Introduction

The dataset is of size 2429x36x6 sorted in 12 different classes with a highly unbalanced repartition. Six time series can be distinguished from every sample within 36-time units. No more information was given, allowing a better understanding of the problem.

Preprocessing

The first step was to visualize the dataset to have a better understanding of the data for preprocessing purposes.

This way, we saw some patterns in the different time series in terms of evolution, but also in terms of amplitude. We first thought about performing a min-max normalization to bring every data between 0 and 1, but it only leads to a still unbalanced range of value inside this interval and so was pointless. Consequently, we tried to standardize to bring mean to 0 and standard deviation to 1 for the different Time series. It improved the Time series representation after replotting. However, this process was not kept during further experiments with models as it was not improving the results.

As provided Time series were having different magnitudes for a given samples, we thought about considering the correlation among the Time series. We tried to remove one signal that was almost always close to zero. This was not changing significantly the results. It decreased the recognition of a class that has a higher correlation with regards to this signal.

Time series had sharp edges along time axis. Different average filters with kernels of different sizes were tested but it did not improve nor worsen the results.

Considering the imbalance of the dataset, we computed the weights of the classes to apply it during training. Unfortunately, it was not improving. This comes from the fact class 9 is overrepresented and results were good recognizing this class. With weights, this class was not well recognized anymore, and results were poor, however the confusion matrix had a cleaner distribution. This was not considered for the final model.

Another way to manage the imbalance was to drop samples from class 9 in both training and testing set. The confusion matrix was looking better at the end and results were slightly better with some initial models. However, it was not allowing to tune correctly the models.

Data augmentation

Data augmentation was tried in different ways: adding gaussian noise for classes with only few samples, mean shift, windowing (with application of zero padding). In all cases it did not improve learning. We did not know exactly which other transformations could be pertinent as we didn't know what exactly the data were.

LSTM

The main model on which we were working was the Long Short Term Memory model was built, and parameter tuning was performed. We reached the best test accuracy as 67%. The bidirectional version of LSTM was also tried but it didn't improve significantly.

Convolutional network

Previous methods applied in the previous cases with methods were tried. A fully convolutional network was tried but it did not perform well (40% accuracy for the 1DCNN) as there is no real correlation between Time series and so the application of kernel is not suitable. Maybe a not fully convolutional network could have fixed this problem, but we decided it was not worth pursuing that aisle given available time.

Transformer

A transformer was implemented, as suggested by the TensorFlow library for times series, but computation time was much higher for not better results, and it was decided to not consider it.

Transformation into data frame for further processing

To perform some further preprocessing and maybe visualize the data in a more efficient way, we converted it to a pandas Data Frame. Because Pandas only receive two-dimensional data, there was an inconsistency. In order to do the conversion, the dimensionality of the dataset was reduced to two by transformations, leading to a Data Frame of 2429*36 rows and 6 columns representing the different channels. A unique ID was assigned to every 36 consecutive rows.

Visualization didn’t give any clue about how to improve past experiments. We tried to feed this new format to a LSTM model, but it didn’t bring any improvement. In contrast, it worsens significantly the accuracy and recognition of the least present classes.

8-crossfold validation and assemble

Our last improvement trial was to perform an 8-crossfold validation, to consider 8 different splits of the dataset and to train 8 instances of the best Bi-LSTM model we obtained previously. In all cases, the models were performing similarly. However, we tried to assemble them all. The final accuracy after submitting this model to test it on new data gave us one additional percentage of accuracy (in total 68%) compared to one model alone and was so the retained model.

Conclusion

To conclude, this assignment allowed us to test a lot of different methods and preprocessing to find a way to improve the results as nothing was really working. We were clueless with respect to the reason why all our attempts were failing to improve or worsen the prediction capability or learnability. Maybe with a deeper understanding of the problem and classes we could have managed to find more accurate processing. One improvement we didn’t have time to implement was to augment the data thanks to a generative adversarial network, that could generate data considering a real pattern determined by the network. Another improvement would be to add a time interpolation method to reduce sharp transition in the Time series.