POLITECNICO
MILANO 1863

Heartbeat classifier for ECG signals

Artificial Intelligence in Biomedicine

Sebastian Edoardo Gerletti
sebastianedoardo.gerletti@mail.polimi.it

Karim Kassem
karim.kassem@mail.polimi.it

Giuseppe Venezia
giuseppe.venezia@mail.polimi.it

I. Abstract

AI applied to heartbeat classification based on ECG signals is an important tool to help doctors identify various types of cardiac arrhythmias. These mainly affect ECG's morphology and/or rhythm which are not always appreciateable from single lead recordings. The data-set used here regards 2 lead ECG signals sampled at 128Hz (70 patients) and 250Hz (35 patients). The goal is to classify normal heartbeats, ones affected by premature atrial contraction (PAC) or premature ventricular contraction (PVC).
Machine learning (ML) and deep learning techniques were utilized both to combine their strengths for the implementation of efficient and generalizable classifiers. To achieve this, respectively handcrafted features and data-driven ones are used, exploring the potentialities of a priori domain knowledge and deep learning.
As regards the latter, best results were obtained transforming heartbeat signals with short-time-Fourier-transform (STFT) and feeding them to a convolutional neural network (CNN), doing so normal heartbeats (N) were distinguished from pathological ones (D) with 91.0% harmonic trace and 86.5% f1 score while PAC (S) vs. PVC (V) classification reached 75.2% harmonic trace and 81.1% f1 score; thus obtaining for the multi-class classification, by serializing the 2 binary ones, 77.4% harmonic trace and 85.2% f1 score. The harmonic trace is a custom metric to deal with multi-class imbalanced problem where biased predictions (towards the majority class) are strongly penalized. As regards the wide features, these were combined with the ECG signal in a hybrid model obtaining 80.7% harmonic trace and 86.0% f1 score; and also exploited alone using KNN (75.9% harmonic trace and 86.8% f1 score). To improve prediction robustness an ensemble between these models was implemented by linearly combining their predictions obtaining 80.7% harmonic trace and 86.0% f1 score.

II. Introduction

The aim of this project is to develop a beat classifier which is able to discriminate between a normal heartbeat, a PAC (Premature Atrial Complex) also called Supraventricular Beat and a PVC (Premature Ventricular Complex) or Ventricular beat. A tool with these capabilities is particularly useful considering the consequences the two anomalies can have in a human-being's health, like Atrial Fibrillation, stroke and myocardium degradation in the case of PAC and left ventricular (LV) dysfunction for PVC as a long-term result. Designing an algorithm which detects whether a patient presents or not either of these conditions for any heartbeat is key for early diagnosis and preventive treatment to further complications; considering also the challenge that manual classification of each heartbeat in ECG recordings would represent for specialized practitioners, effort- and time-wise.
Heart arrhythmias affect ECG signals either as regards rhythm, morphology or both, in this case both PAC and PVC affect rhythm while the former only slightly distorts the P wave and the latter evidently alters the QRS complex shape. Heartbeat-wise classification is done because the most clinically relevant parameter to define PAC and PVC conditions in patients is the burden of each, defined as the percentage of non-physiological heartbeats over the total number of beats during 24 hours.

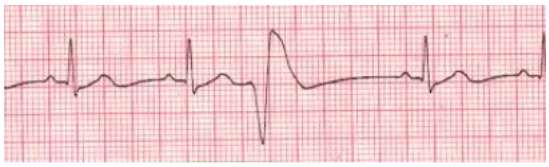

Fig. 1. PVC

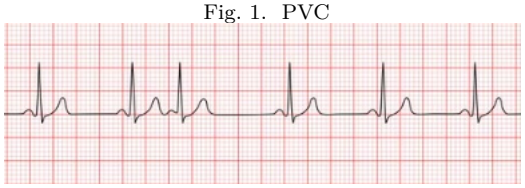

Fig. 2. PAC

III. Materials and methods

A. Materials

The dataset contains 3 files for each of the 105 patients: 2-leads ECG recordings, positions of the R-peaks of the signals and labels corresponding to each R-peak. Signals were sampled either at 128Hz (70 patients) or 250Hz (35 patients) for 30minutes per patient.
Labels for the supervised classification task are N for normal beat, S for PAC and V for PVC. Google Colaboratory was used to exploit the parallel computation capabilities of GPUs, mainly required for deep learning.
Main Python libraries used are: NeuroKit2, an advanced biosignals processing library, HeartPy, SciPy, Numpy, Pandas, MatplotLib and TensorFlow, for building models.

B. Methods

1) Data Loading: Patient specific information was condensed in a list of lists, after importing all .rpk (R-peak positions), .ann (annotations) and .mat (2 lead ECG) files.

2) Data Preparation:

- ***Data visualization and inspection.*** The first step after having loaded the data, was its inspection, starting from signal visualization in order to decide which pre-processing is required. Signals as a whole, single heartbeats and summary statistics were analysed.
  During this phase, 2 signals were removed since corrupted by noise or acquisition artifacts which could not be corrected by filtering, neither custom nor using ecg_clean of Neurokit2.
  As regards inter- R-peak distance, incongruities were encountered both as regards too big distances, that might be due to electrodes detachment, acquisition errors or missed R-peak detection and too close peaks, given by artifacts, noise or wrongful detection.
- ***Train-val split.*** After the initial data cleansing the training dataset was split into training set and validation set, maintaining label-wise stratification: this division has been executed before heartbeat extraction since it is good practice to avoid different instances of the same patient both in the training and in the validation sets. This causes the desired split to be an np-hard task since all possible splits for 105 patients with the given proportions must be evaluated considering for each one the sums of N, S, and V heartbeats per patient.
  This problem was solved using genetic algorithms better explained in appendix[A]. 80 patients were thus considered for the training set and 23 for the validation while testing evaluation will be performed on a new dataset of (hypothesised) same distribution.

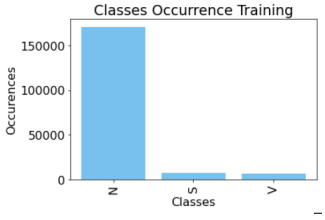

Fig. 3. Training label distribution.

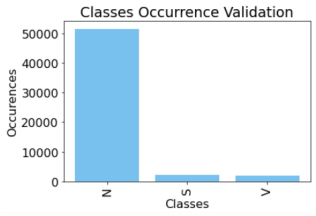

Fig. 4. Validation label distribution.

For each of these two sets, the recordings have been first split between 128Hz recordings and 250 Hz recordings, leading to 4 different subsets and consequently divided in the two individual leads. The second division would have been lately helpful in order to up-sample or down-sample the recordings, to homogenize the sampling frequency of the dataset.

3) Recordings pre-processing:

- ***Filtering.*** Two options were explored and evaluated to clean the ECG signals. On one hand we tried to use the ecg_clean function from the NeuroKit2 library, which removes the baseline shift (high pass filtering) and also reduces the ripple of the signal (low pass filtering). On the other hand we tried to implement a custom band pass filter with cut-off frequencies at 0.05Hz and 40Hz; no notch filter at powerline frequency was used since the Fourier transform of the signals did not show any peak at 50Hz, or any other frequency in the passing band.
  We visualized again entire and windowed signals to whom had been applied the two options, in order to notice how the signal changed. The former solution was chosen after a qualitative analysis of the signals, mainly checking baseline shift and smoothness.
- ***Rescaling.*** Standardization was applied due to the presence of high peaks in the signals, which make it therefore inadequate for normalization, since this would squeeze the smaller values. Mean and the standard deviations were computed for each lead and for each patient of the training set; then each was averaged for patients and leads and respectively subtracted and divided both for training and validation sets.
- ***Heartbeat division.*** Several heartbeat division techniques have been tested before selecting the one we found to be more efficient. The first attempt was with ecg_segment, a function from the NeuroKit2 library which divides the recordings having signals and r_peaks positions as input. However this method chooses the window size based on an iterative estimation of the heart rate in order to include one period per window; this was not adequate for the task at had both for the requirement of having constant input size to the neural networks and for including in the

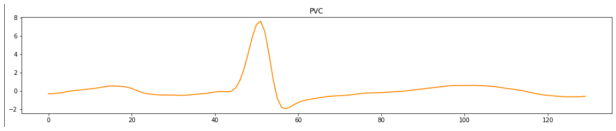

Fig. 5. PVC smaller window

window not only morphological information for each single heartbeat but also rhythm variations.
For these reasons a custom method was implemented: for each sampling frequency the maximum distance among R-peaks was computed and windows were obtained by taking such a number of samples before and after each R-peak. For the head and tail of each patient's signals, end-value-padding was performed.
Considering 128Hz as sampling frequency, the window size obtained was of 476 samples (3.7s) symmetrically centred around the R peak of each heartbeat to classify, this is a much wider field of view than what is usually used in literature since it guarantees the presence of 2 heartbeats around the central one even for low heart rates (conservative approximation) but this provided too much task-irrelevant information which only confused the models. Following literature's indications a much smaller window was used, this time asymmetrical (80 samples before R peaks' positions and 50 after: 1s in total) which is a conservative estimate to obtain 1 whole heartbeat per window, by doing so the network might learn to focus on the morphology better but no information about rhythm is available; this approach therefore also gave poor performances but for the opposite reason: not enough informativeness, as regards deep neural networks. While for the extraction of wavelet coefficients, used later on for ML model this proved optimal since more heartbeat, and not beats, specific, since broader information was then provided by the R-peak distances. The best tradeoff, used then in the deep models, was found for a medium sized window, obtained by analysing the point of convergence for maximum inter-R peak distances among signals sampled at 128Hz: this means disregarding the really high distances (outliers) probably due to miss reading or missed identification; this medium window is centred on the R peak and consist of a total of 264 samples (2s), the idea is for each window to contain also the previous and the next peak with respect to the one in analysis so as to provide timing information while remaining as close up as possible.

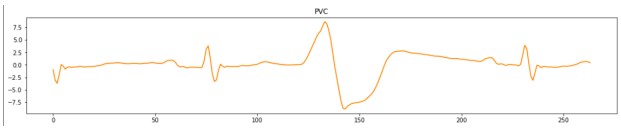

Fig. 6. PVC bigger window

- ***Resampling.*** Once the dataset with heartbeats as samples was obtained, upsampling for the 128 Hz signals and downsampling for the 250 Hz signals was performed using Neurokit2's function, signal_resample, in order to have 2 uniform dataset for all signals. The down/up-sampling method was chosen by qualitatively comparing signals originally sampled at each frequency with the obtained ones. Up-sampled data was used to begin with but this proved to be too cumbersome for Colab's RAM to be uploaded all at once, therefore a data generator for the training set was used, even so RAM got saturated after each model requiring to restart the session every time, other than the fact that training is slower when having to import a batch at a time. Using the down sampled data was considered for these reasons, this decision was taken considering that although less informative in itself this sampling rate has the same information content as higher ones in our case since the frequency content of our signals (remember the low pass filter) is up to 40Hz so any sampling frequency greater than double of 40Hz is appropriate considering Nyquist theorem(from here on all considerations refer to 128Hz sampling frequency).

4) Feature Extraction: In parallel, wide features were extracted from the heartbeats, after filtering but prior to normalization. Contrarily to data-driven feature extraction, a priori domain knowledge was used to find features within whose space the classifier might separate the classes. Rhythm information is provided by R-peaks positions differences,computed between the beat to be classified and the precedent one and with the next one. These are 2 of the most relevant features for rhythm affecting arrhythmias according to literature (Article). As for morphological parameters, we relied on Kurtosis and Skewness, respectively 3rd and 4th order statistics, since PVC beats strongly affect the QRS complex shape.
In order to describe the signal both in the time domain or in the frequency domain, the wavelet transform was used. The wavelets coefficients provide multi-frequency band information of the signals. Since the number of samples in each window was equal to 130 samples (small windows), the maximum level of wavelet decomposition is up to 5, and we took just the 4th and the 5th level, better suited for ECG's frequency content. (Article). In this study, the detailed information at these levels are used to represent morphology-related features of an ECG signal.
Other typically used features regarding ECG signals could not be used since the goal is no to classify it as a whole but rather, its beats. These are parameters like Quality index from NeuroKit which compares each QRS complex to the average QRS and returns a similarity index or ecg_process which returns measures like breathing rate, mean between the RR differences, ECG phase and so on. We thus had 36 variables plus the target one.

We then performed data visualization and summary statistics to look at the distributions of the features extracted and perform some pre-processing. We found in the variables 'Rsucc' and 'Rprec' some R-peak distances very far from the physiological ranges, so we set a superior and an inferior threshold, having found few outlying samples, and most of them belonging to the majority class these were removed with small concern for data loss or biasing. As for the other variables, the values outside the boxplots' whiskers were not considered as outliers since in great quantity and denser towards the mean.
We were limited in removing samples because we tried to avoid to remove samples associated to the labels of the minority classes.
We then tried to do some feature selection with correlation-based method: this suggested removing some variables highly correlated between each other (we kept the ones less correlated with all the others on average, since more informative out of the two). However, trying to train models with this new dataset, the results worsened slightly, specially for neural networks which learn from data the importance to assign to each variable.

5) Dealing with class imbalance for deep models: The main issue this dataset presents is data imbalance, in particular there are many more samples ascribed to the Normal class than to the 2 pathology ones (almost 13:1) while the samples per pathology are more alike; to face this problem, having as main consequence that of biasing the model towards learning to predict too often the majority class various solutions were explored (in order):

- ***Balanced loss.*** In the first case the whole training set was fed to the classifier and balanced loss was used, considering as loss the categorical cross-entropy, this consists in assigning weights, inversely proportional to the class frequency, to the computed loss, and therefore to its gradient and ultimately to the extension by which the networks weights (and biases) are updated. Similar is the mechanism implemented by focal loss which allows to set a hyperparameter defining the entity of weights to assign to counterbalance the classes' distribution. The goal is to learn each class at the same speed, but the result was that the networks were not able to learn the task, simpler models seemed to predict "N" too often while more complex ones seemed to predict "S" or "V" too frequently, the first case can be explained by considering the much higher variability of inputs for the majority class with respect to the other two, while the second case, able to overfit the data thanks to its complexity, did it much more for the classes corresponding to greater weights updates, thus causing the model to perform badly on data containing many more "N"s than otherwise.
- ***Undersampling.*** No adequate trade-off was achievable with such a polarized dataset so reducing the class imbalance by under sampling seemed to be the path to take. The idea was to still use balanced loss but with mitigated weights given by under-sampling the normal class first to 10000 samples (corresponding to "S" count x1.35; "V" count x1.67) and then to 8000; the main issue here is that early stopping occurred too early since while the network is learning to distinguish the 3 classes in equal measure the validation set is composed mainly of "N" samples so them being misclassified causes the loss to increase much more than if the network were misclassifying the other two classes, other metrics were introduced to validate the model's generalization but their trend was sometimes erratic and therefore of difficult interpretation and comparison across models. To resolve this, two alternatives were experimented with: the first was to under-sample also the validation set but just during training while the whole validation set is then used once training is complete to better replicate the model's performances on the original data distribution, which is hypothesised to be equal in the test set; the second to weight the validation loss according to its class distribution (the original one, not the under-sampled one) thus using all of the validation set. At this point losses were comparable and adequate for monitoring during training in order to implement early stopping and reduce learning rate on plateau callbacks although between the 2, the first seemed to be the most efficient solution. At this point the training procedure, whose optimization was guided by models' performances, was deemed adequate enough to actually pass to the models' hyperparameters tuning phase, this was done for all the below configurations but in no way was it possible to satisfyingly avoid mistaking "N" for "S" in many cases; this is probably due to PAC causing only slight morphological distortions (of the P wave) with respect to the clearer QRS complex alteration typical of PVC; and also the 2 pathologies seemed to be confused to often. So, while having concluded the first phase in which models' bias due to the imbalanced dataset was avoided a second phase to improve model's understanding of the task begun: the 2 proposed solutions were, to increase data since deep neural networks are notoriously data greedy, or to simplify the task.
- ***Combining data augmentation and undersampling.*** The minority class ("V") is doubled by replicating it using the most common augmentation technique used for ECG signals, polarity inversion; the same transformation was applied to the other 2 classes for consistency but only in order to perfectly balance them all, no need for balanced loss for the training set in this case but only for the validation set during training for the above reasons of comparability; "S" samples were all used with partial resampling and "N" samples, were all taken different to begin with (did not invert the same ones fed to the network) in order to maximize variability, for this goal

an ensample model would have been developed by training the same architecture on datasets composed by different samples of "N" so that by the end, all of them would have been used but the second approach turned out to be more promising since out of the above two issues only the "S"/"V" distinction was improved with this method. So far all issues presented have been said to be resolved by the sequentially proposed techniques but overall the performances are far from satisfying, best so far is achieved with the LSTM model (although slightly unstable due to reduced batch size to speed up training and considerably overfitted so of scarce generalizability perspective) which, as highlighted in the first row of the confusion matrix mistakes "N" for "S" too often.

- ***2 step classification.*** The multi-class classification problem was divided into 2 binary ones where first a network is used to distinguish Normal heartbeats from pathological ones, which include both PAC and PVC as one category; an other network is then fed the instances predicted as pathological in order 2 distinguish these 2 cases. The advantages of this approach are dual, thanks to this grouping the datasets used result less imbalanced, specially in the second step and the task is greatly simplified, the networks are in fact expected to learn more specific and therefore more efficient features which allow a more accurate classification overall and this can also be promoted by using the architecture or the hyper-parameters for the same architecture which best suit each step. Both validation and training sets were transformed accordingly and they were completely fed to the networks using only weighted loss to deal with data imbalance.

6) Metrics: The task being a multi-class classification with imbalanced dataset proper metric must be defined to evaluate the actual capability of the models to learn the task (separate the labels in the feature space, wether hand-crafted or data-driven).

  - F1 score was defined as the average among F1 scores computed considering each class at a time as positive and the other 2 as negative.
  - Harmonic trace was defined as the harmonic mean of the components on the normalized (with respect to the rows) confusion matrix's diagonal, which corresponds to the harmonic mean of the true positive percentage for each class. this is a custom metric implemented to strongly penalise models biased toward the prediction of the majority class.
  - Balanced accuracy computes the accuracy for each class weighting it with class weights for the training set and sample weights during validation when training a Keras model, thus proving useful for callbacks definition in this case.

7) Model definition and training:

- ***Machine Learning Techniques***
  Different Machine Learning techniques are applied to empirically test which method is the most suitable for this classification task.
  The 2 step classification method cited above is applied here. All machine learning models use a normalized version (obtained with the standard scaler) of the wide features.

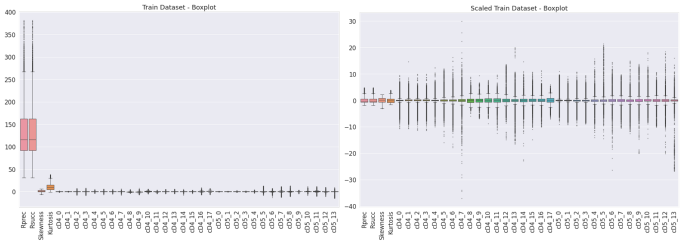

Fig. 7. Training set before and after normalization.

1) ***FFNN*** The Feed-Forward-Neural-Networks tested are quite simple (only 3 hidden layer) to avoid overfitting. From the Confusion matrix it's clear that the 2-Step combined classifier perform better than the single net with the f1 score that increase from 0.839 to 0.881.

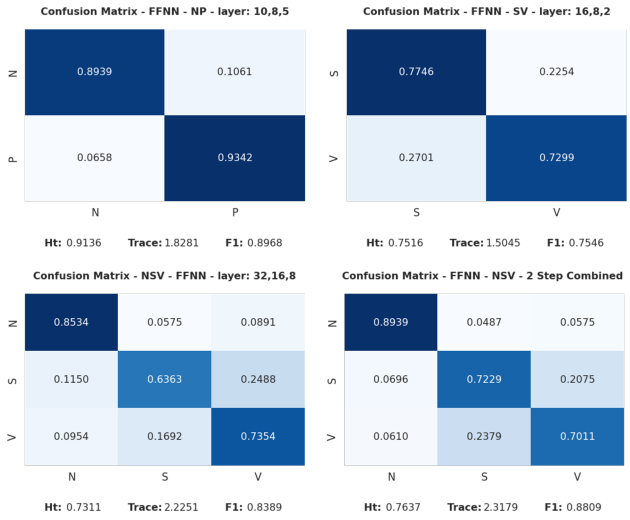

Fig. 8. FFNN Best Performance (On the top-right corner there are the number of hidden units per layer.

2) ***KNN*** Paradoxically, KNN classifier showed a marked performances deterioration after standardization.
Probably this is due to the fact that the first 4 features (next and previous inter-peak distance, kurtosis and skewness) have been excessively compressed compared to the others.

To overcome this problem these features have been multiplied by a weight factor of 14.
This change has remarkably improved the confusion matrix:

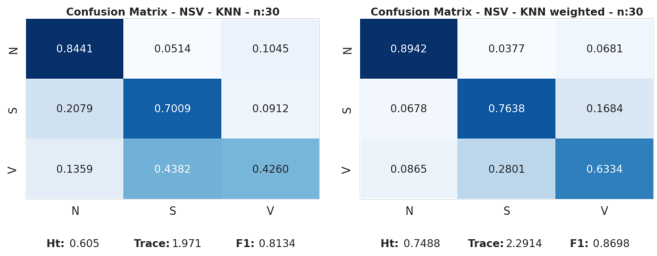

Fig. 9. KNN performances comparison between weighted and non weighted features.

In the following figure it is possible to notice how KNN with the weighted dataset and the 2-step prediction gives results comparable to FFNN.

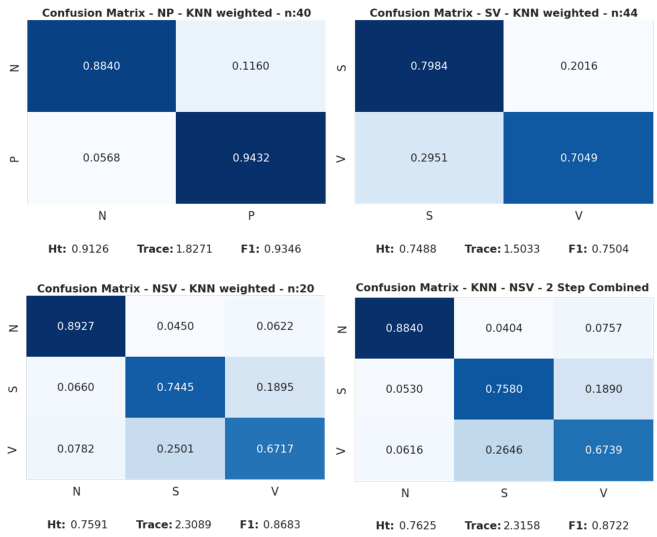

Fig. 10. KNN Performances.

3) ***Decision Tree & Random Forest*** Decision Tree and Random Forest Techniques are very common in literature for this kind of task but unfortunately in this case study their performances are not at the level of FFNN and KNN.

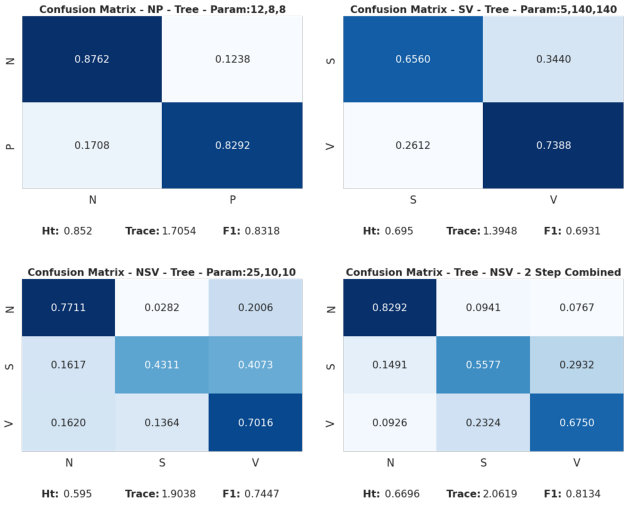

Fig. 11. Decision Tree Best Perfomances.

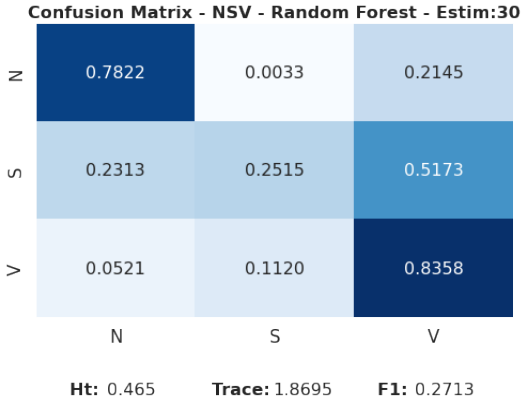

Fig. 12. Random Forest Best Perfomances.

4) ***Adaboost & Gradientboost*** Similarly to decision tree the performances of Adaboost & Gradientboost classifier are not as good as KNN or MLP.

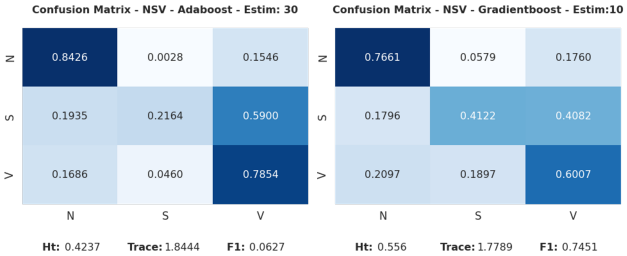

Fig. 13. Adaboost and Gradiendboost Performances.

5) ***Support Vector Machine*** The performances of SVM classifier was better than the ensamble models but not as good as our best classifier (KNN).

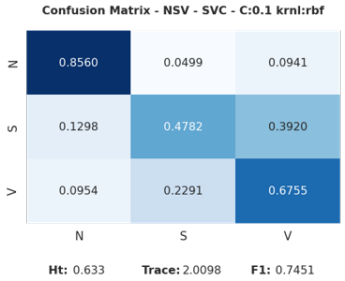

Fig. 14. SVC Performances.

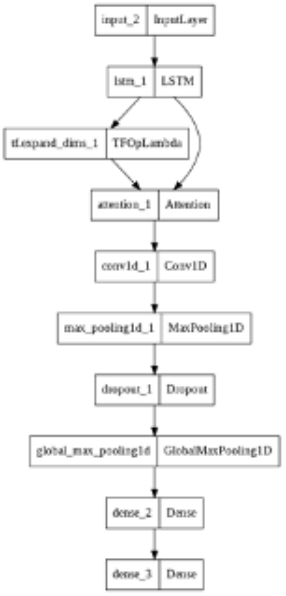

Fig. 15. LSTM classifier

- ***Deep learning***

1) ***LSTM classifier.*** Using recurrent neural networks it was possible to keep track of the signal's evolution in time.
All the models had in common the extraction of features from the LSTM initial layers by means of 1D convolution and classification by means of dense layers taking in input these features; main differences being linear vs non linear classification part, number of convolution-pooling blocks and LSTM variants (hierarchical, bidirectional, both). Hierarchical LSTMs allow the extraction of higher level features similarly to multiple convolutional layers and bidirectional LSTMs process the signal once from start to end and once from end to start and concatenates the obtained hidden states. These models proved capable of learning the task but with one big drawback: to reduce overfitting, dropout, regularization and early stopping, were not sufficient so the final models had to be greatly simplified with respect to the above hyperparameters, including also units and filters numbers, thus reducing performances. An interesting feature to notice was the beneficial use of attention which allows to focus on the states/inputs which are more relevant and this actually suggested, for the use of CNNs to feed them multiple inputs to focus the attention on the actual heartbeat to classify.

2) ***Multi-output classifier*** LSTMs are incapable of efficiently keeping memory of long sequences so rather than feeding to them the signal directly they are often fed a higher level and more compact version of it in output from convolutional layers; interestingly this approach was found combined with a simple dense classifier upon the features extracted by the convolutional part and this approach was seen to provide state of the art accuracy (¿90%) for each output branch) for 6 types of heartbeats' classification on the MIT BIH arrhythmia database thus proving the advantageous use of CNNs for feature extraction and it was also reported that the LSTM top performed slightly better than the traditional dense one. In our case the difference between the 2 output branches' performances was more consistent as observable also by the weights assigned to each by the gridsearch algorithm for the linear combinations of the 2 softmaxs' outputs: 0.96 to the LSTM branch and 0.04 to the dense dense one thus obtaining the following confusion matrix. The main issue here was once again the misclassification of "N" as "S". An interesting future development would be to pretrain this model on the MIT BIH database and then apply transfer learning, so keeping the feature extraction (convolutional) part and training only the 2 classifier branches on our dataset in order to hopefully obtain comparable performances with those obtained in the article.

Fig. 16. Multioutput classifier

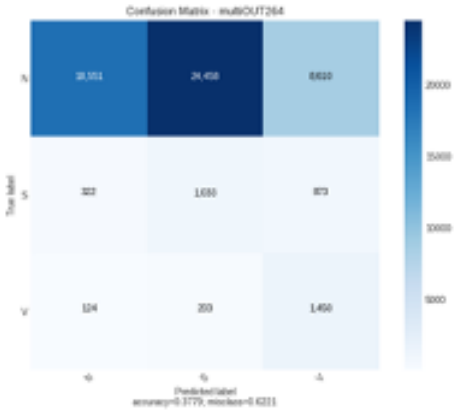

Fig. 17. Confusion matrix multi-output

3) ***Multi-input classifier*** Having realized the great potentiality of feature extraction using CNN a multi-modal approach was adopted taking in input the medium sized window and the small one with the intention to force the model to pay greater attention to morphological differences (close up) between heartbeats to classify and this has been actually achieved, as can be noticed by the very good distinction made between "V" and the other 2 classes but what this network lacks could be a rhythm information such as the wide feature R-R distance to resolve the everlasting issue of "N"-"S" distinction. But also this trial was dropped in favour of the much more interesting results obtained with the next model.

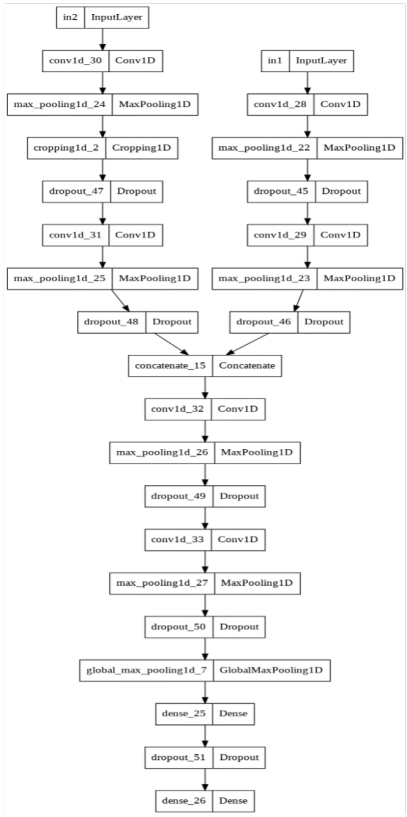

Fig. 18. Multi-input classifier

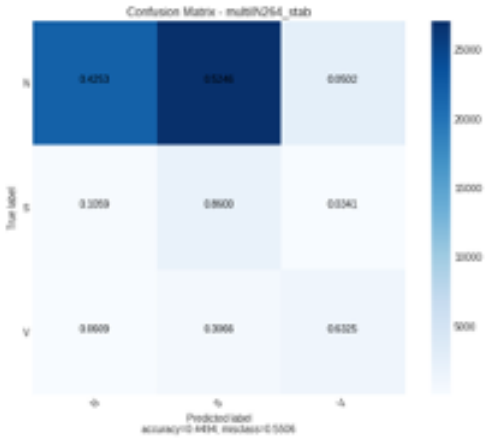

Fig. 19. Confusion matrix multi-input

This decision was mainly taken due to the extremely high overfitting (as can be seen from the plot) of this network and the drastical reduction in performance when reducing the number of

convolutional blocks simplify the model, for this reason this network was not deemed generalizable enough to be used for ensemble methods.

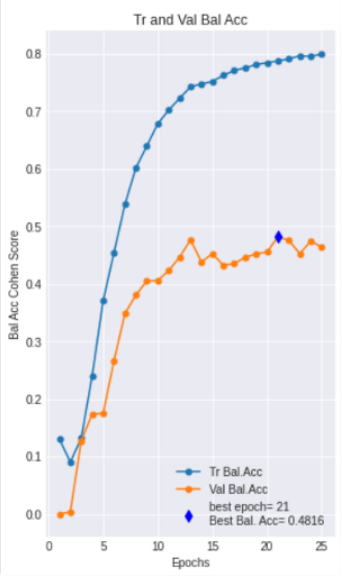

Fig. 20. Accuracy tendency

4) ***ShortTimeFourierTransform CNN***

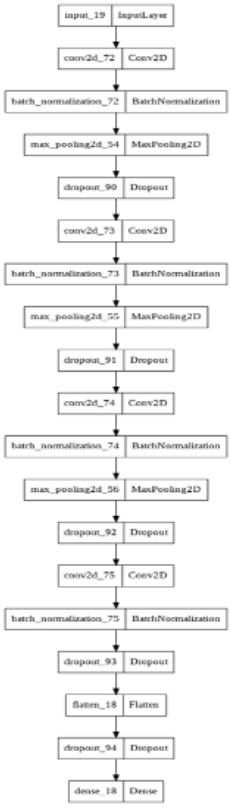

Fig. 21. STFT CNN classifier

CNNs which get fed the 2 lead ECG signal seemed to efficiently extract morphological features, but these are not sufficient to distinguish "N" from "S" so the frequency content of the signal's time sub-windows was used as input; as expected, bigger windows, able to capture the low frequency content defining heartbeat rhythm proved to contain the required information for the task. In STFT the Fourier transform of the signal is computed for each time sub-window in a discrete way with no overlap in this case; in order to minimize input sparsity the frequencies were cropped at 40Hz since this is the cut off frequency of low pass filter used in pre-processing. Being this the most promising approach so far as regards discrimination between all classes, 2 step classification was done using this technique.

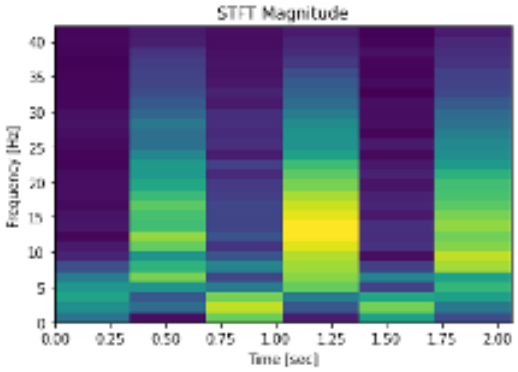

Fig. 22. STFT magnitude

The already high capability of "N" vs. pathological distinction was greatly improved by means of such a task simplification, and although less evidently, also the "S"/"V" classification improved; what is important anyway is the overall result which, while not using wide features, is the best obtained: for this reason, at least for step 2 an ensemble model between the one here presented and a hybrid (wide features - signal) will be created by linear combination of the models' prediction where weights will be assigned to each model in order to maximise the balanced accuracy on the validation set. A close look to the confusion matrix for step 1 allows to notice how the model is just slightly biased towards the prediction of D but it is more convenient for it to be this way than the opposite since in medicine the cost of false negatives is greater than the one of false positive.

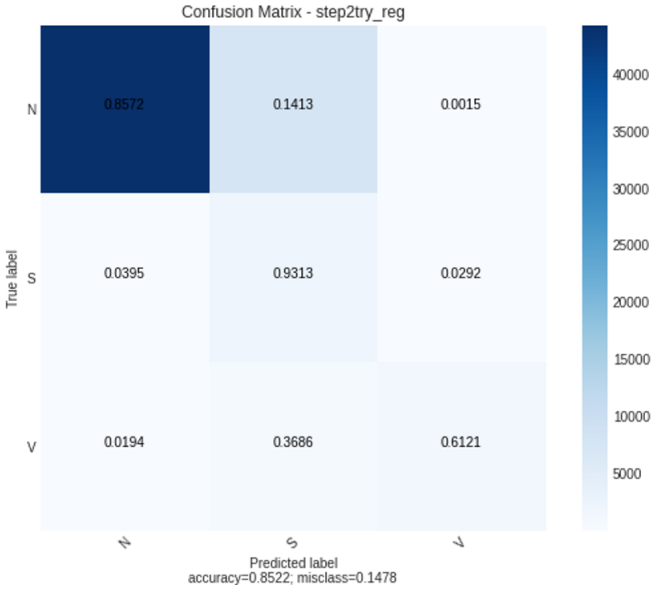

Fig. 23. Confusion matrix 1 - STFT

Here is the confusion matrix relative to the performances on the whole validation set using the 2 step prediction.

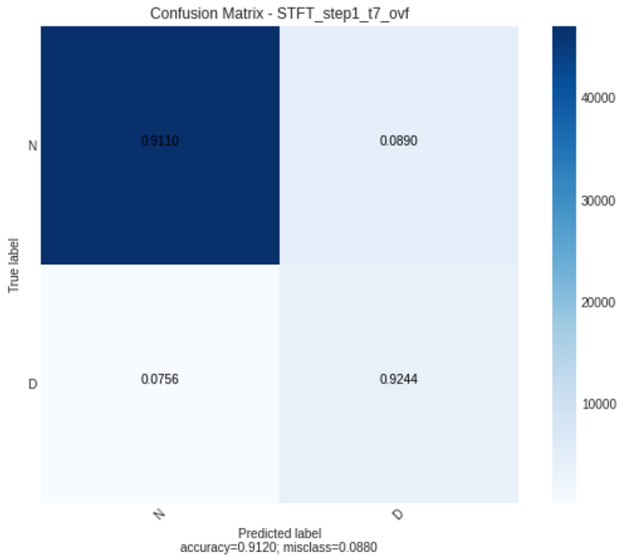

Fig. 24. Confusion matrix 2 -STFT

An interesting aspect to notice is that the same architecture was used both for step 1 and 2, with slight modifications in hyperparameters to improve stability in the first case and overfitting in the second; but actually other models were expected initially to perform better in step 2 than than the one taking in input the STFT of the signal since both the multi-input and the multi-output architectures showed greater capability to extract morphological featured from the signal in time. For this reason all the above models, their variations and combinations of them were retrained and validated on “S”/”V” classification but to no avail, this was probably due to the small amount of data available for this training and the only reliable augmentation technique being polarity inversion, the options were scarce, for instance Gaussian noise was added at random to augment the data but this decreased performance, as did any attempt to simplify the models in order to reduce the number of parameter to train and therefore overfitting. The theoretically best candidate, although not empirically validated is here presented since we think it could potentially be very effective if pretrained on a bigger dataset and then transfer learning could be applied to our own. It is bult considering the multi input (2 parallel branches) advantage of focusing on the morphology of the heartbeat of interest, convolutional processing to extract relevant features for classification and a dense-LSTM-dense output since this was shown to perform better than traditional dense classifier.

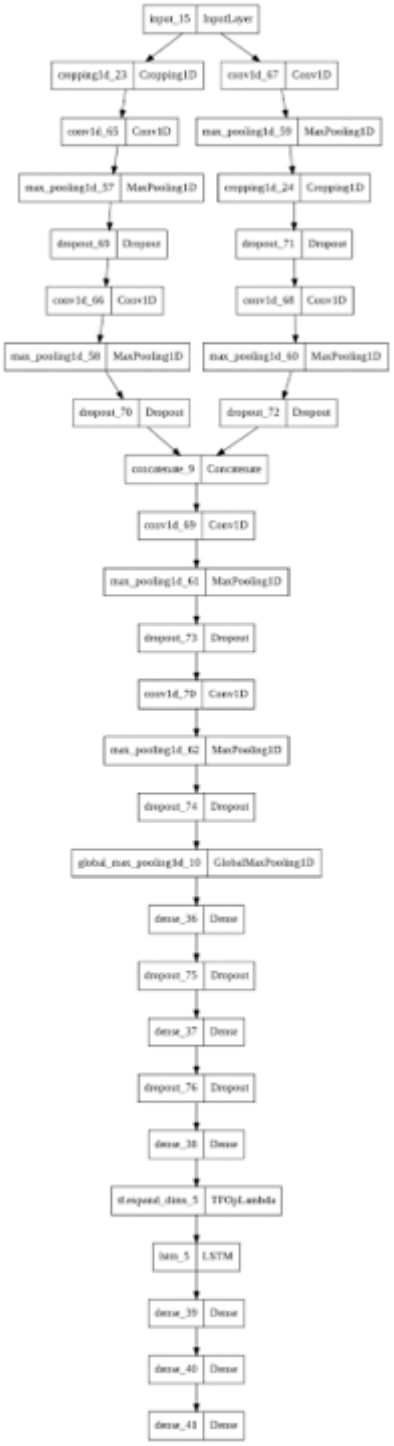

Fig. 25. Conceptual candidate for S/V classification but too complex for the available data.

5) ***Hybrid Model*** Since the feed forward neural network with wide features provide good results, possible improvement consist of combine the information coming both from the features and the signal itself.
To fulfill this aim a possible solution was to use a hybrid model with a dense branch for the classification of the features and a second deep branch for the signal.
The outputs of the branches were then concatenated and fed to a 2 layer classifier.
To deal with class imbalance instead of the classical categorical cross-entropy loss the focal loss was used in combination with different class weights assigned during the training phase.
As for the FFNN case also with the hybrid model the 2-step classification strategy was adopted.
Due to the fact that information about the rhythm (inter-peaks distance) was already present in Wide Features, then the signal window used was made of only 130 sample (sampling frequency of 128 Hz). In such way only one heart beat per window was considered.

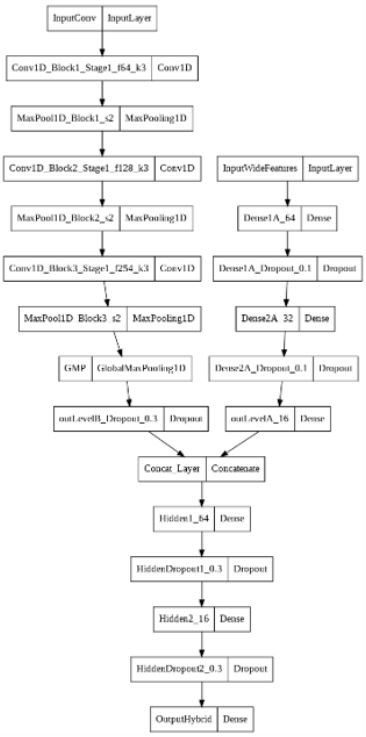

Fig. 26. Hybrid Model NP - Classification of Healthy vs Pathological beats

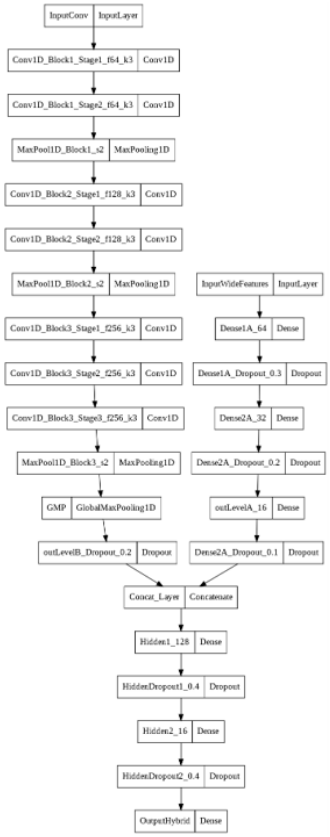

Fig. 27. Hybrid Model NP - Classification of Healthy vs Pathological beats

The results obtained after the combination of the 2 predictor are the best until now.

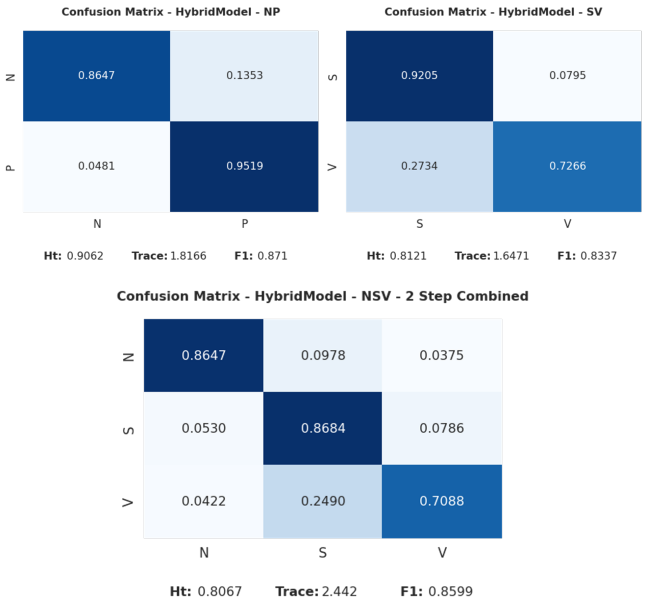

Fig. 28. Performances of the 2-step classification with the hybrid model.

IV. Results

The best models obtained for each type of technique are:

- STFT 2 step deep model
- wide feature KNN 2 step classifier
- hybrid 2 step dense classifier

To exploit the different aspects of informativeness each of these techniques provides, an ensemble model was built using all of them; this allows to increase prediction robustness by implementing majority voting. Actually the linear combination of the three models' outputs was not uniformly weighted but rather, a gridsearch of all compatible weights, with a precision of 0.01, was performed in order to maximize the resulting confusion matrix's trace. Doing so weights are not necessarily assigned proportionally to the performance of each single model on the validation set but maximising the combination potentiality; for this reason KNN receives a greater weight (0.48) than the better performing STFT (0.03), this last model, in fact probably makes similar misclassifications as the best hybrid (weighted with 0.49), while the KNN, which is only fed wide features, learns different aspects which although less performing alone brings an important contribution to the ensemble's performance.

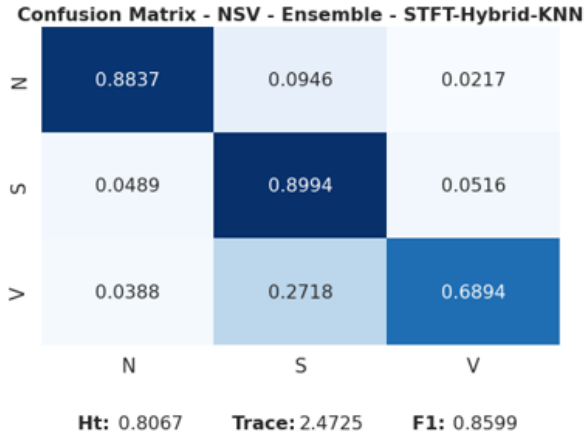

Fig. 29. Hybrid Model NP - Classification of Healthy vs Pathological beats

For more in depth explanations regarding single models we refer to the models in the methods section [III].

V. Discussion and conclusion

During different trials performed with the wide features, and even from PCA, the importance of timing features, namely R_prec and R_succ is evident; this could explain why the deep learning algorithms, which should extract such information from data, perform comparably with the ML ones but overall slightly worse, convolutional layers are in fact feature extractors based on local correlations, therefore more adequate for capturing morphological/sub-window frequency content features rather than rhythmic ones, namely heart rate. The ensemble increases overall

performances and is also expected to improve robustness and therefore generalisability, as will be evaluated on the test set; however it does not resolve the issue, common to all models, that is the lower performance over PVC classification, which is too often mistaken for PAC.
Once again it is important to highlight the greater importance given to the correct classification between healthy heartbeats and pathological ones, since the overall burden of PACs and PVCs rather than just for each single one could be, although of inferior clinical relevance, an important indication towards the need for more specific and accurate diagnostic techniques for the patient.

A. Future developments

Having noticed how the STFT input to the deep CNNs favoured the extraction of more relevant features as classification capability rather than the simple ECG signal in time this approach could be replicated in the hybrid model together with the wide features; and then make en ensemble of this output with that of KNN. Also, the ensemble could be also built in 2 steps, thus combining models for step 1 pf classification, feeding them the pathological samples (predicted by the ensemble 1) and then repeating the linear combination of predictions phase also for step 2 of classification, also using different models, depending on which ones perform better for each phase; this would allow to reduce the error propagation in from step 1 than can weaken the models that perform worse at this step than the second one.

Appendix

To obtain a model able to provide smaller error in the estimation is necessary to perform the stratified random sampling of the dataset.
The problem is quite complex because not only the distribution of heartbeats (N, S, V) must be equal both in the train and validation set but also the subdivision is not “heartbeat based” but “patient based”. This because mixing heartbeats that come from the same patient in the train set and validation one can possibly jeopardize the validation’s performance reliability (too optimistic generalization).
Since scikit learn does not provide a function capable of satisfying all these requirements (and a brute force search would be impossible given the high number of combinations), the most suitable option was optimization based on genetic algorithm.
The problem is coded into a genome: a genome (one possible solution) is defined as a list with length of 80 (the number of patients in the training set). Each element of the list (gene) is a number that represent univocally a possible patient in the training set.

Algorithm execution:

1) A population of 100 possible solutions (genomes) is initialized randomly using random patients from the whole dataset.
2) Each solution is evaluated using a fitness loss that measures how well balanced the splitting is.
3) New solutions are generated from the original population using a 1-point crossover function.
4) The lowest solutions of the population are discarded and the new generation is finished.
5) This process is repeated 2000 times (generations) to reach convergence.
6) To avoid falling in a local minimum, the algorithm was initialized 8 times.

The fitness function is computed as follow:

- Into 3 lists there are the number of N, S, V beats for each patient, stored as weights. The indexes of the lists associate univocally a specific patient of the whole dataset.
- For each genome the total number of N, S, V is computed. The result is the number of beats divided by classes for the candidate training set.
- By subtracting the number of beats of the training set from the total number, the beats classes for the validation are found.
- The percentage of weight distribution in train and validation is computed.
- The fitness loss is defined as the sum of the differences between the percentage of beats of a certain class in training and the percentage in validation for N, S, V.
- Ideally the optimal solution has a fitness of 0, so in this case lower fitness mean better solution (loss).

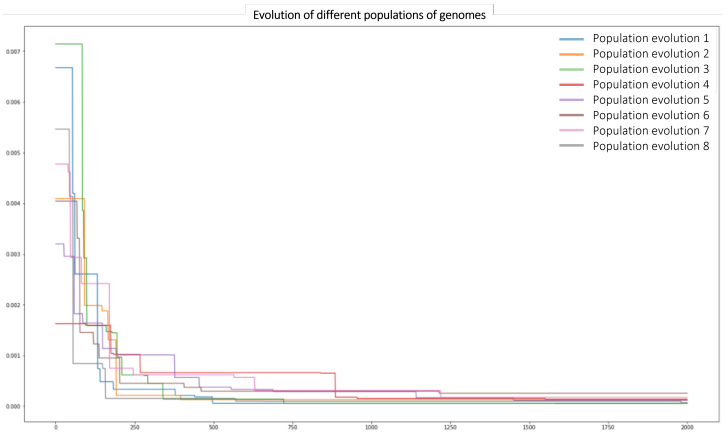

Fig. 30. Genetic Algorithm Evolution