Fatal or Non-fatal: Comparative Study of Classification Algorithms for Cardiac Arrhythmias Discrimination

Team Members:

Alëna Rodionova	(PennKey: nellro	Email: nellro@seas.upenn.edu)
Haochen Han	(PennKey: hanhc	Email: hanhc@seas.upenn.edu)
Haotian Zhu	(PennKey: htzhu	Email: htzhu@seas.upenn.edu)
Po-Yuan Wang	(PennKey: poyuanw	Email: poyuanw@seas.upenn.edu)

Assigned Project Mentor:

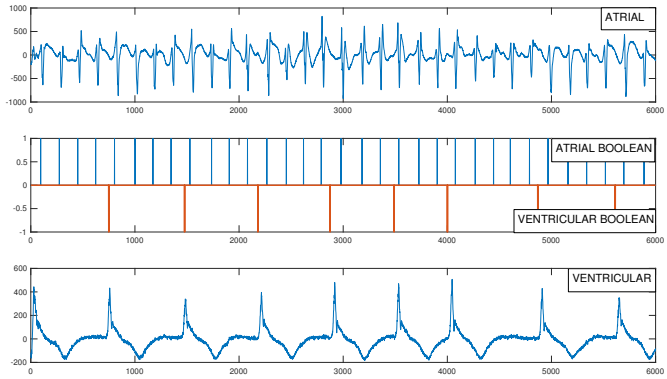
Brandon Lin

Team Member Contributions:

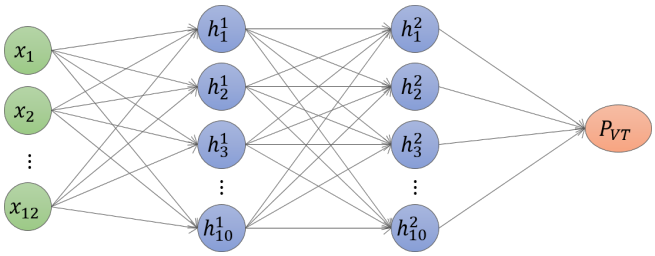
Team member	Contributions
Alëna Rodionova	Problem formulation, data pre-processing (feature extraction), DNN methodology
Haochen Han	Decision tree methodoloy, result analysis
Haotian Zhu	SVM methodoloy, FFT feature extraction, FFT approach
Po-Yuan Wang	kNN methodology, modified kNN approach

Code Submission:

`https://github.com/nellro/CIS520_project`



(a) EGM recording during SVT. Bottom panel represents V-signal, top panel shows A-signal, middle panel presents corresponding boolean beat channels.



(b) DNN Model Architecture

Figure 1: EGM signal example (left), proposed DNN model (right).

1 Introduction

According to the World Health Organization (WHO), heart disease is the leading cause of death around the world. More than 1000 Americans died each day in 2016 from heart attacks. Although a single arrhythmia heartbeat will not have a serious impact on life, continuous arrhythmia can result in fatal circumstances. For instance, *Ventricular Tachycardia* (VT) is a potentially fatal arrhythmia. On the other hand, *Supraventricular Tachycardia* (SVT) is not considered to be dangerous for most of people.

In this project we compared classification methods for electrogram (EGM) arrhythmia discrimination. We consider two arrhythmia types: ventricular tachycardia (VT) and supraventricular tachycardia (SVT). Therefore, we viewed this problem as a binary classification task to discriminate whether a patient has VT (fatal tachycardia) or SVT (non-fatal tachycardia).

The set of methods we considered, implemented and analysed include Deep Neural Network (DNN), Support Vector Machine (SVM), k-Nearest Neighbours (k-NN) and Decision Tree.

2 Related work

Due to high importance of the problem, hundreds of methods for arrhythmia classification have been presented in the literature, many of them use machine learning techniques. For instance, in [8] authors perform multi-class classification of ECG signals using convolutional neural network (CNN) to discriminate between different tachycardia types. Authors in [2] present a 95.16% accurate k-NN classifier. In 2012, other group of researchers reported a 99.34% accurate multi-class arrhythmia classification algorithm using *ensemble decision trees*. In [11] authors show that under specific conditions decision trees perform better than SVM classifiers.

3 Data set

For our problem we used the **Cardiac Model EGM Database** [7]. The EGM (electrogram) signals in the database were generated by the heart model that has been validated for realism by cardiologists [7]. The database consists of 1920 EGMs, equally split into 960 VTs and 960 SVTs. Each recording has two real-valued channels, depending on the heart chamber where they were measured: Ventricular channel

(V-signal) and Atrial channel (A-signal). Each recording also has two boolean channels, representing the detected heart beats in V- and A-signals.

An example SVT EGM recording is shown in a Figure 1a.

3.1 Data pre-processing and features extraction

Following the methodology from [5], data pre-processing includes the following steps:

1. Resample each signal with a sampling frequency of 300Hz.
2. Filter the signal with a fourth-order bandpass Butterworth filter with low cut-off frequency of 0.4Hz and high cut-off frequency of 40Hz to avoid baseline drift and reduce high-frequency noise.
3. Segment the signal into 10s data lengths.
4. Extract 6 features from each signal (6 for each V-signal and 6 for each A-signal: 12 in total): four features are derived from image-based phase plot analysis (these features represent nonlinear and non-stationary dynamics of the signal, see [5]), one derived in the frequency domain (number of frequencies which have higher amplitude than the mean value), and the last feature is Shannon Entropy [12] (The SE value is higher for VT than for SVT).

The final ML problem for our project is a 12-dimension-input binary classification problem.

4 Solution methods

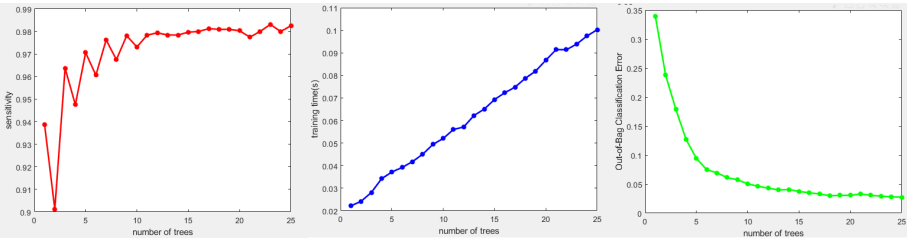
The set of methods we implemented, analysed and compared include:

- Deep Neural Network (DNN, see Section 4.1),
- Decision Tree (Section 4.2),
- Support Vector Machine (SVM, Section 4.3),
- k-Nearest Neighbours (k-NN, Section 4.4).

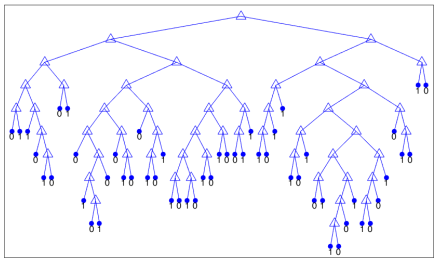
Related work review shows that decision trees can achieve high accuracy results in arrhythmia classification based on ECG (electro-cardiogram) signals. As ECG and EGM representations are topologically close to each other, decision tree is a valid candidate for distinguishing fatal/non-fatal tachycardias based on EGMs. Compared with neural networks, decision trees have an advantage that they are easy to interpret. Unlike neural network, the transparency of decision tree gives us a better understanding of the intermediate processes. k-NN algorithm is known for its implementation simplicity and the absence of the training phase. It is robust to noisy data, but sensitive to irrelevant features. The biggest disadvantage of k-NN is its expensive computations in the testing phase. Moreover, in case of high-dimensional data, the distance between two data points becomes less meaningful and the accuracy may decrease [3].

We used the following performance measures (See Section 5):

- Accuracy (0-1 loss).
- Confusion matrix measures (TP, TN, FP, FN).
- Precision, F1, Sensitivity or TPR (# correctly detected VT/# true VTs), Specificity or TNR (# correctly detected SVT/# true SVTs). Those are very important performance measures in case of tachycardia discrimination. Though FP (misclassifying SVT as VT) might lead to a process of further examination or treatment episodes which are painful for the patient, FN (misclassifying VT as SVT) might put the patient in the risk of death.
- We also measured training and test times.



(a) Tree Estimations: Trade-off between the number of trees and (left subplot) Sensitivity, (middle subplot) training time, (right subplot) OOB.



(b) Learned tree graph structure.

Figure 2: Decision tree plots

Now we will present each analysed approach in more details.

4.1 Deep Neural Network

The proposed DNN model consists of 2 hidden layers with 10 neurons each, 12 neurons in the input layer (we use 12 features input data representation, see Section 3) and one node in the output layer, see Figure 1b. These hyper-parameters were obtained by running the set of experiments and comparing the performance, no cross-validation was performed.

Relu activation functions were used, output layer uses logistic sigmoid activation function to produce an estimate of the probability of VT label. At the training phase we minimize the cross-entropy loss $l_{\log}$: $\{0,1\} \times [0,1] \to \mathbb{R}_+$:

$$l_{\log}(y,p) = -y \log p - (1-y) \log(1-p). \tag{1}$$

We trained the proposed DNN model for 3000 epochs (one forward and backward pass of *all* the training examples) with a batch size of 128 (number of training examples in one forward/backward pass).

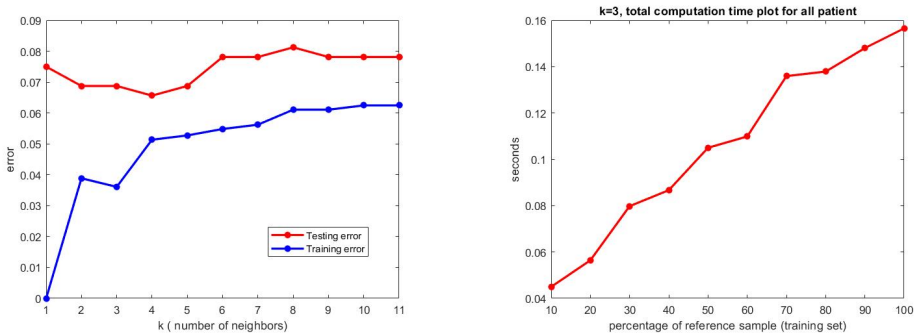
Implementation. We used Keras 2.2.4 and Python Deep Learning library which runs on top of TensorFlow. The Python version is 3.6. We ran experiments using Ubuntu 18.04 LTS with no GPU support. Implementation results are presented in the Table 1.

4.2 Decision Tree

As a first step, a simple classification tree was generated from the training data. The tree was built using CART (Classification and Regression Trees) algorithm [10]. CART constructs a binary tree, where each node has two output leaf nodes. Gini gain selection was used for the splitting. The final tree graph is presented in the Figure 2b. See Table 1 for the performance results.

As a next step, to improve performance of the model, we tried Bootstrap Aggregation (Bagging) [9] on trees. Bagging algorithm creates subsets of data and trains decision trees on those subsets. It has been shown in that such methodology can effectively improve the stability of the algorithm [4] and avoid overfitting [6]. To decide on the number of trees for the Bagging algorithm, three performance measures were analysed: (1) sensitivity, (2) training time and (3) OOB. Figure 2a shows the trade-off between the number of trees and three measures listed above.

Implementation. We used the MATLAB class *TreeBagger*, from Statistics and Machine Learning Toolbox [1] to generate our classifiers. See MATLAB code for more details.



(a) Error Plot (b) Computation Time

Figure 3: k-NN performance: training/test errors (left) and computation time (right).

4.3 Support Vector Machine

We used a polynomial kernel:

$$G(x_j, x_k) = (1 + x_j x_k)^q$$

Cross-validation procedure was performed to determine the best parameter value $q \in \{2, 3, 4\}$. Cross-validation was based on the accuracy measures. The performance results are presented in the Table 1.

In addition to the basis SVM model trained on the 12-features extracted using methodology from Section 3, we trained three more models to analyze possible improvements. First, naïve approach, uses a sequence of the heart-beat timings as an input to the model. Second approach uses *Fast Fourier Transform* (FFT) representation of the initial EGM signal. In this case, the features are the amplitudes and phases of $k = 10$ frequencies with highest amplitude. Third approach performs *Principal Component Analysis* (PCA) on the FFT spectrum to do the frequency-choosing part: rather than pick k frequencies with highest amplitude, pick the ones that are ortho-normal and maximize the variance. Our experiments showed that FFT with PCA model achieves the highest accuracy of 94.3%, but the training time is high, 7.89 sec. On the other hand, the basis SVM model showed the second best accuracy of 93.4%, with training time being only 0.09 sec.

Implementation. The MATLAB built-in function `fitcsvm` was used to train an SVM model and do predictions.

4.4 k-Nearest Neighbours

One of the advantages of using k-NN is that it is intuitive and simple. Moreover, k-NN algorithm has no training phase. However, k-NN stores all training data in the memory space, therefore, the method will become slower as the training data increase, see Figure 3b. We used a standard binary 0-1 loss to determine the overall accuracy and confusion matrix of the model, see Table 1. Figure 3a shows the training and test errors with respect to the number of neighbours k (X-axis).

In addition to the standard k-NN, we implemented a second version of k-NN algorithm that is more robust to unstable features. This implementation uses Minkowski distance with parameter $p = 3$. More details can be found in the provided MATLAB code.

Implementation. MATLAB was used to program the k-NN algorithm. No additional libraries or tools required.

		DNN	Decision tree	SVM	k-NN
Accuracy 0-1		0.9219	0.9313	0.9344	0.9437
Sensitivity		0.9688	0.9438	0.9313	0.90625
Specificity		0.875	0.9188	0.9375	0.94375
Precision		0.8857	0.9423	0.9371	0.9412
F1		0.9254	0.9430	0.9342	0.9201
Confusion matrix	**TP**	155/160	151/160	149/160	144/160
	TN	140/160	147/160	150/160	151/160
	FP	20/160	13/160	10/160	9/160
	FN	5/160	9/160	11/160	16/160
Train time (overall)		46.88s	0.02s	0.09s	-
Test time (per patient)		0.06ms	0.21ms	0.06ms	0.97ms
Probability estimation		+	-	-	-

Table 1: Performance measures for four presented classification algorithms

5 Experimental results and Discussion

To compare algorithms we used the same dataset (see Section 3) for all algorithms: 1600 training data points and 320 testing data points. The performance measures are presented in a Table 1. All algorithms show high accuracy above 92% and high sensitivity above 90%.

Among all algorithms k-NN shows highest accuracy, but the lowest sensitivity which is undesired in medical research. k-NN algorithm can respond quickly when adding a new training point or taking out a training point from the set. However, k-NN is sensitive to outliers and does not perform well on the imbalanced data. Moreover, k-NN suffers from the curse of dimensionality: when the data dimension increases, the computation time grows rapidly.

Decision tree achieves the second best classification sensitivity after DNN while keeping test time low. The fact that this algorithm requires relatively small computations, and that it represents a set of simple classification rules, are the reasons why decision tree performs well with respect to test time. The disadvantage of decision tree approach is overfitting which occurs often. Moreover, when the input data has inconsistent sample size, its information gain tends to be characterized by the features which has more numerical values. However, those downsides can be avoided using Bagging trees (see Section 4.2).

DNN achieves the best sensitivity result, which is very important in medical applications, but it also has the lowest accuracy value, which is also important for medical CPS. One also can see that DNN model makes very fast predictions in comparison with all other presented algorithms. This can be an implementation artefact, since Keras was developed with a focus on enabling fast experimentation and therefore, performs better in comparison with MATLAB libraries.

SVM approach achieves comparable accuracy, sensitivity, and other performance measures with other approaches. However, when the dataset gets large, SVM requires more RAM and costs more time, which is a potential problem for medical applications.

Acknowledgments

The authors would like to thank Kuk Jin Jang for the valuable discussion on the existing arrhythmia monitoring algorithms and publicly available ECG databases.

References

[1] Matlab statistics and machine learning toolbox, R2016a – R2019a. The MathWorks, Natick, MA, USA.

[2] I Assadi, A Charef, T Bensouici, and N Belgacem. Arrhythmias discrimination based on fractional order system and knn classifier. 2015.

[3] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is nearest neighbor meaningful? In *International conference on database theory*, pages 217–235. Springer, 1999.

[4] Yves Grandvalet. Stability of bagged decision trees. *NeurIPS*, 2006.

[5] Shirin Hajeb-Mohammadalipour, Mohsen Ahmadi, Reza Shahghadami, and Ki Chon. Automated method for discrimination of arrhythmias using time, frequency, and nonlinear features of electrocardiogram signals. *Sensors*, 18(7):2090, 2018.

[6] Foroozand Hossein and Weijs Steven V. Entropy ensemble filter: A modified bootstrap aggregating (bagging) procedure to improve efficiency in ensemble model simulation. *entropy*, pages 1–16, 2017.

[7] Zhihao Jiang, Houssam Abbas, Kuk Jin Jang, Marco Beccani, Jackson Liang, Sanjay Dixit, and Rahul Mangharam. In-silico pre-clinical trials for implantable cardioverter defibrillators. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 169–172. IEEE, 2016.

[8] Tae Joon Jun, Hoang Minh Nguyen, Daeyoun Kang, Dohyeun Kim, Daeyoung Kim, and Young-Hak Kim. Ecg arrhythmia classification using a 2-d convolutional neural network. *arXiv preprint arXiv:1804.06812*, 2018.

[9] Beriman Leo. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[10] Breiman Leo, Friedman Jerome H, Olshen Richard A, and Stone Charles J. Classification and regression trees. *Boca Raton*, 1984.

[11] MONALISA MOHANTY, PRADYUT BISWAL, and SUKANTA SABUT. Ventricular tachycardia and fibrillation detection using dwt and decision tree classifier. *Journal of Mechanics in Medicine and Biology*, page 1950008, 2018.

[12] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.