ANNDL Assignment 2 - Alessia Cotroneo, Alessandro Del Vitto, Michele Di Sabato

Reference Notebooks:

- AR_Model.ipynb
- DirectForecasting_Cotroneo_DelVitto_DiSabato.ipynb

Introduction

In this report we describe the steps we followed and the issues we encountered while conducting experiments for the second competition of the course Artificial Neural Networks and Deep Learning. At first, we tried to implement **Direct Forecasting**: we tuned the hyperparameters, designed the model architecture and tested different loss functions. Then, we passed to **Autoregressive Forecasting** and we observed an increase in the accuracy in both the training and test sets. We **conclude** with the description of our best model.

1. Direct Forecasting

At first we decided to use a direct approach based on bidirectional LSTM models. Because of the length of the prediction horizon, we expected to obtain moderate results.

Hyperparameter tuning

The most influential hyperparameters were *windows* and *stride*: we tuned them according to the plot of the 7 features in the dataset. In particular, we wanted the window to be large enough to capture recurrences in the time series, such as sequences of peaks or frequent patterns. After having fixed the window, we moved to the stride: we found that the best results were reached when two subsequent sequences shared roughly 90% of the overlap. The best values for *window* and *stride* were, respectively, 100 and 10.

Model definition

The direct model is based on bidirectional LSTM models: after the input layer and a bidirectional LSTM layer (128 units), we added 3 couples of layers, each composed of a one dimensional convolution followed by one dimensional MaxPooling. Then we decided to include an additional bidirectional LSTM layer (256 units), followed by a Global Average Pooling layer, a dense layer and a final one dimensional convolution layer. We also added dropout layers to reduce overfitting. We found that the variation in the accuracy both in the training and test set were more influenced by the *window* and *stride* parameters, than by the layer choices. In addition to the Mean Squared Error (MSE) loss function, we also trained the network using the Mean Absolute Percentage Error (MAPE) and the Mean Absolute Error (MAE) loss functions: the results on the test set proved that the neural network trained on the MAE achieved the best results.

Test set performance

To assess the performance of the network, we split the provided data into two parts: one strictly used for training, and the remaining for testing (we opted to use the last 4000 values of the time series for testing). We decided, in accordance with the lab sessions, to call this last part of the dataset "test set". According to Figure 1, the network seems to perform well for most of the features, except for *Sponginess* and *Meme Creativity*.

Future prediction

Finally, to simulate the performance of the network on the actual test set, we decided to plot the prediction on the last sequence available, called "future prediction" in the laboratory sessions. As Figure 1 clearly shows, the orange bands do not seem to widen as time progresses: as seen during the exercise sessions, this is an indicator of the fact that the ability of the LSTM to learn the time series is quite poor.

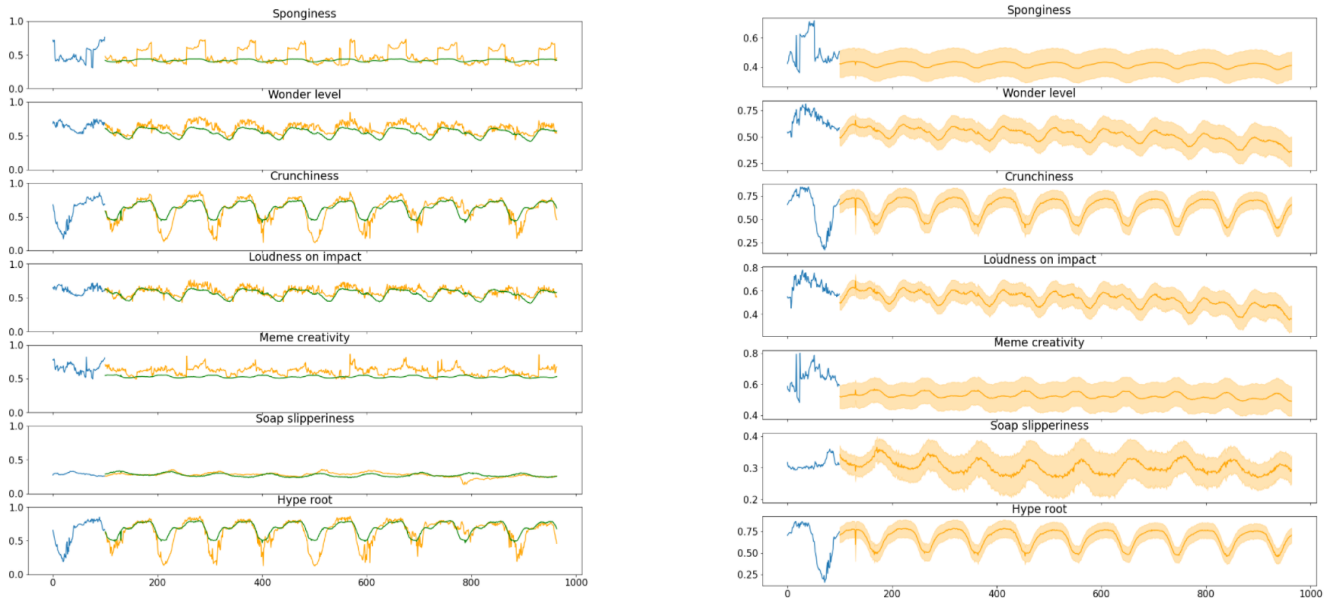

Figure 1: on the **left**, the **test set** prediction plot: the green line is the predicted curve, the orange one is the true curve. On the **right**, the **future** prediction plot.

Because of this reasons and since we thought that the problem could be better handled by training the network so that it is able to reproduce the time series progressively, predicting a small time interval at a time and including sequentially this estimate to obtain the next, we moved to an Autoregressive model.

2. Autoregressive Forecasting

Our first attempt with an Auto Regressive model has been a complete failure. We quickly realized that the task of predicting 864 steps ahead in the future one by one was extremely difficult. We then tried to reduce the number of predictions that the model had to perform, augmenting the parameter *telescope* in such a way that it was still a divider of the total regression telescope. This gave great results. The proper AR model with a suitable *window* and *telescope* pair was able to predict the test set significantly better than all the direct forecasting models we have tried. We highlight that a suitable *window-telescope* pair is needed because there must be a balance between the length of the prediction period and the amount of data the network can exploit. Moreover the *telescope* parameter must be set, as explained before, as a divider of the regression telescope number. We have made several attempts and in the end we noticed that the best results were given by the models with window, stride and telescope around 200, 10 and 16 (so that the AR model had to be applied 54 times). In all our trials we used a model composed of a bidirectional LSTM layer followed by a Conv1D and MaxPool1D layers (see Notebook AR_Model.ipynb).
The AR forecasting technique works well with periodic signals, since it incorporates its prediction on the output at the previous step to forecast the output in the next period of length *telescope*. We think this is the reason why we are able to better predict the peaks and the valleys of signals like *Crunchiness* and *Hype Root*. At the same time, this property can have side effects: whenever the periodic signal changes, the model presents significant errors.
From the solution of the AR model on the samples of the test set we noticed that the behaviours of the first and fourth signals, namely *Sponginess* and *Loudness on impact*, were the most difficult to learn. This seems reasonable, since they are characterized by small and sudden variations that are harder to replicate, given the fact that the other features of the dataset present smoother periodical variations. We attempted to make the prediction of these signals more accurate, but unfortunately we ended up losing precision on the prediction of the other features. Moreover, we noticed that the highest error on the test set was reached when predicting features number 3 and 7 (although their trend was quite well captured), instead of *Sponginess* and *Loudness on impact*. For these reasons, we decided that improving the prediction performance of the network on *Sponginess* and *Loudness on impact* was not worth sacrificing the precision on the others.

3. Conclusions

In conclusion, the highest level of accuracy in the test set has been reached when using the Autoregressive model with values for *window*, *stride* and *telescope* respectively equal to 200, 10 and 16.
Finally, the following plots show the learning rate decrease, the mean absolute error (MAE) and mean squared error (MSE) loss functions in the training phase of the Autoregressive model (see Figure 2). Moreover, Figure 3 shows the performance of the model on the test set and on the last sequence available (future prediction).

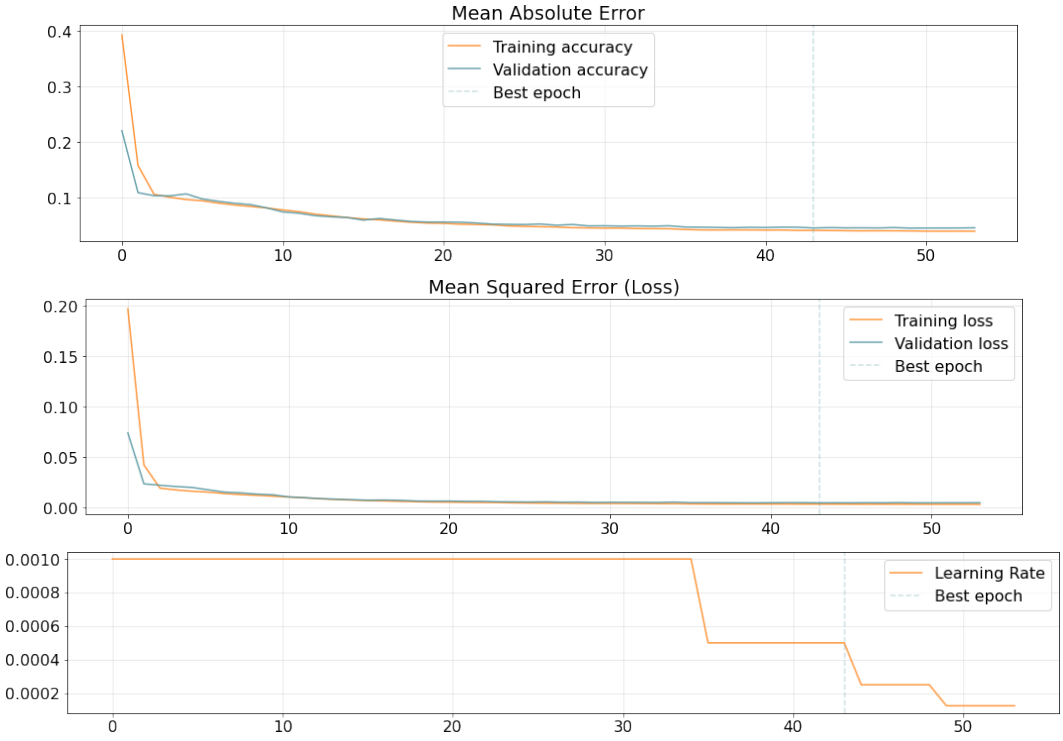

Figure 2

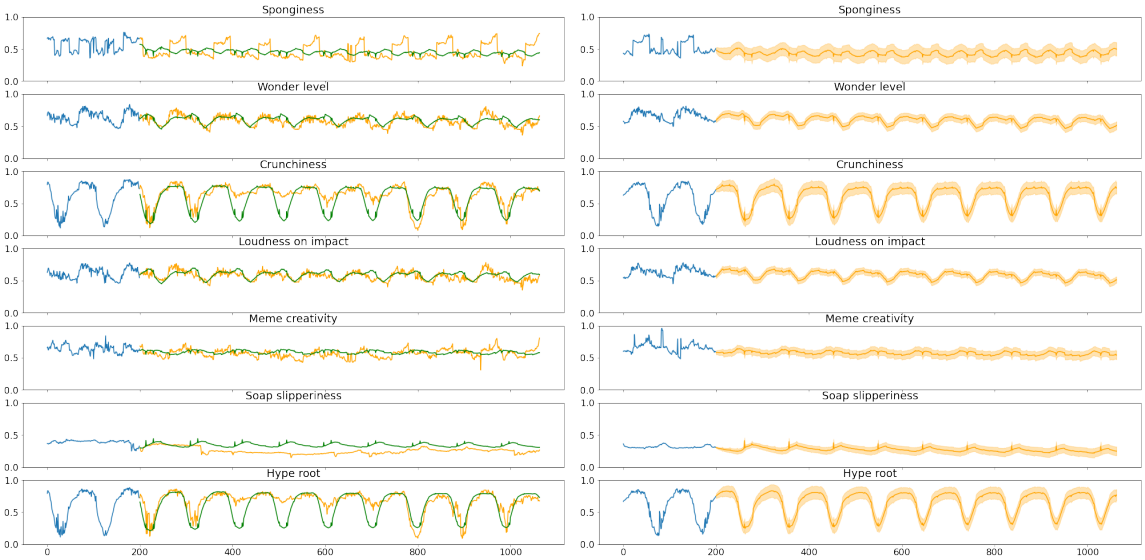

Figure 3