Implementation of Kohonen Self Organizing Maps for Color Recognition

By

Kyle Zeller

April 30, 2017

A composition of multiple machine learning methodologies,
submitted to the Faculty of the Undergrad School of the University at Oswego, State University of New York in discretional fulfillment of the Artificial Intelligence & Heuristics Course, as an

Undergrad in Electrical & Computer Engineering and Computer Science

Department of Computer Science

Contents

Abstract

Kohonen SOMs are an unsupervised machine learning technique discovered by Professor Teuvo Kohonen in the 1980s. Kohonen SOMs are generally used for taking n-dimensional information and mapping it down to a 2D representation of the input space $\mathbb{R}^n$. In this project, this machine learning technique will be applied to clustering a set of n-dimensional input and clustering the input together by common characteristics and then categorizing the input to have a way to interact with the program. The application will be a form of color recognition utilizing Kohonen Self Organizing Maps. The colors can be represented as vectors of (red, green, blue) components and the SOM will categorize the colors and you can then ask what color a given color vector best represents.

Chapter 1

Introduction

The main goal of this project is to be able to accurately label clusters of RGB input vectors by giving a sample vector and getting an accurate description of the color it is best represented by. This process is satisfied through the use of Kohonen Self Organizing Maps (SOMs) to train/cluster the initial RGB vectors in a table using a set of RGB sample vectors, an initial radius size, and a specified number of iterations. The latter part of the process consists of utilizing a post clustering technique in order to label all the clusters in the SOM. This phase consisted of utilizing K-means clustering with K being the number of RGB sample vectors that were used to train the data, then letting the winning RGB vector to the sample vectors be the initial centroids to minimize the convergence time, since each centroid is theoretically being placed at a local minimum. The next phase of the process was prompting the user to input some arbitrary RGB vector and a label for the the cluster that is found to be most similar to the given RGB vector (as long as the cluster isn't already labeled). After successfully labeling the clusters and getting accurate responses to querying the system using a sample RGB vector, the last stage to the project was to take two of clusters selected at random and to then randomly sample the clusters to render a checkerboard.

It is also worth noting that there were three different initializations of the Kohonen Self Organizing Maps, and that each initialization used three different similarity metrics in an attempt to see which map initialization and which similarity metric performed the best overall. There was a random initialization of the RGB vectors on the map, RGB circles radiating outward & equidistant from the center of the map, and RGB & black fading inwards from the corners of the map. The similarity metrics used in the project, were euclidean distance, the cosine similarity, and the pearson correlation coefficient. At each stage of the project renditions of the RGB vectors on the map were necessary to provide useful visual information to gauge the performance of the SOMs. There were renditions of the three map initializations, nine final renditions of each of the three map initializations using each of the three similarity metrics, GIFs generated of all nine maps adjusting from beginning to end, and a rendering of all nine output maps in grayscale, whereas the distance between every weight vectors and their neighbors is calculated to average all these distances to be a color which is then set to the respective location.

Chapter 2

Background

To understand this application of Kohonen Self Organizing Maps for color recognition, it is necessary to define some of the methods and terminology used. Kohonen Self Organizing Maps, are an unsupervised machine learning technique that “takes an n-dimensional input space and maps it down to a 2-dimensional output space or verbatim a map” [2]. This project utilized three different similarity metrics for measuring the similarities between any two given n-dimensional vectors, returning a scalar quantity. The similarity metrics used were euclidean distance, the cosine similarity, and the pearson correlation coefficient. E.g. using the “similarity metrics to determine the winning RGB vector to a training sample RGB vector” [6]. The project also used the “Gaussian function to determine the neighbors of a winning RGB vector in the map within a certain radius and how much to scale those neighboring RGB vectors the further out from the winning RGB vector they are” [7]. Other suggested methods to perform a role as the gaussian function, include “concentric squares, hexagons, or using the mexican hat function” [13]. Another important feature of the project was the “post-clustering that was performed after the clustering by using SOMs” [11]. K-means clustering was used for this secondary clustering, so the user could label the individual clusters, to which they could later use to accurately classify different RGB vectors. Among “K-means clustering being used for post-clustering, other resources suggested the use of bottom-up or top-down hierarchical agglomerate clustering to create the clusters of the colors, structurally making the shape of a dendrogram” [8]. “Bottom-up hierarchical agglomerate clustering, initially takes every RGB vector in the map and treats it as it’s own cluster and then combines the clusters one at a time, till the clusters form one single cluster. Visa Versa, top-down hierarchical agglomerate clustering is the opposite; taking all the points as one cluster and then breaking them into separate clusters one at a time till every point is a cluster” [12]. One very important aspect of the project came when selecting the value of K for the number of clusters for the K-means clustering. The value of K would directly affect what and how many RGB vectors went into each cluster. K-means clustering requires that the number of clusters to be created is known a priori, so picking the right value of K is sine qua non. This is an issue which isn’t unique to this application of K-means clustering alone. Many real-world problems e.g. clustering and classifying population census data [1], depend on knowing the value of K. As a result some sophisticated approximation techniques to determine the optimal value for K have been developed. Some of the “recommended methods to approximate the optimal value of K, are to use a technique called the elbow method, where different values of K are chosen and plotted with a certain cost function. This consists of the distance from the points within each cluster are from each other and is plotted whereas an point of a high decrease in the cost function exists and the value of K associated with that rapid decrease in the cost is the optimal approximated number of K" [9]. Two common cost functions used when utilizing this elbow method, is Dunn's index, which is aimed at finding sets of clusters that are compact and are well separated from each other. The other; Davies-Bouldin index, is aimed at finding clusters that are compact and their centers are far away from each other. Another technique for approximating K, and which is a fairly new concept, is to use the kernel trick from Support Vector Machines to approximate the number of groupings of data that are found and to take that number as the value for K. Finally, the other technique that can be applied and is the most preferred if it is possible, is when the data has been trained on a finite set of samples. Given this to be the case, it is assumed that the exact number of clusters created by the SOM, is the number of samples that were used to train it. The benefit to this is even greater if the SOM clustered the data very well, because at this time, "if the initial centroids for K-means clustering are picked to be the winning vectors on the map that best correspond to the training samples themselves, the performance of the post-clustering is guaranteed to be approximately ideal, since in theory the winning weight vectors of these training samples are supposed to be located at the local minimum for each cluster" [10]. This in turn leads to quicker convergence while using K-means clustering, and dramatically shortening the computation time to create these clusters.

Chapter 3

Program Description

The first part of the project was to create a symbolic representation of the each of the Red, Green, & Black map initializations. One would be a map of RGB vectors selected at random, another would be of a map where Red, Green, Blue, & Black are initialized at the corners fading inwards, and lastly would be a map with RGB circles equidistant from the center that fade out radially.

The second part of the project was to use similarity metrics e.g. euclidean distance, the pearson correlation coefficient, and the cosine similarity to determine the most similar RGB vector in the map at any time, given some other RGB vector. All three similarity metrics were used at this stage to provide another way to determine their effects on the each initialization of the map, from beginning to end. These similarity metrics were primarily used at this stage in the project to "determine the winning RGB vector found in the map when comparing some training sample RGB vector with every RGB vector in the map" [14].

The third stage of the project was to determine all the neighboring RGB vectors within a certain radius of the winning RGB vector that was found from the previous part of the project. Over time the radius would shrink by a certain amount and new neighbors would be found with respect to the winning RGB vector.

The fourth stage of the project was scaling creating a function that would scale the winning RGB vector in the map to be more similar to the training sample RGB vector. Likewise, all the neighboring RGB vectors to the winning RGB vector would also have to be scaled to be similar to the sample vector. However, the neighbors would scale pro rata with respect to the Gaussian function, the current radius size, and with the current time. This formula in turn would provide the basis of the learning function whereas the more time that passes the less the outermost neighbors will learn.

The fourth stage of the project was to take the previous parts of the project and set a number of training samples to be generated, a certain initial radius, and an amount of iterations that the Kohonen Self Organizing Maps would run for. Each of these parts would play a climacteric role in what the final clusters of the data would look like. E.g. the number of training samples would affect the number of overall clusters, the RGB vectors for each of the training samples would affect the color & the clustering, the radius would have the potential to affect the size of the clusters, and the number of iterations to run the program would affect how discretely separated the clusters are from each other & how the SOM clustered the data as a whole.

The fifth stage of the project was to create renditions of the map, ergo making it possible to visualize the performance of the Kohonen Self Organizing Maps. This was made possible by integrating a clisp library which would allow for the ability to render the RGB map as a png file. It was important to render every stage of the map adjusting to show the initial states of the map, to see how different similarity metrics affected the final RGB map for each map initialization, and it was important to see what how well the SOM technique clustered the data, so a grayscale method to average a point the distances with all its neighbors was made and was rendered for each of the map initializations using each of the similarity metrics.

The sixth stage of the project was to take the final clustering from the RGB map that clustered the the initial data the best and put all the data into clusters that can be labeled by a user to test later on for color accuracy / recognition. This was done using "K-means clustering, whereas the number of K clusters chosen for this post clustering, inter alia was set to be the number of training samples, and the initial centroids were set to be the winning RGB vectors from the map that best corresponded to these training samples at the stage in the program. This in turn was a quid pro quo, that would ideally decrease the time it takes to perform K-means clustering" [15]. After the completion of the secondary clustering, the user would be able to label the clusters by checking to see what cluster best corresponds to an RGB vector provided by the user and if the cluster isn't already labeling the user can provide a color label to that specific cluster. It is at this time that the user can perform accuracy tests to see how well the clusters performed with respect to the colors that they provide to the clusters in the RGB map itself.

The last stage of the project was to take two randomly selected clusters from the final stage of the RGB map and to render a checkerboard by alternatingly selecting at random from each cluster some RGB vector. This is also a decent benchmark test for the performance of the secondary clustering, since it shows visually the error in color that was rendered from each of the two clusters.

Chapter 4

Demos

The ability to gauge the performance of the Kohonen Self Organizing Maps being used throughout every stage of the project was a challenge, since in its raw form it is simply a large table of numbers that is difficult to directly interpret. One of the solutions to this was to create renditions of the different frames of the RGB map as it was adjusting, so that the results could be graded for performance and accuracy. The renditions for the performance of the clustering using Kohonen Self Organizing Maps, were the three map initializations, the nine final frames from each RGB map initialization using each of the three similarity metrics, the nine final frames grayscaled for a secondary clustering performance benchmark test, the nine GIFs generated for the RGB maps adjusting from beginning to end, and finally the checkerboard created form the clusters after the secondary clustering was performed using K-means clustering.

The project tested different map initializations such as the random RGB map initialization, the RGB & black initialized from the corners fading inward map initialization, and the RGB circles initialized equidistant from the center & fading radially outward map initialization. These different map initializations would play a role in the performance of how well the clusters formed as the RGB map is adjusting.

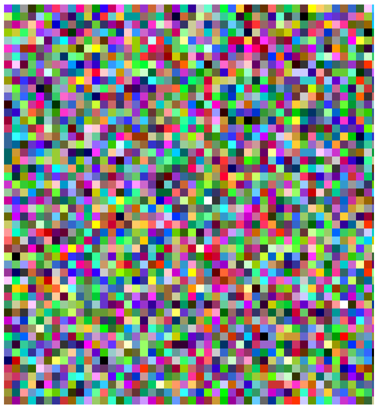
Figure 1: Randomly initialized RGB map.

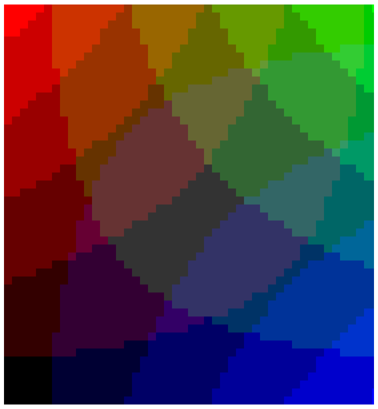
Figure 2: RGB & black initialized from the corners fading inward.

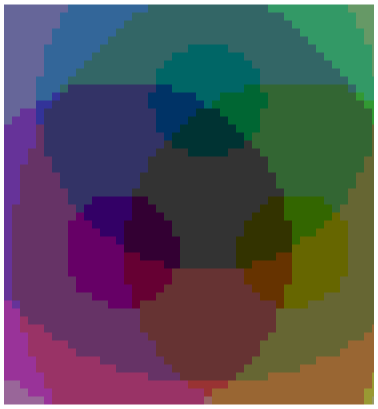
Figure 3: RGB circles initialized equidistant from the center & fading radially outward.

The project tested different similarity metrics on each of the three RGB map initializations. These nine different alternative ways of using the RGB maps would play a role in the performance of how well the clusters formed as the RGB map is adjusting. It would also help determine which map initialization and similarity metric performed the best overall.

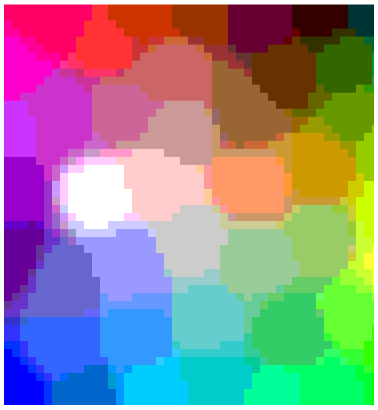
Figure 4: Random RGB initialization, using euclidean distance.

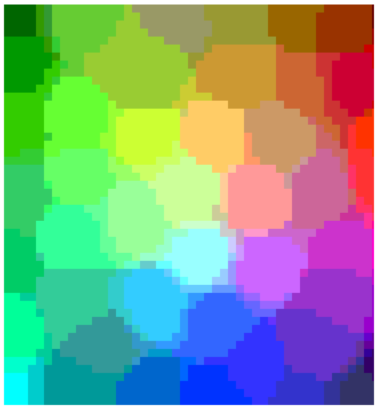
Figure 5: Corner RGB initialization, using euclidean distance.

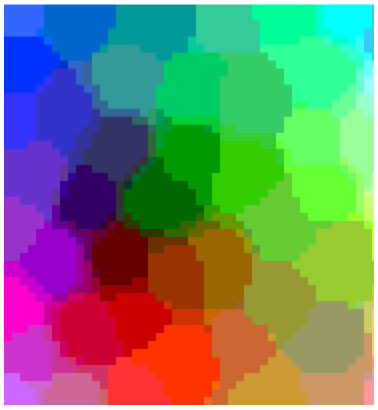
Figure 6: Center RGB initialization, using euclidean distance.

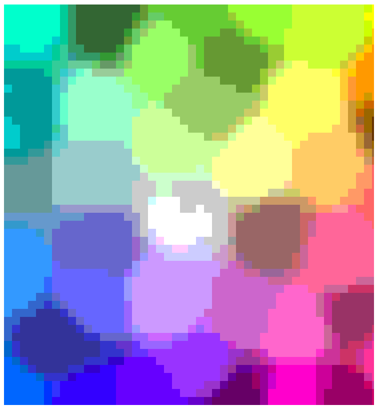
Figure 7: Random RGB initialization, using the cosine similarity.

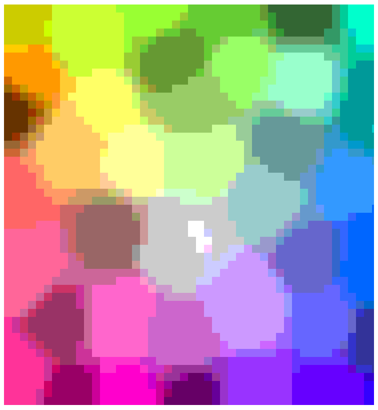
Figure 8: Corner RGB initialization, using the cosine similarity.

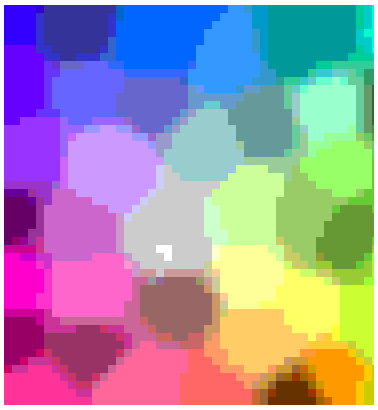
Figure 9: Center RGB initialization, using the cosine similarity.

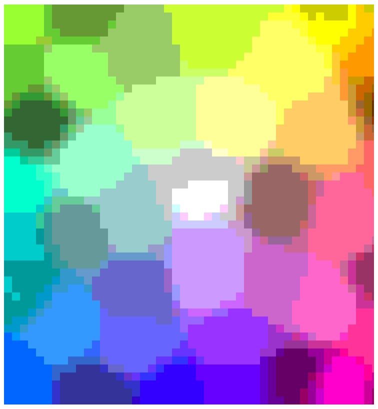
Figure 10: Random RGB initialization, using the pearson correlation coefficient.

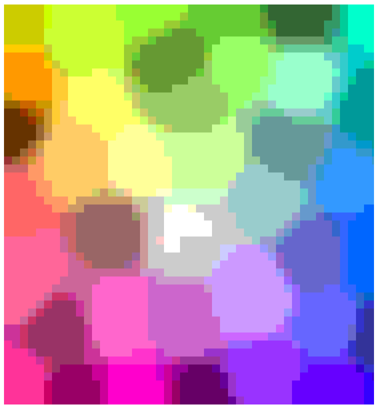
Figure 11: Corner RGB initialization, using the pearson correlation coefficient.

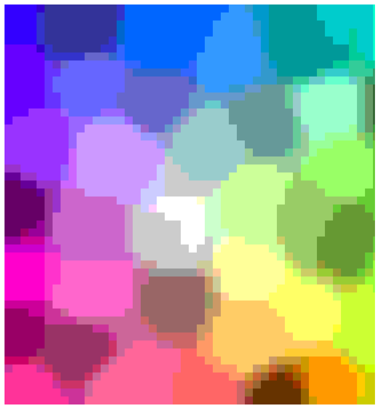
Figure 12: Center RGB initialization, using the pearson correlation coefficient.

The project tested different similarity metrics on each of the three RGB map initializations. These nine different alternative ways of using the RGB maps would play a role in the performance of how well the clusters formed as the RGB map is adjusting. In addition to this it would be nice to put the clusters into a grayscale to see how well the SOM is clustering the data.

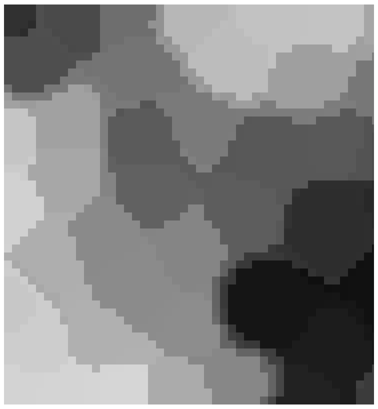
Figure 13: Random RGB initialization, using euclidean distance.

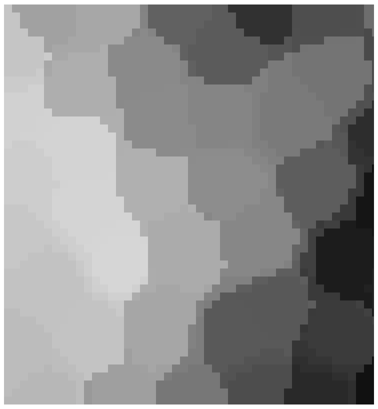
Figure 14: Corner RGB initialization, using euclidean distance.

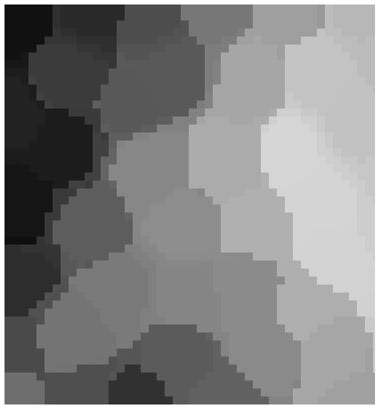
Figure 15: Center RGB initialization, using euclidean distance.

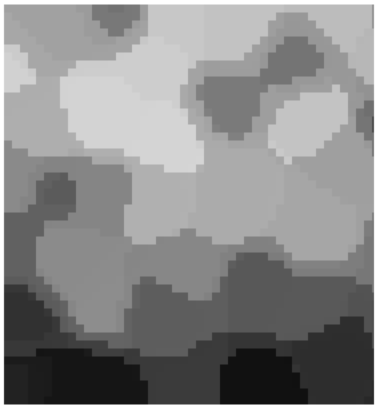
Figure 16: Random RGB initialization, using the cosine similarity.

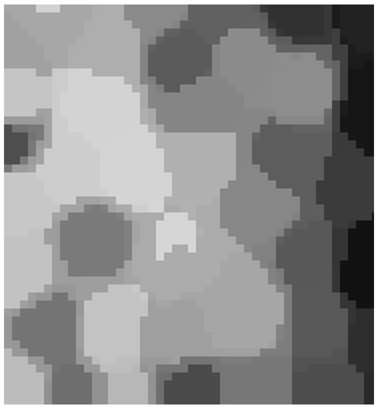
Figure 17: Corner RGB initialization, using the cosine similarity.

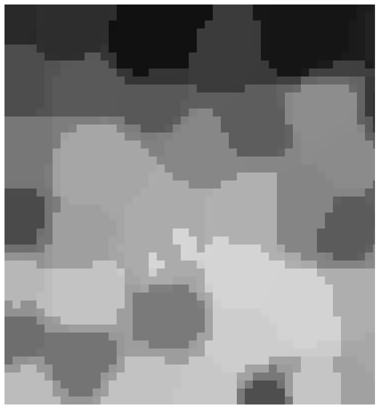
Figure 18: Center RGB initialization, using the cosine similarity.

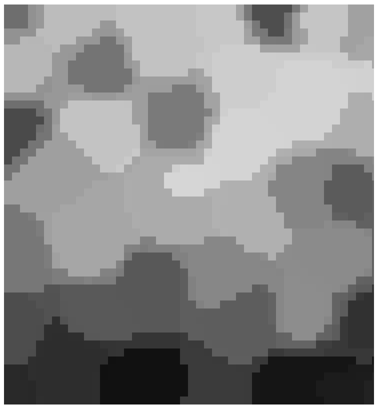
Figure 19: Random RGB initialization, using the pearson correlation coefficient.

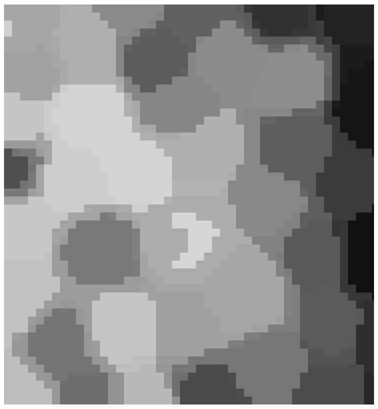
Figure 20: Corner RGB initialization, using the pearson correlation coefficient.

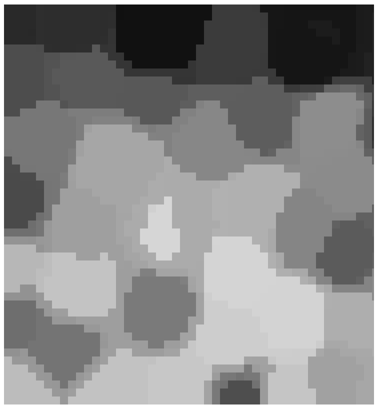
Figure 21: Center RGB initialization, using the pearson correlation coefficient.

The latter part of the project was to take two randomly selected clusters from the final stage of the RGB map and to render a checkerboard by alternatingly selecting at random from each cluster some RGB vector. This is also a decent benchmark test for the performance of the secondary clustering, since it shows visually the error in color that was rendered from each of the two clusters.

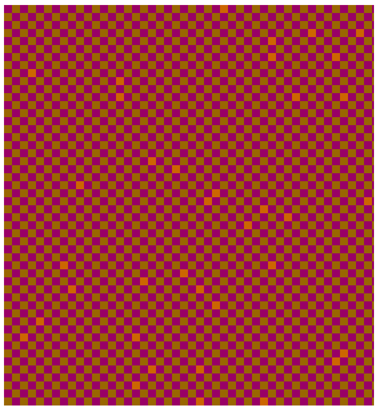

Figure 22: Two randomly selected RGB clusters rendered in a checkerboard fashion whereas the points are also randomly selected from the clusters.

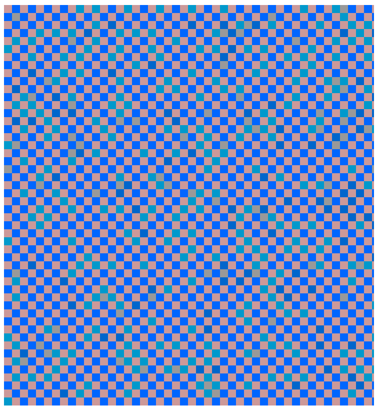

Figure 23: Two randomly selected RGB clusters rendered in a checkerboard fashion whereas the points are also randomly selected from the clusters.

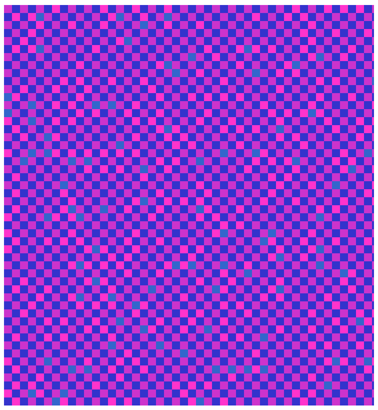

Figure 24: Two randomly selected RGB clusters rendered in a checkerboard fashion whereas the points are also randomly selected from the clusters.

Chapter 5

Reflections And Conclusions

The original goal for the project was to implement Kohonen Self Organizing Maps for clustering RGB data and then using K-means secondary clustering to make it possible for a user to label the clusters and perform a form of accuracy testing which would be the equivalent of testing for color recognition. In the process of creating the initial clusters generated by the Kohonen Self Organizing Maps, I learned of how vast variety in the methods used in the process of creating the clusters affected the final clustering results. E.g. the map initializations, similarity metrics, neighbor finding function, learning function, amount of training data, the training data per se, and the number of iterations to run the Kohonen Self Organizing Maps method for all had a different, but decently large effect on the final clustering output of the SOMs. The other realization I had about my plan of the project, was re the post clustering that was sine qua non to the project, because without it, labeling the data would be virtually impossible. In this stage of the project I had to come up with a clever way to cluster the already visually clustered data from the final stage of the Kohonen SOMs, without impacting the accuracy of RGB vectors that would go in each cluster, so that the user could label the data and perform color recognition tests, with high accuracy. In other words, I had to pick a secondary clustering technique that wouldn't jeopardize the original already clustered RGB vectors. The technique that was chosen for the post clustering was chosen to be K-means clustering out of simplicity, but there was another challenge and also unique feature to use this technique. The original challenge presented to me using K-means clustering, was picking an optimal number K for the amount of clusters to be formed and after a lot of researching online, I discovered a clever trick. If I was to "pick the number of K to be the number of training RGB samples, and the initial centroids for the K-means clustering to be the winning RGB vectors in the map best corresponding to these samples at this latter part of the program" [10]. Then not only have I found the exact number of K, but those winning RGB vectors that best corresponding to these samples at this latter part of the program, are also the local minimum for the actual clusters. This meant that the convergence time for the centroids in the K-means clustering algorithm has speed up drastically and the total computation time for creating this clusters has decreased by a large amount. Overall I found that the RGB map initialization that had a lot of random perturbation in situ, utilizing simple euclidean distance similarity metric outperformed the other map initiations and the similarity metrics used.

I'm very confident that I accomplished a lot from this project. It would have been interesting to see what the affects would have been on each stage of the clustering, to try and use other types of learning functions. As well as trying and comparing the results of the K-means clustering to the hierarchical agglomerate clustering, or even changing the values of K for the K-means clustering. I feel extremely fortunate to have gotten the opportunity to learn such a plethora of machine learning methodologies, and I look forward to see what other opportunities await me, either as an undergraduate or as a graduate.

Appendices

A Rules for Implementing

Kohonen Self Organizing Map Algorithm

1. Initialize three RGB maps
    1.1. One initialized with random RGB vectors
    1.2. One initialized with RGB & black at the corners fading inwardly
    1.3. One initialized with RGB circles fading radially outward
2. Utilize three different similarity metrics for finding the comparing the similarities of any two RGB vectors
    2.1. Euclidean Distance: $d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)}$
    2.2. Cosine Similarity: cosim(a,b) = $\frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$
    2.3. Pearson Correlation Coefficient: r = $\frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}\sqrt{n\sum_{i=1}^{n} y_i^2 - \left(\sum_{i=1}^{n} y_i\right)^2}}$
3. Utilize a function to scale the neighbors of a winning RGB vector on a map within a certain radius and as a function of time, whereas the further away the neighbor the less it is scaled to be like the sample vector
    3.1. Gaussian Function: f (x) = $a \times e^{\left(-\frac{(x-b)^2}{2\times c^2}\right)}$
4. Initialize Sample RGB vectors
5. Initialize radius
6. Initialize number of iterations
7. ***Remainder* of The Kohonen Self Organizing Map Algorithm:**

```
For each iteration:
    For every RGB vector in the map:
        Compare with a sample vector for finding the winner RGB vector
    Scale the winner RGB vector and the neighboring RGB vectors to be more like the
    sample vector; the further away from the winner RGB vector the less it is scaled
    Update the time, radius, and iteration count
```

The K-Means Clustering Algorithm

8. Initialize K to the number of training sample RGB vectors
9. Initialize the centroids to be the winning RGB vectors in the map that best correspond to each of the samples at this latter stage of clustering from using KSOMs
10. Initialize the number of iterations to use K-means clustering
11. ***Remainder*** **of The K-Means Clustering Algorithm:**

```
For each iteration:
    For every RGB vector in the map:
        Use similarity metrics to compare the RGB map data to the centroids to see what
        clusters to put the corresponding RGB vector from the map into
    For every each cluster:
        Update the centroids by averaging together all the data from each cluster
    Check for convergence
```

Algorithm for Labeling Clusters

12. Initialize an empty list that will contain the labels the user will give corresponding to a specific cluster
13. Prompt the user for an RGB vector
14. Using similarity metrics, see if the cluster that best corresponds to the input vector is labeled
15. If it is labeled, return the label. Otherwise prompt the user for a label for the cluster.

Algorithm for Rendering a Checkerboard using the Clusters

16. Pick two clusters at random & render a checkerboard by randomly selecting points from each cluster alternatingly

References

[1] Shanmugapriya, B., and M. Punithavalli. "A Modified Projected K-Means Clustering Algorithm with Effective Distance Measure." *International Journal of Computer Applications* 44.8 (2012): 32-36. Web. 4 Apr. 2017. <https://www.researchgate.net/figure/220579196_fig6_Fig-6-k-means-clustering-of-n-dimensional-block-group-vectors-projected-into-SOM>.

[2] Germano, Thomas E. "Self Organizing Maps." *Self-Organizing Maps*. N.p., n.d. Web. 04 Apr. 2017. <http://davis.wpi.edu/~matt/courses/soms/index.html#Quality>.

[3] "Self-organizing maps." *Self-organizing maps*. N.p., n.d. Web. 04 Apr. 2017. <https://www.viscovery.net/self-organizing-maps>.

[4] Dozono, Hiroshi. "Application of Self Organizing Maps to Multi Modal Adaptive Authentication System Using Behavior Biometrics." *Applications of Self-Organizing Maps* (2012): n. pag. Web. 4 Apr. 2017. <https://www.intechopen.com/books/applications-of-self-organizing-maps/application-of-self-organizing-maps-to-multi-modal-adaptive-authentication-system-using-behavior-bio>.

[5] Galhardo, Carlos Eduardo Cardoso, and Werickson Fortunato De Carvalho Rocha. "Exploratory analysis of biodiesel/diesel blends by Kohonen neural networks and infrared spectroscopy." *Analytical Methods*. The Royal Society of Chemistry, 26 Mar. 2015. Web. 04 Apr. 2017. <http://pubs.rsc.org/en/content/articlelanding/2015/ay/c4ay02995j#!divAbstract>.

[6] "An Introduction to Self-Organizing Maps." *University of Manitoba - Information Services and Technology - Personal Homepage Server*. N.p., n.d. Web. 04 Apr. 2017. <http://home.cc.umanitoba.ca/~umsidh52/PLNT7690/presentation/SOM.html>.

[7] *Neuro AI*. N.p., n.d. Web. 04 Apr. 2017. <http://www.learnartificialneuralnetworks.com/kohonen.html>.

[8] *Unsupervised Learning Clustering*. N.p.: n.p., n.d. Http://www.mit.edu/~9.54/fall14/slides/Class13.pdf.

[9] "Determining the number of clusters in a data set." *Wikipedia*. Wikimedia Foundation, 29 Apr. 2017. Web. 04 May 2017. <https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set>.

[10] "What is the difference between SOM (Self Organizing Maps) and K-Means?" *Artificial intelligence - What is the difference between SOM (Self Organizing Maps) and K-Means? - Stack Overflow*. N.p., n.d. Web. 04 May 2017. <http://stackoverflow.com/questions/15754178/what-is-the-difference-between-som-self-organiz

ing-maps-and-k-means>.

[11] Champandard, Alex. *Self-Organising Maps For Colour Recognition*. N.p., n.d. Web. 04 May 2017. <http://ai-depot.com/Tutorial/SomColour.html>.

[12] Hindawi. "Self-Organizing Map-Based Color Image Segmentation with k-Means Clustering and Saliency Map." *International Scholarly Research Notices*. Hindawi Publishing Corporation, 07 June 2011. Web. 04 May 2017. <https://www.hindawi.com/journals/isrn/2011/393891/>.

[13] N.p.: n.p., n.d. Https://arxiv.org/ftp/arxiv/papers/1206/1206.0104.pdf.

[14] *Self-organizing Maps*. N.p., n.d. Web. 04 May 2017. <https://www.cs.hmc.edu/~kpang/nn/som.html>.

[15] N.p.: n.p., n.d. Http://www1.inf.tu-dresden.de/~ds24/lehre/bvme_ss_2013/pr_04_clust.pdf.