Machine Learning - CIE 417

Cardiac Arrhythmia

Project Report

By:

Anhar Hassan **201601171**

Khaled Osama **201600515**

Moustafa Elsayed **201600848**

Sara Elbesomy **201601849**

Table of contents

Abstract

The goal is to predict a given ECG Signal coming from a patient and classify it to 5 different types. we used the MIT-BIH Arrhythmia Database. In order to speed up the process of training data, we used the wavelet transform to reduce the size of the data and neglect noisy data. We managed to reduce its size to ⅛ of its original size and keep the most important features at the same time. We have tried neural network models such: LSTM, GRU, 1D-CNN, combination between CNN and GRU ,and classical machine learning such support vector machine. As a result, we can say that all the models work well on the data and yield good accuracy but svm with RBF kernel and LSM are the best models we have tried.

Problem Statement

Heart Arrhythmia is a medical condition where the electrical signals that control the heart do not work properly [1]. These impulses for arrhythmia patients may cause the heart to beat too fast, too slow, or irregularly. Arrhythmia contributes with a significant percentage, about 15% of the world 's death [2]. The problem is that arrhythmia has no certain symptoms, and the doctors do electrocardiography to diagnose it [3]. Hence, we are training models to classify people into normal and into one of four types of arrhythmia according to their ECG signals.

Database

- MIT-BIH Arrhythmia Database[4]:

  The MIT-BIH Arrhythmia Database contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979.

  - Number of Categories: 5

- Number of Samples: 109446
- Sampling Frequency: 125Hz
- Data Source: Physionet's MIT-BIH Arrhythmia Dataset
- Classes: ['N': 0, 'S': 1, 'V': 2, 'F': 3, 'Q': 4]

Literature Review

Different literature review that we read:

- **First Model: CNN[5].**

  In this paper the researchers used Convolutional neural networks to detect the arrhythmia and their model got of 93.4% on arrhythmia classification.

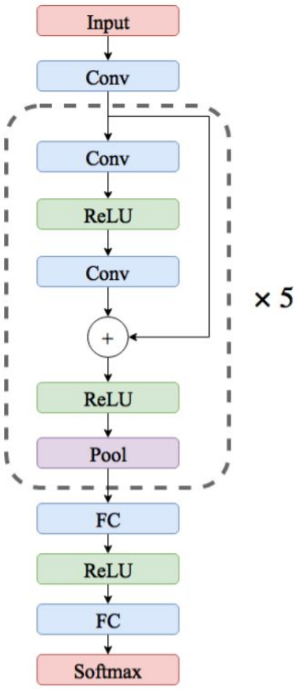

Figure1:Architecture of CNN[5].

- **Second Model: 2-D CNN[6].**

Also using CNNs but more complex one with more preprocessing like only using gray scale images:

ECG arrhythmia classification using a 2-D convolutional neural network. The proposed classifier achieved 99% average accuracy[6].

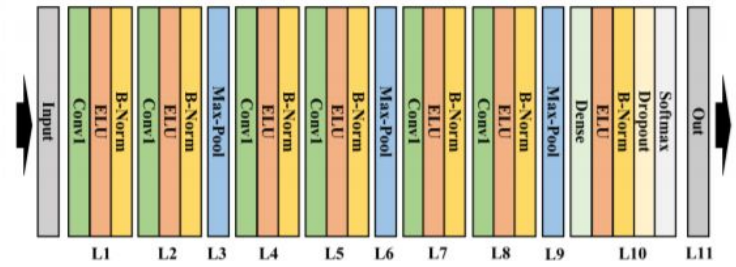

Figure 2:Architecture of 2-D CNN[6].

- **Third Model: LSTM[7].**

in this paper the proposed architecture is a type of Recurrent neural network called LSTM (long short term memory)[7].In this paper, they did something really innovative; instead of doing preprocessing to the data to extract features, they passed it to a 1D CNN model then after few layers they took that layer parameters as the features. (so basically the feature extractor was a CNN)

The proposed classifier achieved 99% average accuracy[7]

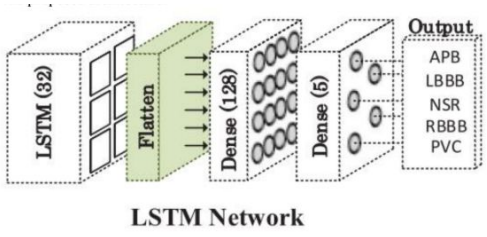

Figure 3: The architecture of the proposed classifier[7].

Data Preprocessing

We used pywt library to apply a 3-level discrete wavelet transform to extract important features of the data.

We used the wavelet transform to divide the data into two sections by filtering the data by low pass filter one time and by high pass filter another time. Then, we ignore the high-frequency part and divide the low-frequency part into two sections by the same method.

We do so because we are interested in the low-frequency part because by intuition the information is concentrated in the low-frequency as it is assumed that there are no such large variations in the data to have a high frequency. The high-frequency parts are usually noise so we want to neglect them.

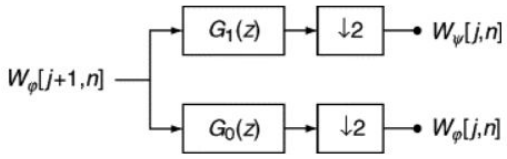

Figure (4)

Figure (4) [8] shows how the data is divided into two parts where G1 is a low pass filter and G2 is a high pass filter.

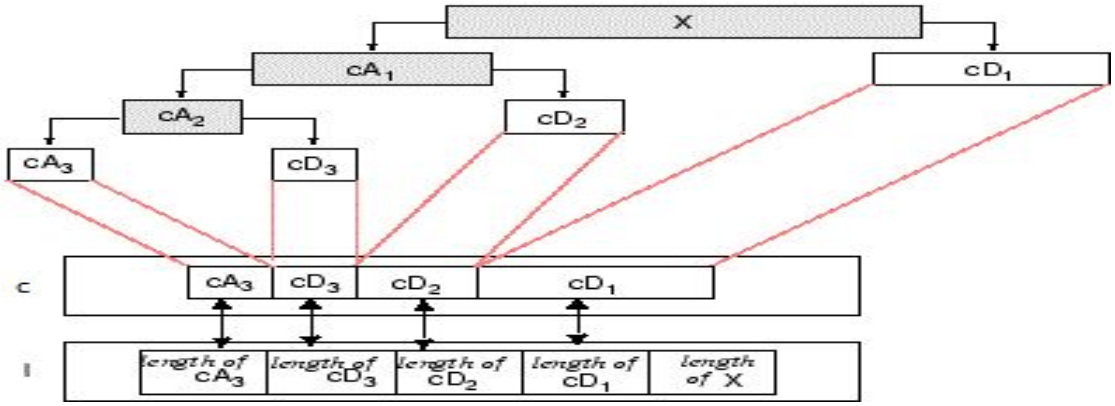

Figure (5)

Also, **Figure (5)** shows the same process but on many levels.

In our model we 3 levels so we cut down to almost ⅛ of its size.

Classification models

We used both neural networks and classical machine learning models as classifiers for the dataset. From neural networks, we used:

- LSTM
- GRU
- 1D -CNN
- Combination of CNN & GRU

And from classical machine learning models, we used:

- Support vector machine

LSTM Model:

In this model, we used two LSTM layers one with output ‘64’ and the other with ‘32’ we flatten its shape to enter a normal layer with 128 neurons and activation function “Relu”. Finally, we used a dropout layer to avoid overfitting.

```
if model_type == 'LSTM':
    model.add(LSTM(64, return_sequences=True, dropout=0.1, input_shape=(f_size, 1)))
    model.add(LSTM(32, return_sequences=True, dropout=0.1))
    model.add(Flatten())
    model.add(Dense(128, activation='relu'))
    model.add(Dropout(0.2))
```

GRU:

In this model, we used two identical GRU layers with output ‘32’ and dropout factor ‘0.5’. Then we flatten the output shape to enter a normal neural network with ‘32’ neurons then dropout to avoid overfitting. Finally, a normal layer with output ‘10’.

```
elif model_type == 'GRU':
    #model.add(Input(256,256))
    model.add(GRU(32, dropout=0.5,recurrent_dropout=0.5,return_sequences=True))
    model.add(GRU(32, dropout=0.5,recurrent_dropout=0.5, return_sequences=True))
    model.add(Flatten())
    model.add(Dense(32))
    model.add(Dropout(0.5))
    model.add(Dense(10))
```

1D-CNN:

In this model, we used two layers of 1D convolutional neural network followed by a Maxpooling with one dimensional. Both CNN layers have the same activation function which is “Relu” but different convolution length which is ‘18’ in the first and ‘9’ in the second. After those four layers, we flatten the shape and use a normal neural network “Dense” with activation function “Relu”.

```
if model_type == '1DConv':
    model.add(Conv1D(18, 4, activation='relu', input_shape=(f_size,1)))
    model.add(MaxPooling1D(2))
    model.add(Conv1D(9, 2, activation='relu'))
    model.add(MaxPooling1D(2))
    model.add(Flatten())
    model.add(Dense(15, activation='relu'))
```

Combination of CNN and GRU:

In this model, we used the two types to make use of their advantages to make a very strong model. Firstly, we used a 1D-CNN layer with length '64' and activation function "Relu". Then we dropout with factor '0.2' to avoid overfitting, then we used Maxpooling1D to reduce the spatial size of the data to reduce the number of parameters and computation in the network. Then we used the GRU layer with output '64', then a dropout layer with factor '0.1'.

```
elif model_type == 'CNN_GRU':
    model.add(Conv1D(64, 3, activation='relu'))
    model.add(SpatialDropout1D(0.2))
    model.add(MaxPooling1D(4))
    model.add(CuDNNGRU(64))
    model.add(Dropout(0.1))
```

At the end of each model, we add a normal neural network with activation function "Softmax" because its output is the probabilities and this is what we want to reduce the test error.

Support vector machine:

In this model, we used two different kernels so, we considered that we have two different models.

1. **SVM with Linear kernel:**

In this model, we used kernel “Linear” and regularization factor C = 1.

```
# training a linear SVM classifier
from sklearn.svm import SVC
svm_model_linear = SVC(kernel = 'linear', C = 1).fit(X_train, Y_train)
```

2. **SVM with “RBF” kernel:**

   In this model, we used the “RBF” kernel with regularization factor C = 1 and gamma = 0.001

```
svm_model_linear = SVC(kernel = 'rbf', C = 1, gamma = 0.001).fit(X_train, Y_train)
```

Results and Discussion

Model	Training accuracy	Test accuracy
LSTM	0.9887	0.9887
GRU	0.9334	0.9442
1D- CNN	0.8678	0.8722
CNN-GRU combination	0.9867	0.9809
SVM- Linear	-	0.96
SVM- RBF	-	0.99

It is obvious that the combination of CNN and GRU improved the performance of the model, especially if we compare it with the model of CNN only.

We also can notice that the performance of CNN-GRU model is very similar to LSTM model because they almost have the same accuracy.

The odd thing that we do not have reasons for it is how the SVM with RBF gave a better results that neural network models which are assumed to larger in capacity than SVM.

References

[1]C. Dietrich, "Decision Making: Factors that Influence Decision Making, Heuristics Used, and Decision Outcomes", *Inquiries Journal*, 2010. [Online]. Available: http://www.inquiriesjournal.com/articles/180/decision-making-factors-that-influence-decision-making-heuristics-used-and-decision-outcomes. [Accessed: 14- Dec- 2019].

[2]N. Srinivasan and R. Schilling, "Heart arrhythmia - Symptoms and causes", *Mayo Clinic*, 2018. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/heart-arrhythmia/symptoms-causes/syc-20350668. [Accessed: 15- Dec- 2019].

[3]"Arrhythmia | National Heart, Lung, and Blood Institute (NHLBI)", *Nhlbi.nih.gov*, 2019. [Online]. Available: https://www.nhlbi.nih.gov/health-topics/arrhythmia. [Accessed: 15- Dec- 2019].

[4]"Arrhythmia | National Heart, Lung, and Blood Institute (NHLBI)", *Nhlbi.nih.gov*, 2019. [Online]. Available: https://www.nhlbi.nih.gov/health-topics/arrhythmia. [Accessed: 15- Dec- 2019].

[5]M. Kachuee and S. Fazeli, "ECG Heartbeat Classification: A Deep Transferable Representation", *Arxiv.org*, 2018. [Online]. Available: https://arxiv.org/pdf/1805.00794.pdf%3Fsource%3Dpost_page----------. [Accessed: 15- Dec- 2019].

[6]T. Jun and H. Nguyen, "ECG arrhythmia classification using a 2-D convolutional neural network", *Arxiv.org*, 2018. [Online]. Available: https://arxiv.org/pdf/1804.06812.pdf. [Accessed: 15- Dec- 2019].

[7]O. Yildirim, U. Baloglu, R. Tan, E. Ciaccio and U. Acharya, "A new approach for arrhythmia classification using deep coded features and LSTM networks", *Computer Methods and Programs in Biomedicine*, vol. 176, pp. 121-133, 2019. Available: 10.1016/j.cmpb.2019.05.004.

[8] Sciencedirect.com. (2019). Wavelet Transforms - an overview | ScienceDirect Topics. [online] Available at: https://www.sciencedirect.com/topics/computer-science/wavelet-transforms