Bono Federico
Cecere Nicola
Zanutto Davide

# Artificial Neural Networks and Deep Learning

## Time-series classification challenge

The second challenge of this year was a time-series classification problem. The objective is to correctly classify samples in the multivariate time series format. As we saw in the laboratories during courses, our team decided to start experimenting with some basic networks. In particular, we concentrated on three different network layers: LSTM, BiLSTM, and 1D Convolutional Neural Network. We next attempted to preprocess and standardize the data in a variety of ways, and build a network mixing layers of different networks.

## Preprocessing

In this challenge, we experimented with various preprocessing techniques, however eliminating outliers and performing global and class-specific normalization and standardization did not significantly increase accuracy. First of all, using the ratio 0.8 - 0.1 - 0.1 we split the dataset into training, test and validation set. Then we tried some external libraries we tried, such sklearn RobustScaler, but they did not work with our dataset. We created a model that includes feature selection (i.e., eliminating some features from the time series) and one that attempted to adjust the window size by following the tips in the course Google Drive. We attempted (failing) to adjust the class weights in the network since some classes had fewer datapoints. Additionally, we attempted to add some batch normalization layers to reduce the dataset width, however the end result was a loss of accuracy. One of the most recent experiments we performed involved adding noise using an external library, but the results weren't much better. Eventually our best model was built without any specific data preprocessing.

## 1D Convolutional Neural Network

In order to get familiar with the dataset, we started with the more fundamental and well-known model that was recommended in the laboratories. We began by experimenting with different convolutional and dense layer combinations on the network that was taught in class. We tried switching the activation function from Relu to LeakyRelu in particular, and we saw some benefits. After that, we attempted some of the preprocessing stated earlier, but they all performed poorly

on this kind of network. Additionally, we saw significant overfitting, so we added some dropout layers and increased the batch size to 512. The final score we obtained using the 1D convolutional neural network is 0.6239.

# Lstm

We also experimented with an LSTM-based network design. As always, we started with the laboratory's notebooks, reaching almost 0.6 in accuracy from the beginning. Given such an interesting starting point, we tried to optimize the network, changing the layers parameters and tuning the architecture. At the end the best result with this network was reached with a more complex topology, with multiple LSTM layers and higher number of parameters from the classification layers.

# BiLstm

In this section we will talk more in depth about the second network we tried to use in this challenge: BiLstm. We first tried to run the network as it was in the laboratory's notebook, but we didn't reach a high score. We then tried to vary some of the layers' parameters, including the dropout and the shape, and we managed to reach some decent results. Unfortunately, they were not as high as the ones obtained with the LSTM network. Afterwards, we also tried to add some layers and combine them in different ways but nevertheless all these tries did not improve our score.

# 1DCNN-BiLSTM

After we experimented with three different networks we decided to attempt to combine their respective strengths. The initial attempt used convolutional layers in the model's first section, lstm layers in the second section, bilstm in the third section, and some dense layers for the classification section. In this way we reached a score of 0,68. After deciding that it would be better to not use all of the tree kind of layer at once, we first decided to delete the bilstm section, for which we received a score of 0.69. We next removed the lstm part and readjusted the bilstm component, resulting in our highest score of 0.7052. We choose to utilize the LeakyRelu activation function in this network as well because it seemed to operate more effectively.

# Final submission

The final model we submitted was the 1DCNN – BiLstm with dynamic learning rate.

The network was built with the following layers:

- Two 1D convolutional layers;

- Two Bidirectional lstm layers;
- Three dense layers;
- One dropout layer.

Graphs generated with best model: