Northeastern University
EECE – Pattern Recognition and Machine Learning
April 2018, Asha Chen-Phang, Emily Dutile, Nate Otenti, Tristan Sweeney

Classifying High-Resolution Brain Scans

1 Prediction Program

Various models were training and evaluated in order to receive the highest accuracy of predictions on the high resolution brain scans[1]. Linear SVM, Decision Tree, Random Forest, and Ada Boost were all analyzed, but the model that gave us the best results was a Decision Tree model.

1.1 Data Analysis

We first started with visualizing the data for our analysis by using Jupyter Notebooks, numpy, and matplotlib. To do so, we took one of the csv files, found the rows of the file that is labeled as 1, found the rows of the file that has 0 as the label, and generated a file with 100 "foreground" images and 100 "background" images to visualize one of the images. The output of the following below can be viewed in the Appendices.

```
#!/usr/bin/python3
import matplotlib.pyplot as plt
import numpy as np
data = np.loadtxt("../../input/10true10false.txt", dtype=int, delimiter=",")
input_data=np.array([(l[:-1]).reshape(21,21,7, order='F') for l in data]) # 'F' for Fortran like indexing
input_labels=np.array([(l[-1]) for l in data])
img_index = 0 # select image number '0'
for z_dim in range(7): # prints all slices of the z axis
    plt.figure()
    print("Center Pixel value: ",input_data[img_index][10,10,z_dim])
    plt.imshow(input_data[img_index][:,:,z_dim], cmap='gray', interpolation='nearest')
```

1.2 Feature Extraction

From here image pre-processing and feature extraction was performed as seen in Appendix B and C, showing the comparison of images of foreground sample and background sample. After analyzing the images, histogram and statistics on all of planes(xy, xz, yz) slices for the two images, we were able to determine important features along with a threshold to tell the difference between a foreground and background image. This lead us to the conclusion of not training complicated models on the full image but to use the following features and consequently the feature vector:

- center pixel value
- average of a window of pixels around the center pixel
- number of pixels with intensity greater than a threshold

1.3 Methodology

From our data analysis we performed a classification comparison of several classifiers in a Jupyter Notebook on a smaller sample of the dataset using scikit-learn to better visualize the data and the nature of the decision boundaries of different classifiers with respect to the dataset. The feature vector was made up of 10 features and the label.

[1] https://drive.google.com/drive/u/0/folders/1EJBgJFmp-FQf2czw9LGImoOhEO2OvOoo

```
# Feature Vector
CenterPixel,...
xyImgSliceMean,xyFFTSliceMean,xyCntOverThres,...
xzImgSliceMean,xzFFTSliceMean,xzCntOverThres,...
yzImgSliceMean,yzFFTSliceMean,yzCntOverThres,...
label
```

In Appendix D, the classifier comparison can be viewed along with the classification accuracy on the test set in the lower right. The test set contained a balanced scenario with 100/100 samples, proving that the Decision Tree had the best accuracy with a 60/40 train/test sample using the feature vector without parameter tuning.

1.3 Design

Model	Run Time (mins)	Accuracy
SVM	0	0
Decision Tree	0	0

1.3.1 Preprocessing

1.4 Evaluation

TODO - show accuracy numbers for different parameter settings we explored.

1.5 Testing

2 Distributed Comutation

TODO - show steps of distributed computation. How data and computation repartitioned and assigned to workers.

TODO - num tasks created during each stage of the model training process

TODO - data being shuffled

TODO - num iterations executed during model training

TODO - show how changes of parameters controlling partitioning affect the running time (i.e: for bagging, was it better to partition the model file and test data or both)

3 Performance

The running time and speed up results for the model training and prediction phase are show below:

3.1 Model Training

Machines	Running Time (mins)
11	0
20	0

Speed up = running time on 11 machines / running time on 20 machines

$$= 0/0$$

$$= 0 \quad (1)$$

3.1 Model Prediction

Machines	Running Time (mins)
11	0
20	0

Speed up = running time on 11 machines / running time on 20 machines

$= 0/0$

$$= 0 \quad (2)$$

References

[1] Map-Reduce for Machine Learning on Multicore,
http://www.andrewng.org/portfolio/map-reduce-for-machine-learning-on-multicore/

[2] Cross Validator Model,
https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-mllib/spark-mllib-CrossValidato

[3] Classifier Comparison,
http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

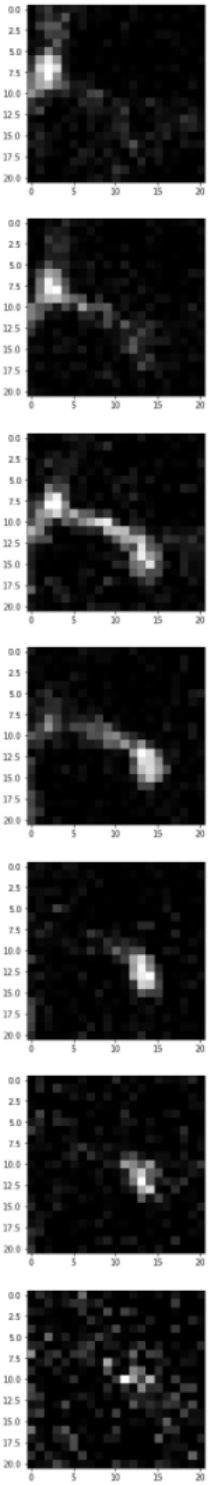
[4] Classification by using Ensembles of Classifiers,
https://grzegorzgajda.gitbooks.io/spark-examples/content/classification/rf-classification.html

[5] MLlib: Scalable Machine Learning on Spark
https://web.stanford.edu/ rezab/sparkworkshop/slides/xiangrui.pdf

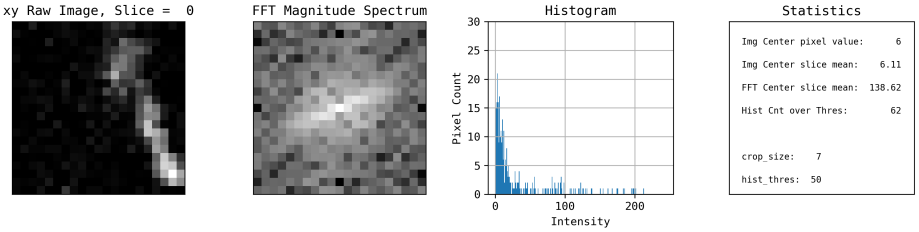
[6] Ensembles - RDD-based API
https://spark.apache.org/docs/latest/mllib-ensembles.html

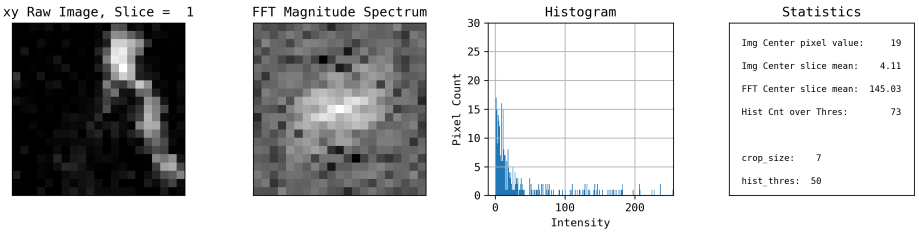
Appendices

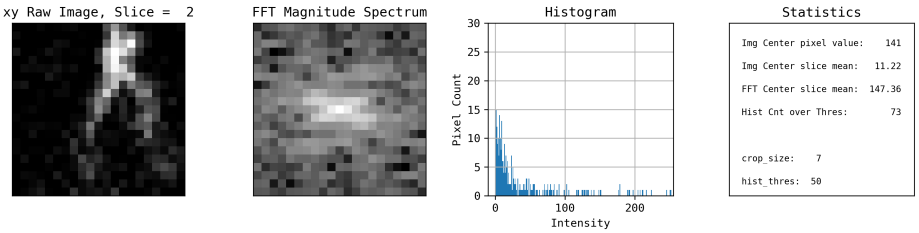
A Visualizing the Input

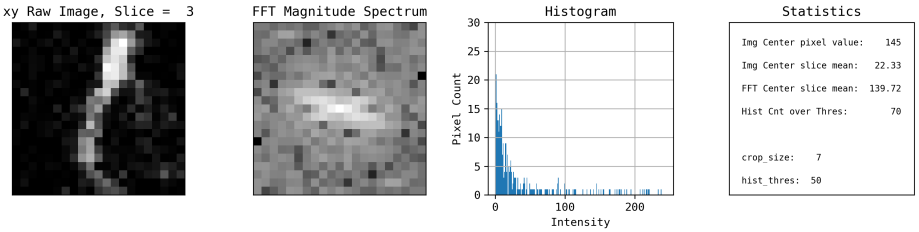


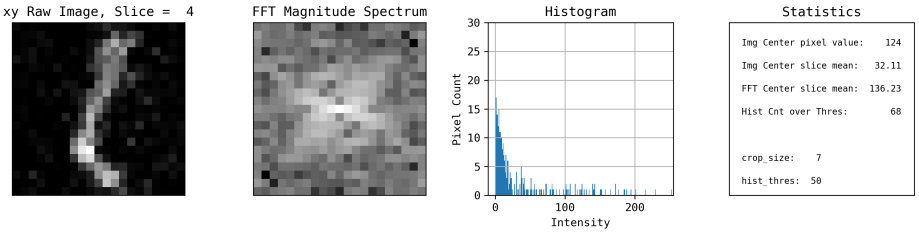
B Foreground Analysis

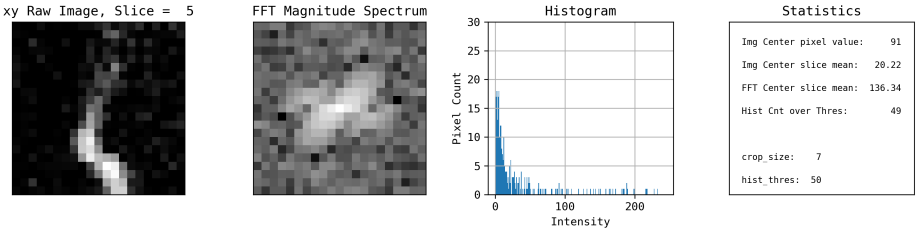


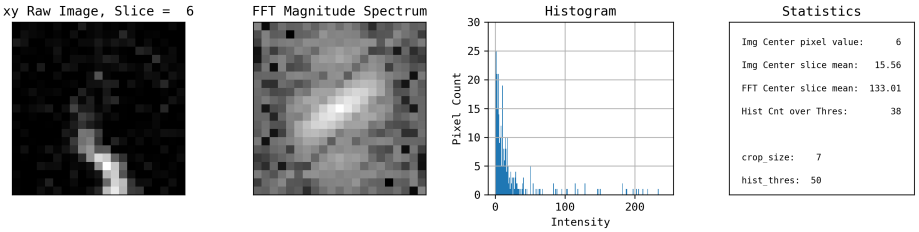




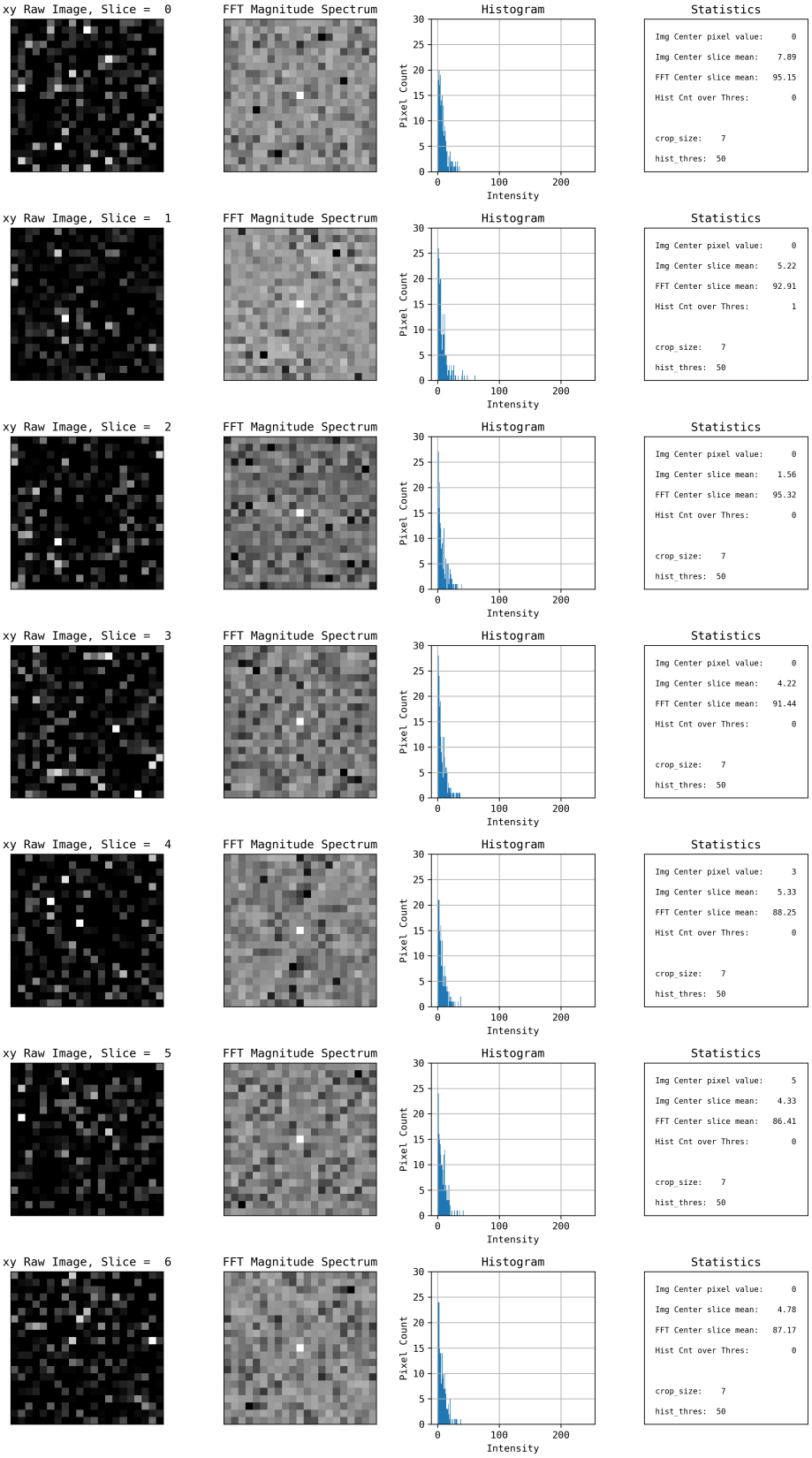








C Background Analysis



D Classification Comparison Test