ECG SIGNALS CLASSIFICATION

Luca Galli
905236

Enrico Mannarino
850859

Christian Persico
829558

TABLE OF CONTENTS

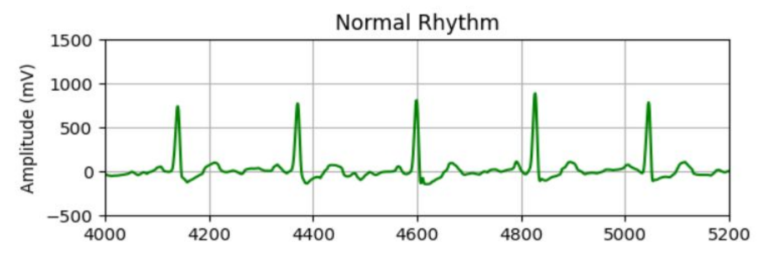
01.

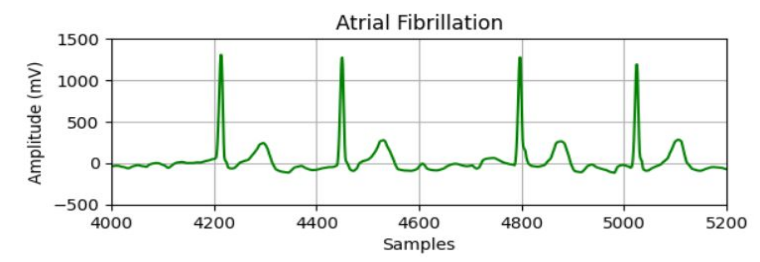
GOALS AND PRESENTATION OF THE DATASET

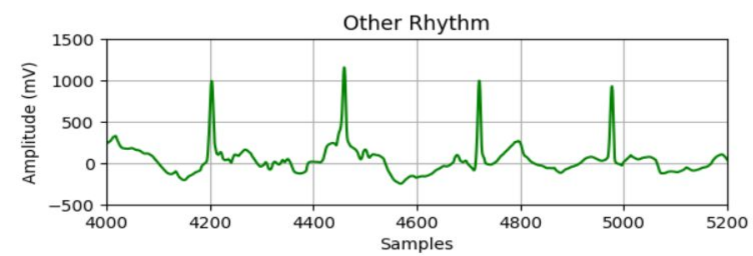
Dataset

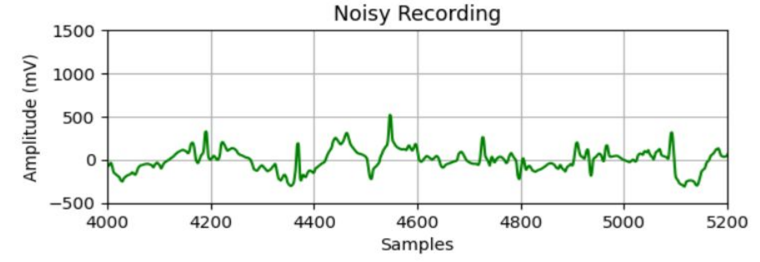
The **PhysioNet 2017** dataset consists of 8528 electrocardiogram (ECG) recordings, collected using the AliveCor device, sampled at 300 Hz and divided by a group of experts into four different classes (all data are provided in MATLAB):

Normal Rhythm (N), Atrial Fibrillation (A), Other Rhythm (O), Noisy Recording (~)





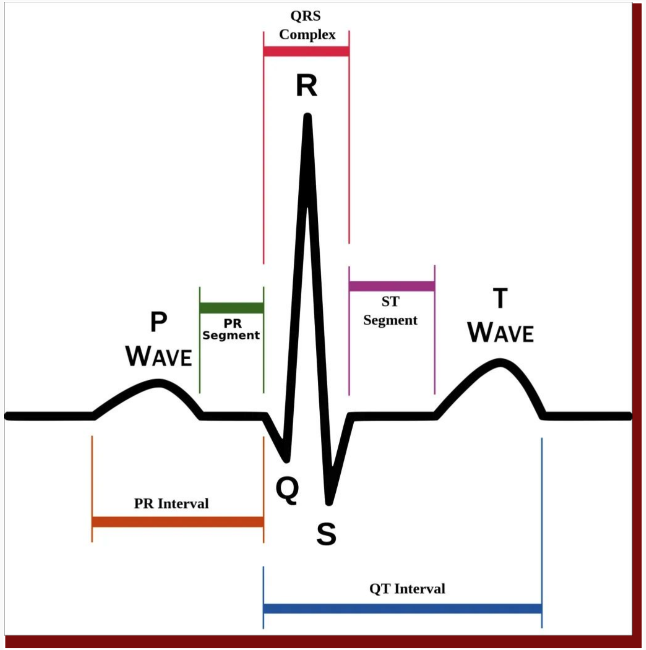




Goals

The aim of the project is to build a neural network that is able to classify ECGs to their respective class with a good degree of accuracy.

In particular, atrial fibrillation is a type of irregular heartbeat that occurs when the heart's upper chambers, the atria, beat out of coordination with the lower chambers, the ventricles.
It is the most common sustained cardiac arrhythmia, occurring in 1-2% of the general population and is associated with significant mortality and morbidity through association of risk of death, stroke, hospitalization, heart failure and coronary artery disease, etc.



Source: https://a-fib.com/

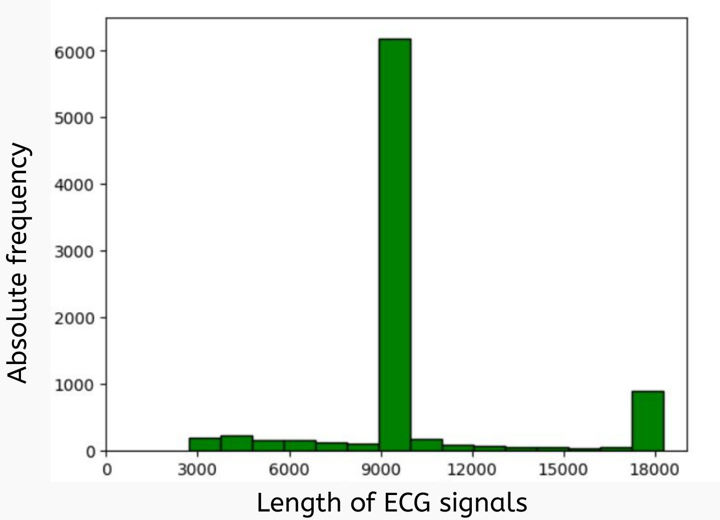
02.

DATA EXPLORATION AND PRE-PROCESSING

The observations in the dataset vary in signal length from 2714 to 18286 samples (approximately 9 to 60 seconds).

SOLUTION

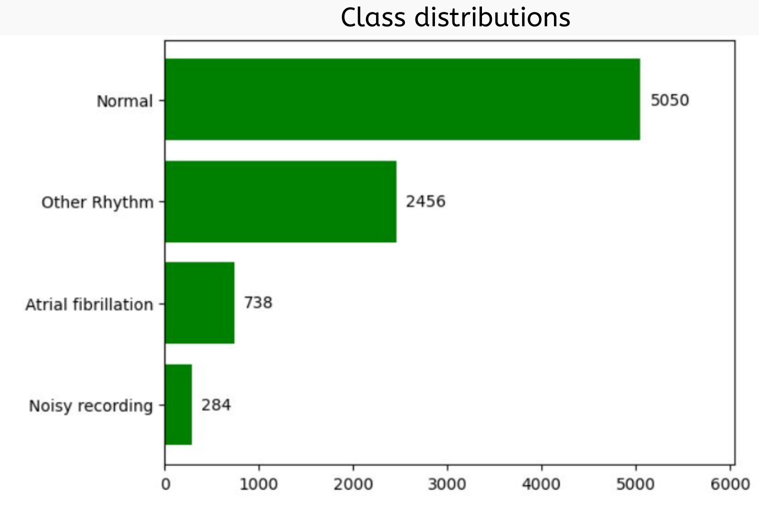
Signal truncation by multiples of 9000 values.



Imbalanced classes

SOLUTION

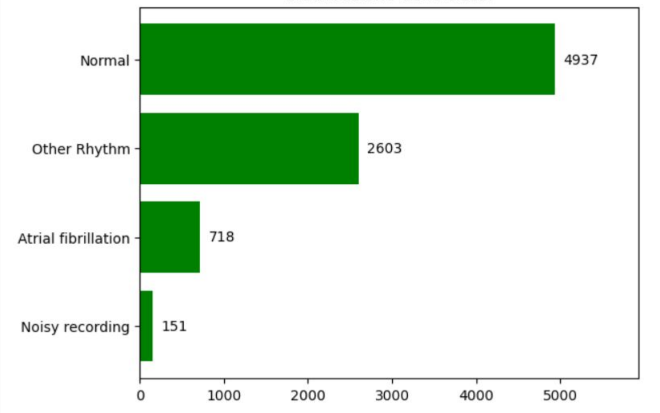
- Assigning different weights for each class
- Data augmentation



Signals selection and division

	Discarded (< 9000)	Splitted (≥ 18000)	Total
Normal Rhythm	521	408	**- 113**
Atrial Fibrillation	113	93	**- 20**
Other Rhythm	194	341	**+ 147**
Noisy Recording	139	6	**- 133**
	- 967	**+ 848**	**- 119**

- New total of observations: **8409** (from 8528).
- New class distribution:



Descriptive statistics

	Overall average	Mean standard deviation	Max	Min
Normal Rhythm	7.85	199.37	8318	- 10636
Atrial Fibrillation	7.72	183.25	6342	-6787
Other Rhythm	6.89	196.31	8257	-7655
Noisy Recording	1.66	397.23	7309	-6646

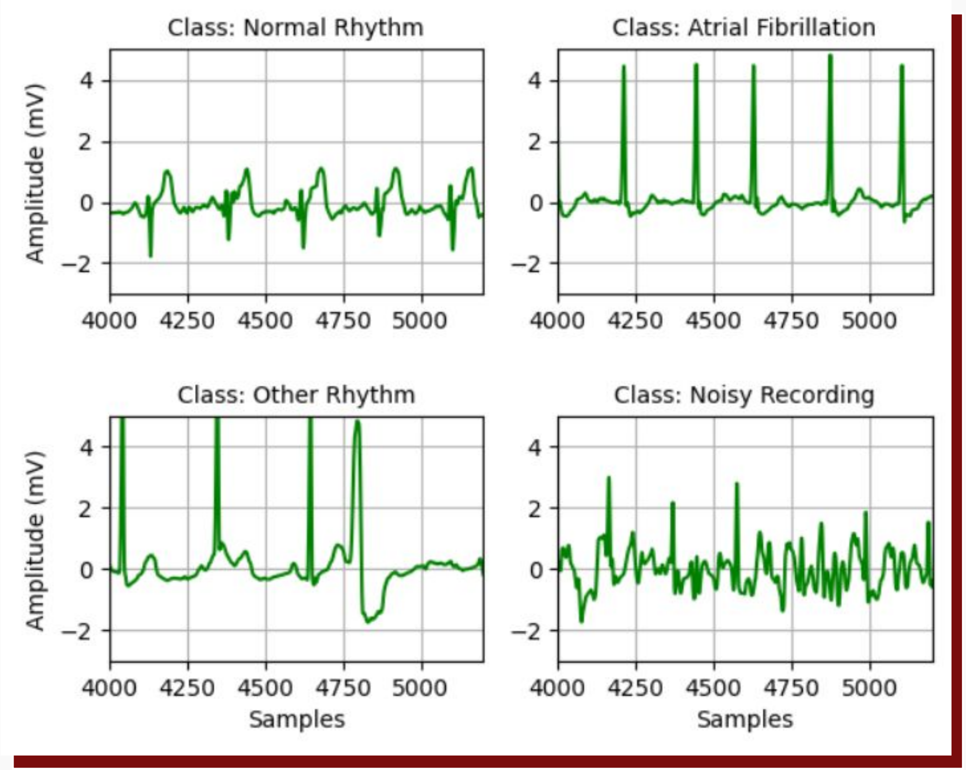
Dataset division

The dataset was divided into three parts, with stratified sampling by class:

- **60%** training set (5045 obs.)
- **20%** validation set (1682 obs.)
- **20%** test set (1682 obs.)

+

- **Standardization**
- **One-hot encoding**



Standardised samples

03.

DEEP LEARNING MODELS

Hyperparameters

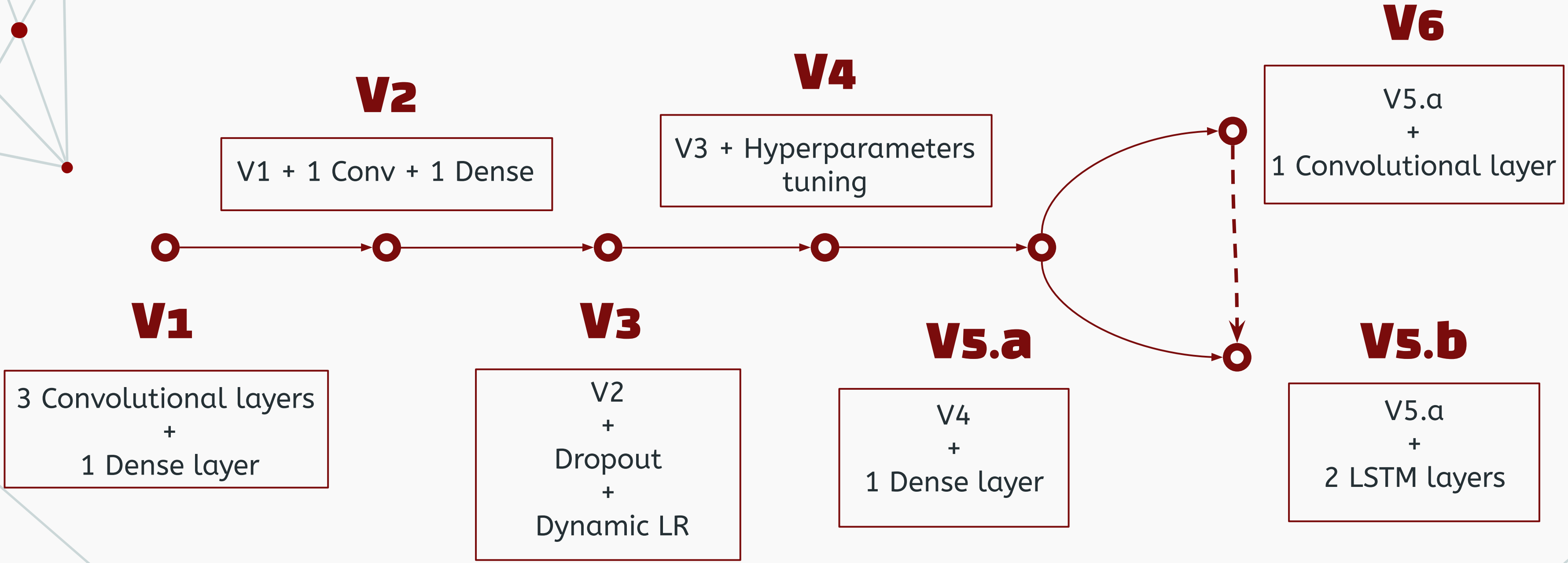
- **EPOCHS**: 50
- **LOSS**: Categorical cross-entropy
- **ACTIVATION FUNCTIONS:** ReLU and Softmax
- **OPTIMIZER**: Adam
- **LEARNING RATE**: from 0.001
- **CALLBACKS**
  - **EARLY STOPPING:** patience of 10 on validation loss value
  - **REDUCE LR ON PLATEAU:** patience of 5 on validation loss value

WEIGHTED LOSS APPROACH:

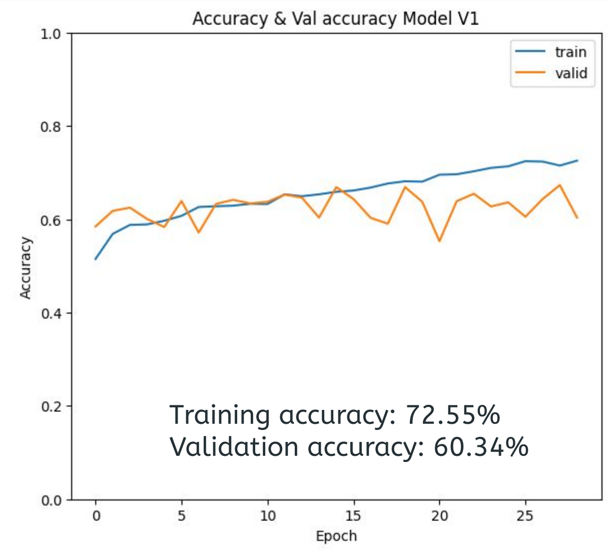
Assign different weights for each class in the classification process as follows:

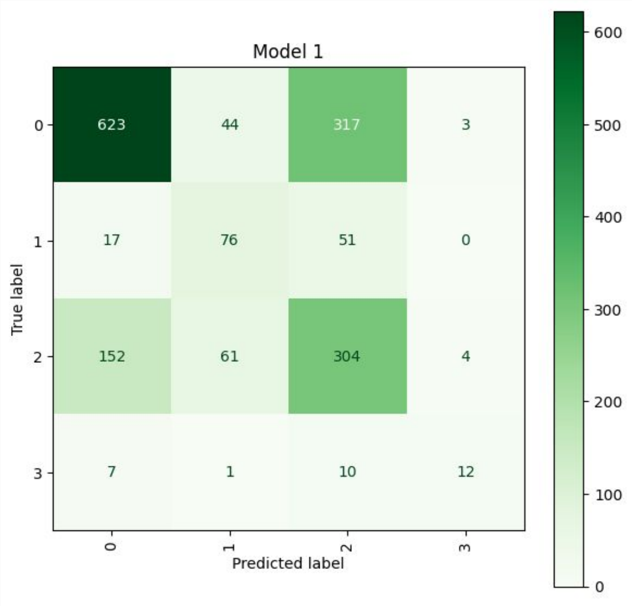
$$1 - \frac{\text{number of samples present}}{\text{total number of samples}}$$

Timeline


V1
3 Convolutional layers
+
1 Dense layer
V2
V1 + 1 Conv + 1 Dense
V3
V2
+
Dropout
+
Dynamic LR
V4
V3 + Hyperparameters tuning
V5.a
V4
+
1 Dense layer
V5.b
V5.a
+
2 LSTM layers
V6
V5.a
+
1 Convolutional layer

Model V1





OBSERVATIONS

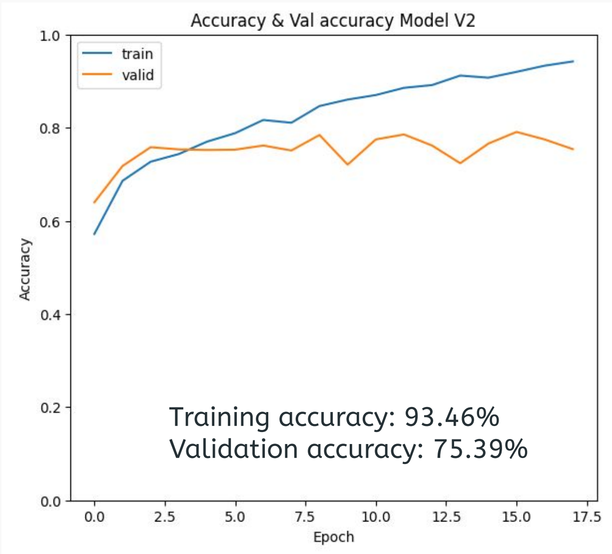
- Accuracy is low
- Model is not complex enough

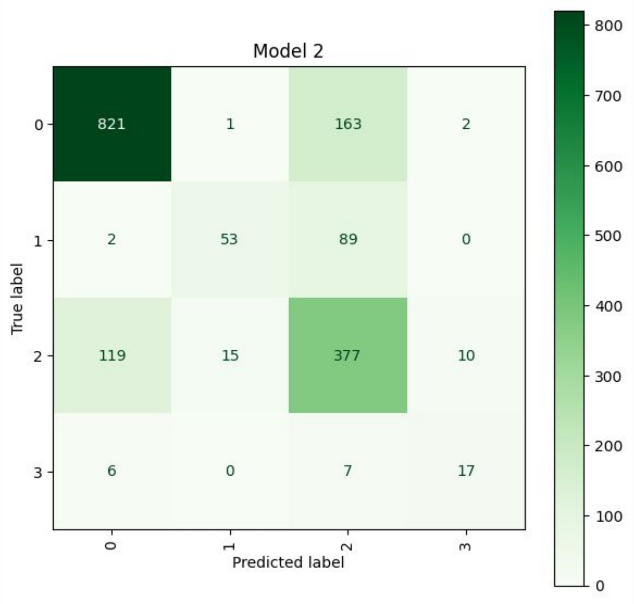
SOLUTION

- Let's add a Convolutional layer, a Dense layer and set dense layers' kernel initializer to normal

Total params: **39 556**

Model V2





OBSERVATIONS

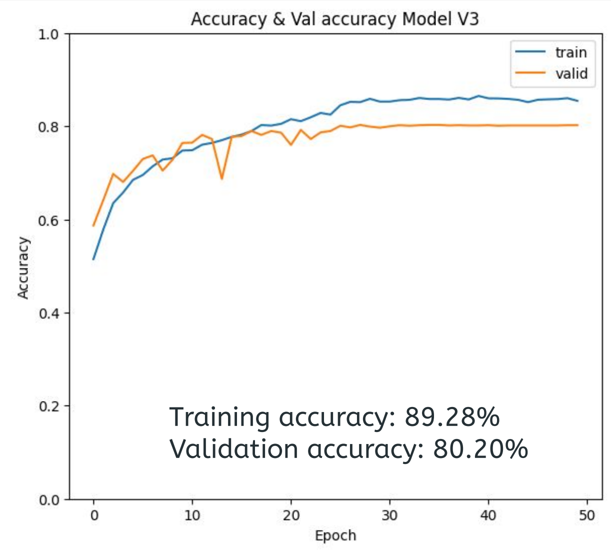
- Overfitting
- Accuracy fluctuations

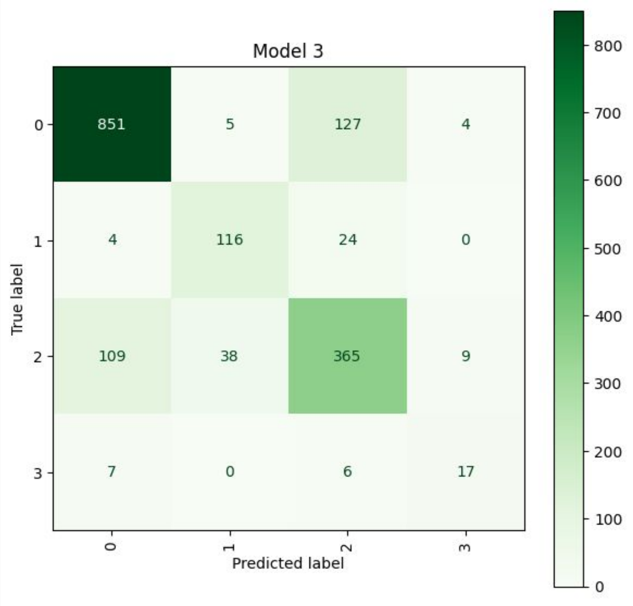
SOLUTION

- Let’s add Dropout layers
- Introduce dynamic LR

Total params: **66 756**

Model V3





OBSERVATIONS

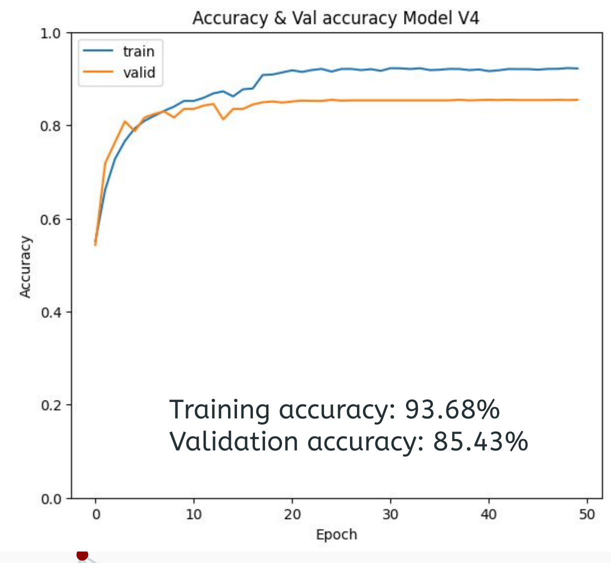
- Good performance

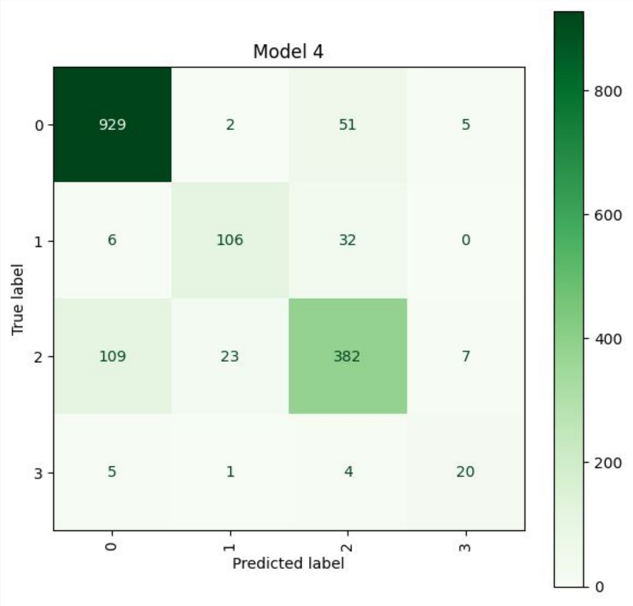
SOLUTION

- Let’s try changing filters and kernel size to improve performance

Total params: **66 756**

Model V4





OBSERVATIONS

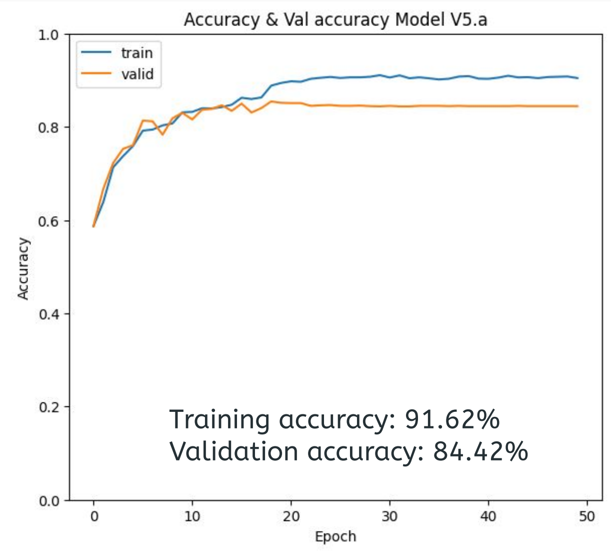
- Better performance

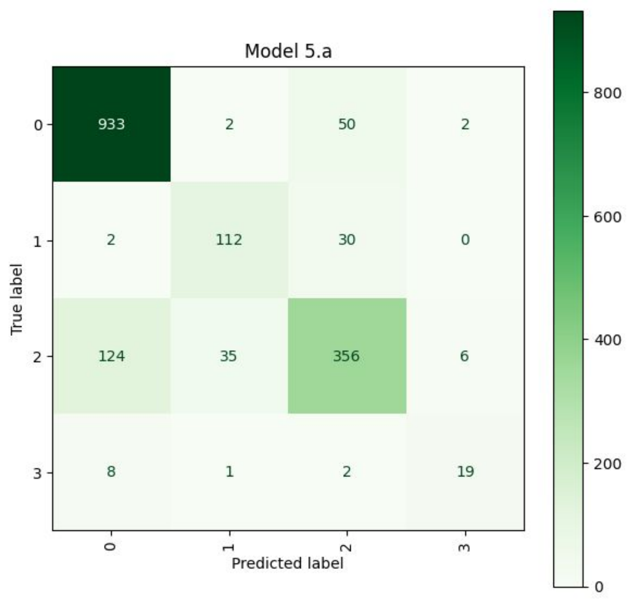
SOLUTION

- Let's try increasing complexity by adding a Dense Layer

Total params: **593 028**

Model V5.a





OBSERVATIONS

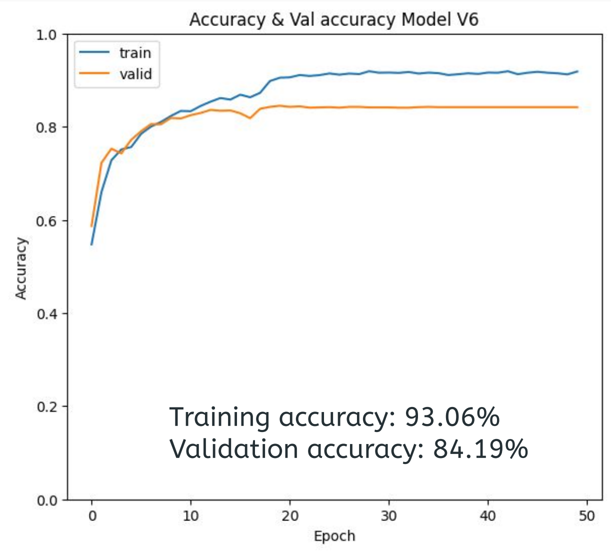
- Slightly worse performance than V4

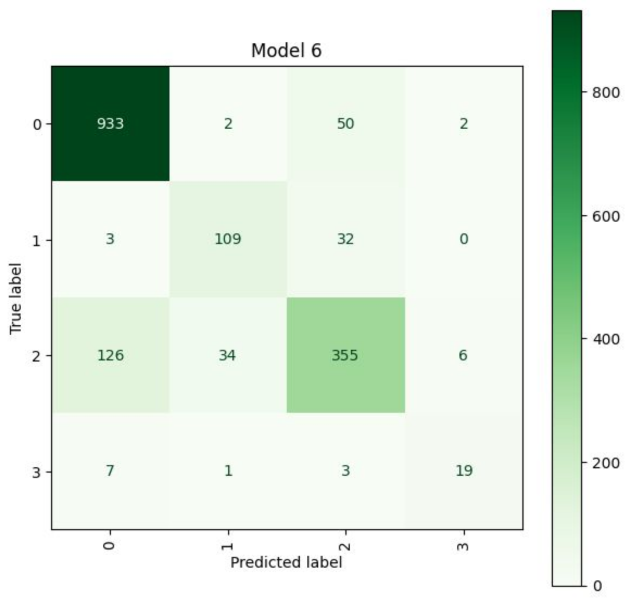
SOLUTION

- Let's try increasing complexity by adding a Convolutional layer

Total params: **606 228**

Model V6





OBSERVATIONS

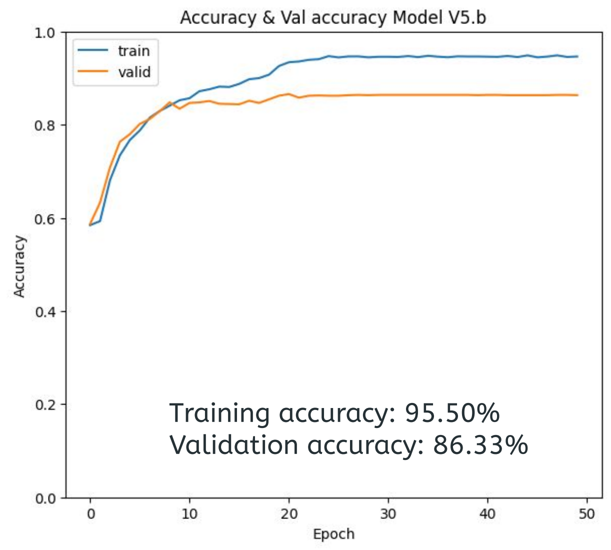
- Slightly worse performance than V4

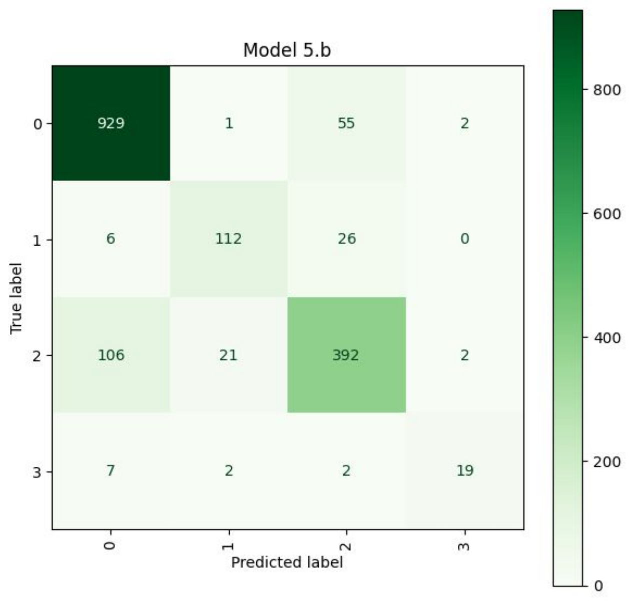
SOLUTION

- Let’s try adding two LSTM layers to the V5.a to find a better solution

Total params: **886 164**

Model V5.b





OBSERVATIONS

- Best performance overall

Total params: **844 164**

Metrics of the models

	Loss	Accuracy	AUC	Precision	Recall
V1	0.8613	0.6034	0.8654	0.6343	0.5589
V2	0.9900	0.7539	0.9215	0.7551	0.7533
V3	0.5266	0.8020	0.9509	0.8073	0.7943
V4	0.4906	0.8543	0.9624	0.8571	0.8484
V5.a	0.4892	0.8442	0.9638	0.8500	0.8424
V6	0.5415	0.8419	0.9583	0.8466	0.8365
V5.b	0.4497	0.8633	0.9648	0.8646	0.8615

04.

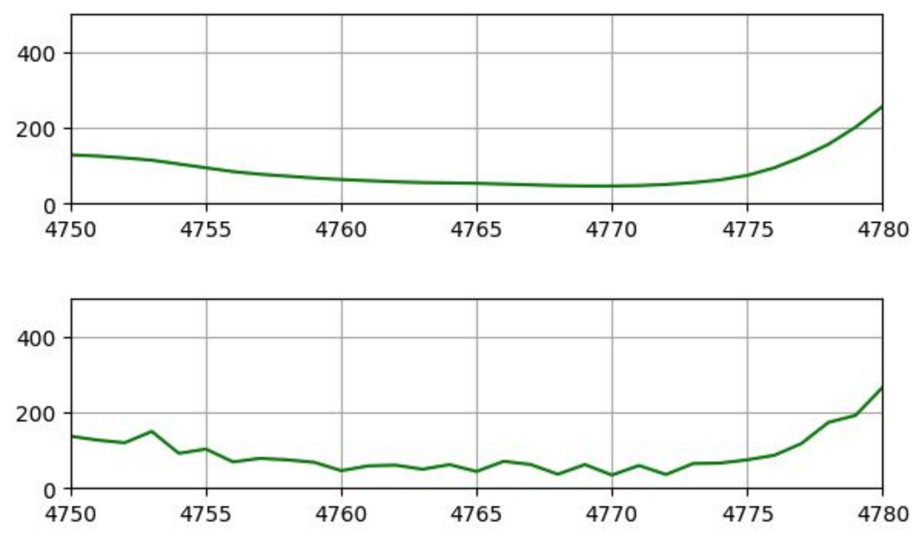
DATA AUGMENTATION AND RESULTS

Data augmentation

The four classes have very unbalanced values between them, respectively numbering 2962, 431, 1562 and 90 elements in the **training set**, so we want to balance them out by bringing them all to 2962 components (numerosity of the majority class).

To accomplish this process, a technique called **Jittering** is used, which involves adding a small random constant distributed as a standard Normal distribution to duplicate signals until class balancing is achieved.

+ Standardization



Example of zoomed jittered signal with sd = 10 to make the transformation more obvious

CLASSES	Non-augmented class sizes	Augmented class sizes
Class 0	2962	2962
Class 1	431	2962
Class 2	1562	2962
Class 3	90	2962
Total	5045	11848

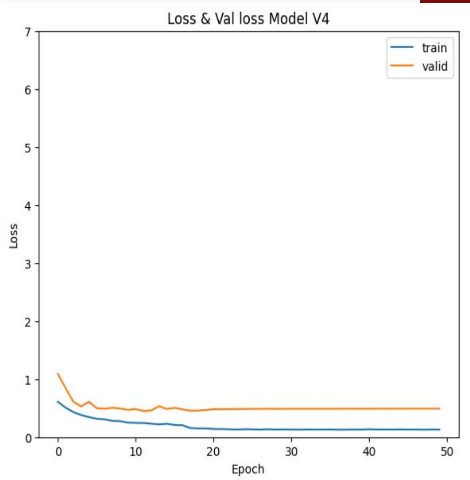
Metrics with class weights

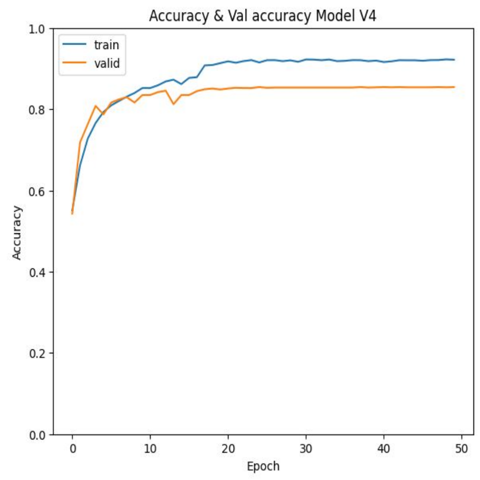
	Loss	Accuracy	AUC	Precision	Recall
V4	0.4906	0.8543	0.9624	0.8571	0.8484
V5.a	0.4892	0.8442	0.9638	0.8500	0.8424
V5.b	0.4497	0.8633	0.9648	0.8646	0.8615

Metrics on augmented data

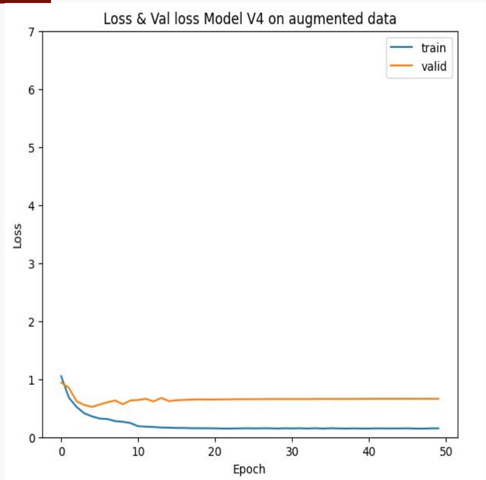
	Loss	Accuracy	AUC	Precision	Recall
V4	0.6607	0.8466	0.9525	0.8479	0.8454
V5.a	0.6479	0.8436	0.9522	0.8451	0.8430
V5.b	0.5991	0.8347	0.9478	0.8376	0.8341

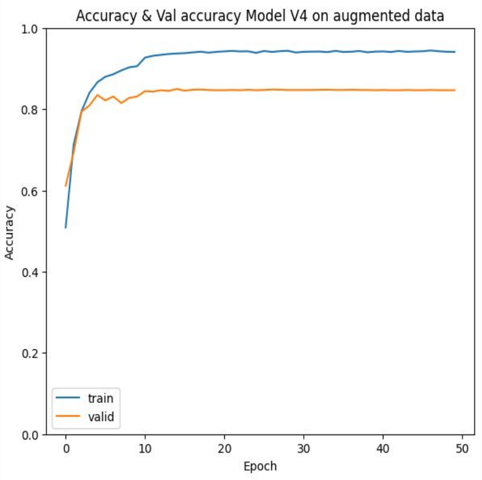
V4





Class weights naive method

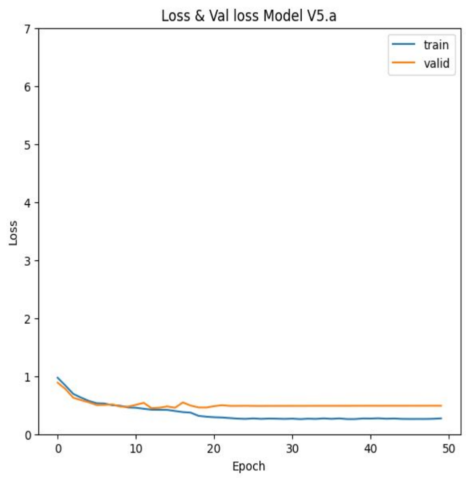


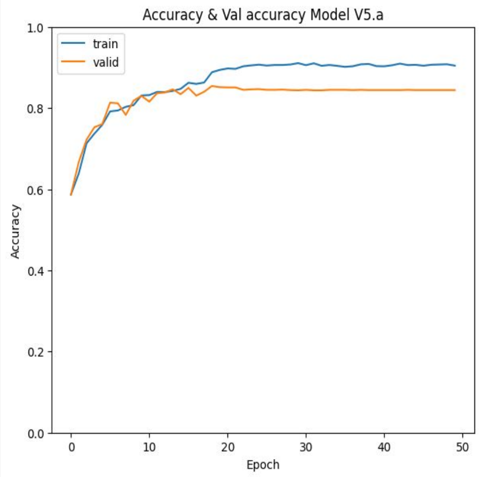


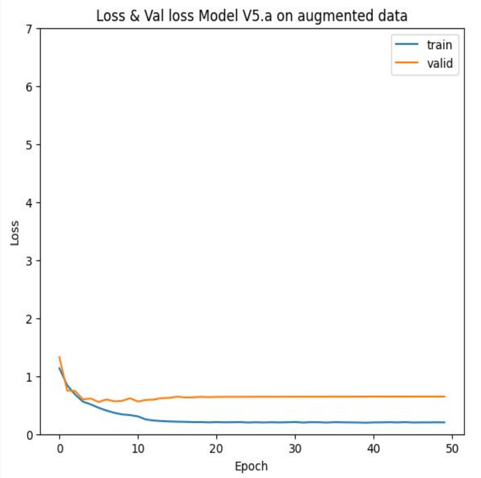
Augmented data with Jittering

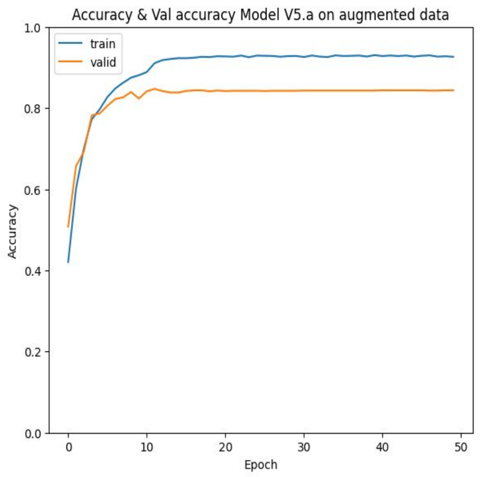
V5.a

Class weights naive method





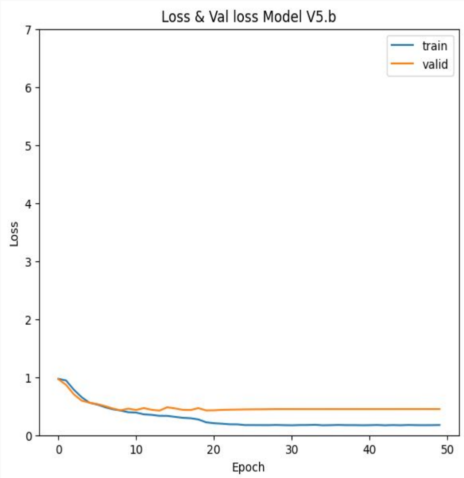


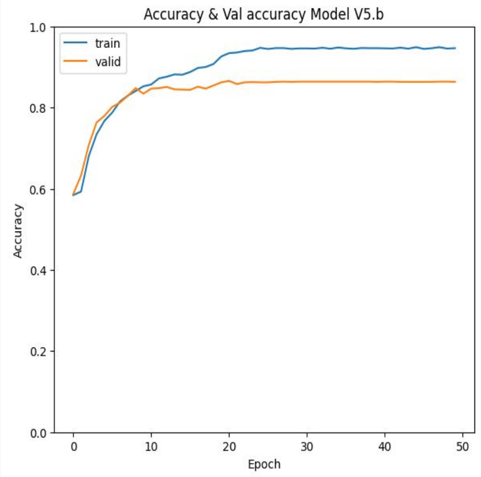


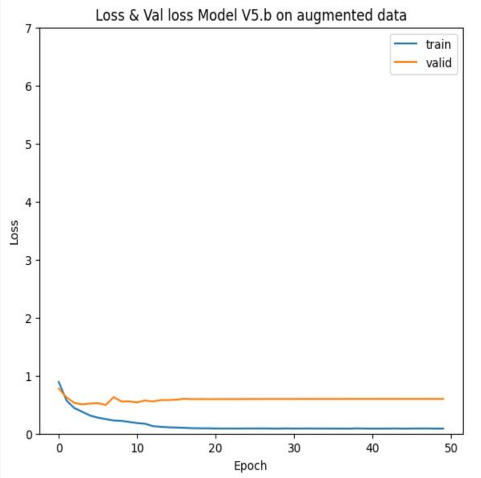
Augmented data with Jittering

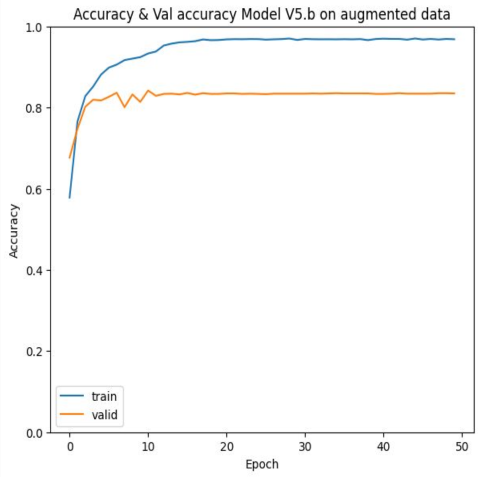
V5.b

Class weights naive method



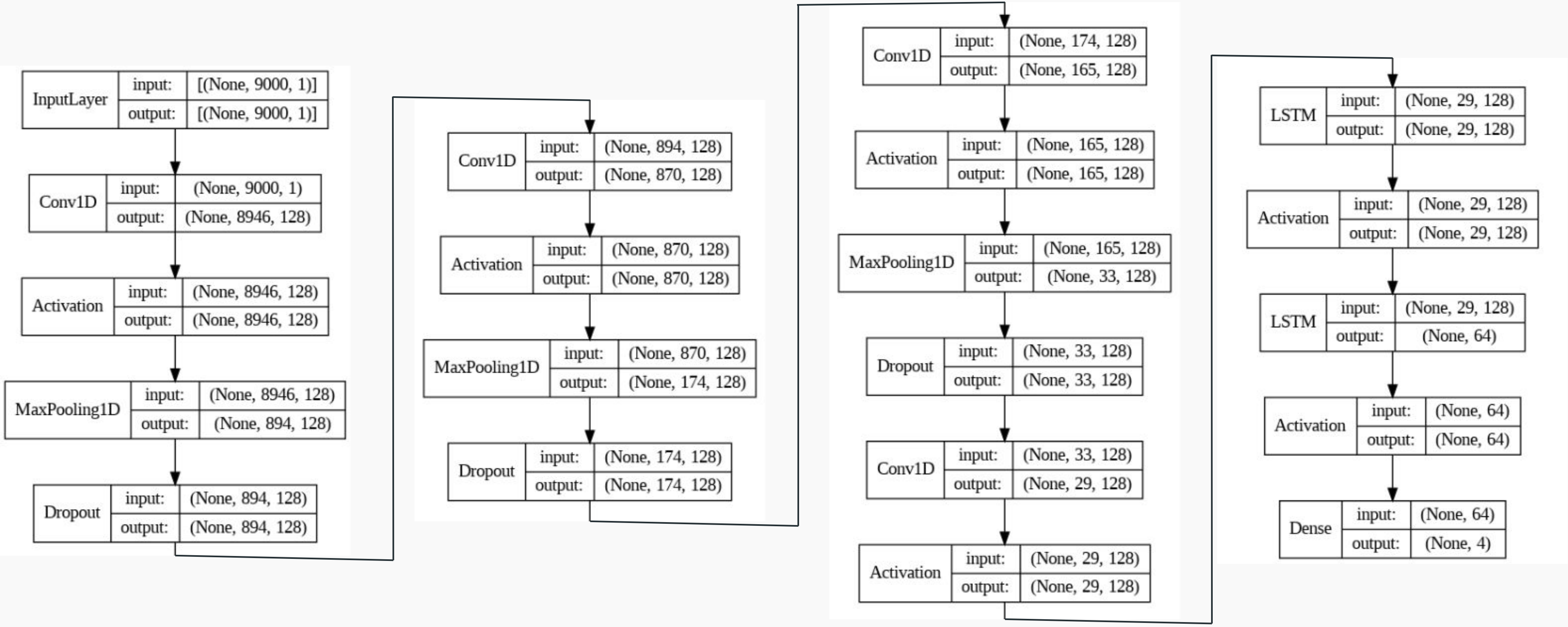






Augmented data with Jittering

Architecture of Model V5.b

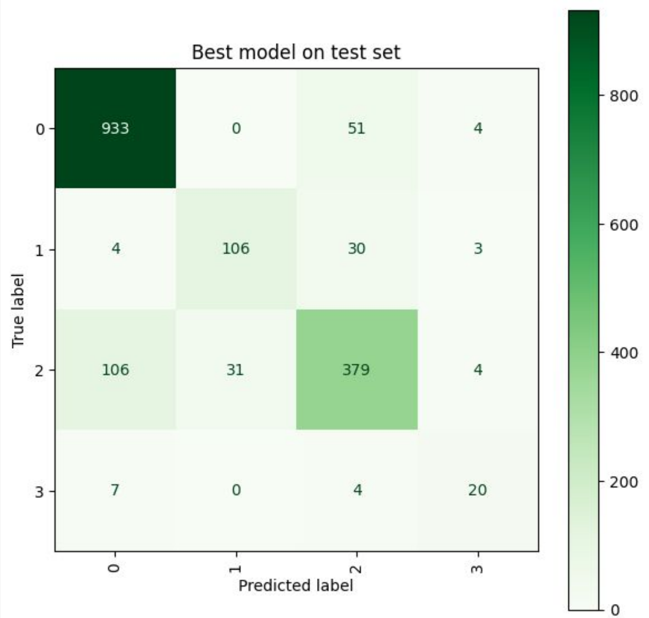


05.

EVALUATION ON TEST SET

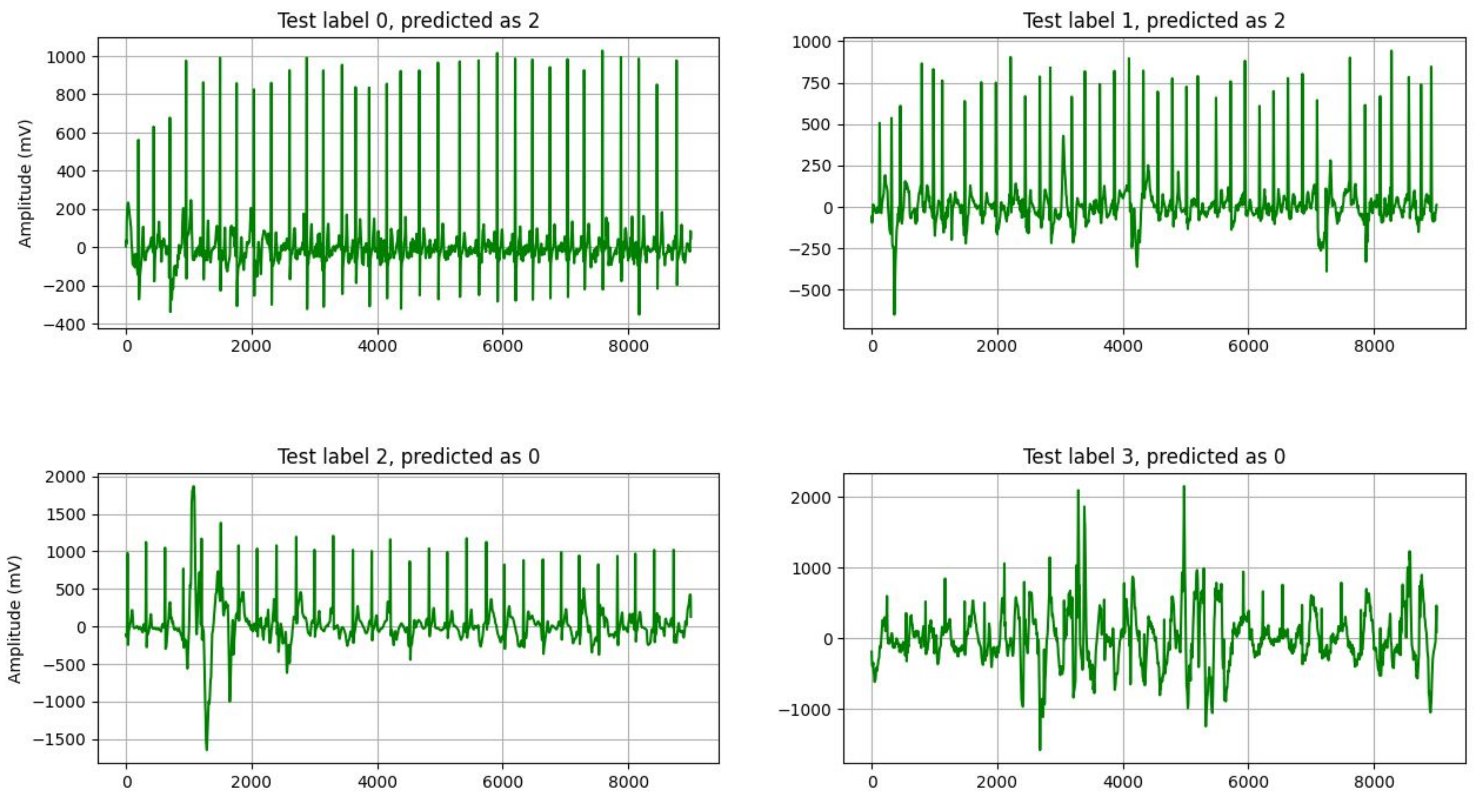
Metrics on test set

	Precision	Recall	F1-score	Support
Normal Rhythm	0.89	0.94	0.92	988
Atrial Fibrillation	0.77	0.74	0.76	143
Other Rhythm	0.82	0.73	0.77	520
Noisy Recording	0.65	0.65	0.65	31



Accuracy	Loss	AUC	Precision	Recall
0.8549	0.4685	0.9626	0.8603	0.8532

Examples of misclassified samples



06.

FUTURE DEVELOPMENTS

As future improvements could be considered:

- Alternative way of managing signal lengths

- Implementation of further data augmentation methods for the type of data at our disposal (e.g. **GAN**).

- Implementation of signal handling and **transformation methods** (Short-Time Fourier Transform/Wavelet Transform and so on).

- **Transfer learning** of already proven models with fine-tuning of hyperparameters → e.g. MobileNet/ResNet (involve transformation of signals into images) or specialised models for ECG classification (e.g. neural network trained on the Icentia11k dataset).

REFERENCES

- https://it.mathworks.com/help/deeplearning/ug/classify-ecg-signals-using-long-short-term-memory-networks.html
- https://physionet.org/content/challenge-2017/1.0.0/
- https://keras.io/examples/timeseries/eeg_signal_classification/#prepare-tfdatadataset
- http://103.82.172.44:8080/xmlui/bitstream/handle/123456789/614/Thesis%20Book%20154404_154407.pdf?sequence=1&isAllowed=y
- https://arxiv.org/pdf/1706.00527.pdf
- https://arxiv.org/pdf/2206.13508.pdf
- https://www.tensorflow.org/tutorials/structured_data/imbalanced_data
- https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8759878

THANKS

l.galli40@campus.unimib.it

e.mannarino1@campus.unimib.it

c.persico4@campus.unimib.it