Capstone Project 2 (ECG IMAGES): Final Report

1. Problem statement

We want to predict behaviour of the EKG signals. Categories are normal beat, supraventricular ectopic beat, ventricular ectopic beat, and fusion beat.

Our client is healthcare providers and hospitals. Some patients need to be kept under observation and those patients are connected to EKG machines. It is impossible to put a person 7/24 to watch them. It will be better if we can obtain and analyze EKG data simultaneously to check if everything is in normal condition.

By analyzing EKG data we can inform healthcare providers immediately so they can take immediate action. Seconds might become important to save one's life.

2. Dataset

I will be using data from a Kaggle project. The source link:

https://www.kaggle.com/analiviafr/ecg-images

The original data is coming from Physionet's MIT-BIH Arrhythmia Dataset. There are 109445 samples in the dataset. Image Resolution is 196x128. They classified the data in five categories. Number of pictures in each category are as follow:

N (Normal beat)	: 90.589

S (Supraventricular ectopic beat)	: 2.779
V (Ventricular ectopic beat)	: 7.236
F (Fusion beat)	: 803
Q (Unknown beat)	: 8.038

I will use a convolutional neural network to solve this problem.

My deliverables will be the Jupyter notebook that includes codes and some notes. I will also report my method and findings in a final report.

3. Data Wrangling

This dataset has no separate train and test set. I downloaded the original dataset to my computer and created my own train and test sets. I chose a 0.2 split ratio. To make consistency I separated 20% of each category as a test set. This notebook is in my github. I made EDA on this notebook.

After separation I uploaded my test and train sets as a private dataset into the kaggle.

4. EDA

This data set is very clean. I plotted 5 images for each category as an example.

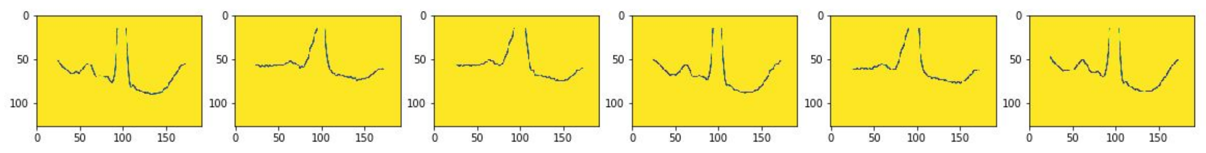

Figure 1. F Fusion beat images

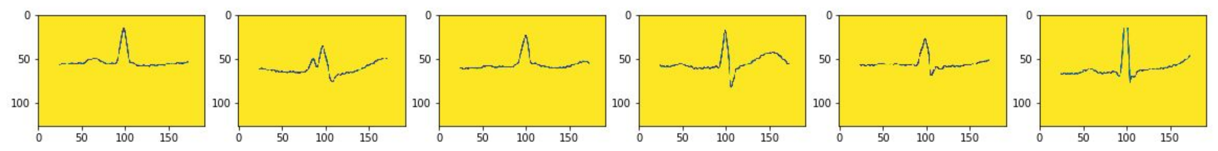
Figure 2. N Normal beat images

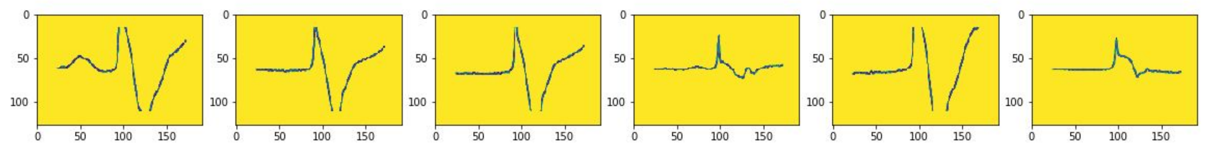
Figure 3. Q Unknown beat images

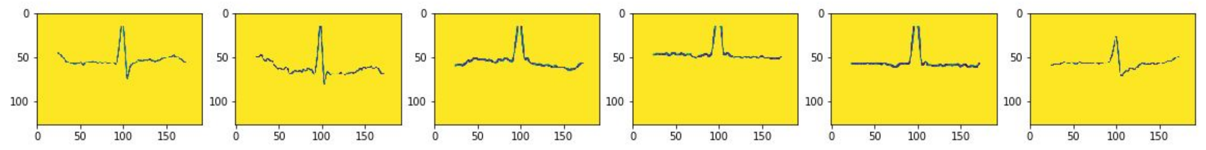
Figure 4. Supraventricular ectopic beat images

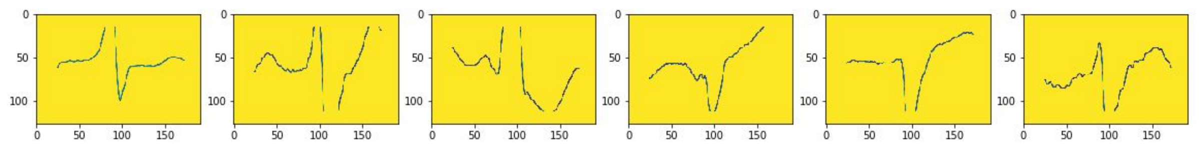
Figure 5. Ventricular ectopic beat images

All images have RGB information. I plotted one of them below.

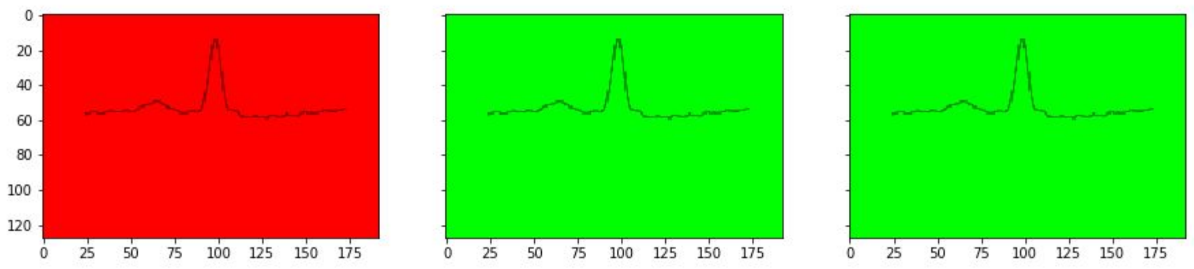

Figure 6. RGB images of one image

For training only one channel will be used.

5. Predictive Model

Predictive models are run at Kaggle servers. I created a new dataset with my train and test sets at Kaggle. The link is: https://www.kaggle.com/mmtazmuratark/ecg-data

The number of files in each directory:

data - 2

Train - 5:

F: 642

N: 72471

Q: 6430

S: 2223

V: 5789

Test - 5:

F: 161

N: 18118

Q: 1608

S: 556

V: 1447

Convolutional Neural Network is used for classification. I used the PyTorch package.

5.1. Model DataLoader

Original image resolution is 196x128 pixels. Image resolution is reduced to 120x120 pixels. I used one channel and I normalized the images with mean=0.5 and std =0.5.

The batch size is 32.

5.2. Model Architect

I tried 4 different models and compared them.

5.2.1. First Model

My first model is a very basic model composed of two convolutional layers. Architect is as follow:

1. First Convolutional Layer: 4 convolutional filters with kernel size (F*F) = 3*3,

stride (S) = 1, padding (P) = 1.

Aim of convolutional layers is to detect features in the given image. In this layer we have 3*3 filter. Input image size is 120*120 and output image size is also 120*120:

$$\text{Output} = \frac{I+F+P+P}{S} + 1 = \frac{120-3+1+1}{1} + 1 = 120$$

Our network will learn 4 filters in this process

2. Activation function: Rectified linear unit (ReLU)

After the coming layer after convolutional is the activation layer. f(x) = x^{+} = max(0,x). This relatively new activation function has strong biological and mathematical motivations and it is the most popular one nowadays. Its advantages are biological plausibility, sparse activation, better gradient propagation, efficient computation, scale-invariant. (Ref: https://en.wikipedia.org/wiki/Rectifier_(neural_networks))

3. Max pooling with kernel size 2

The problem with output of a convolutional layer is it is sensitive to locations of the features in the input image. One solution to this sensitivity is pooling. The next layer after the activation layer is pooling. Two common pooling methods are average and maximum pooling. We used max pooling. The kernel size is 2*2.

After max pooling, data size reduced from 120*120 to 60*60.

4. Second Convolutional Layer: 4 convolutional filters with kernel size=3*3, stride=1, padding=1

Number of inputs and outputs are the same. So our network will continue to learn the same filters in this layer.

5. Activation function: Rectified linear unit (ReLU)

We used the same activation function.

6. Max pooling with kernel size 2

After this Second max pooling layer input size reduced from 60*60 to 30*30.

7. Dropout(p=0.5)

Neural networks are inclined to overfit a training dataset. For this reason some nodes are dropped randomly. In this layer we assign to 0 to nodes with a probability of 0.5 to overcome overfitting.

8. Linear layer: input 4*30*30 output: 5 (Because we have 5 classes)

In this layer all coming input is converted to one single layer. We have 5 classes so this flattened single layer is connected to the last layer which has 5 nodes.

Criterion is CrossEntropyLoss. This criterion combines nn.LogSoftmax() and nn.NLLLoss() in one single class. This criterion is a good choice for us because we do classification. (Ref:https://pytorch.org/docs/master/generated/torch.nn.CrossEntropyLoss.html?highlight=crossentropyloss#torch.nn.CrossEntropyLoss)

Optimizer is stochastic gradient descent(SGD) with learning rate=0.001 and momentum=0.9

Number of epochs is 5.

These models are trained on the Kaggle gpu.

During training after 300 steps, Loss and Accuracy is printed on the screen.

Confusion matrix, recall, after each epoch. The confusion matrix after 5 epoch:

Confusion Matrix:

	ACTUAL				
PREDICTIONS	N	Q	S	V	F
	71650	546	1451	1042	373
	150	5781	3	99	3
	234	2	734	35	1
	412	100	35	4567	75
	25	1	0	46	190

class name: N, total number of class: 72471, Correctly predicted: 71650, Recall: 0.99%, Precision: 0.95%, F1-Score: 0.97%

class name: Q, total number of class: 6430, Correctly predicted: 5781, Recall: 0.90%, Precision: 0.96%, F1-Score: 0.93%

class name: S, total number of class: 2223, Correctly predicted: 734, Recall: 0.33%, Precision: 0.73%, F1-Score: 0.45%

class name: V, total number of class: 5789, Correctly predicted: 4567, Recall: 0.79%, Precision: 0.88%, F1-Score: 0.83%

class name: F, total number of class: 642, Correctly predicted: 190, Recall: 0.30%, Precision: 0.73%, F1-Score: 0.42%

Accuracy of the Trained Model on Test Data

Confusion Matrix:

```
[17959,   127,   381,   299,    96],

[   26,  1467,     0,    43,     1],

[   49,     0,   169,     5,     0],

[   80,    14,     6,  1093,    33],

[    4,     0,     0,     7,    31]],
```

class name: N, total number of class: 18118, Correctly predicted: 17959, Recall: 0.99%, Precision: 0.95%, F1-Score: 0.97%

class name: Q, total number of class: 1608, Correctly predicted: 1467, Recall: 0.91%, Precision: 0.95%, F1-Score: 0.93%

class name: S, total number of class: 556, Correctly predicted: 169, Recall: 0.30%, Precision: 0.76%, F1-Score: 0.43%

class name: V, total number of class: 1447, Correctly predicted: 1093, Recall: 0.76%, Precision: 0.89%, F1-Score: 0.82%

class name: F, total number of class: 161, Correctly predicted: 31, Recall: 0.19%, Precision: 0.74%, F1-Score: 0.31%

5.2.2. Second Model

This model is almost the same as the first model. For comparison I only changed filter numbers in the second convolutional layer(CL). Number of filters is 8 in the second CL.

Confusion Matrix:

	ACTUAL				
P R E D I C T I O N S	N	Q	S	V	F
	20164	1870	635	1620	175
	19649	1661	614	1631	165
	16852	1496	503	1275	143
	15620	1388	464	1248	155
	186	15	7	15	4

class name: N, total number of class: 72471, Correctly predicted: 20164, Recall: 0.28%, Precision: 0.82%, F1-Score: 0.42%

class name: Q, total number of class: 6430, Correctly predicted: 1661, Recall: 0.26%, Precision: 0.07%, F1-Score: 0.11%

class name: S, total number of class: 2223, Correctly predicted: 503, Recall: 0.23%, Precision: 0.02%, F1-Score: 0.04%

class name: V, total number of class: 5789, Correctly predicted: 1248, Recall: 0.22%, Precision: 0.07%, F1-Score: 0.10%

class name: F, total number of class: 642, Correctly predicted: 4, Recall: 0.01%, Precision: 0.02%, F1-Score: 0.01%

5.2.3. Third Model

In this model I introduced a more complication to the model.

9. First CL : 4 convolutional filters with kernel size = 3*3, stride = 1, padding = 1.
10. Activation function: Rectified linear unit (ReLU)
11. Max pooling with kernel size 2

12. Second L: 8 convolutional filters with kernel size = 3*3, stride = 1, padding = 1
13. Activation function: Leaky Rectified linear unit (LeakyReLU)
14. Batch Normalization
15. Max pooling with kernel size 2
16. Dropout(p=0.25)

17. Linear layer: input 8*30*30 output: 100
18. Activation function: Rectified linear unit (ReLU)
19. Dropout(p=0.5)
20. Linear layer: input 100 output: 5

Model is trained with 5 epochs.

	ACTUAL				
P R E	N	Q	S	V	F
	585	76	22	80	3

DICTIONS	24813	2232	700	2242	253
	2100	166	72	186	19
	26868	2547	850	1985	234
	18105	1409	579	1296	133

class name: N, total number of class: 18118, Correctly predicted: 17959, Recall: 0.01%, Precision: 0.76%, F1-Score: 0.02%

class name: Q, total number of class: 1608, Correctly predicted: 1467, Recall: 0.35%, Precision: 0.07%, F1-Score: 0.12%

class name: S, total number of class: 556, Correctly predicted: 169, Recall: 0.03%, Precision: 0.03%, F1-Score: 0.03%

class name: V, total number of class: 1447, Correctly predicted: 1093, Recall: 0.34%, Precision: 0.06%, F1-Score: 0.10%

class name: F, total number of class: 161, Correctly predicted: 31, Recall: 0.21%, Precision: 0.01%, F1-Score: 0.01%

5.2.4. Forth Model

1. First CL : 4 convolutional filters with kernel size = 3*3, stride = 1, padding = 1.
2. Activation function: Rectified linear unit (ReLU)
3. Max pooling with kernel size 2

4. Second L: 4 convolutional filters with kernel size = 5*5, stride = 1, padding = 2

5. Activation function: Rectified linear unit (ReLU)
6. Max pooling with kernel size 2
7. Dropout(p=0.5)

8. Linear layer: input 4*30*30 output: 5

Confusion matrix:

	ACTUAL				
PREDICTIONS	N	Q	S	V	F
	27781	2426	832	2165	248
	3994	317	133	335	35
	5705	471	164	406	51
	11495	1100	341	1044	117
	23496	2116	753	1839	191

class name: N, total number of class: 18118, Correctly predicted: 17959, Recall: 0.38%, Precision: 0.83%, F1-Score: 0.52%

class name: Q, total number of class: 1608, Correctly predicted: 1467, Recall: 0.05%, Precision: 0.07%, F1-Score: 0.06%

class name: S, total number of class: 556, Correctly predicted: 169, Recall: 0.07%, Precision: 0.02%, F1-Score: 0.04%

class name: V, total number of class: 1447, Correctly predicted: 1093, Recall: 0.18%, Precision: 0.07%, F1-Score: 0.10%

class name: F, total number of class: 161, Correctly predicted: 31, Recall: 0.30%, Precision: 0.01%, F1-Score: 0.01%

Accuracy of the Trained Forth Model on the Test Data

Confusion matrix:

```
[6892,  595,  224,  510,   43]

[1060,    69,    30,    70,   10]

[1463,  124,    52,   113,     7]

[2803,  304,    71,   263,   43]

[5900,  516,  179,   491,   58]
```

class name: N, total number of class: 18118, Correctly predicted: 6892, Recall: 0.38%, Precision: 0.83%, F1-Score: 0.52%

class name: Q, total number of class: 1608, Correctly predicted: 69, Recall: 0.04%, Precision: 0.06%, F1-Score: 0.05%

class name: S, total number of class: 556, Correctly predicted: 52, Recall: 0.09%, Precision: 0.03%, F1-Score: 0.04%

class name: V, total number of class: 1447, Correctly predicted: 263, Recall: 0.18%, Precision: 0.08%, F1-Score: 0.11%

class name: F, total number of class: 161, Correctly predicted: 58, Recall: 0.36%, Precision: 0.01%, F1-Score: 0.02%

6. Conclusion

I tested four different models. My first is the basic one and gives the best results. Reported scores for classes: N, Q, S, V, F are Recall: 0.99, 0.90, 0.33, 0.79, 0.3; Precision: 0.95, 0.96, 0.73, 0.88, 0.73; F1-Score: 0.97, 0.93, 0.45, 0.83, 0.42. Forth model reported scores for classes: N, Q, S, V, F are Recall: 0.38, 0.04, 0.09, 0.18, 0.36; Precision: 0.83, 0.06, 0.03, 0.08, 0.01; F1-Score: 0.52, 0.05, 0.04, 0.11, 0.02.

Second model and third models' Recall, Precision, and F1-Scores are less than first and forth models' scores.