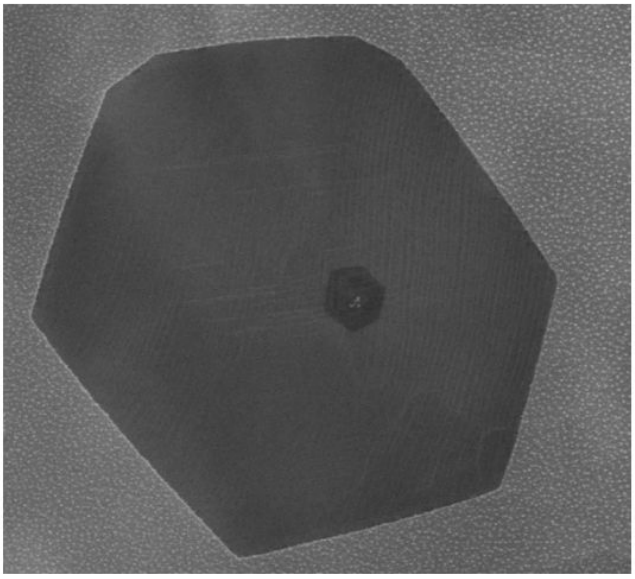
ME498 Final Report

Topic 2: Automatic Identification of Graphene



Team 1: Dingyu Peng, Liu Hong, Thomas Phelps, Tyler Tran

I. Introduction

Graphene synthesis is a very promising field. Graphene is the strongest material ever tested, one of the highest thermal conductivities, and is nearly transparent. Graphene fabrication is very involved, so there is high demand for finding a way to identify graphene in electron microscope images.

Our objective is to detect graphene surface coverage in SEM (scanning electron microscope) images to quantify the efficacy of different synthesis recipes. We will use the tool provided by the nanoMFG node to generate training data by manually determine the areas covered by graphene. Eventually, we will develop a machine learning algorithm to automatically scan SEM images to determine the locations and features of graphene.

II. Proposed Solution

Our solution is to use the provided package software to manually mask the graphene area then read the binary image to be processed as our test labels. We will also read the raw image to be processed for feature generation.

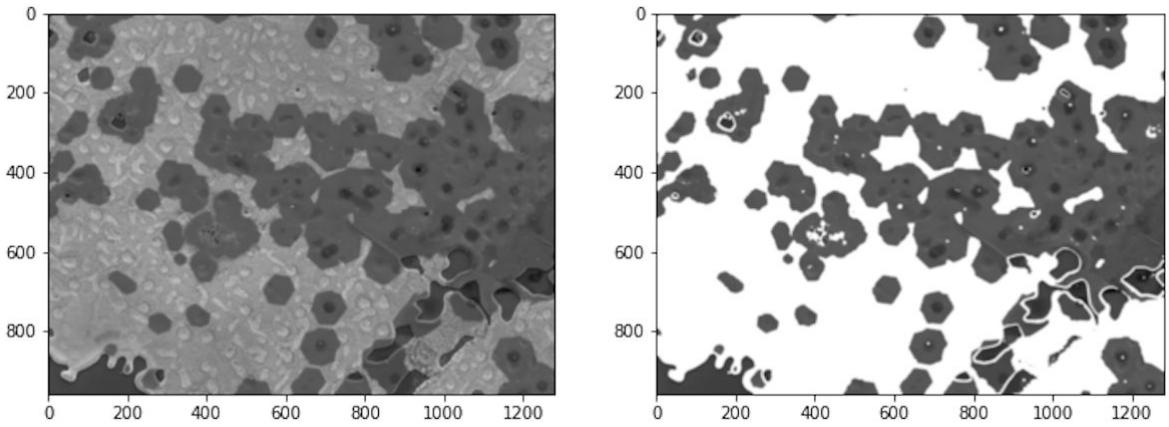


Figure 1: Raw image and masked image

For feature generation, we use an 8x8 sliding window to slide by 4 pixels at a time to generate numerous 8x8 patches for the raw image as matrices. We then categorize the color intensities represented by numbers in the matrices into 127 bins as our 127 features. Each 8x8 patch makes up one layer of training data so one .TIF image provides us with large amounts of training data.

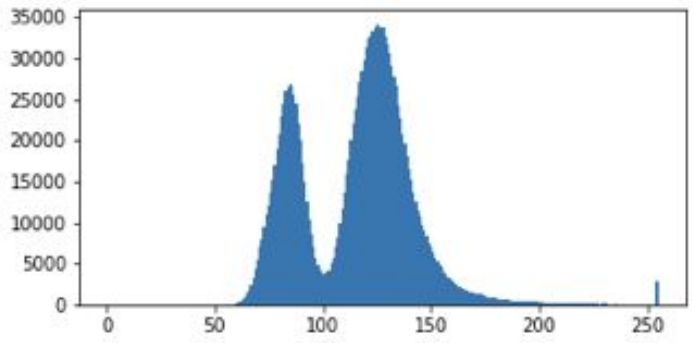


Figure 2: Histogram for a single patch

The histogram in Figure 2 shows two peaks, one corresponding to the graphene sections of the image and the other corresponds to the rest. Most of the histograms are able to seperate the peaks fairly easily. For every two steps in intensity we fit the data into one feature, so for all 0-255 intensity levels we obtain 127 features.

For label generation, we use the same sliding window algorithm to generate an 8x8 binary matrix for each patch. We then perform max-pooling for each patch, obtaining a zero or one label for each patch, 0 representing graphene, 1 representing background.

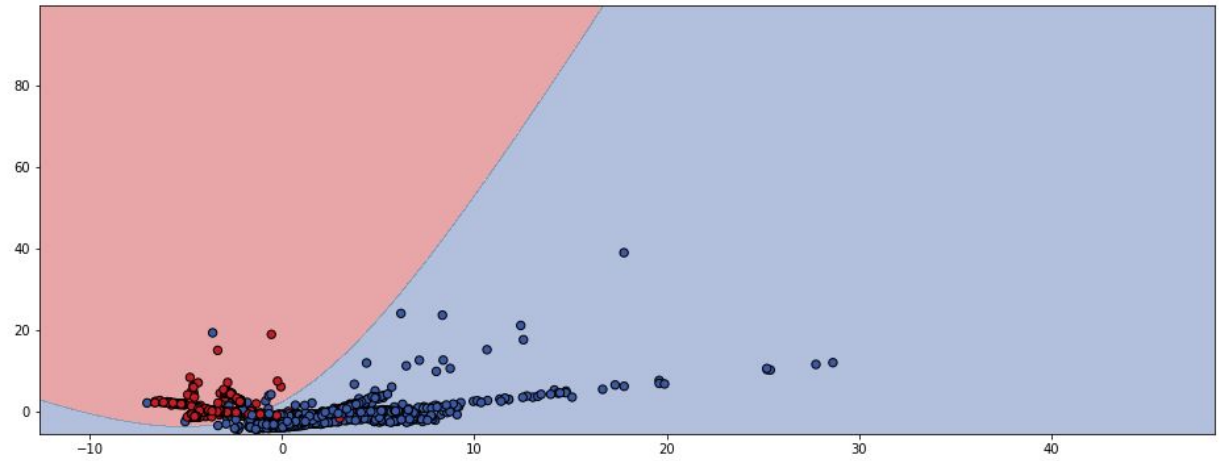


Figure 3: QDA classification visualized (Red being graphene)

We used five .TIF raw images and their masked images for the training. We performed standardization on the data to compensate for lighting differences between images. We then use

PCA to select two best features. We used Gaussian kernel and QDA classifier and found QDA classifier returning the better result, visualized in Figure 3.

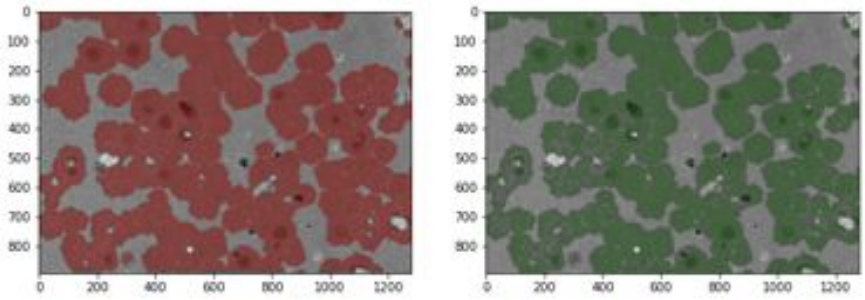


Figure 4: Training image and predicted image

The predicted image on the right of Figure 4 shows our image 1 training result, as compared to the training image on the left.

III. Summary and Conclusion

	Actual	Predicted	
		Graphene	Not Graphene
Training 1	Graphene	0.889	0.111
	Not Graphene	0.025	0.975
Training 2	Graphene	1	0
	Not Graphene	0.062	0.938
Training 3	Graphene	0.997	0.003
	Not Graphene	0.054	0.946
Training 4	Graphene	1	0
	Not Graphene	0.022	0.978
Training 5	Graphene	1	0

	Not Graphene	0.022	0.978
Self Test	Graphene	0.979	0.021
	Not Graphene	0.013	0.987
Final Test	Graphene	0.982	0.018
	Not Graphene	0.003	0.997

Table 1: Confusion matrices

As shown in Table 1, the results of the training data produced a high degree of accuracy when it came to predicting graphene. The source of most of the misidentification came from areas of major deformities or substrates. In addition, since a grayscale is implemented, images that had overlapping peaks from the intensity histograms produced less than average results like in training dataset 1. Regardless, the large amount of data used for training helped to predict graphene correctly ~97% of the time.

IV. Future Work

Our 8 by 8 patch size window may cause information lost around the edge of the graphene because classification result only based on the majority of pixels from the patch window. For future work, we can try to use a multi-size sliding window to patch the raw image to generate training features. Multi-size sliding window can guarantee to transfer all size of graphene's geometric information into training feature. As for training labels, we can define more classes such as 0, 1, 2, 3 and so on. (corresponding to graphene in patches center, left corner, right corner and not at all) Then, the edge information of graphene can also be identified during the training and testing. What's more, we can also work on the intensity recording in order to distinguish the overlapped graphene from single-layer graphene, because overlapped graphene region will have higher intensity compared to the single layer graphene. During the analyzing, we also found some SEM image contained significant pollution, which may influence the result of graphene identification. We can tune the training data to collect the feature for pollution area and predict if it is graphene or non-graphene.

V. Reference

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.