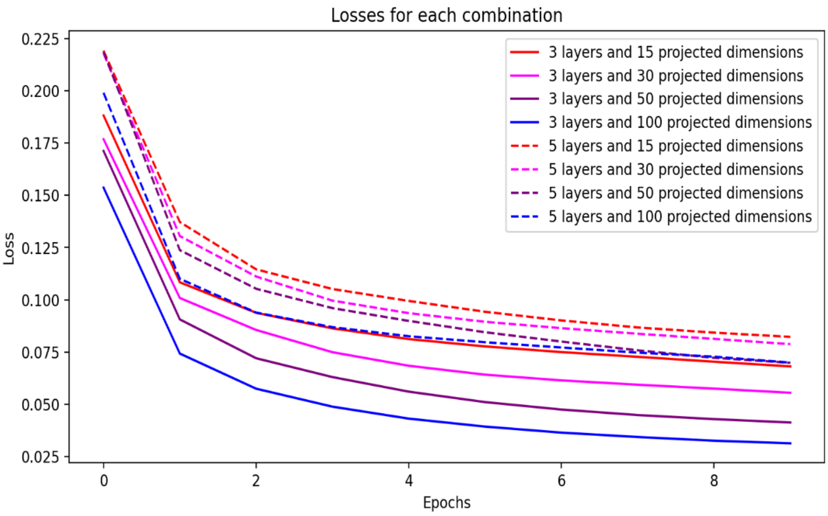
With each dataset, MNIST and FMNIST, we intend to validate the network performing a study and a selection of hyper-parameters in terms of the number of hidden layers and number of projected dimensions. For this, we train a model with each architecture and **10 epochs**.

2.1 MNIST

For the MNIST dataset we obtained the following results:

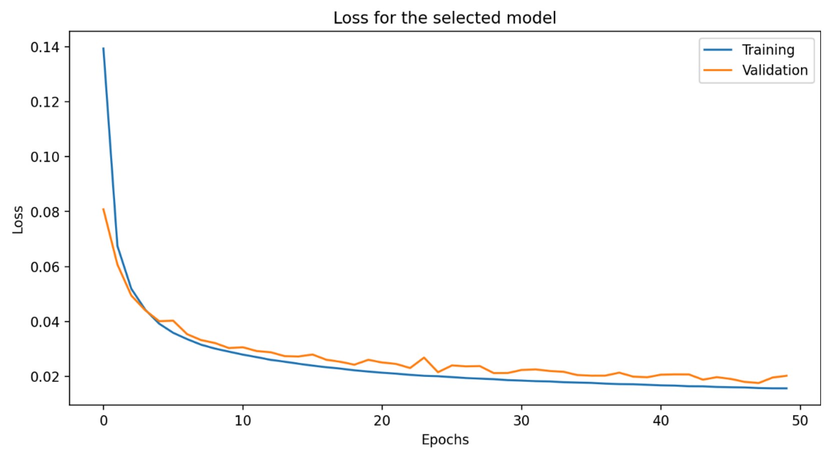
Projected Layers	3 Hidden Layers	5 Hidden Layers
15 layers	0.0681	0.0823
30 layers	0.0555	0.0788
50 layers	0.0412	0.0698
100 layers	0.0313	0.0699

Table 1: MSE loss for different architectures at the end of 10 epochs



By observing this table, it can be stated that the **best combination** of hyper-parameters is **100 projected layers** with **3 hidden layers**. Furthermore, a study of the learning rate has been performed in which we concluded that the best alpha for the model is α = **10e-3**.

If we try to **overfit** the model with the **best architecture** with a training of **50 epochs** it gives a resulting **average loss of 0.0157**, which is a reasonably good loss for an autoencoder. But as we can see, though the **validation set does not increase** in any moment, it is always slightly **above the training set**, which can be seen as **poor generalization**.

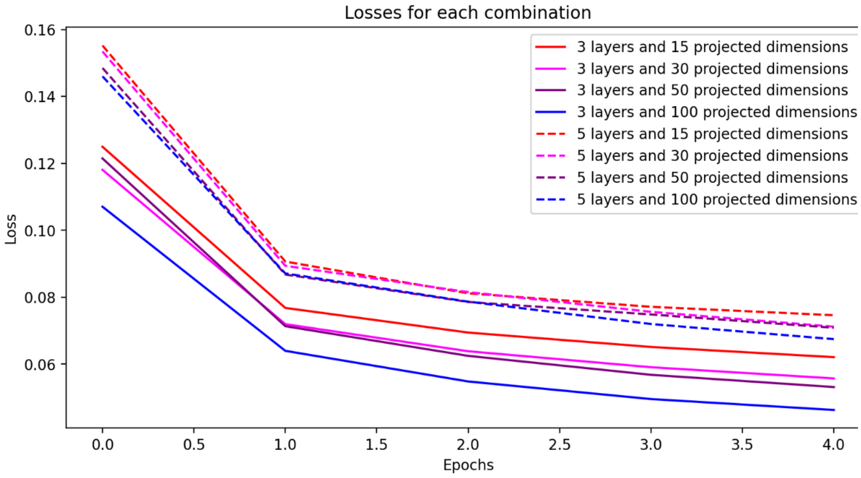


2.2 FMNIST

For the FMNIST dataset the results are:

Projected Layers	3 Hidden Layers	5 Hidden Layers
15 layers	0.0649	0.0702
30 layers	0.0561	0.0729
50 layers	0.0522	0.0718
100 layers	0.0463	0.0736

Table 2: MSE training loss for different architectures at the end of 5 epochs



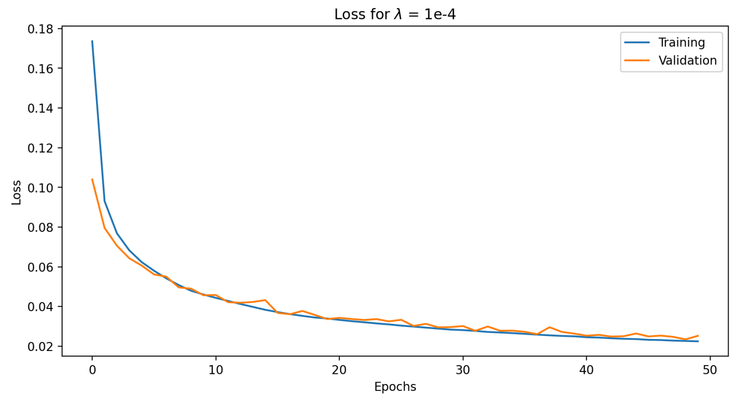
Again, the best combination of hyper-parameters for our model is **3 hidden layers** and **100 projected dimensions**. If we try to **overfit** the selected model we obtain a **0.0287 MSE training loss** and **0.0312 validation loss** after training for **30 epochs**. The behavior is the same as in the MNIST dataset, it does not really overfit, but it has **poor generalization**.

3. Implementation of regularization techniques

Now we will explore different regularization techniques. The most common technique for forcing sparsity in a sparse autoencoder (SAE) is Lasso regularization, but we will also try Dropout. In the following table we will compare the results of the model with Lasso regularization with λ=1e-4, which was selected by hyperparameter selection, after training for 50 epochs and dropout regularization after training for 10 epochs.

Regularization technique	Lasso	Dropout	None
MSE	0.0225	0.9259	0.0157

Table 3: MSE training loss with different regularization techniques

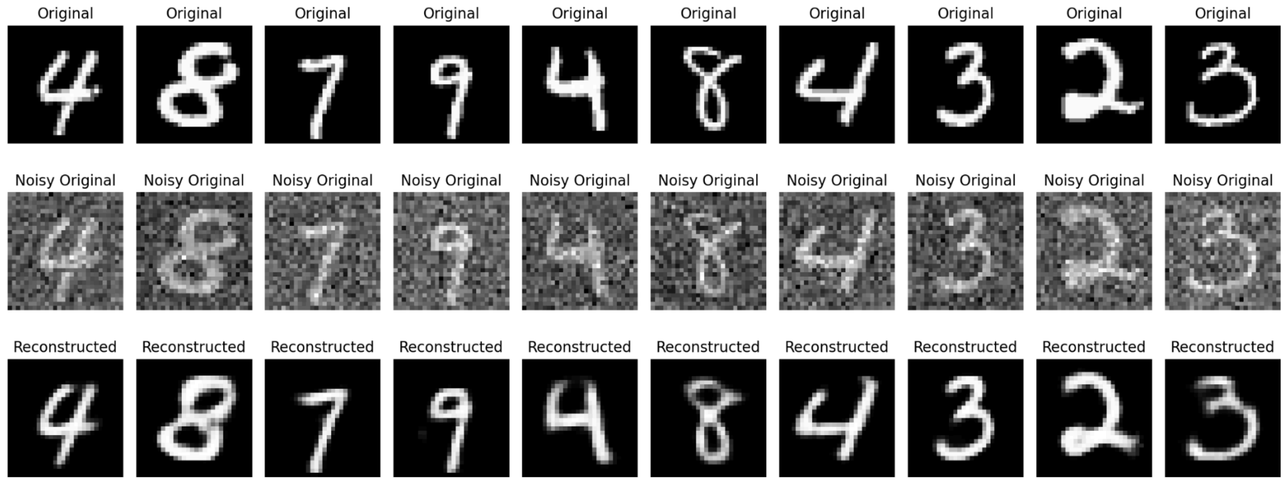


As we can see, Dropout performs poorly, that is because the network does not really overfit at any moment. On the other hand, as we have seen since the model has bad generalization, we achieve a better generalization, which can be observed in the image, where the validation loss is only slightly higher than the original model.

4. Denoising autoencoder

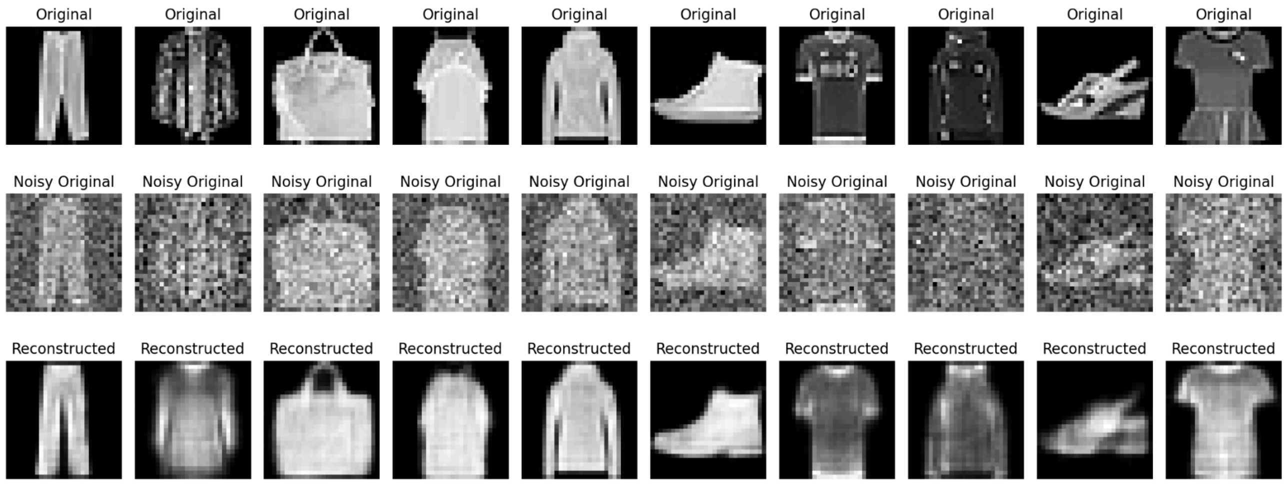
4.1 MNIST

Finally, applying constructing our **denoising** autoencoder with the same parameters as the selected model and trained with the **MNIST** dataset with added noise ϵ, where $\epsilon \sim N(0, 0.6)$. It gives quite **good results**.



4.2 FMNIST

Applying the same noise to the **FMNIST** dataset we achieve **slightly worse results** but it still performs good, which makes sense since the **images have more details that are lost with the noise**.



On these two images, we can see the **denoiser performance** using the **Peak Signal-to-Noise Ratio** (PSNR). It is fairly good and gives an idea of the quality of the reconstruction.