Assignment 2, Multi-class Classification problem with dense neural network and Autoencoders

Lorenzo Famiglini - 838675 - lorenzofamiglini@gmail.com

Abstract: The assignment consists on the prediction of gray-scale images of letters P - Z with a full connected neural network. Consequently, try a dimensional reduction application with autoencoders and then apply them in a classification task.

1. Introduction

The purpose of this study is to compare two different models, one without applying a dimensional reduction technique and one with the application of the autoencoder. The aim of the assignment is to evaluate the performance of the two models and to understand if an application of the autoencoders could lead us a same result with less trainable parameters.

2. Preprocessing

The data set is composed by 28*28 (pixel) features and 11 target variables (handwritten letters from P to Z). Normalization technique was applied for all the features (in order to have values from 0-255 to 0-1). After analyzing all the levels that make up the target variable, a problem of class imbalance was found: [1295, 1265, 1346, 1329, 1336, 1297,1269, 1327, 1322, 1321, 893]. Therefore, an oversampling technique called SMOTE was applied to the training set. Subsequently, the y variable was transformed into categorical feature.

2.1. Train, Test split

The training data set was split into train and test set with respect to 80% and 20% proportions. I used the test set (during training phase) for the validation part (also for tuning the parameters).

3. Dense neural network without autoencoder

The first task was based on developing a full dense neural network. The Softmax function was chosen for the output layer for the multi-class classification (11 neurons output layer) and the categorical crossentropy for the loss function to be minimized. After the optimization process, a model with 256 neurons for the first hidden layer and 128 neurons for second hidden layer were built. Rectified Linear Unit was chosen as the activation function for both layers. Furthermore, Dropout has been applied in order to avoid the overfitting: for the first hidden layer the percentage of Dropout was 0.5 and for the second one was 0.5. Despite the use of this technique, there was still overfitting. For this reason , regularization techniques were applied specifically the l1_l2 regularization in such a way to regularize the model for reducing the generalization error. A value of 0.0001 was chosen for the l1 regularization and 0.01 for the l2 (for both of the hidden layer). The Stochastic Gradient Descent was applied for the loss minimization phase with different parameters: at the learning rate, a value of a 0.01 was selected, the decay with 0.001 and the Nesterov Momentum at 0.9. In conclusion, for the training phase a batch size of 512 elements was applied for a number of 200 epochs.

4. Grid Search technique and choice of parameters

To choose the number of neurons, the learning rate, what kind of gradient optimization algorithm to use and how many layers to insert, I applied the technique of Grid Search for both of the models. The table 1 below shows all the parameters' combination that have been tested:

Parameter	Combination
Learning rate	0.2,0.1, 0.001, 0.0001, 0.00001
Decay	1e-1,1e-2,1e-3,1e-4,1e-5,1e-6
Opt	SGD, ADAM, NADAM, ADAMAX
Batch size	64,128,256,512
Epochs	50,100,150,200
Dropout	0.1,0.2,0.3,0.4,0.45,0.5
Activation F.	ReLu, LeakyReLu, Tanh
Layers	1,2,3

Tab 1.

5. Autoencoder

The aim of the Autoencoder is to copy the input variables and try to retrieve latent variables with dimensionality reduction techniques. A specific neural network was developed for this purpose. It was built into these main parts: the encoder architecture with 784 neurons for the input, 256 neurons for the first hidden layer and 128 for the bottle neck. This means that an image is compressed with a factor of 6.1. The ReLu activation function was chosen for the encode part. The last section of the autoencoder is composed by the decoder: the hidden layer with 256 neurons with Relu activation function and the output layer with 784 neurons with the Sigmoid function (and the binary crossentropy for the loss function). In the training phase the same SGD of the neural network previously explained was selected. The batch size was 512 and the number of epoch was set at 300. The result of this Autoencoder is showed in the figure 1.

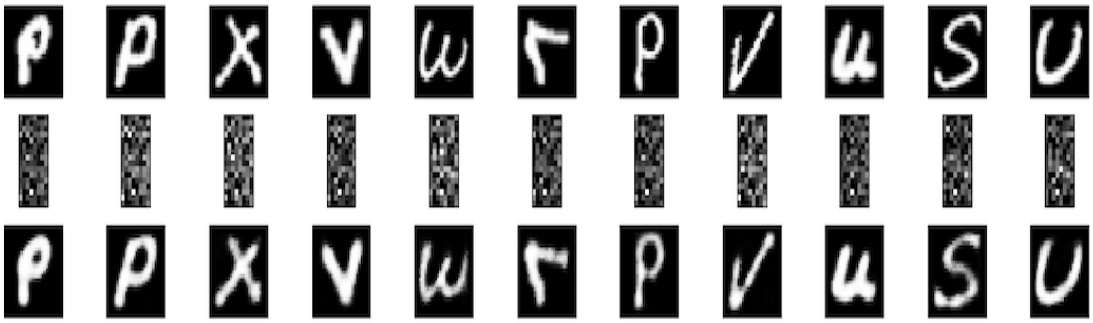


Figure 1: A sample of the Autoencoder to the handwritten letters

5.1. Classification NN with Autoencoder

The aim of the last task was to build a neural network with the Autoencoder for the dimensionality reduction and then try to classify the letters. After the encode part (of the Autoencoder) was trained, in order to classify two new hidden layers were added to the encode and bottleneck network. The first one was created with 128 neurons, the second one with 64 neurons and then applied to them the ReLu activation function. The output layer was composed by 11 neurons and the Softmax function (with the categorical crossentropy loss function to minimize). For reducing the overfitting the Dropout and l1_l2 regularization were added to these hidden layers. For the first parameter with a percentage of 0.4 and 0.3 and for the second one with a value of 0.0001 (l1) and 0.01 (l2). During the training phase only the last layers (not the encoded and bottleneck parts) were trained. For this reason the number of parameter to train (from 334,992 to 58,379 parameters) was reduce by the Autoencoder technique. The same optimization algorithm (with same parameters) for the loss function, batch size and the number of epochs of the previous model were chosen.

6. Results

6.1. Dense neural network without autoencoder results

Through the Early Stopping (40 epochs of patience) and Model Checkpoint techniques (best model for the validation loss) have been obtained results that are shown in Figure 2:

	precision	recall	f1-score	support
0	0.99	0.98	0.98	1077
1	0.97	0.97	0.97	1077
2	0.98	0.97	0.98	1077
3	0.99	0.99	0.99	1077
4	0.98	0.99	0.98	1077
5	0.95	0.97	0.96	1077
6	0.94	0.96	0.95	1077
7	0.99	0.99	0.99	1077
8	0.99	0.97	0.98	1077
9	0.97	0.95	0.96	1077
10	0.99	0.99	0.99	1077
accuracy			0.98	11847
macro avg	0.98	0.98	0.98	11847
weighted avg	0.98	0.98	0.98	11847

	precision	recall	f1-score	support
0	0.97	0.98	0.98	259
1	0.98	0.98	0.98	253
2	0.99	0.97	0.98	269
3	1.00	0.98	0.99	266
4	0.98	0.99	0.99	267
5	0.94	0.98	0.96	259
6	0.94	0.98	0.96	254
7	1.00	0.98	0.99	266
8	0.99	0.96	0.97	264
9	0.97	0.95	0.96	264
10	0.99	1.00	0.99	179
accuracy			0.98	2800
macro avg	0.98	0.98	0.98	2800
weighted avg	0.98	0.98	0.98	2800

Figure 2: Left figure is referred to the Train set and right figure is referred to the Validation set

From the above results the model achieved an optimal performance both in the training set and in validation set. The loss function in the train set was 0.31 and 0.30 for the validation set. This has been achieved thanks to the regularisation and optimisation techniques set up to solve this type of task. The test set subset (without labels) was taken into account in order to understand if these results in the validation set were reliable. Through the visualization of these test images and the predictions of the model, the accuracy has been confirmed.

6.2. Autoencoder results

The aim of the application of the Autoencoder is to reduce the dimensionality characterized by the features space. After training this algorithm the binary crossentropy loss was minimized with a value equal to 0.14, as shown in the figure 3:

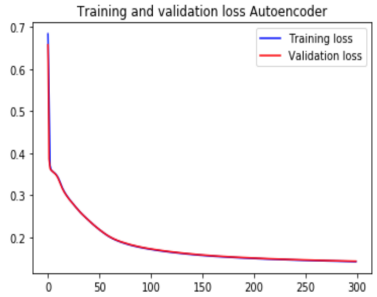
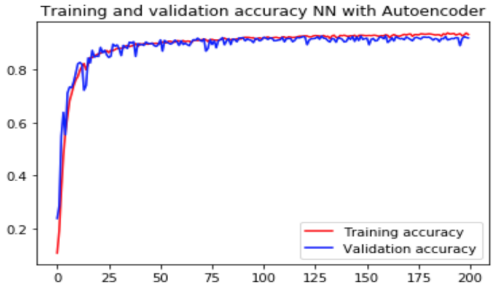


Figure 3: Trend of loss function during training

6.3. Classification NN with Autoencoder results

Let's see how the autoencoder affects the results obtained. For train set: accuracy 0.95, loss 0.28, F1-measure weighted 0.96. For the validation set: accuracy 0.93, loss 0.41, F1-measure weighted 0.92. There is a presence of overfitting but it's not so relevant. From these results we can conclude that the dimensionality reduction task with classification achieved a good outcome.



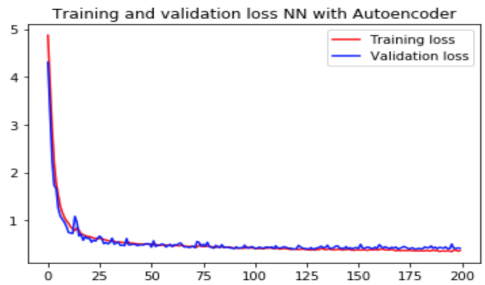


Figure 4: