Unsupervised Satellite Image Segmentation Via Silhouette Score Optimized K Means Clustering

SURYA KESWANI, The Augmented Design Laboratory @ The University of California, Santa Cruz

This paper dives into the techniques used to classify and segment satellite images at great levels of details. Techniques covered in this paper include: classification via color distribution of pixels, Canny Edge Detection optimized via Otsu's Method and a zero parameter method, and K Means Clustering optimized via the Elbow Method and a Silhouette Score. Each of the techniques are explained and the options for satellite image segmentation are weighted. All the code and documentation for this project can be found by clicking here.

1 INTRODUCTION

The National Highway Traffic Safety Administration conducted a study that concluded 94% of serious automotive accidents in the United States are caused by dangerous driver behavior or just plain human error [1]. These risks of loss of life and serious injury have led to a vast amount of research and development in the autonomous vehicles (AV) market over the last decade. The AV industry was worth $54.23 billion in 2019 and it is projected to grow to $556.67 billion by 2026 (a compounding annual growth rate of 39.47 %) [2].

This explosive amount of research and development has brought the industry into the public's eye. Some skeptics have raised ethical and safety concerns in relation to AV testing. There have been a handful of fatalities as a result of testing AV's on the road. Waymo, Alphabet's self driving car company, has proven that vehicle testing can be just as effective and informative in virtual simulations. The company is well known for its highly intelligent set of autonomous vehicle taxis that operate in the United States. Waymo has credited the safety of its AV fleet to the 10 billion miles of simulation testing that it has conducted [3].

As the industry continues to rapidly grow, more and more virtual simulations are being conducted to test the safety and robustness of AV algorithms. These virtual simulations are extremely labor intensive to build as they require the creation of complex graphics in a 3 dimensional space. Furthermore, many virtual simulations are built using raw data collected from vehicles with on board sensors. This limits virtual testing environments to the scope of data. Virtual environments become directly dependent on where and for how long these data collecting vehicles drive.

The objective of this paper is to explore and test segmentation of images at high levels of detail and use said segmented photos as a training data set for a neural network architecture. Segmentation of these images would mean isolating desired features and labelling them. In the context of satellite images, relevant features that would be isolated include roads, intersections, sidewalks, crosswalks, and lane markings, to name a few.

If segmented in high levels of detail, satellite images would be used to understand road infrastructure. Satellite images contain a plethora of information such as the frequency distribution of the angles of incidence of roads coming into an intersection, sidewalk geometry, and other unique features such as the presence of islands and medians. Furthermore, virtual road infrastructure could be updated in real time as new satellite images are retrieved. This would allow the built simulations to account for construction, road closures, and based on the frequency of updates, possible traffic scenarios, and accident scenarios.

The methodologies for image segmentation covered in this paper include: classification via color distributions of pixels, Canny edge detection optimized via Otsu's method and a zero parameter

Author's address: Surya Keswani, sukeswan@ucsc.edu, The Augmented Design Laboratory @ The University of California, Santa Cruz.

method, and K means clustering optimized via the elbow method and the Silhouette Score. Each of the techniques are explained in detail and the options for satellite image segmentation are weighted.

2 RELATED WORKS

In this section, two related papers will be discussed. The first paper, HD Maps ... , uses a residual network based on a convolutional neural network to yield promising results. The key difference between this related work and the research in this report is the raw data used. HD Maps uses a hybrid data set of high definition satellite images and data from on-board vehicle sensors. As previously stated, data collected from vehicles is expensive and restrictive. This project concentrates solely on using satellite images as input data.

The second piece of adjacent research, Aerial LaneNet, uses discrete wavelet transformation and a fully convolutional neural network architecture to achieve promising results. This supervised technique of using a neural network architecture is able to segment satellite images with detail. Due to a lack of open source, annotated ground truth data, an unsupervised architecture had to be explored for this paper and the Aerial LaneNet architecture could not be leveraged.

2.1 HD Maps: Fine-grained Road Segmentation by Parsing Ground and Aerial Images [4]

This paper segments satellite images using a residual based convolution neural network. The results of the segmented satellite images are then cross-referenced with with data collected from stereo cameras mounted on top of a car. To align the map data as well as the vehicle data the center lines of roads are retrieved from Open Street Map (OSM). The cross referencing of maps, car data, and OSM data can categorize sidewalks, road lanes, and parking spots. The architecture's accuracy reduces at intersections because of the complexity of the turning lanes at intersections. The accuracy is also negatively affected in dense pedestrian areas as pedestrians crowd the raw data and there are many unique crosswalks. This variation in both intersection types and crosswalk types makes segmenting these features challenging. The effects of cross referencing and aligning the data can be seen in Figure 1. Some of the shortcomings mentioned can be seen in Figure 2.

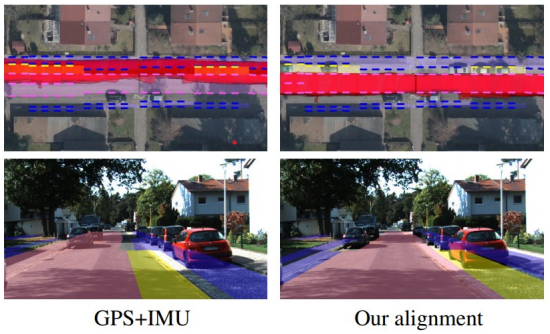

Fig. 1. Effect of reasoning about alignment: (left) alignment given by GPS+IMU, (right) alignment inferred by model. (top) Ground road classifier projected into the aerial image (shown in red). (bottom) The semantic classes projected on the ground image. The joint reasoning significantly improves alignment.

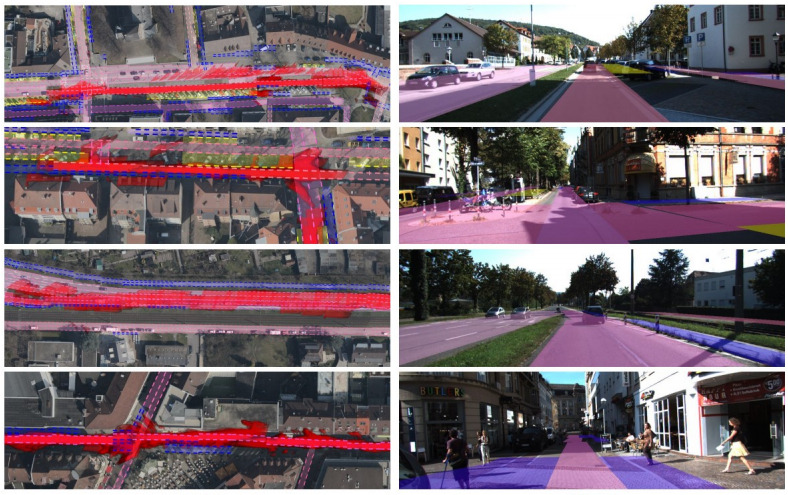

Fig. 2. Left: The ground road detection with red projected into the aerial image after alignment and road layout estimation. Right: The semantic lanes projected back into the aligned ground image. These scenes are all challenging with parallel roads, parking spots and intersections. The bottom image is especially difficult since it is an urban pedestrian area. Note that the aerial and ground images were taken with several years difference in different seasons. Pink is road, blue is sidewalk and yellow marks parking spots.

2.2 Aerial LaneNet: Lane Marking Semantic Segmentation in Aerial Imagery using Wavelet-Enhanced Cost-sensitive Symmetric Fully Convolutional Neural Networks [5]

For reasons very similar to this paper, researchers at the the Department of Earth Observation Center, Remote Sensing Technology Institute, Photogrammetry and Image Analysis, German Aerospace Center built a supervised architecture to segment satellite images. Aerial LaneNet extracts details such as lane markings from road infrastructure solely from satellite images. Figure 3 shows the architecture used and Figure 4 shows some of the results generated by this supervised architecture.

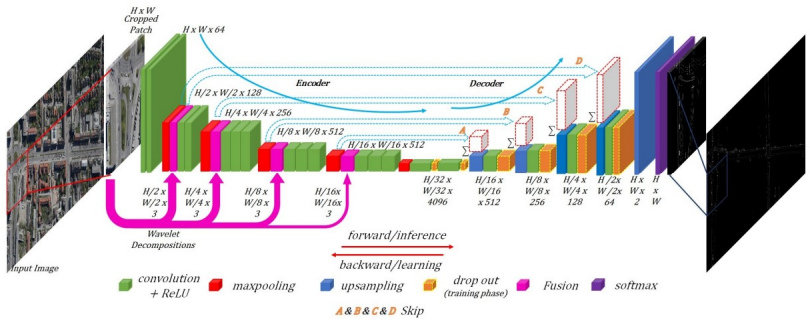

Fig. 3. Aerial LaneNet. Overview of lane marking segmentation approach using Wavelet-enhanced symmetric cost-sensitive fully convolutional neural networks. The input image is a high resolution aerial image. It is cropped first and segmented using Aerial LaneNet network. In the end, segmented patches are stitched together. H and W represent height and width and third number is number of feature maps.

Aerial LaneNet, consistent with the conclusions in this paper, finds that segmenting satellite images comes with many complications. The two most prominent complications being image occlusions and shadows. Occlusions include partial or full changes or appearances in lane markings. This includes the cars obscuring lane markings, degrading lane markings, etc. Moreover, shadows large enough and dark enough mask lane markings and skew the results of the segmentation.

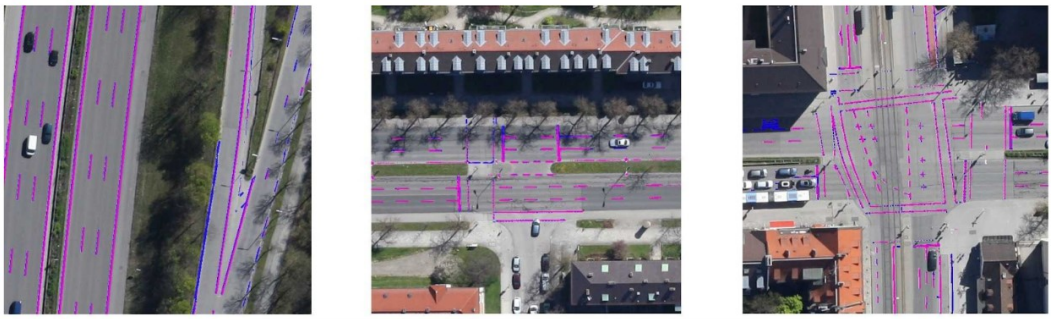

Fig. 4. Aerial LaneNet. Sample results from model.

3 THE DATA: LOW ALTITUDE INTERSECTION SATELLITE IMAGES

In an effort to release this this research as an open source project, the data used for this project has to open openly available. Accessing free and detailed data is difficult. This required the procedural creating of a satellite image data set. Additionally, the data set had to be consistent and preferably at a low range altitude. Many of the readily available satellite image data sets were collected at a high altitude and road details such as lane markings and medians could not be captured. To solve this, a script was built to generate low altitude satellite images. The script takes latitude and longitude coordinates as inputs and centers the output images around these coordinates. The script utilizes the Mapbox API* [6] to retrieve the satellite data. In addition to creating a data set of low altitude sat images, the script overlays the Mapbox Building Segmentation Mask over the images. The final outputted images segment buildings in red. This improves the clustering model run on the images. An example of the data can be seen in Figure 5.

* Note: Mapbox is a platform for mobile and web applications that allows developers to easily incorporate location based features such as maps, search and navigation to any experience.

4 METHODOLOGY

This section will cover the different techniques that have all been tried and tested in the process of segmenting the data set using unsupervised learning algorithms. The process of successfully building a segmentation model required many iterations of trial and error. Many of the techniques failed but brought upon valuable conclusions.

4.1 Image Segmentation via Color Distributions [7]

When initially tackling this project, the first idea was to segment the images via color distributions. Though this is the era of deep learning and big data, color spaces are still surprisingly powerful and hold a plethora of information about an image. Color spaces are still widely used in many applications of image analysis. Figure 6 shows an image of a clown fish and Figure 7 is the HSV color distribution of the clown fish image.

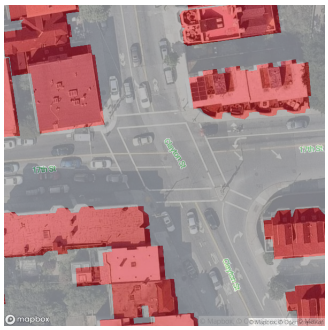

Fig. 5. Example of data. The buildings are masked in red using Mapbox's Building Segmentation Mask. This image was taken at the coordinates: -37.761783, -122.446816

Fig. 6. Image of a brightly colored orange clown fish against dark blue ocean background.

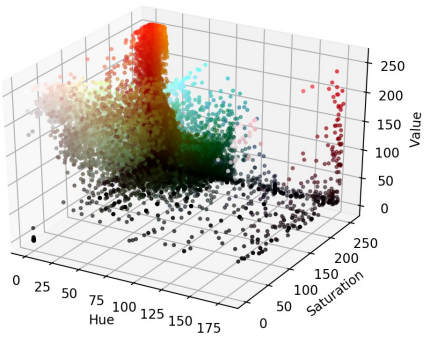

Fig. 7. The Hue, Saturation, and Value color distribution of the clown fish image from Figure 6. The color distribution in the HSV space shows in the top back corner, the cluster of orange pixels are easily separable from the rest of the colored pixels, a majority of them being teal, white, and dark blues and blacks, all representative of the background and small details in the image.

Note that an HSV space is used over and RGB space. The HSV color space is a cylindrical color space which makes the pixels more visually and computationally separable.

Using OpenCV, a open source computer vision library [8], an upper and lower bound can be set for the colors white and orange. These two sets of bounds are then used to filter the colored pixels in those ranges. Figure 8 shows the white pixel segmentation mask of the clown fish. Figure 9 shows the orange segmentation mask of the clown fish. Figure 10 shows the overlay of the orange pixel segmentation mask and the white pixel segmentation mask.

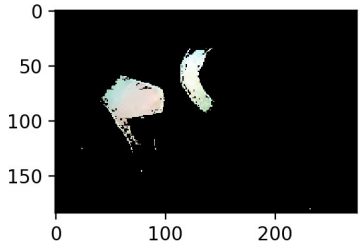

Fig. 8. The white pixel segmentation mask of the clown fish.

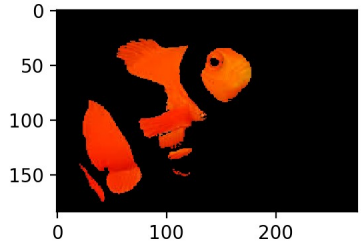

Fig. 9. The orange pixel segmentation mask of the clown fish.

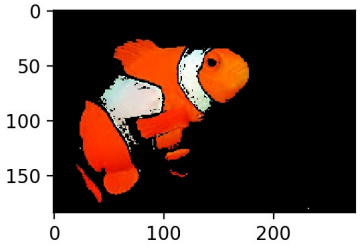

Fig. 10. The overlay of the orange pixel segmentation mask and the white pixel segmentation mask.

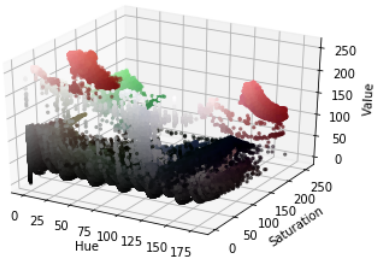

Fig. 11. A sample color distribution of the satellite image data set.

While this segmentation method works very well for this image of a clown fish, the results for the satellite image data set are underwhelming. What you will notice about the clown fish image

is the high level of contrast in the image. The dark background along with yellow colored plants allow for the fish to stand out in the photo. Satellite images lack these levels of contrast, even with the already masked buildings. Sidewalks, cars, and outlines of building all contain gray scale values that distort the resulting segmentation. Figure 11 shows the color distribution from a sample image from the data set. The distribution clearly shows the gray scale pixels are too heavily clustered. Because of this tangled distribution, the data set is poorly segmented using the color distributions.

4.2 Image Segmentation via Threshold Optimized Canny Edge Detection [9, 10]

The second approach taken to segment the image data set was to implement the Canny Edge Detection algorithm from the OpenCV Library. Canny Edge Detection is a well known edge detection algorithm that was developed by John F. Canny in 1986. The multistage algorithm can be broken down into 5 steps:

(1) **Noise Reduction:** Edge Detection is susceptible to noise because the math involved in edge detection is based on derivatives. By default, a 5x5 Gaussian filter is is applied to the image to reduce noise and blur in the image.

(2) **Gradient Calculation:** The intensity and direction of an edge is calculated using the gradient in both the vertical and horizontal direction. To do this, the smoothed imaged is filtered through a Sobel Kernel to get the first derivative in both the horizontal(x) and vertical(y) directions. These two images are then used to find the edge gradient and direction for each pixel using the two following formulas:

$$EdgeGradient(G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle(\theta) = \tan^{-1}(G_y/G_x)$$

(3) **Non-Maximum Suppression:** The previous step will create some edges that are thick and some edges that are thin. To thin out all the edges, Canny Edge Detection performs non-maximum suppression. The algorithm iterates through all the points on the gradient intensity matrix and keeps the pixels with the maximum intensities with respect to their local neighbors. When Canny Edge Detection in run on an image, edges are highlighted in white while the rest of the image, the background, is kept black. On the gray scale spectrum , white holds the maximum value of 255 and black holds the minimum value of 0. Therefore, when a thick white edge is run through the non-maximum suppression algorithm, only the most intense white pixels are kept, and the duller white pixels are suppressed to black. This thins out the thicker edges created by the gradient calculations in Step 2.

(4) **Double Threshold:** The Canny Edge Detection algorithm requires minimum and maximum threshold inputs to classify pixels. These 2 thresholds create 3 categories in which pixels are classified:
   (a) Strong Intensity Pixels: These pixels have intensity gradients that are *higher than* the value of the *maximum* threshold. We are sure these pixels will contribute to the final edge.
   (b) Weak Intensity Pixels: These pixels have intensity gradients that are *lower than* the value of the *minimum* threshold. We are sure these pixels will not contribute to the final edge. These pixels are considered irrelevant.
   (c) Mid-Range Intensity Pixels: These pixels have intensity gradients that fall between the specified thresholds. It is unclear if there are strong intensity pixels or weak intensity pixels. The Hysteresis mechanism in the following step will determine if these in-range pixels should contribute to the final edge they lie on.

(5) **Edge Tracking Hysteresis:** The Hysteresis determines if mid-range intensity pixels will contribute to the final edges. If a mid-range pixel is connected to a strong pixel then the mid

range pixel is now classified as a strong intensity pixel. Otherwise, the mid-range pixel is classified as a weak intensity pixel and therefore discarded.

To optimize the Canny Edge Detection Algorithm for the satellite image data set, two optimization functions (Otsu's Method and a Zero Parameter Method) were implemented to calculate the minimum and maximum threshold values. The reason the thresholds have been optimized is to account for the vast variation of satellite images than can be run through this model. For instance, a satellite image taken at night versus a satellite image taken during the day will require very different threshold values for meaningful edge detection to occur.

4.2.1 Optimization of Edge Detection using Otsu's Method [11]. : Otsu's method is an adaptive threshold algorithm for binary image classification. Otsu's method finds optimal threshold values by calculating and analyzing the weighted within-class variance for every possible pixel value (0-255). Otsu's Method was implemented using the OpenCV Library.

4.2.2 Optimization of Edge Detection using a Zero Parameter Method [12]. Another threshold optimization applied relies on a simpler method that utilizes the median value of the single channel of pixel intensities. This median, along with an optional argument sigma(σ), is used to calculate the lower and upper thresholds for the Canny Edge Detection algorithm. Sigma is used to vary the percentage thresholds (range 0-1). The thresholds were calculated as follows:

$$lower = max(0, (1 - \sigma) * median))$$

$$upper = min(255, (1 + \sigma) * median))$$

These thresholds are constructed based on the percentages controlled by sigma. A lower sigma percentage value will create a tighter threshold range and a higher sigma percentage value will create a wider threshold range. As stated, to create a truly zero parameter optimization, sigma can be set to a default value, such as 0.4. Otherwise, sigma can be adjusted by the user.

4.2.3 Results of Threshold Optimized Canny Edge Detection. Despite the different optimizations performed, the results of running Canny Edge Detection Algorithm on the satellite image data set were not ideal. As Figures 12 and 13 show, Canny Edge detection still picks up a lot of noise in the photo despite the blurring effect of the Gaussian filter. The edge detection picks up details such as crosswalk paint and roads well, but the noise picked up from cars, trees, the shadows of trees, and other miscellaneous features of the image make this form of edge detection impractical. Figure 12 shows a sample image from the data set optimized using Otsu's Method. Figure 13 shows a sample image from the data set optimized via the zero parameter method.

4.3 Image Segmentation via K Means Clustering [13]

The final unsupervised method tried on the satellite image data set was a K Means Clustering model. The K Means algorithm is a popular and powerful unsupervised Machine Learning algorithm that has many use cases. The algorithm clusters data into k clusters. For instance, if K is set to 3, the algorithm will cluster the data into 3 groups. If K equals the number of data points, each data point will be its own cluster. K means initializes K center points and then works through the following iterative steps:

(1) Assign each data point the cluster center it is closest to
(2) Recalculate the cluster centers based on the data points assigned to it

These two steps are repeated until the clusters converge and there is no change. The key to this model is choosing the optimal number for K. As K increases in value, towards the maximum value of the number of data points, the accuracy of the model will go up as each data point will slowly become isolated and therefore its own cluster. K values that are too small will not effectively

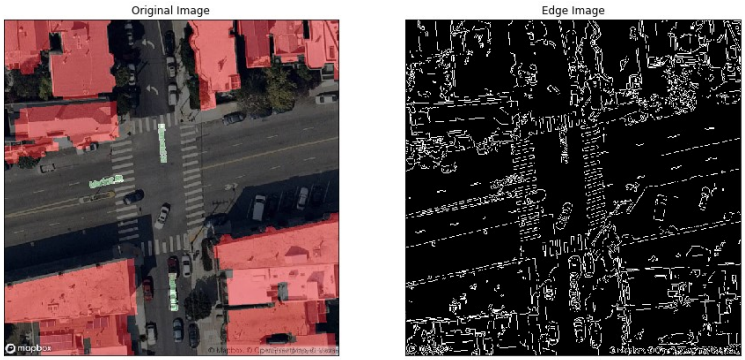

Fig. 12. Otsu's Optimized Canny Edge Detection with a lower threshold of 100 and an upper threshold of 200.

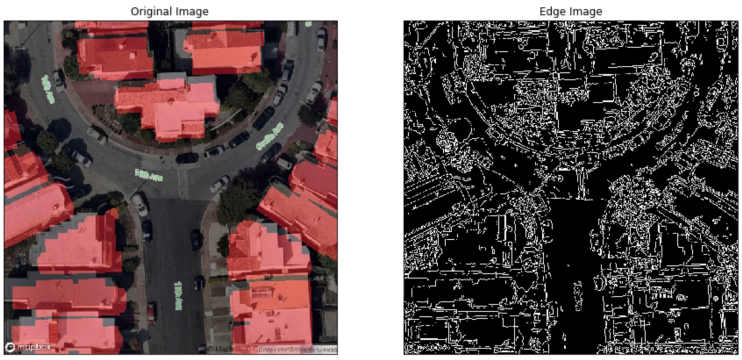

Fig. 13. Zero Parameter Optimized Canny Edge Detection with a lower threshold of 48, an upper threshold of 95, and 0.33 as sigma.

cluster the data. To solve this, two optimization functions were tested so each image has the optimal number of clusters.

4.3.1 Optimization of Clustering using The Elbow Method [14]. The first k optimization algorithm that was considered was the elbow method. This method picks a K value that optimizes for the number of clusters based on the size of each cluster.

The idea is to pick the k value that outputs the smallest clusters *before* the cluster sizes begin to converge with each increasing k . For instance, Figure 14 shows K values mapped to WCSS, or within-cluster sum of squares. WCSS is a measure of cluster size. The graph shows that the best K value to pick for this specific data set is 5. It is the largest number of clusters before the WCSS begins to converge.

In many cases, it is unclear what would be the optimal K value using the Elbow method. Figures 15 and 16 show a separate data set where the optimal K can be argued to be 3, 4, or 5. It is not entirely clear the exact point at which K should be chosen. If K is assigned a value of 3, that is before the cluster sizes converge. A K value of 5 would result is much better clustering but the K value is slightly after the convergence. Weighing these options procedurally is difficult. Therefore, the Elbow method was not implemented for this reason of ambiguity.

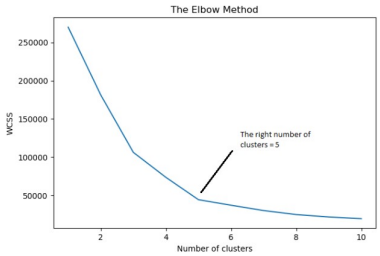

Fig. 14. Example of Elbow Method successfully

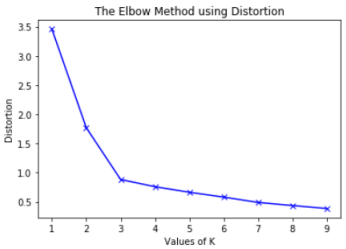

Fig. 15. Example of Elbow Method ambiguity based on distortion of clusters. K can be chosen from range 3 to 5. Distortion is calculated as the average of the squared distances from the cluster centers of the respective clusters. Typically, the Euclidean distance metric is used.

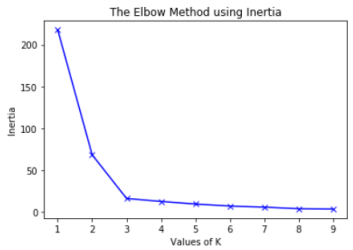

Fig. 16. Example of Elbow Method ambiguity based on Inertia of clusters. K an be chosen from range 3 to 5. Inertia is the sum of squared distances of samples to their closest cluster center.

4.3.2 Optimization of Clustering using Silhouette Score [15–17]. The second approach taken to optimize K is the Silhouette Score. Rather than attempting to decide which K value is optimal based on where the cluster sizes converge, the most optimal Silhouette Score is the score with the largest value. The Silhouette Score ranges from -1 to 1. A value close to 1 suggests that data point is close to its cluster and is part of said cluster. A value close to -1 means the data point is assigned to to wrong cluster and should be re-evaluated. The Silhouette Score for a data point is calculated as follows:

$$Score = (x - y)/max(x, y)$$

Y is the mean intra-cluster distance (the mean distance to the other instances in the same cluster) and X is the mean distance to to the all the instances of the nearest cluster. The Silhouette Coefficient

is a much clearer metric when optimizing for K but the score is computationally very expensive and slows down the clustering model tremendously. Images can take up to 15 minutes to cluster. A majority of that time is spent by the optimization function.

5 CONCLUSIONS

This paper has discussed many different techniques to solve the problem at hand, detailed segmentation of satellite images using unsupervised machine learning. Many of these techniques proved to be unsuccessful except the K Means Clustering model optimized by the Silhouette Coefficient. Figure 17 shows an image taken from the data set and clustered with 2, 4, 6, 8, and 10 clusters. The model, like the Canny Edge Detection, captures excess data such as shadows, tree, cars, etc but the clustering model captures much less noise.

The image that is segmented with 2 clusters primarily segments any dark shadows and dark patches of grass. This is clearly too small of a K value to use.

The image segmented with 4 clusters does a much better job in segmenting roads, builds, and dark shadows. For the purposes of isolating the road features in the images, a K value of 4 will seem to work very well.

The image segmented with 6 clusters goes a step further and is able to segment shadows, sidewalks, and the different gray values of the roads in the image.

Clustering with more than 6 clusters results in additional details in the photo being segmented. For the purposes of this research, this is not necessary as we look to isolate and label larger features such as roads, sidewalks, etc.

After running the Silhouette Score optimizer on this data set, it is clear the time and computational power the optimizer needs is outweighed by the results of a standard and chosen K value. In most cases, a standard k value of 6,7, or 8 will work just fine. This can been seen in Figure 18 where there is a second sample image from the data set. The optimizer for other data sets will most likely be much more beneficial in comparison to this data set.

With some slight manual adjustments and refinements to the segmentation, the roads, medians, and some crosswalks will be segmented from these images. To build the finalized training data set for a neural network architecture, the satellite image data set will be segmented using the clustering model. The results of the clustering model will this be manually smoothed and refined for the neural network architecture.

ACKNOWLEDGMENTS

I would like to thank Professor Jim Whitehead for overseeing this project and providing insight into different leads. I would like to thank Professor Marilyn Walker for sponsoring and supporting my research. I would like to thank Donald Stewart, for his continued assistance and advice with this project.

REFERENCES

[1] *Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey.* Feb 2015. Online; Accessed 13-March-2020.

[2] Akshay Jadhav. Autonomous vehicle market size, share and analysis: Forecast 2026. https://www.alliedmarketresearch.com/autonomous-vehicle-market, May 2018. Online; Accessed 13-March-2020.

[3] Darrell Etherington. Waymo has now driven 10 billion autonomous miles in simulation. https://techcrunch.com/2019/07/10/waymo-has-now-driven-10-billion-autonomous-miles-in-simulation/, Jul 2019. Online; Accessed 13-March-2020.

[4] Gellért Máttyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Hd maps: Fine-grained road segmentation by parsing ground and aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3611–3619, 2016. Online; Accessed 13-March-2020.

[5] Seyed Majid Azimi, Peter Fischer, Marco Körner, and Peter Reinartz. Aerial lanenet: Lane-marking semantic segmentation in aerial imagery using wavelet-enhanced cost-sensitive symmetric fully convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 57(5):2920–2938, 2018. Online; Accessed 13-March-2020.

[6] Mapbox. https://docs.mapbox.com/api/. Online; Accessed 13-March-2020.

[7] Rebecca Stone. Image segmentation using color spaces in opencv python. https://realpython.com/python-opencv-color-spaces/, Oct 2018. Online; Accessed 13-March-2020.

[8] Opencv. https://opencv.org/, Mar 2020. Online; Accessed 13-March-2020.

[9] Alexander Mordvintsev and Abid K Revision. Canny edge detection. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html, 2013. Online; Accessed 13-March-2020.

[10] Sofiane Sahir. Canny edge detection step by step in python-computer vision. https://towardsdatascience.com/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123, Jan 2019. Online; Accessed 13-March-2020.

[11] Alexander Mordvintsev and Abid K Revision. Image thresholding. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_thresholding/py_thresholding.html#otsus-binarization, 2013. Online; Accessed 13-March-2020.

[12] Adrian Rosebrock. Zero-parameter,automatic canny edge detection with python and opencv. https://www.pyimagesearch.com/2015/04/06/zero-parameter-automatic-canny-edge-detection-with-python-and-opencv/, Apr 2015. Online; Accessed 13-March-2020.

[13] Nagesh Singh Chauhan. Introduction to image segmentation with k-means clustering. https://towardsdatascience.com/introduction-to-image-segmentation-with-k-means-clustering-83fd0a9e2fc3, Jul 2019. Online; Accessed 13-March-2020.

[14] Alind Gupta. Elbow method for optimal value of k in kmeans. https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/, Jun 2019. Online; Accessed 13-March-2020.

[15] Selecting the number of clusters with silhouette analysis on kmeans clustering. https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html, 2019. Online; Accessed 13-March-2020.

[16] sklearn.metrics.silhouette-score. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html, 2019. Online; Accessed 13-March-2020.

[17] Jyoti Yadav. Selecting optimal number of clusters in kmeans algorithm(silhouette score). https://medium.com/@jyotiyadav99111/selecting-optimal-number-of-clusters-in-kmeans-algorithm-silhouette-score-c0d9ebb11308, Aug 2019. Online; Accessed 13-March-2020.

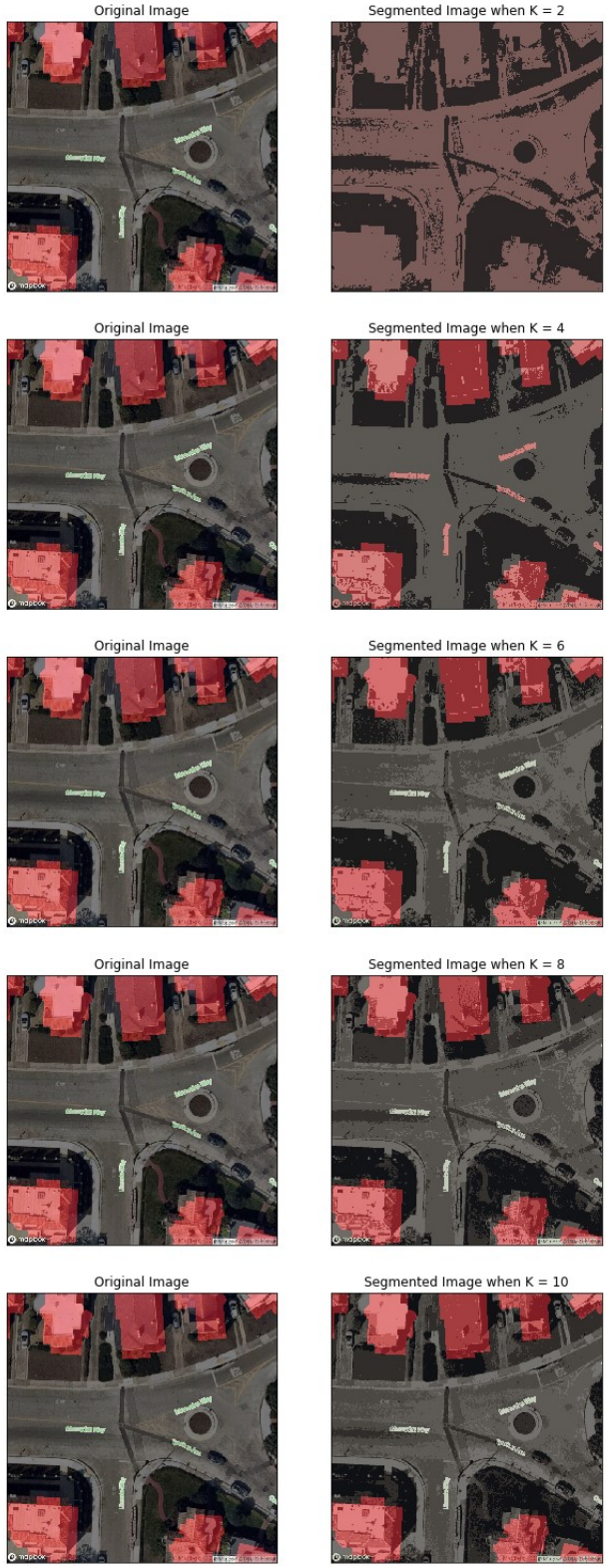

Fig. 17. Sample 1 data set image with clustering model run for different K values.

Fig. 18. Sample 2 data set image with clustering model run for different K values.