ECG Heartbeat Classification for Arrhythmia Detection: A Machine Learning Approach

Nguyen Hai Dang - 22BI13073
Department of Information and Communication Technology
University of Science and Technology of Hanoi

***Abstract*—Classifying ECG heartbeat automatically for diagnosing arrhythmia has gain the attention of researchers recent years. In this report, I represent an approach to the problem using random forest, a popular ensemble model used in supervised classification task. Although the result is comparable to others, it still requires more works to be done on preprocessing aspect.**

***Index Terms*—ECG, arrhythmia, machine learning, signal classification**

I. Introduction

Arrhythmia is a term refers to any problem that relate to abnormal heartbeat rhythm. There are mainly 2 types of arrhythmia. A heart beats too fast (more than 100 beats per minute when resting) is called tachycardia [1] (Figure I). A heart beats too slow (less than 60 beats per minutes when resting) is called bradycardia [2]. Tachycardia can cause fainting and thrombosis (i.e., blood clots blocking blood vessels). Bradycardia, although it can be normal for most cases, especially for healthy people or athletes, it could lead to many symptoms such as chest pain, confusion, memory problems, etc. Therefore, being able to classify heartbeat from ECG signal is crucial. Unfortunately, classifying ECG signal is a difficult and time-consuming task that require the consultation of professional physicians. Thus, being able to achieve the task automatically is of great importance.

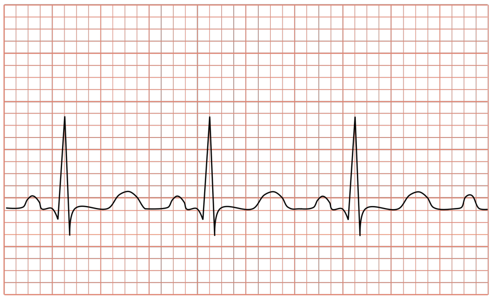

Fig. 1. A sample of fast heartbeat rhythm

There are a number of researches have been done on classify ECG signals automatically. [3] preprocessed data using bidirectional infinite impulse response filter with band-pass filter with non-linear blood pressure - pulse transit time (BP-PTT) model. [4] provides a preprocessing procedure to extract beat and a deep learning model using 1D CNN to classify. [5] used low-dimensional denoising embedding transformer (LDTF) with the same preprocessing steps as [4]. In this report, I will apply random forest on MIT-BIH dataset, which is the same as [4], [5]

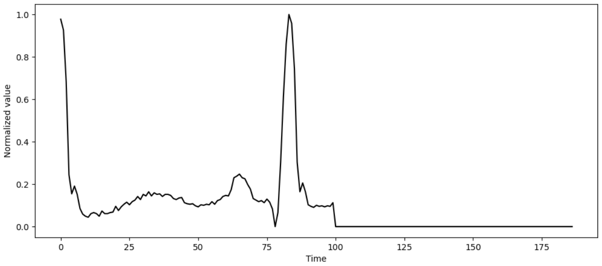

Fig. 2. A sample of processed ECG signal

The report is structured as follow: Section 2 describes the dataset: metadata and data analysis, section 3 describes the experiment protocol and section 4 is conclusion.

II. Dataset

In this section, I represent how the data were collected, preprocessed and some explanatory data analysis on the train dataset.

ECG signals consists of 48 ECG recordings, collected using a two-channel ambulatory ECG monitoring from 47 patients. These recordings are preprocessed using the following procedures [4]:

1) Split ECG signals to 10-second windows. For each of these windows, do the following steps.
2) Normalize signal to the range 0 and 1
3) Find all local maximums
4) Find all ECG R-peak candidates.
5) Calculate nominal heartbeat period T = median(R-R time intervals)
6) For each R-peak candidate, select a signal sequence of length L = T
7) Padding each sequence with 0s to a predefined fixed length (in this case is 187)

The result is a table containing 109446 heartbeat samples (see Figure 2), each has 187 features with 1 label. There are 5 classes: N, S, V, F, Q, which are encoded to integer from 0 to 5, respectively. Figure 3 displays the distribution of labels in training dataset. It is observed that the data is highly imbalanced. The dataset is further divided into train set (87554 samples) and test set (21892 samples).

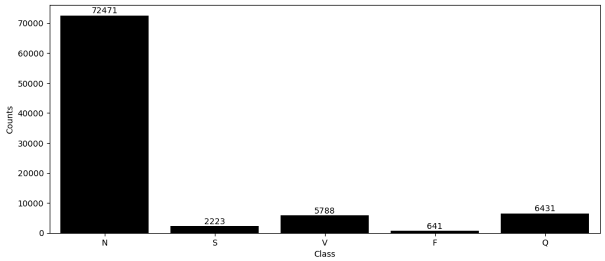

Fig. 3. Label distribution

III. Experiment

In this section, I describe the training and hyperparameter tuning process for the random forest model, evaluate model's performance and compare it to existing works.

A. Training and Hyperparameter Tuning

In my approach, I use random forest, an ensemble model constructed by combining multiple small decision trees. During training and tuning process, I focused on 2 hyperparameters: maximum depth of the tree and the number of trees in the forest (see Table I for the range I used). The process is done sequentially, begin with maximum depth first since it is set to infinity by default of scikit-learn library. This can easily lead to overfitting. The optimal values are found using 5-fold cross validation with grid search approach.

TABLE I
Grid of parameters for grid search

Parameters	Range of values
max_depth	1,3,5,...,19
n_estimators	50, 100, ..., 500

Table II summarizes the parameters obtained from grid search. One thing worth mentioning is that I set class weight is ”balanced”, i.e., each class is automatically set a weight as:

$$w_c = \frac{N}{C * count(c)}$$

where w_c is the weight associated with class c, N is the total number of samples and C is number of classes. If a class has small samples compare to other, the weight will be more significant, and vice versa. This is to cope with imbalance dataset, as shown in Figure 3.

TABLE II
Final random forest hyperparameters

Parameters	Values
n_estimators	400
max_depth	9
class_weight	balanced

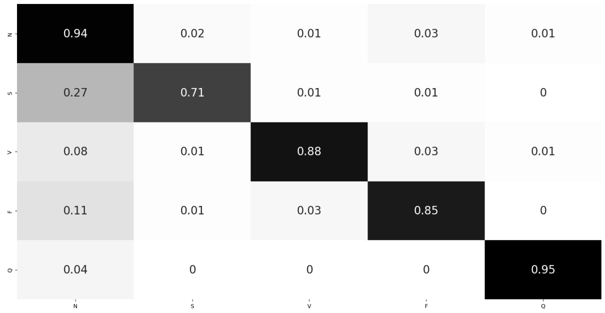

Fig. 4. Normalized confusion matrix

B. Evaluation

Beside the confusion matrix shown in Figure III-B, for evaluating the model, I use 3 metrics precision, recall (both using macro method to account for imbalance), and overall accuracy. Macro precision is calculated by averaging precision for each class:

$$Macroprecision = \frac{1}{C} \sum_{c \in C} \frac{TP_c}{TP_c + FP_c}$$

where TP_c, FP_c is number of true positive, false positive when consider c as the positive class. Similarly, macro recall is calculated as:

$$Macrorecall = \frac{1}{C} \sum_{c \in C} \frac{TP_c}{TP_c + FN_c}$$

Accuracy is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Table III-B shows the result of the model. Though it is observe that my accuracy is similar to [4], when considering macro precision and recall, the metrics is not impressive.

TABLE III
My result compare to other

	Precision	Recall	Accuracy
[4]	-	-	93.4
This	71.9	86.7	93.3

IV. Conclusion

In this report, I show an approach to the ECG heartbeat classification problem. The overall accuracy is competitive, but other metrics that account the imbalance is not good enough. This suggest that there are more opportunities for improvement. The future works will focus on further preprocess the data using either traditional digital signal processing techniques or deep learning architectures that is capable of processing sequential data such as 1D CNN, RNN or LSTM.

References

[1] Awtry EH, Jeon C, Ware MG (2006). ”Tachyarrhythmias”. Blueprints Cardiology (2nd ed.). Malden, Mass.: Blackwell. p. 93. ISBN 9781405104647.

[2] Hafeez Y, Grossman SA (9 August 2021). ”Sinus bradycardia”. StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing. PMID 29630253. Retrieved 16 January 2022.

[3] A. Esmaili, M. Kachuee, and M. Shabany, “Nonlinear cuffless blood pressure estimation of healthy subjects using pulse transit time and arrival time,” IEEE Transactions on Instrumentation and Measurement, vol. 66, no. 12, pp. 3299–3308, 2017.

[4] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, “ECG Heartbeat Classification: A Deep Transferable Representation,” in 2018 IEEE International Conference on Healthcare Informatics (ICHI), Jun. 2018, pp. 443–444. doi: 10.1109/ICHI.2018.00092.

[5] J. Guan, W. Wang, P. Feng, X. Wang and W. Wang, ”Low-Dimensional Denoising Embedding Transformer for ECG Classification,” ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 2021, pp. 1285-1289, doi: 10.1109/ICASSP39728.2021.9413766.

[6] Moody GB, Mark RG. The impact of the MIT-BIH Arrhythmia Database. IEEE Eng in Med and Biol 20(3):45-50 (May-June 2001). (PMID: 11446209)