Neural Network Project Report:
Convolutional Neural Network for Electrocardiogram Classification

Francesco Fabbi   Federico Kieffer

Contents

1 Introduction

The present project focuses on the implementation of a Convolutional Neural Network(CNN) for the Electrocardiogram(ECG) classification. In particular, the main objective of the work is the atrial fibrillation recognition task, as it was proposed in the 2017 PhysioNet/CinC Challenge [1]. Atrial Fibrillation(AF) is the most common type of cardiac arrhythmia. AF affected patients are more prone to serious complications such as ischemia or stroke and therefore its early detection is crucial for more effective prevention and treatments. At this end, we propose an 8-layer 2D CNN for the classification of the spectrograms that are associated to the original ECG recordings provided in the dataset of the Challenge. More precisely, the workflow that we have followed is organized in three fundamental steps (fig.1): signal pre-processing (section 2), signal transformation (section 3) and signal classification (sections 4,5,6,7).

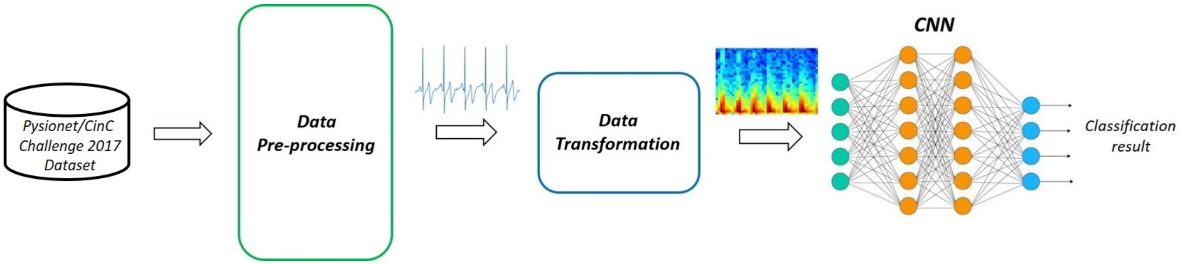


Figure 1: *Workflow*

1.1 The Dataset

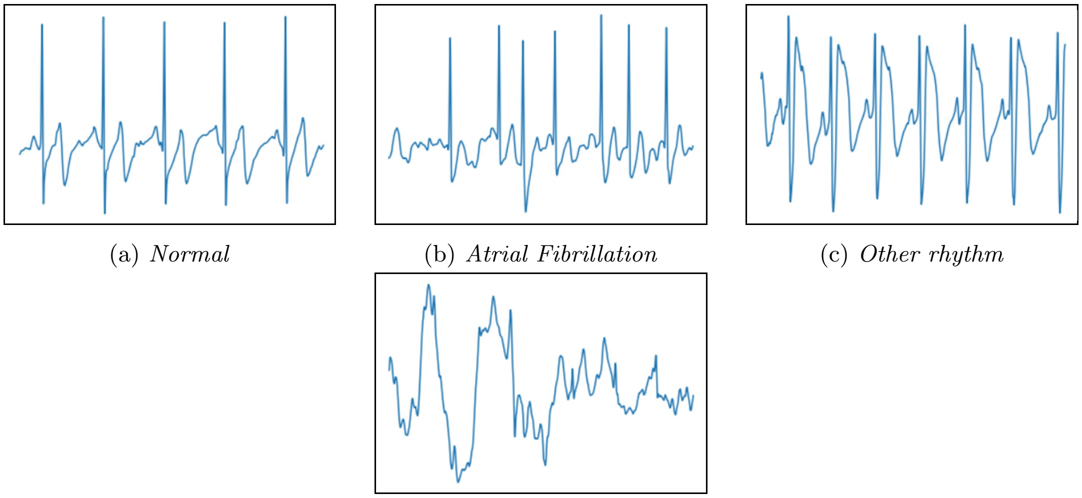
The dataset includes 8528 single lead ECG signals recorded from real patients with an AliveCor$^{\circledR}$ Device. The signals are sampled at 300 Hz and have duration ranging from 9 to 60 seconds. All ECG signals are labeled with one of the following four classes: normal sinus rhythm(N), atrial fibrillation(A), other rhythm(O) and noise($\sim$).

Class	Samples nbr.	Proportion
Normal	5076	59.52%
Atrial Fibrillation	758	8.89%
Other	2415	28.32%
Noise	279	3.27%

Table 1: *Dataset summary*

A recording example for each of the four classes is reported in fig.2. Several aspects of this dataset render it quite challenging. Firstly, the classes are markedly unbalanced (table 1). Secondly, the duration of the recordings, and hence the number of samples representing each ECG, have a high variability. Finally, the ECG quality of a non-negligible part of the records is rather poor, as evident episodes of noise superposition are present even in those recordings labeled as normal.

For this reason we have decided to tackle the problem proposed in the challenge gradually. In particular, at a first stage, we have focused on the main objective of the challenge (i.e. the atrial fibrillation recognition) and we have considered a binary classification problem in which we have kept only the two classes N and A. The proposed architecture has in this context exhibited a classification accuracy of about 83%. Secondly, we have enriched this problem by adding as third class the recordings classified as noise, yielding a classification accuracy of approximately 78%. Finally the entire 4-class classification problem has been tackled. The aforementioned CNN has allowed in this case an overall classification accuracy of 63%.



(a) *Normal*

(b) *Atrial Fibrillation*

(c) *Other rhythm*

(d) *Noise*

Figure 2: *Examples of ECG recordings*

2 Data Pre-Processing

First of all, a series of pre-processing operations have been carried out on the original signals so as to facilitate the CNN in the feature extraction process. The first problem that we had to tackle in this context was the different length of the ECGs. To this purpose we have performed, as suggested in [2], a data segmentation operation. In particular, we have selected a segment length of 1500 samples, which correspond to 5s of registration. This operation has a triple scope. Firstly, the inputs' sizes are made consistent with each other. Secondly, a data augmentation procedure is also established since an average number of 6 different training samples are obtained from a single original recording. Finally, the segmentation procedure has feature extraction purposes as well. In fact, the most relevant feature that distinguishes a normal ECG from an AF-affected one can be retrieved in the different distances between the consecutive peaks in the corresponding waves. In other words, while in normal ECGs this distance tends to be constant and regular, in AF-affected ones this regularity is completely lost (fig.2). In this respect, the ECG segmentation allowed us to zoom-in and better capture this geometric feature. This process has been particularly useful in the perspective of the realization of the spectral images associated to the pre-processed signals. After the segmentation operation, the resulting segments have been collected in a new *augmented* dataset, in which an equal number of samples for each class is as well allocated (balanced dataset).

Three more pre-processing steps have been performed on this augmented dataset. Firstly, a median filter has been applied to the signals so as to remove the *floating bias* effect from the leads. Secondly, all the recordings have been re-scaled to the same dynamic range [-1,1] (input normalization). Finally, a band-pass filter with 0.5 hz and 40 hz as cut-off frequencies as been as well applied to the ECG waves. In fact, although the samples provided in the original dataset had been already band-pass filtered, the first spectrograms that we have realized highlighted that the signals were still affected by undesired high frequency noise components.

3 Data Transformation: ECGs' spectrograms generation

After having pre-processed the original signals, the resulting ECGs have been turned into spectrograms. A spectral image is a visual way of representing the signal strength over time at the various frequencies that are present in the waveform. Moreover, to better highlight the dynamic range of the frequency content, a logarithmic transformation has been applied to the obtained spectrograms. In particular, the *signal.Spectrogram* module of the *Scipy* python library has been used to convert the 1D signals into 2D spectrograms. The function hyperparameters have been chosen accordingly with the guidelines provided in [3]. In particular, the spectrograms are computed using a Tukey window of size 64 with shape parameter 0.25 and 50% overlap (i.e. setting the *noverlap* parameter to 32). Furthermore, the colormap '*jet*' has been applied to the resulting spectral images to better highlight the variations of the frequency intensities within the signal. The logarithmic spectrogram of an example ECG signal of the class N is shown in fig.3

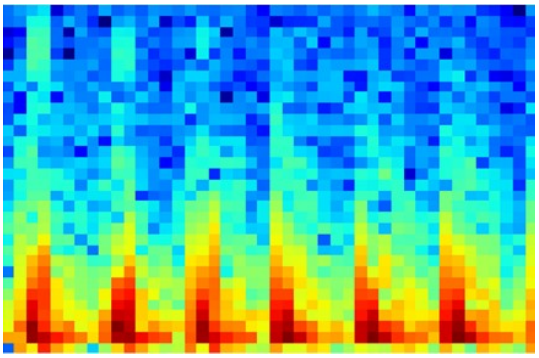


Figure 3: *Logarithmic spectrogram of an example ECG signal*

4 Deep Learning: The Network

In this section we give a detailed description of the proposed network architecture as well as the training procedure followed.

4.1 Network architecture

Many different network architectures were analysed and tested during the experimental phase related to the three different aforementioned problems. Regarding the layers' organization, we followed the guidelines provided in [3]. In particular, we started from a network with only four convolutional layers, divided into 2 blocks of two layers each. From a block to another, the number of filters of the next block were increased by 32. Apart from the number of filters, the two blocks have the same structure: the kernel size

is always set to (3,3), all layers use the same activation function (**'ReLu'**) and the second convolutional layer of each block is provided with a (2,2) **max pooling** operator.
However, this firts structure of network didn't lead to fully satisfactory results, so the structure of the network has been progressively complicated and in the end we built up an architecture (fig. 4) with four convolutional blocks, each having the same structure above described.

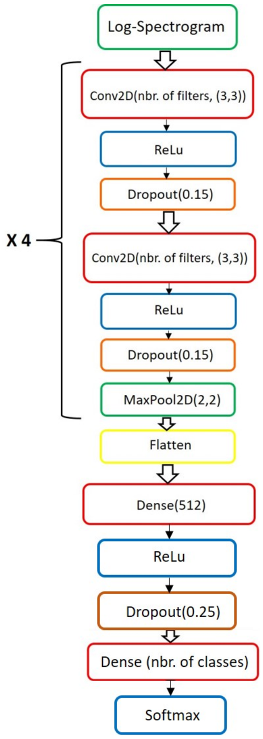


Figure 4: *Network Structure*

Furthermore, the convolutional layers are followed by a dense layer provided with 512 Relu units and, finally, by an output **softmax** layer having a number of units equal to the number of classes of the particular problem considered.
We have also noticed that trying to further increase the number of convolutional blocks was useless since the network was no more able to increase the classification performances. More precisely, the introduction of additional layers was disadvandageous since the model was stopping to learn general features from the training samples and started sooner to overfit to the dataset. This is not surprising, since a model with too much capacity compared with the data availability tends to memorize the training samples rather than generalizing and finding the most suitable mapping function needed to classify the input data.

4.2 Training procedure

First of all, before feeding the data to the network, an image reshaping and normalization has been performed and, after that operations, the dataset has been divided in training and test sets using the **sklearn** function *train_test_split()*. In particular, in the Binary Classification problem we had 2000 samples for the N class and 2000 for the A class, while in Three and Four classes classifications we had 1000 samples for each class. The percentage of data used for testing has been set to 0.2 in the binary classification case and to 0.1 for the remaining two problems. Moreover, during the train-test split operation, we made sure that the balance between the classes that characterized our datasets was as well kept in the resulting training and test set. The choice of a smaller percentage of validation samples in the three and four classes classification problems was due to the much lower availability of data of the class $\sim$. For this reason, we wanted to keep more samples for the training phase in order to reduce the overfitting of the model. In our approach to the project, a lot of attention has been paid to the problem of overfitting, which could affect the generalization ability of the convolutional neural network. Therefore, we took into account not only the Data Augmentation procedure performed during feature extraction phase, but we have also employed Dropout and Early Stopping techniques. In particular, we added a Dropout rate of 0.15 to each convolutional layer and a Dropout rate of 0.25 for the first Dense Layer (fig.4). For what concerns the Early Stopping, we implemented it by monitoring the **val_acc** value obtained at each epoch, with a *patience* of 4. This means that if the value of this parameter does not increase within a window of 4 epochs, than the training procedure is *early* stopped, so as to prevent the network from overfitting to its training samples.

Regarding the optimizer choice, we proceeded by trial and error and we finally found that the most suitable one for our particular problem was the **SGD (Stochastic Gradient Descent)**. In fact, all the other optimizers that have been tested tent to get immediately stuck in local minima, thus leading to poor classification performances.

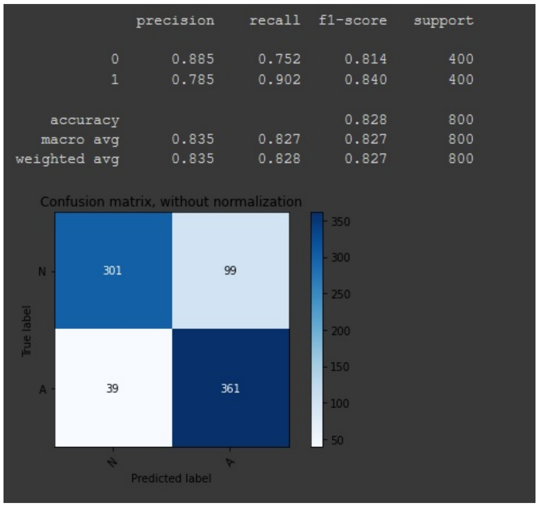
In this training framework, we set the maximum number of training epochs to 50, although the training procedure was often interrupted much earlier due to the Early Stopping command. In addition, we decided to consider 20 as the value for the *batch_size parameter*. Thus, during each epoch, the number of training samples in each single forward/backward pass was equal at most to 20. This choice seemed to allow a good compromise between an efficient network's training procedure, which would require smaller batches, and an accurate estimate of the gradient, which would instead require an higher number of samples composing each batch.

5 Simulation results

In this section we illustrate the classification performances of the network above described in all the three sub-problems that have been considered in the work.

5.1 Binary Classification Case

The network has been first of all tested in a binary classification problem, in which only the two classes N and A have been taken into account. The performance metrics that have been monitored in the testing phase are below reported, together with the associated confusion matrix.



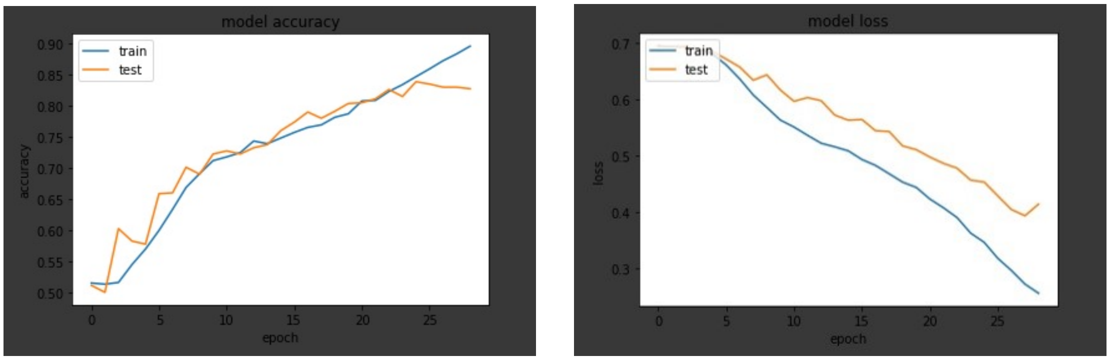
```
              precision    recall  f1-score   support

           0      0.885     0.752     0.814       400
           1      0.785     0.902     0.840       400

    accuracy                          0.828       800
   macro avg      0.835     0.827     0.827       800
weighted avg      0.835     0.828     0.827       800
```

Figure 5: *Performance metrics and confusion matrix plot (Binary case)*

As anticipated, the network allows a classification accuracy of about 83%. A deeper analysis of the results reported allows also to highlight the presence of an higher number of false positives (FP), namely normal ECGs wrongy classified as pathological, with respect to false negatives (FN), i.e. those AF-affected patients that are confused with healthy ones. This kind of balance between FP and FN is indeed preferable with respect to the opposite situation. In fact, a missed diagnosis of the pathology would bring much worse consequences compared with a wrong classification of a normal patience. This result is also confirmed by an higher precision of the N class and, correspondingly, by an higher recall of the class A. The accuracy and loss graphs, which are representative of the binary classifier training history, are below reported.



(a) *Accuracy*

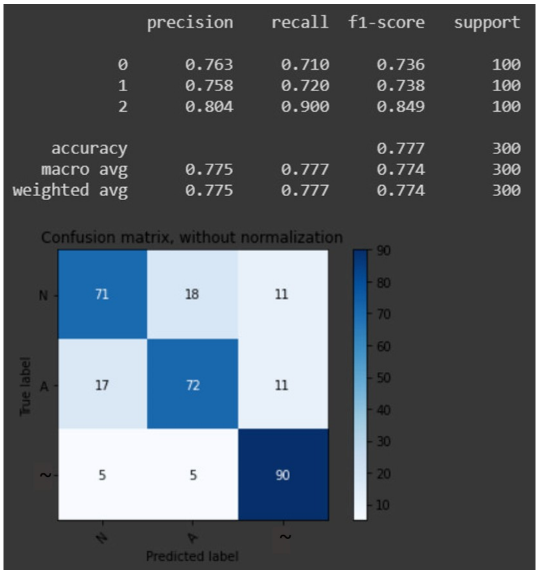
(b) *Loss*

Figure 6: *Accuracy and loss graphs (Binary case)*

It can be observed that the accuracy and loss curves have respectively an almost monotonically increasing and decreasing behaviours, which indeed reflects a correct cost function minimization performed during the training phase. In particular, thanks to the above discussed techniques, the resulting model is just slightly affected by *overfitting*, as highlighted by the moderate offset between the train and test accuracy curves reported at the end of the training process.

5.2 Multi-Class Case: Three classes

As a second step, we have added to the classification problem the recordings labeled as *Noise*, yielding a three-class classification problem. Those recordings represent signals that are believed to be too noisy to be associated with a reliable diagnosis. As expected, the introduction of this third class does not affect much the resulting classification performances of the CNN, which allows an accuracy of about 78%. The performance metrics relevant to this case are in the following reported:



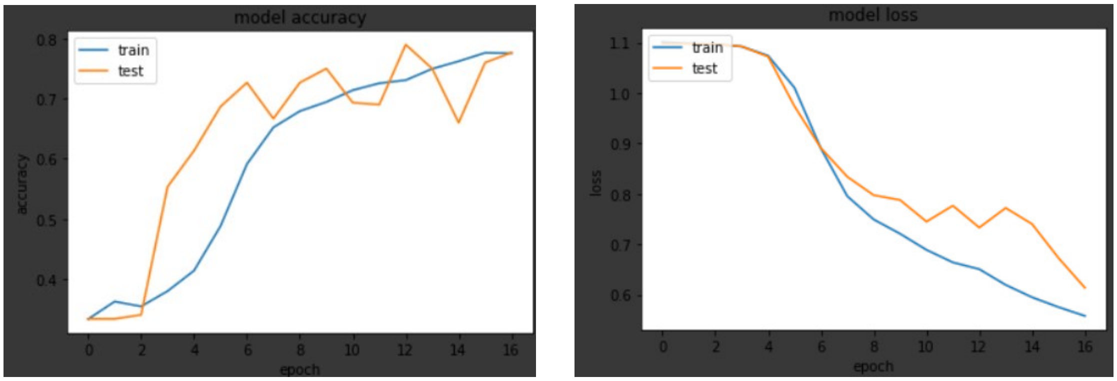
```
              precision    recall  f1-score   support

           0      0.763     0.710     0.736       100
           1      0.758     0.720     0.738       100
           2      0.804     0.900     0.849       100

    accuracy                          0.777       300
   macro avg      0.775     0.777     0.774       300
weighted avg      0.775     0.777     0.774       300
```

Figure 7: *Performance metrics and confusion matrix plot (Three-classes case)*

As it can be observed by a direct inspection of the confusion matrix, the most remarkable confusion still occurs between the two classes N and A. This result was quite predictable as the recordings classified as *Noise* are characterized by evident irregularities in the wave morphology (fig. 2), which render those ECGs easier to be recognized by the classifier. This outcome is also confirmed by the higher values of all the performance metrics relative to the third class, compared with the other two. Unlike the binary classifier, the most remarkable defect of this ensemble is represented by the almost perfect balance of FP and FN relative to the N and A classes that it generates.
The accuracy and loss graphs obtained in this case are reported in fig.8. The fundamental role of the implemented *early stopping* strategy in avoiding the model *overfitting* is in this case particularly evident. It is in fact clear that after the 6th epoch the test accuracy started to oscillate between two values, while the train accuracy kept a monotonically increasing behaviour. This means that further epochs would have for sure led the model to overfit to its training samples. However, thanks to the techniques employed, the final train and test accuracy are stabilized to the same final value.



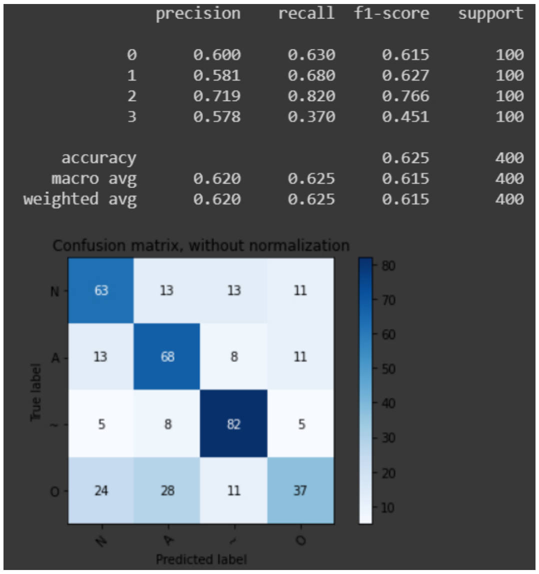
(a) *Accuracy*

(b) *Loss*

Figure 8: *Accuracy and loss graphs (Three-classes case)*

5.3 Multi-Class Case: Four classes

Finally, we have considered the full problem proposed in the challenge by adding to the training set an equal number of samples labeled as O. The introduction of this fourth class causes a remarkable drop of the overall classification performances of the network, which reaches an accuracy of about 63%.



```
              precision    recall  f1-score   support

           0      0.600     0.630     0.615       100
           1      0.581     0.680     0.627       100
           2      0.719     0.820     0.766       100
           3      0.578     0.370     0.451       100

    accuracy                          0.625       400
   macro avg      0.620     0.625     0.615       400
weighted avg      0.620     0.625     0.615       400
```

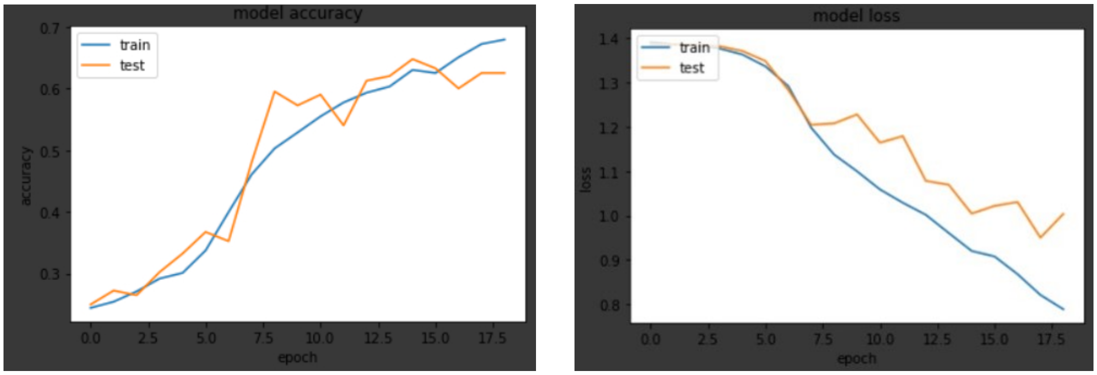
Confusion matrix, without normalization

True label \ Predicted label	N	A	~	O
N	63	13	13	11
A	13	68	8	11
~	5	8	82	5
O	24	28	11	37

Figure 9: *Performance metrics and confusion matrix plot (Four-classes case)*

In particular, if we look at the confusion matrix, we can see that network is still pretty much able to distinguish the three classes considered in the previous section, while the images associated to the **Other Rithms** ECGs tend to be more frequently confused with the two other classes N and A. This can be also seen by comparing the performance metrics related to the fourth class with those of the other three, which are all characterized by a remarkably higher f1-score. This result was quite predictable as many samples labeled as O tend to be morphologically similar to those belonging to the class N (fig.2). This

geometric feature similarity (which is reflected into the similarity of the corresponding spectrograms) partially explains the confusion between the samples of these two classes. Moreover, those samples labeled as O are likely to include different kinds of arrhythmias, which are not clearly specified in the challenge. This aspect introduces an additional difficulty at the feature extraction phase since, in this case, there are no precise guidelines to correlate the ECGs' geometries with their corresponding class. On the other hand, the samples belonging to the third class still remain the most-accurately classified, as confirmed by both an higher precision and recall.
By looking at the Accuracy and Loss Function curves we have, for both of them, a behaviour which is qualitatively similar to the training one but, differently from the previous case (Three-classes classification), the resulting model is slightly affected by Overfitting. However, the phenomenon is not very much accentuated, thus leading to the conclusion that the usage of anti-overfitting techniques such as DropOut, Early Stopping and Data Augmentation have been fundamental in this last case as well.



(a) *Accuracy*

(b) *Loss*

Figure 10: *Accuracy and loss graphs (Four-classes case)*

6 Conclusions

In this project we developed a Convolutional Neural Network for the classification of ECGs affected by different kinds of arrhythmias and, in particular, for the AF recognition. A series of pre-processing and transformation operations were preliminarily carried out on the signals so as to render the original data more suitable to the classification phase.
Although the proposed network is not particularly complicated, we managed to obtain quite satisfactory classification performances when considering the binary classification problem with the two classes N and A (83% of accuracy) and the three-classes classification problem, where the class of noisy signals was as well considered (78% of accuracy). However, performances tend to get markedly worse when the class O is added too, as confirmed by an accuracy drop to 63%.
In order to obtain better results in the four-classes classification problem, we would either need a much higher computational power (i.e. a much deeper network [3]) or more precise guidelines in the feature extraction procedure, especially for the samples labeled as *Other* (O).

References

[1] G. Clifford et Al. *AF classification from a short single lead ECG recording: The Physionet/Computing in Cardiology Challenge 2017,* Computing in Cardiology, 2017.

[2] F. N. Hatamian et Al. *The Effect of Data Augmentation on Classification of Atrial Fibrillation in Short Single-Lead ECG Signals Using Deep Neural Networks,* arXiv, 2020.

[3] M. Zihilmann et Al. *Convolutional recurrent neural networks for electrocardiogram classification,* IEEE, 2017, pp. 1–4..