Assignment 2
Advanced Machine Learning
University of Milano-Bicocca
M.Sc. Data Science

Alessandro Riboni
October 28, 2019
ID number: 847160
a.riboni2@campus.unimib.it

Prediction of gray-scale images of letters P - Z

The assignment consist on the prediction of gray-scale images of letters *P-Z*. The provided data comprises the training set that can be used for the training (and eventually for the validation) and the unlabelled balanced test set. In this report will be presented three different tasks:

- Resolution of the problem of supervised classification with a traditional neural network;
- Visual investigation of the reconstruction abilities of an auto-encoder architecture;
- Use and evaluation of the encoded representation generated by the auto-encoder to solve the problem of supervised classification.

1 Supervised classification with a traditional neural network

To develop a neural network, it was necessary to divide the training data into training and validation sets. Subsequently, the class attribute was binarized using the get_dummies function, thus obtaining 11 boolean variables. Each image in the dataset is 28x28 in size, and, for this reason, the input layer of the neural network is composed of 784 neurons. Three hidden layers have been created with the function of activation "relu" (Rectified Linear Unit). The hidden layers are composed of 256, 512, and 256 neurons, respectively. Having eleven different classification possibilities, the output layer is formed by 11 neurons with "softmax" activation function. To reduce the risk of overfitting, dropouts have been inserted after each hidden layer. Dropout consists of randomly setting a fraction rate of input units to 0 at each update during training time. Also, to avoid overfitting during the training phase of the model, the EarlyStopping callback function was used, which stops the training when a monitored quantity has stopped improving (in this situation, the quantity to be monitored was val_loss). In the following figure, it is possible to observe the trend of accuracy and loss function during the 75 epochs on training set and validation set. These results were obtained with the following parameters: batch size = 32, optimizer = "adadelta" and loss function ="binary_crossentropy".

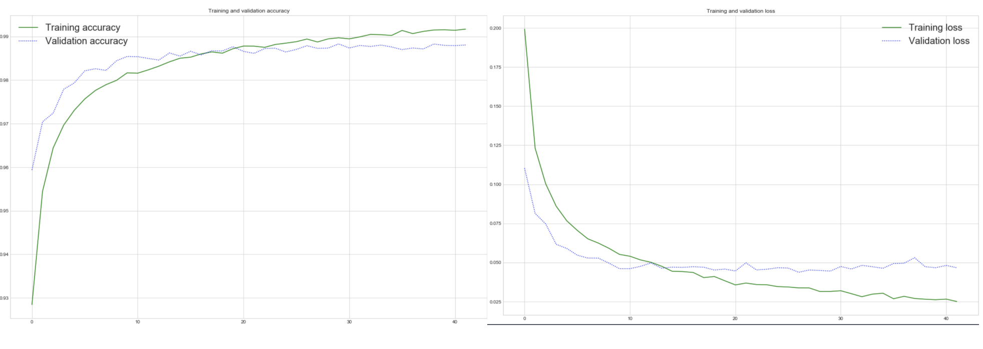

Figure 1: Accuracy and loss function on training set and validation set.

As it can be seen from the figure, after 40 epochs the neural network reaches an accuracy greater than 0.98 and the loss function approaches zero. Comparing the trend on the training set and on the validation set it can be noticed that the developed network does not overfit.

2 Visual investigation of the reconstruction abilities of an auto-encoder architecture

For solving this second task, it was necessarily to implement an autoencoder architecture. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. It is composed by three different parts:

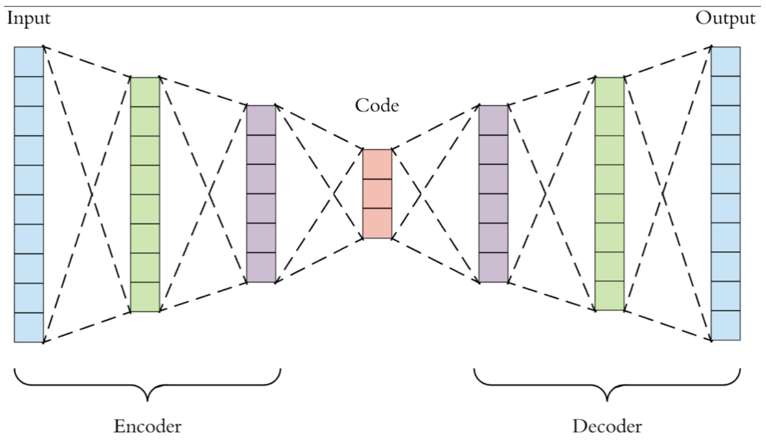

Figure 2: Graphic representation of an autoencoder architecture

- **Encoder**: it is composed by an input layer of 784 neurons (28x28) and two hidden layers of 512 and 256 neurons. For all of them, it is used the activation function ”relu”;
- **Code**: it is the central layer with 128 neurons and activation function ”relu”;
- **Decoder**: it is composed by two hidden layers with activation function ”relu” and one output layer. The hidden layers are composed by 256 and 512 neurons respectively. The output layer is made with 11 neurons (equal to the number of classes) and activation function ”sigmoid”.

In order to train this neural network, it was used ”adam” as optimizer and ”binary_crossentropy” as loss function. After several tests to choose which parameters to use, it was decided to use a batch size equal to 64 and 50 epochs. The following figure shows the original images present in the dataset and those reconstructed after the autoencoders process.

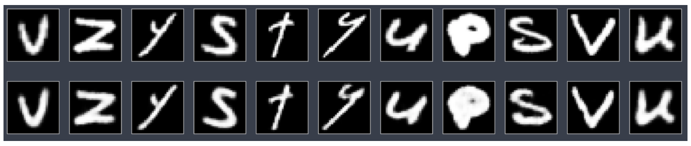

Figure 3: Original images vs reconstructed images by auto-encoder

The results obtained are very similar to the original images. The letter that is most difficult to identify after decoding is ’P’.
In order to test the quality and performance of the network, a noisy coefficient equal to 0.4 has been inserted. From the following figure it is possible to observe that the classification remains quite accurate for some letters while it is more difficult for others.

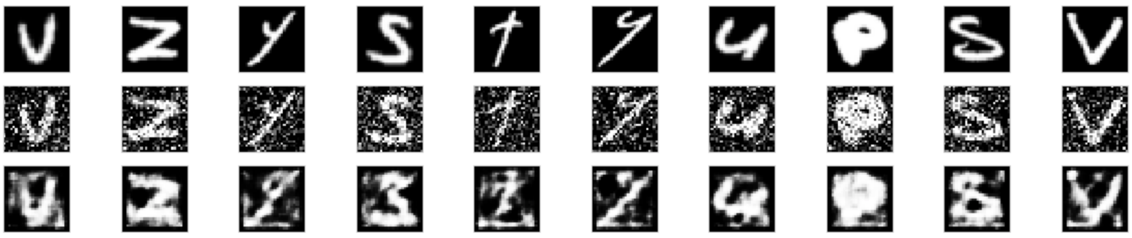

Figure 4: (1) Original, (2) Original + noise, (3) Reconstructed by auto-encoder from (2)

3 Use and evaluation of the encoded representation generated by the auto-encoder to solve the problem of supervised classification

A different way to solve the problem of supervised classification is to build a model with the same architecture as the autoencoder presented previously.
The weights of the three encoder layers (one input + two hidden layers) are set through the function "set_weight" so that they are equal to the weights of the auto-encoder of the previous task. Then "layer.trainable = False" is set in such a way as not to train the parameters associated with the neurons of such layers.
After the layer code with 128 neurons, a new layer with 256 neurons is inserted with the activation function "relu" and a kernel_regularizer =l2(0.01). Regularizers allow applying penalties on layer parameters or layer activity during optimization. These penalties are incorporated in the loss function that the network optimizes. Finally, an output layer with 11 neurons and "softmax" as activation function is added at the neural network. The model is trained with an "adam" optimizer and "binary_crossentropy" as the loss function. The batch_size is set to 32 with 75 epochs, such as in the first task.
As can be seen from the following figure, the results obtained are good: the accuracy is greater than 0.97 already after about 10 epochs, and the loss function approaches zero. In spite of this, the difference between the training set and validation set is more visible in comparison to the accuracy of first task and the trend of the validation set is less constant.

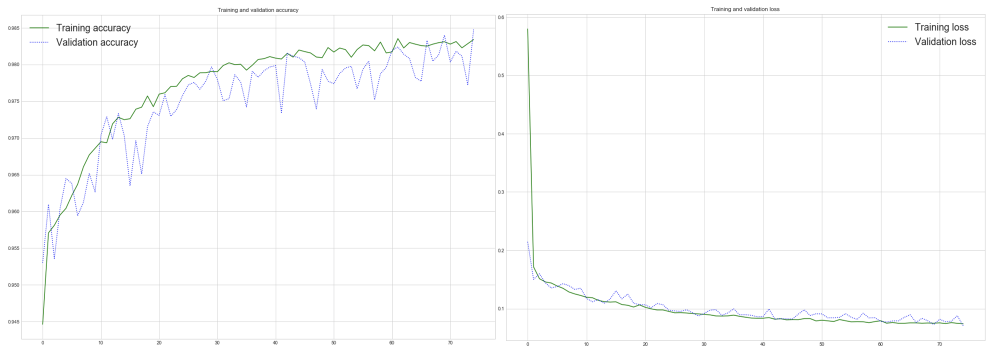

Figure 5: Accuracy and loss function on training set and validation set.