PROJECT REPORT

Project Title: ECG-based Classification with Deep Neural Networks & Support Vector Machines

Date: May 4, 2025

Name: Sean Kiser (801289595)

Primary Papers:

- Authors: Giovanna Sannino and Giuseppe De Pietro
- Title: A deep learning approach for ECG-based heartbeat classification for arrhythmia detection
- Year: 2018
- Link: A deep learning approach for ECG-based heartbeat classification for arrhythmia detection

- Authors: Saira Aziz, Sajid Ahmed & Mohamed-Slim Alouini
- Title: ECG-based machine-learning algorithms for heartbeat classification
- Year: 2021
- Link: ECG-based machine-learning algorithms for heartbeat classification

1. Literature Review

1.1 A deep learning approach for ECG-based heartbeat classification for arrhythmia detection

This paper discusses the usage of analyzing Electrocardiograph data in order to categorize heart beats into normal and abnormal. This analysis is very important in the medical industry as Arrhythmia, an irregularity in the rate of rhythm of the heart beat, can be life threatening. It can cause ventricular fibrillation and tachycardia which can trigger cardiac arrest. The deep neural network model was developed using Google's tensor flow framework. They used the model to categorize heart beats into abnormal beats, the true positive class, and normal beats, the true negative class. Some drawbacks to automatic heartbeat classification is that classifiers tend to produce over-optimistic results. In clinical practice, the performance of the classifier declines because of inter-individual variation. A proposed solution was to use training and testing sets from different ECG recordings in order to account for inter-individual variation. This would allow for the classifier to have a better generalization ability. Another solution proposed was to train a global classifier and then fine tune a local classifier. The model proposed in this paper was trained using the MIT-BIH Arrhythmia Database. It consists of a collection of 24-hour ECG recordings from 47 subjects. There were four preprocessing steps discussed in this paper to improve performance. The four steps were denoising, peak detection, signal segmentation, and temporal features extraction. Also due to the dataset being imbalanced they removed 14,828 items. This imbalance caused the model to be biased towards the majority class. The final dataset consisted of 4576 data points. 2288 normal beats and 2288 abnormal beats. The actual deep neural network model was composed of seven hidden layers, with 5, 10, 30, 50, 30, 10, and 5 neurons. The activation function used for the hidden layer was ReLU. The output layer used the softmax function, and the cost function used was cross entropy. The accuracy achieved on the test set using the proposed model was 99.09%.

1.2 ECG-based machine-learning algorithms for heartbeat classification

This paper discusses many of the same things in the first paper. It discusses how Electrocardio signals are super important in the health industry as cardiovascular diseases are the leading causes of death globally. Analyzing these waves can help to diagnose many cardiac diseases. They similarly remove the noise from the signals to better improve the models performance. The first step is to account for baseline drift using DWT. Next high frequency noise is removed. The last preprocessing step is to find the peak detection from the signal. Then the paper discusses different feature extractions used for the model. The two models proposed in this paper was support vector machine classifier and multi-layer perceptron classifier. Classification is

performed by finding hyperplanes to differentiate the classes. The accuracy of the SVM model was 99.6%

2. Methods

2.1 Method A (Deep Neural Network)

For method A I loaded both the ECG5000 training and validation set. The ECG5000 dataset comes from the UCR Time Series Classification Archive, Maintained by the University of California, Riverside. The training set consists of 500 samples and the validation set consists of 1500 samples. I then split the training and validation set into its features and labels. For preprocessing I made sure to normalize the features using sklearn's StandardScaler library. I then implemented the paper's deep neural network model.

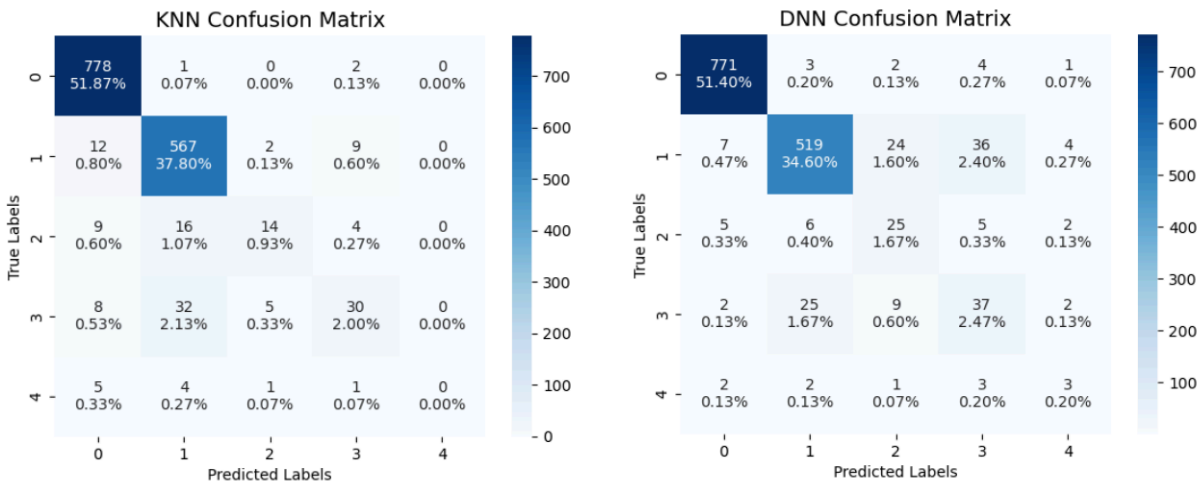
2.2 Method B (Support Vector Machine)

For method B I followed similar steps to method A. I loaded the data and then split the training and validation set into its features and labels. Then I normalized the features using sklearn's StandardScaler library. After that I implemented the paper's Support Vector Machine model.

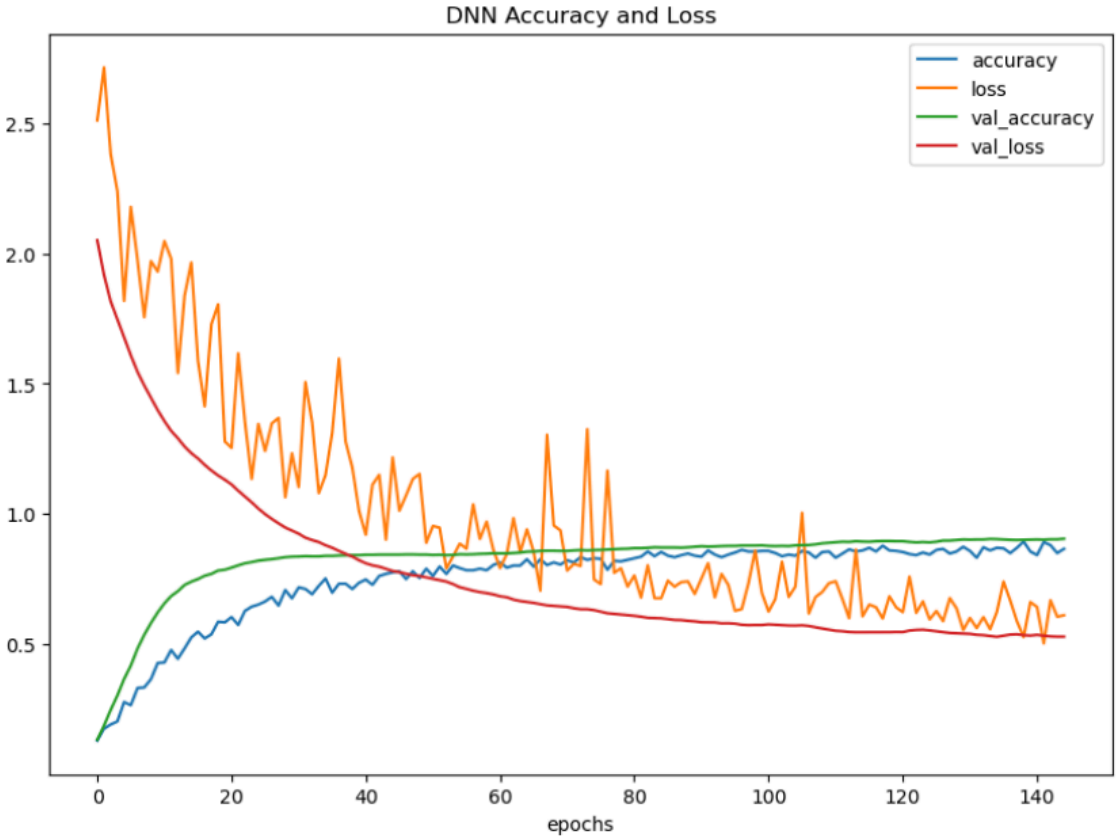
3. Experiments

3.1 Method A Results

After implementing the paper's model using the same deep neural network as described, I was not able to get the same results that they were getting. I was getting an accuracy of around 85%. This likely was because I did not use the same preprocessing steps that they did. Also it is likely that the deep neural network that they used was too powerful and was causing the model to overfit. So I decided to use fewer hidden layers as well as change the number of neurons per layer in order to improve the performance of the model. As a result I got an accuracy around 90%. While this accuracy is not nearly as good as the paper nor the KNN model, it does have a better precision and recall. The KNN model struggled to generalize the data to the validation set. It correctly classified class 2, 14 times, class 3, 30 times , and class 4, 0 times. Whereas the DNN model correctly classified class 2, 25 times, class 3, 37 times, and class 4, 3 times.



This means that the DNN does a much better job of generalizing the data. If the validation set was more balanced the DNN would likely perform much better than the KNN model. I also think that if I preprocessed the data a little better I would get results more similar to the ones in the paper. Since the paper wasn't specific on some of the hyper parameters I had to fine tune them on my own. I used a learning rate of 0.0001 and 145 epochs for my model. I also used early stopping to make sure that my model didn't overfit. Here is the accuracy and loss of my model:

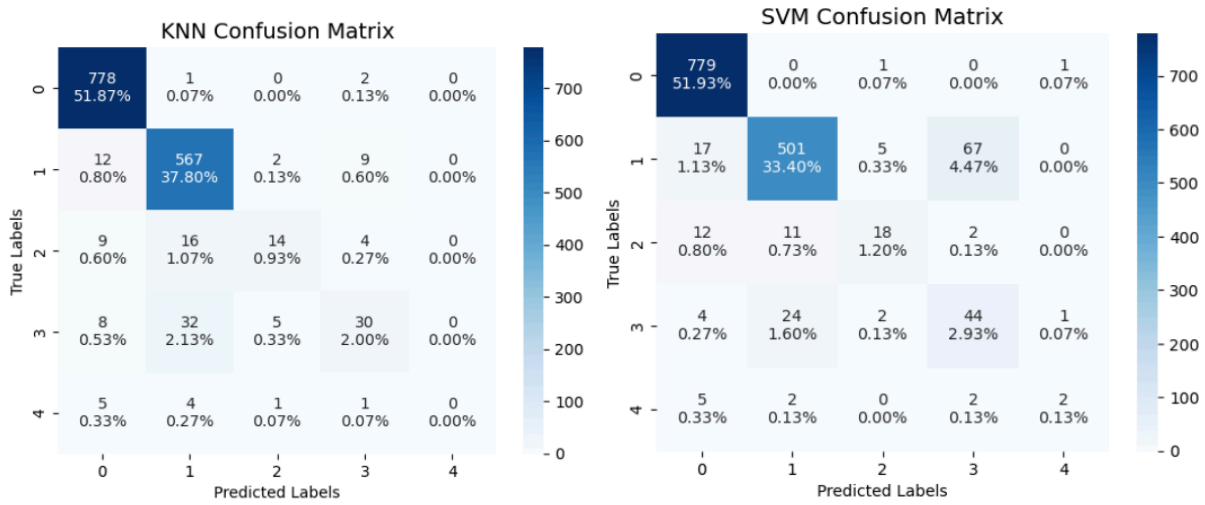


Here are the metrics of DNN model:

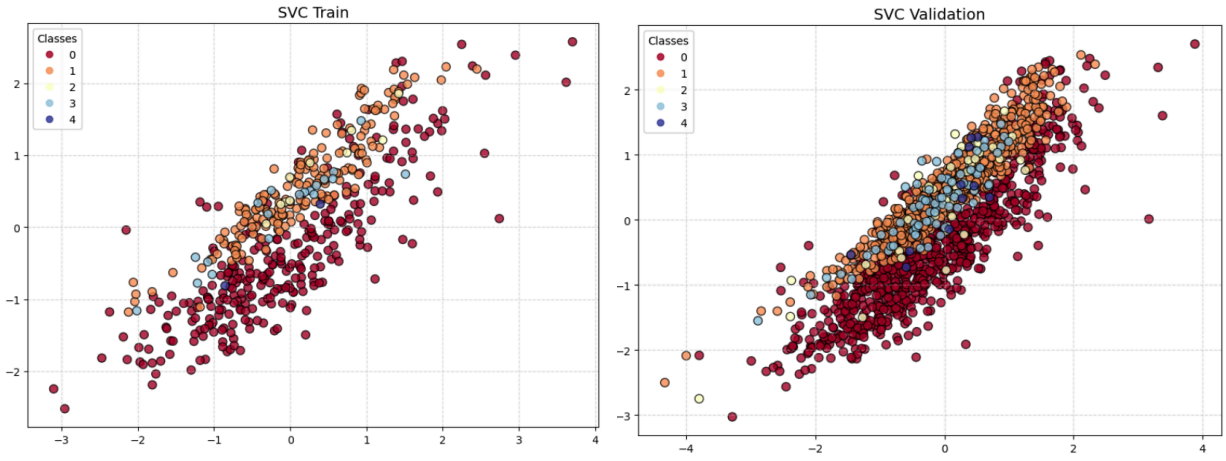
```
Accuracy: 0.9033
Precision (macro): 0.6020
Recall (macro): 0.6429
F1 Score (macro): 0.6188
AUROC (macro, ovr): 0.8901
```

3.2 Method B Results

With method B I was also not able to get a similar result as described in the paper. With that being said the support vector machine model did generalize better than the KNN model. This model also had an accuracy of around 90%, so while lower than the KNN model it did perform better at predicting classes 2, 3, and 4. As said above these models struggle to perform due to there being such a large class imbalance in both the training set and validation set. The KNN model is performing better accuracy wise because it is picking the majority class most of the time. But the trade off is that its precision and recall are much lower than the SVM model. Here is the confusion matrix of the SVM model compared to the KNN model:



Similarly, the code for this paper was not provided so I had to fine tune the parameters myself. For this model I used ‘rbf’ for the kernel, 10 for C, and 0.001 for gamma. And here is what the model looks like plotted:



Here are the metrics of the SVC model:

```
Accuracy: 0.8960
Precision (macro): 0.6919
Recall (macro): 0.6067
F1 Score (macro): 0.6230
```

4. Conclusion and Discussion

In this paper I reimplemented 2 different ECG-based classification models proposed by other researchers. While both papers did explain their methodologies the code was not included so I had to reimplement the models on my own. That meant tuning the hyper parameters for better performance, as well as totally changing the structure of the deep neural network model. Unfortunately I was not able to get the same results from the papers. This could be attributed to many things. For one my preprocessing was not as in-depth. If I wanted to improve the performance I would have to remove the noise from the data and extract some features from the data. Another reason that my model did not perform as well likely has to do with the training set and validation set being highly imbalanced. This caused the models to have a more likely chance of predicting the majority class. That being said I can definitely see an improvement in these two models compared to the KNN model. These two approaches were better at generalizing the data than the KNN model. The next steps for these two models would be to improve the preprocessing step and try and reimplement the strategies done in the paper.

Citations

G. Sannino, G. De Pietro, A deep learning approach for ECG-based heartbeat classification for arrhythmia detection,mFuture Generation Computer Systems, Volume 86, 2018, Pages 446-455, ISSN 0167-739X, https://doi.org/10.1016/j.future.2018.03.057.

(https://www.sciencedirect.com/science/article/pii/S0167739X17324548)

Aziz, S., Ahmed, S. & Alouini, MS. ECG-based machine-learning algorithms for heartbeat classification. *Sci Rep* 11, 18738 (2021). https://doi.org/10.1038/s41598-021-97118-5

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo,
Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis,
Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow,
Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia,
Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster,
Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens,
Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker,
Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas,
Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke,
Yuan Yu, and Xiaoqiang Zheng.
TensorFlow: Large-scale machine learning on heterogeneous systems,
2015. Software available from tensorflow.org.

Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

Code:
https://github.com/SeanKiser/Machine-Learning-Final