Classification of ECG Arrhythmia Using a Recursive Convolutional Neural Network Optimized with Manifold Learning Algorithms

Manogna Vemulapati
mvemulap@andrew.cmu.edu

Gregory Nielsen
ghn@andrew.cmu.edu

Abstract

The project presents a modified approach to automated classification of heart arrhythmia in an electrocardiogram recording. The model trains both a convolutional neural network and a long-short term memory network to classify the ECG recordings, then merges the resulting weights to create a new neural network with both image-recognition and context-based classification capabilities. This novel construction was found to have an average of a 7% increase in classification accuracy of test data from the baseline CNN model. Additionally, the project experiments with the use of manifold learning algorithms to improve the efficiency of training. The findings show that isomap embedding of the input data decreases accuracy slightly, but speeds up training nearly 12-fold.

1 Introduction

Cardiac arrhythmia diseases are a representative type of cardiovascular disease that are indicated by any irregular change from normal heart rhythms. They are typically diagnosed by analysis of electrocardiogram (ECG) recordings of a patient's heart rhythms over a period of time. Human diagnosticians are frequently incorrect in their classification of ECG recordings, and sometimes miss irregularities entirely. Early and accurate detection of irregularities in heart rhythms is essential for the patient's survival. Many automated decision systems have been designed to tackle the problem of classifying cardiac arrhythmia diseases based on ECG recordings, and some of these systems have had higher accuracy rates than the average diagnoses of cardiologists. The MIT-BIH database provides annotated ECG recordings, and is one of the most prominently used databases for this line of research. For each recording in the dataset, multiple cardiologists have annotated each beat (a peak in the image), and also have provided the class of the arrhythmia in the recording as a whole. The goal of an automated system is to take in the annotated recording as input, process the data into features as an image, and to learn a model that can predict one of 7 classes of heart arrhythmia (including "normal" or no arrhythmia) in a new ECG recording. The six classes of irregular heartbeat that this project's model learns to classify are: Atrial premature beat, left bundle branch block, right bundle branch block, paced beat, premature ventricular contraction, and ventricular escape.

2 Background

The baseline model that we constructed as a point of reference is a convolutional neural network, with design choices influenced by preexisting work on the same dataset [2]. Our final architecture is illustrated in Figure 1.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

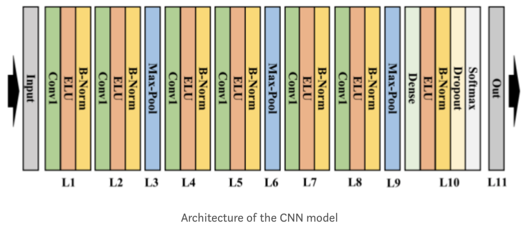

Figure 1: Convolutional neural net architecture

Below we provide the training accuracy over 10,000 epochs for the baseline model, as well as the model's accuracy for test data in the first 100 epochs. We observed that training loss converged to nearly 0 cross entropy, but the best performance achieved by the model was approximately 87% accuracy on test data.

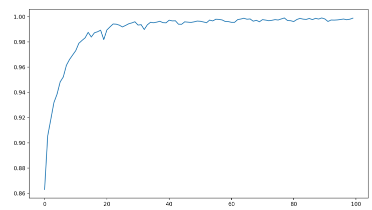

Figure 2: Training accuracy for baseline model

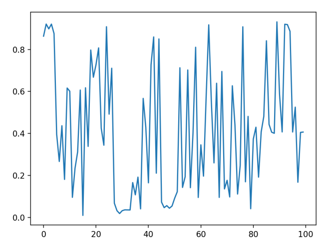

Figure 3: Test accuracy over first 100 epochs

Another important finding in the midway report was that it took thousands of epochs for the model to converge to its optimal performance. Figure 3 shows the model's erratic performance over the first 100 epochs on test data. The baseline model falls short both in its accuracy of predictions on test data, and in how inefficient it is to train the model.

3 Related Work

Given the MIT-BIH recordings as inputs, the arrhythmia classification problem is reduced to an image classification problem, which can be modeled effectively by convolutional neural networks [1]. Many research papers have experimented with different CNN architectures to classify ECG recording images, and but many of these models suffer from overfitting and prohibitively long training times. Another approach seen in some research is that the inputs can be interpreted as a sequence of different types of beats, in which earlier peaks in the recording provide context for features that appear later. This type of input data can be effectively modeled by a recursive neural network, such as a Long-Short Term Memory network (LSTM) [4]. Both of these models have had approximately the same level of success (by fine-tuning hyperparameters, both models can attain similar performance). They typically converge to a test accuracy of under 90%, even after thousands of epochs of training. Additionally, there has been research on the use of correlation feature selection to improve the speed of training the neural network and the accuracy of the final model [3]. While there is research that has shown correlation feature selection can improve efficiency of training, no existing research has

attempted to directly transform the dimensionality of the data.
In summary, the current field of research for classifying heart arrhythmia in ECG readings has shown that an ideal model would be able to: (1) efficiently identify significant peaks in the ECG image, (2) be able to connect the significance of these features over several time steps, and (3) ignore noisy features in the data, like outlier peaks.

4 Methods

4.1 Neural network architecture

To improve upon the baseline, our final model attempts to learn new features of the data using a Long-Short Term Memory Network (LSTM), in addition to the convolutional neural network. Inspiration for the use of both kinds of networks in a single algorithm was drawn from research done for the problem of blood cell image classification [6].

- The raw input from the MIT-BIH Arrhythmia database are ECG recording files, which is preprocessed using the Python "biosppy" package. We first segment the recording into feature signals. The correct labels (arrhythmia classes) are extracted from the recording annotations using the "wfdb" package.
- The segmented recordings are then parsed into numpy array representations of 128x128 images with a single input channel (the images are black and white). These are passed as input to the convolutional neural network with the baseline architecture. The convolutional neural network is trained on a set of 12,000 inputs.
- We then flatten the image representations of the data points into single-dimensional arrays, giving each data sample 256 features, and pass this format of the input to a long-short term memory network. The LSTM is trained on the same inputs as the CNN.
- Both networks use a softmax activation at the output layer to output the predicted class of the input data, and are independently trained to minimize cross-entropy loss over 200 epochs.
- We then “merge” the learned weights of the two networks. We take the outputs of the layers in the learned model before the softmax activation and combine them with an element wise multiplication. This result is then passed through an affine layer, followed by a softmax activation to output the final classification. These two layers of the model are also trained for 200 epochs by minimizing cross entropy loss.

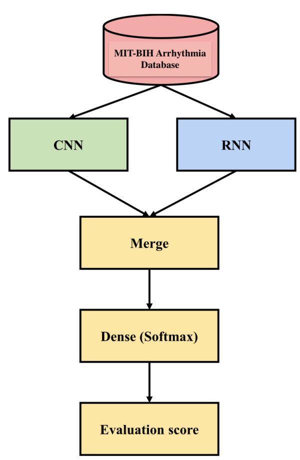

Figure 4: Architecture of final model

4.2 Dimensionality reduction

To make training the model more efficient and potentially reduce noise in the data, we tested the use of dimensionality reduction techniques on the input. We used three manifold learning algorithms implemented by the “scikit-learn” package: (1) Isomap Embedding, (2) Locally Linear Embedding, and (3) Spectral Embedding. These three algorithms were chosen to account for different possible shapes of the data’s features in the hyperplane. We used the algorithms to project the 128x128-dimensional input onto a 256-dimensional hyperplane — that is, each algorithm selected 256 components of the original 16384 features. An example is shown below for an Isomap embedding of a sample signal.

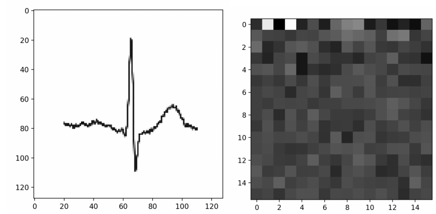

Figure 5: Result of Isomap embedding on sample data point

We see the white and black (lightest and darkest) pixels in the embedded image capture the most significant features of the original image, which are the irregular peaks.

4.3 Experimental procedure

First, the CNN and RNN models were trained independently on the full-sized training data set (before dimensionality reduction). Then, the best dimensionality reduction algorithm was determined by plotting the training and validation accuracies of training the model on each of the three transformed datasets. The smaller data was then used to train and test the model again using 5-fold cross-validation. The training set is initially 15,000 randomly selected recordings from the database. In each batch of training, the data are shuffled and 12,000 samples were selected to train the model. The remaining 3,000 samples were then used as validation data.

5 Results

The most meaningful metrics from our experiments are the training and validation classification accuracies, which we plot with respect to the epoch of training. First, we show the classification accuracy for our model trained on the full-sized dataset.

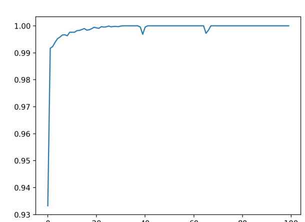

Figure 6: Training accuracy

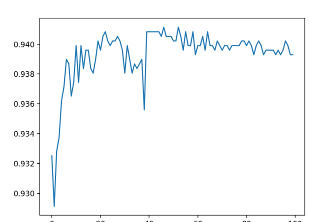

Figure 7: Validation accuracy

We see that our model converges to nearly perfect classification of the training data in around 50 epochs, and converges to about 95% validation accuracy within 100 epochs of training.
Below, we provide the same metrics for the model trained on each of the transformed datasets (after application of each of the manifold learning algorithms).

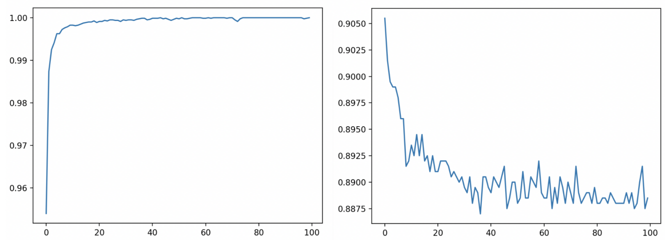

Figure 8: Training and validation accuracy after locally linear embedding

We see above that training the model on LLE-transformed data causes it to overfit much faster. The model converges to approximately 88% validation accuracy after 100 epochs. While the model did have much faster training time per epoch, the test accuracy is much worse than that of the model trained on the full-size data.

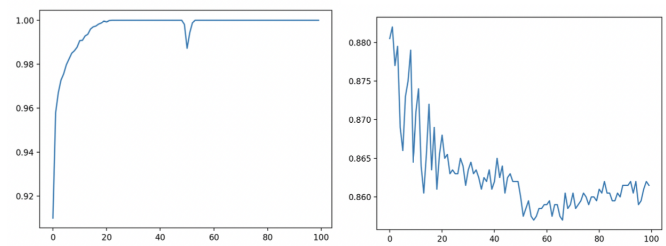

Figure 9: Training and validation accuracy after spectral embedding

We see above that training the model on the data after performing spectral embedding has somewhat unpredictable results. It may not retained necessary features in some of the input data, causing the model to initially have oscillating results on the validation data. The model converges to approximately 86% validation accuracy after 100 epochs.

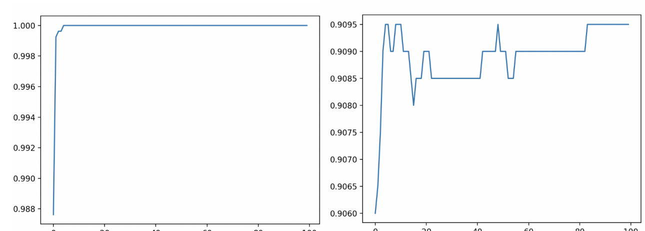

Figure 10: Training and validation accuracy after isomap embedding

We found that isomap embedding reduced the dimensionality of the data in a way that still retained the most important features, and allowed for the best performance of the model after training for 100 epochs. The model converges to approximately 90% validation accuracy, and does not overfit within this timeframe. All of the above manifold learning algorithms reduced training time from about 2 minutes per epoch to about 10 seconds per epoch.
Because the use of isomap embedding for dimensionality reduction had the best validation performance above, we used the data transformed through isomap to train and test the model with 5-fold cross validation. For each batch, the model attained nearly perfect classification accuracy on test data, so we will only report the validation accuracies. The results of this procedure are summarized in the table below.

Table 1: 5-Fold cross-validation results

Batch	Validation Accuracy
1	0.9336
2	0.9419
3	0.9245
4	0.8714
5	0.9207

The best test accuracy achieved by the model after dimensionality reduction with isomap embedding was 94%, and from the cross-validation process we see that the average effectiveness of our model is 92.242% classification accuracy. We see that this is a distinctive increase in accuracy from the baseline model, and additionally the model is far more efficient, as training time per epoch is nearly 12 times as fast as the baseline.

6 Discussion and Analysis

The performance of the CNN and RNN models alone tend to plateau after around 200 epochs of training, and the optimal accuracy that either of them can converge to on validation data was consistently under 90% (regardless of finetuning hyperparameters). This observation indicates that neither model is independently capable of learning all of the features of the ECG recording needed to make a diagnosis. We observe that while each of these networks have reasonably good classification capabilities for this dataset, the additional layer that merges the two models' learned weights is capable of crossing the 90% accuracy threshhold. In fact, the validation accuracy plotted in Figure 7 starts out at above 90%, because the model begins learning with weights that result from the two pre-trained networks. This shows that the model successfully combined the context-learning capabilities of the LSTM with the image-recognition capabilities of the CNN to make it more effective overall.

The results of the manifold learning algorithms were more inconsistent. Both LLE and spectral embedding were found to be ineffective in identifying the key components of the input data. This probably contributed to the early overfitting trend we observed in the models trained on the two resulting datasets. While we found isomap embedding to be fairly effective, it still caused a slight decrease in the model's performance on test data, when compared to the model trained on the full-sized data. This leads us to believe that the raw input data isn't very noisy – that is, most of the features are probably significant, and do need to be learned by the model for it to achieve optimal performance. However, the tradeoff we face here is the learning time. While we see a marginal loss in classification accuracy when we transform the data with isomap embedding, we see an exponential increase in the efficiency of training the model.

The architecture of this model does face some limitations. There is quite a bit of overhead in the process of training two separate neural networks – since the final output layer depends on the results of the two trained networks, the top 2 layers of the network can only be trained afterwards. The model could be improved by giving the convolutional neural network itself a recursive structure, as suggested by [5]. We could improve both efficiency and accuracy by incorporating the LSTM memory cell state structure into the CNN. It would be more efficient because having only one network to train would mean less overhead.

The key observations we take away from our results are:

- The ECG recordings in the MIT-BIH Arrhythmia database do not necessarily suffer from the curse of dimensionality, but the features of this dataset can still be reduced by using isomap embedding to isolate key components.
- Both the use of convolution kernels and the use of memory cell states are useful in the construction of a deep learning model to classify this dataset, and combining these two mechanisms makes a more powerful network.

- There exists an accuracy-speed tradeoff in this model, because to attain signficantly faster training time through dimensionality reduction, the model has to sacrifice classification accuracy.

References

[1] Ng, Andrew, et al. “Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks.” Stanford ML Group. 2017 (July 6). https://arxiv.org/pdf/1707.01836.pdf.

[2] Singh, Ankur. “ECG arrhythmia classification using a 2-D convolutional neural network.” https://medium.com/datadriveninvestor/ecg-arrhythmia-classification-using-a-2-d-convolutional-neural-network-33aa586bad67. 2018 (July 3).

[3] Mitra, Malay, et al. “Cardiac Arrhythmia Classification Using Neural Networks with Selected Features.” Procedia Technology, Vol. 10, pages 76-84. 2013.

[4] Singh, Shraddha, et al. “Classification of ECG Arrhythmia using Recurrent Neural Networks.” Procedia Computer Science, Vol. 132, pages 1290-1297. 2018.

[5] Kim, Jiwon, et al. Deeply-Recursive Convolutional Network for Image Super-Resolution. 2016 (Nov 11). arxiv.org/pdf/1511.04491.pdf.

[6] Liang, Gaobo, et al. “Combining Convolutional Neural Network with Recursive Neural Network for Blood Cell Image Classification.” IEEE Access, Vol. 6, pages 36188-36197. 2018 (July 3).