Big Data Project #4: Transforming How We Diagnose Heart Disease

by Svea Joaquino & Andrea Seet

Summary of Findings

In order to improve the automatic measurement of the images of the cardiac MRIs, we preprocessed our data into training and testing sets to receive validation scores. This determines the accuracy that our training test has in order to predict the MRIs. We also determined from our live loss plot that lower values were outputs for both loss and validation loss. We also ran a random forest test, which addresses the balance of overfitting and underfitting to measure the performance of our tests. From these extremely small values, we have evidence to state that our tests have produced a low mean absolute error, mean squared error, and root mean squared error.

Problem

Declining cardiac function is a key indicator of heart disease. This can be measured by end-systolic and end-diastolic volumes, both of which can derive the ejection fraction. Magnetic Resonance Imaging is considered the gold standard to accurately test the heart's squeezing ability. The process of analyzing the MRIs is slow and finding a faster way to enhance the doctor's ability to diagnose heart conditions early will advance heart disease treatment. The problem is to automatically measure end-systolic and end-diastolic volumes in cardiac MRIs from the over 1000 patients in the data set. We will ask the questions, what are our validation and loss scores from our training set and what are the parameters that perform the best from running random forests within our data set?

Processing the Data

First, we loaded labels for the training dataset and since the images in the dataset are different sizes and resolutions, we wanted to process the images to be the same size. With the dicom format, we extracted the compressed files from the zip and made the images uniform. We dealt with intervening directories and added relevant directories into a list called slices. The train data only contains the final determined volumes of the left ventricle. Then, we measured the distance between slices. These slices are a series that cover the entire heart and are from the SAX views or multiple short-axis views whose planes are perpendicular to the long axis of the left ventricle. To simplify the data, we are only utilizing the SAX images and have resized and taken the average of these images per patient.

Method

To analyze this processed data, we began by feeding that data into a deep neural network model, a Convolutional Neural Network (CNN). A CNN is a deep learning algorithm which takes an input image and assigns weights and biases to the image. Because we only processed the "Train" data, we split the data into a training and testing subset and ran these into the model. Given that end-systolic and end-diastolic volumes are necessary to predict the ejection fraction, we ran the CNN model for systolic and diastolic values. Following this model, we ran a live loss training plot to get a better idea of loss, accuracy, validation loss, validation accuracy being updated at each epoch. Lastly, to strengthen our prediction model, we created a random forest that joins different types of modeling techniques to form a prediction model of multiple decision trees.

Results

Diastolic

```
In [24]: model = tflearn.DNN(network, tensorboard_verbose=0)
         model.fit({'input': X_train}, {'target': y_train}, n_epoch=1,
                   validation_set=({'input': X_test}, {'target': y_train}),
                   snapshot_step=100, show_metric=True, run_id='convnet')

Training Step: 5  | total loss: 6.15769 | time: 1.984s
| Adam | epoch: 001 | loss: 6.15769 - acc: 0.0144 -- iter: 320/330
Training Step: 6  | total loss: 6.07829 | time: 3.117s
| Adam | epoch: 001 | loss: 6.07829 - acc: 0.0052 | val_loss: 5.49528 - val_acc: 0.0184 -- iter: 330/330
```

Systolic

```
In [17]: model = tflearn.DNN(network, tensorboard_verbose=0)
         model.fit({'input': X_train}, {'target': y_train}, n_epoch=1,
                   validation_set=({'input': X_test}, {'target': y_train}),
                   snapshot_step=100, show_metric=True, run_id='convnet')

Training Step: 5  | total loss: 5.90758 | time: 2.212s
| Adam | epoch: 001 | loss: 5.90758 - acc: 0.0036 -- iter: 320/330
Training Step: 6  | total loss: 5.60530 | time: 3.322s
| Adam | epoch: 001 | loss: 5.60530 - acc: 0.0314 | val_loss: 5.23992 - val_acc: 0.0245 -- iter: 330/330
--
```

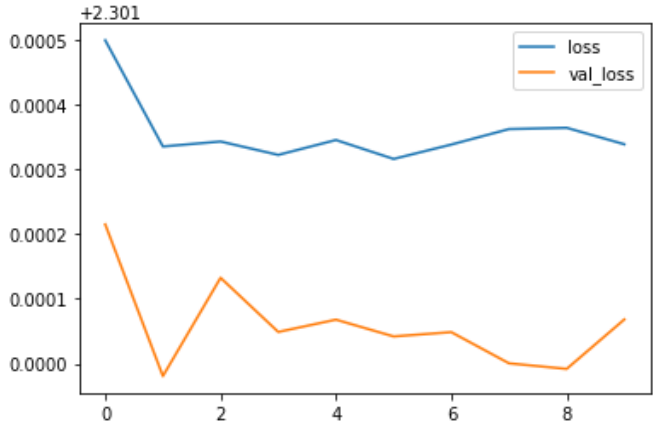
Validation Loss and Accuracy

When we ran the training model for the systolic values, we received a total loss number of 5.90758 for training step 5 and the accuracy was 0.0036. For the diastolic values, we received a total loss number of 6.15769 and the accuracy was .0144. In training step 6, for systolic, we received a total loss of 5.60530 and the total accuracy was 0.0314. The total validation loss for diastolic was 6.07829 and the accuracy was 0.0314. A small number of patients could be responsible for the biggest loss in total performance. The lower the loss, the better a model (unless the model has over-fitted to the training data). Accuracy is a summation of the errors made in the training and validation sets. The main objective in a learning model is to minimize the loss function's value in terms of the model's parameters by changing the weight vector values through different optimization methods.

The loss value implies how well or poorly a certain model behaves after each iteration of optimization so when comparing training step 5 to training step 6, we received a lower validation loss number. This means that with more steps and feeding more data into the training set, we have a lower loss value. We ideally expect the reduction of loss after each iteration, which we observed when inputting our training set. However, with reducing the loss value, we could possibly encounter the problem of overfitting in which the model memorizes the training examples that were fed in and results in becoming ineffective for the test set.

Live Loss Plot

When we ran our live loss plot, we received a loss that started at 2.3015, which dropped and leveled off to around 2.3013. For the validation loss, we start off with a value of 2.3012 which drops as well, but increases and wavers in increases and decreases. However, it does not ultimately reach its starting point of 2.3012. Using this type of plot allows us to see multiple training runs and how the loss and validation loss changes after being updated after each epoch. In this static plot, we do not observe how training our parameters changes, but in the process of running the plot, we observed that it grows step by step and outputs this plot at the end.



Random Forests

For regression problems, random forests evaluate the performance of the algorithm using the mean absolute error, mean squared error, and root mean squared error. When we concatenated the training and testing data sets, we received a mean absolute error of 0.00283, mean squared error of 1.369e-05, and a root mean squared error or 0.0037. With these small values, the error values should decrease with the increase in the number of estimators. For the mean absolute error, we are measuring the absolute value of the difference between the forecasted value and the actual value. It tells us how big of an error we can expect when we are running the validation test as a performance and quality measure.

Discussion

After finding the total loss, validation loss, and validation accuracy scores, we observed that there were decreases in the losses and accuracy values from training step 5 and training step 6 for both diastolic and systolic. By feeding more data into our training steps, we are able to predict the cardiac MRI images better. From both the live loss plot and the random forest evaluation, we find that our errors are decreasing and are of low numbers, which potentially can be effective in transforming how we diagnose heart disease to speed up the process to automatically measure systolic and diastolic volumes for patients.