# AN2DL - Competition 2 - Report

Bergonzoli - Bonfanti - Giovannini

## 1. Introduction

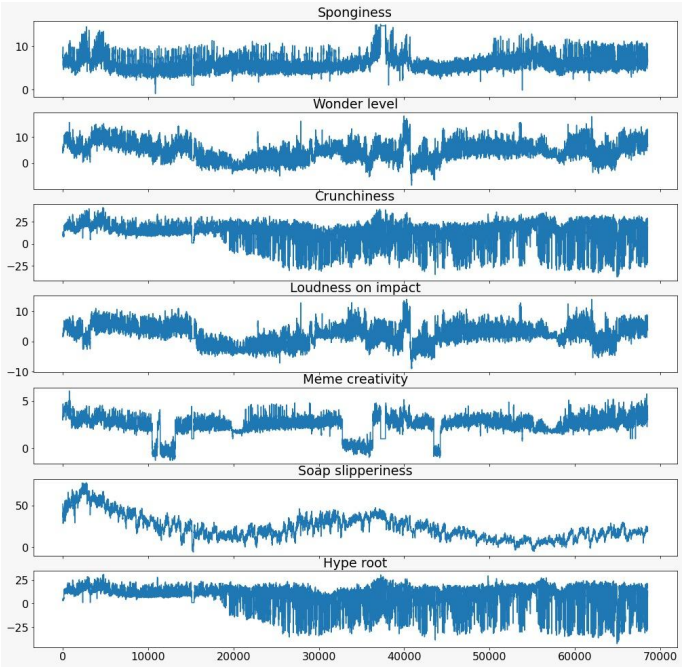
Our case study is a Time Series Forecasting, in which we have to predict future samples of a multivariate time series. The dataset is made of 68528 data, with 7 labels: sponginess, wonder level, crunchiness, loudness on impact, meme creativity, soap sliperness, hype root.

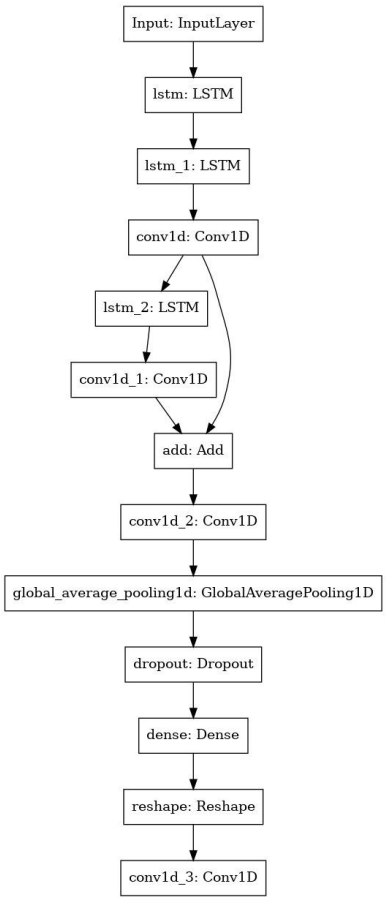
### 2.1. Solution: Direct Forecasting

We preprocessed the data applying a normalization over the observations in each different label.

We started with a trivial model, composed by some convolutional layers followed by some LSTMs, applying it to the direct forecasting method. We directly predicted 864 points, using a telescope of 864.

We also tried to change two other parameters: window and stride, paying attention to the fact that the stride should be a divider of the window. We also noticed that the window should be approximately of the same order of magnitude as the telescope, to have enough points to make a good prediction.




### 2.2. Solution: Autoregressive Forecasting

To improve our predictions, we implemented the autoregressive forecasting method (with telescope = 1) to the same model above. However, we didn't notice a significant reduction of the RMSE, and for this reason we increased the telescope, and this was actually a good choice.

Also this time we made some changes on window and stride in order to discover the best values for this model. Since we reduced the telescope up to 8, we set the window value to 200 (combined with a stride of 5). Indeed, at the beginning we had a window of thousands of units, but we noticed that with such telescope it worked better a smaller value.

At this point, we thought of introducing skip connections. The model we designed was composed by LSTMs, convolutional layer and Global Average Pooling, connected to the last FC part. *(left image)*

Knowing that a valid alternative for the Long Short Term Memory is the Gated Recurrent Unit, we decided to replace the LSTM layer in the skip connection with the second one. This was actually a good idea!

Seeing previous results, we substituted all the LSTM layers of the model with GRU: but the result wasn't what we expected. In fact, even if it was better than the very first one, the model with both components was still the best one.

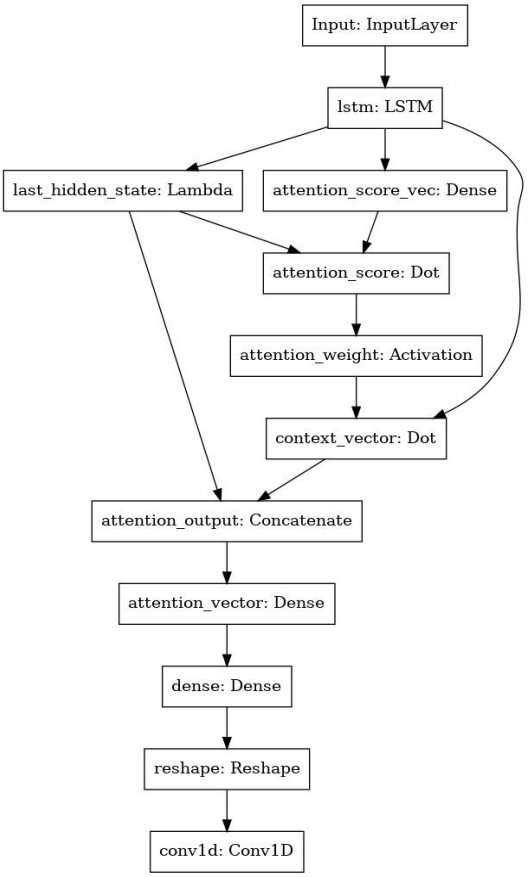
Up to this point, we had always kept the same loss function and metric: MeanSquaredError and MeanAbsoluteError respectively; so we tried other combinations, introducing also the MeanAbsolutePercentageError. However, in the end, the average RMSE was still better with the original setting.

We used the callback function ReduceLROnPlateau, that decreases the learning rate, when a metric stops improving, during the training phase. Starting from the default lr value of 1e-3, we reduced it to 1e-4 and 1e-5. The overall best was 1e-4, but the last gave us a better score in the 'Sponginess' label.

Our best model so far had a low number of epochs, and we wanted a longer training phase, so we increased the patience value (which determines after how many epochs without improvements the learning can be stopped) in the EarlyStopping callback function, and this step actually enhanced the accuracy of our predictions, giving us our best result. (RMSE 3,8450)

### 2.3. Solution: Attention Mechanism

Meanwhile, we started trying to design a brand new model based on the attention mechanism. For the very first attempt we simply concatenated an LSTM layer and an attention layer, importing the attention library, which gave us interesting results. *(right image)*




We attempted to change the parameter of the units in the LSTM, and we saw that the best one was 64. Working on this model we also added some convolutional and max pooling layers, but without any particular result. For this reason we thought it would be a good idea to add the attention to our best model previously analyzed, before the last dense layer.

With this combination we obtained a RSME of 4,0269, which is better than the one with only LSTM and attention, but the best predictions were still made by the model in the 2.2 paragraph.

### 2.4 Solution: Combining different models

Eventually we noticed that all the different labels of the dataset reached the minimum RMSE in different models; our network wasn't actually the best one with respect to all the features. For this reason we drafted another one which computed predictions using seven different models, one for each label; this gave us a RMSE of 3,4269.

## 3. Conclusion

To sum up, the model that carried out the minimum RMSE was the one with the best RMSE in all the labels, as we expected.

We combined the following models:

| Name Submission | Codalab profile | RMSE | sponginess | wonder level | crunchiness | loudness on impact | meme creativity | soap slipperiness | hype root |
|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| nobid200e5 | giuliabergonzoli | 3,8668 | 1,7466 | 1,2975 | 6,6234 | 1,2017 | 0,6719 | 2,6826 | 6,8534 |
| nobid200e5NOteste-5 | giuliabergonzoli | 4,1673 | 1,4845 | 1,6075 | 6,5720 | 1,3373 | 0,7909 | 5,4273 | 6,4587 |
| 388modificamax | giuliabergonzoli | 4,4347 | 1,7681 | 1,3909 | 6,4534 | 1,0957 | 0,5231 | 6,7813 | 6,5952 |
| sub_model12 | ilariabonfanti | 4,4984 | 2,0951 | 3,6337 | 7,3453 | 2,7246 | 0,8702 | 2,5356 | 7,4494 |
| IdeIdeAttention8Fuori2 | chiara_giovannini | 5,4056 | 1,7327 | 1,8248 | 7,5439 | 1,4334 | 0,4827 | 9,7302 | 6,6577 |
| SkipSenzaBidnuovaPat | chiara_giovannini | 3,8450 | 1,6335 | 3,6412 | 5,9691 | 2,7877 | 0,8333 | 2,9396 | 5,9013 |