Cluster Classification/Optimization through KMean and Winner Take All Classification to Reduce the Color Range of .ppm Images

Timothy Ryan Lovvorn

April 22, 2017
ECE 471 Pattern Recognition
Undergraduate
tlovvorn@vols.utk.edu

Abstract

Often in classification algorithms problems while it is possible we are aware of the number of necessary classes, we are unaware of any prior data classifications (no training set). In this form of unsupervised learning we may classify a data set based upon an agglomerative method. The pixels of an image may be one such example and may be reclassified into clusters to map the image to a lower color representation. Through KMeans and Winner Take All agglomerative methods we may classify the pixels within a given image to a Kth designated cluster of pixels to map the image to this lower representation of color. Though visual image analysis and analysis of each epoch's data we may draw comparison between the two methods of classification based upon their conversion speed, image clarity, and overall reclassification numbers per epoch. With these methods I was successfully able to map the given image to successively smaller color spaces with a minimal loss of resolution and image clarity.

1.Introduction

It is often an issue in attempting to classify data that we are presented with the issue of having no formal reference point, no training set from which we may build our classifier. Because of this, a form of unsupervised learning is necessary, with work on the subject spanning back decades. An early paper recommend a form of clustering, but which you may note the spatial placement of data points and make inferences concerning their classification from such data [2]. This agglomerative method has been expanded upon to have many real world applications such as classifying of handwriting, reducing images to lower resolution, and the classification of biological cellular bodies based upon cell structure [1]. For our purposes we focus mainly on the clustering methods of WTA and KMeans.

The KMeans method seeks to randomly attribute a given number of clusters, which is assumed to be known, and then map datums to said clusters, adjusting the cluster center as necessary based upon each round until no more reclassification is necessary. This method is powerful as it means we need to have no prior knowledge about how any similar data would be classified, only the number of cluster and data set. The weakness of needing to know the number of clusters has also be addressed in past research, though the mean shift application or brute force guess and check methods [4]. Based upon this method of classification, we would expect a less than optimal solution on an issue as well as possible different solutions yielding convergence, as this does not account for data variance.

The Winner Take All method takes the approach of KMeans a step further, and reclassifies the cluster values each time it has a new datum classified to it, in addition to recalculation at the end of every epoch. I expect this to be a slightly more powerful method of classifier as this is also reclassifying based upon the positon of each pixel, variance is having an effect here. This method

is more specialized and may be extended to data sets where clustering is not as dense, with greater variance in the data set [3]. There are still limitations, as the learning rate will have a direct impact on the convergence of the space, as well as wether or not it is capable of converging.
The C++ source code is contained within the appended KCC.cpp file, and all relevant information on to run the experiment via script and read the created files in within the appended README.txt file appended to this report. All graphical representations were created through 3rd party software in Microsoft Excel spreadsheets as well as the imagemagick conversion software in Hydra.

2. Technical Approach

2.1 KMeans

We begin by generating a set of K pixels, which I created by randomly selecting values between 0 and 255. After this, we computate the distance between each pixel in our image set, and each cluster center in our randomly generated set of K pixels, with the minimum distance having the image pixel be assigned to said K pixel. After this is performed for all pixels in the image. The K clusters are recalculated. This is done by summing the R, G, and B values for each pixel classified to a particular K cluster, then averaging them by the total number of pixels classified to said cluster:

$$K_{new} = \frac{\sum_{i=0}^{n} p_i}{n}$$

Where P_i is a pixel classified for cluster K, n is the number of P pixels, and this is performed for the respective R, G, B values of each K cluster point. This is performed for all K clusters. This process and the classification process is then repeated until the number of pixels reclassified for a given iteration is 0.

2.2 Winner Take All

This method is performed in a manner very similar to KMeans. As with KMeans, a set of K pixels is randomly generated, and distances between each K cluster and pixel in the image is calculated. The closest K value to a pixel is declared to be the "winner," similar in manner to the above and the given pixel is classified to the given K value. The difference now is for each pixel, when it has been assigned to a given K pixel, the K cluster value is recalculated based upon the

distance the cluster is from the given pixel:

$$K_{i+1} = K_i + \alpha(P_i - K_i)$$

Where alpha is the learning rate. To create faster convergence, I assigned a dynamic learning rate, wereby if the number of reclassified pixels was larger and the previous number of reclassified samples, the learning rate was halved:

$$if(New\ number\ of\ \ Resorted\ Pixels > Old\ number\ of\ Resorted\ Pixels)$$
$$then\ \alpha = \frac{\alpha}{2}\ else\ \alpha = \alpha$$

This ensures as there is less of a margin of change as fewer pixels require reclassification. This is performed every time a pixel is classified. As with KMeans, after all pixels are classified, the cluster center is recalculated in the same manner as KMeans. This operation is also continued until the number of reclassified pixels in an iteration is 0.

2.3 Degradation Metrics

As there are various smoothing effects given by our methods for our method of generating our reduced images, using a strictly empirical method of distinguishing between the images would likely yield uncharacteristic results. Based upon this, I utilized the method recommend by Dr. Qi: I did a visual comparison between the newly generated reduced image, and the original image. Based on the differences between different K values of image reductions as well as I made my observations on the effects of each method.
In addition, to measure the difference between the Classification methods, for each iteration though all the pixels, the number of pixels reclassified was tabulated. The total number of iterations necessary for the image space to converge was also tabulated. For different methods, as well as different cluster numbers, these will be compared.

3. Experiments and Results

3.1 The Data Set

The data set consisted of a singular image file of the .ppm type, as illustrated below in figure 1. The image consisted of 480 x 480 RGB pixels capable of representing 256^2 different colors.

Figure 1: The original Unaltered Image

The image was originally in a .ppm format, but for all subsequent presentation purposes the image has been converted into a .jpg format for platform ease.

3.2 KMeans

3.2.1 Empirical Metrics

For each iteration, the number of misclassified samples was tabulated, as demonstrated by figure 2. In addition, the total number of epoch taken to reach a converge of 0 reclassified images is also displayed in figure 3.

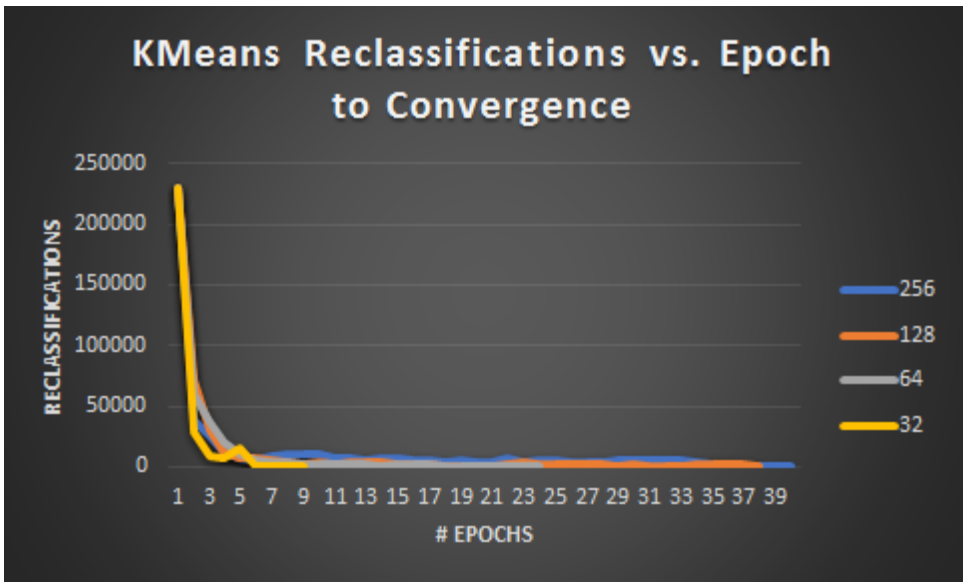

Figure 2: The number of reclassifications relative to number of clusters for KMeans Classification

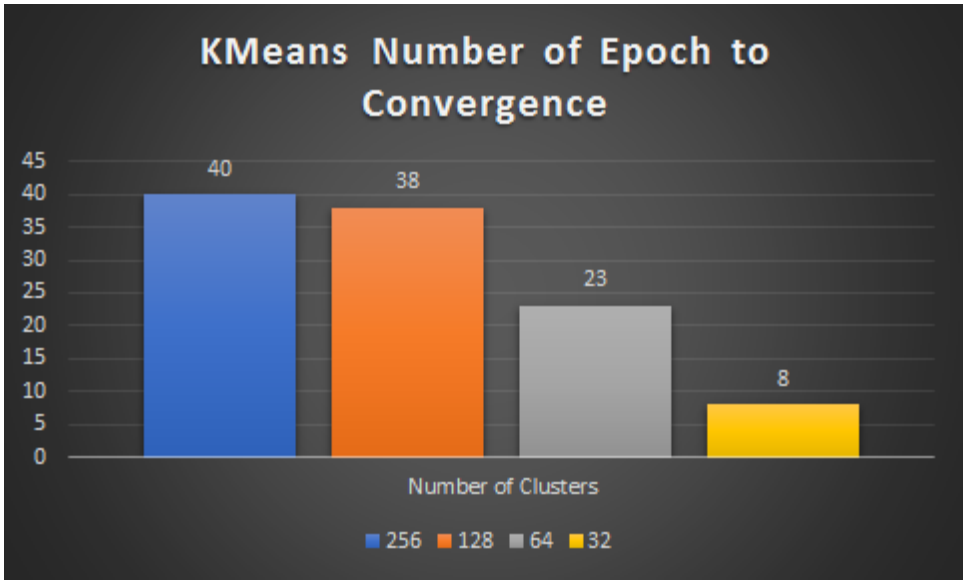

Figure 3: The total number of epoch necessary for KMeans Convergence

A few interesting inferences may be made, the most obvious being the more colors, the longer period of time it takes to reach convergence. Interestingly, this is not always the case. As on a separate run of the algorithm I reached convergence of the 256 cluster trial with only 17 epochs. This follows with our knowledge however, as the initial cluster points are being determined randomly. If the random points happen to align rather well with the actual clustering of the similar pixels locations, fewer epochs will be necessary for convergence. The downward trend in time is also expected, as the fewer classifiers which exist, the wider the margin between each,

therefore it would give wider ranges for a pixel to classify in. Also this shows the sheer weight of computational time taken with this algorithm, as p^k, where p is the number of pixels and k is the number of clusters, must be computed each time. Figure 2 also reveals an interesting phenomenona where occasionally the number of reclassified pixels will increase between epochs, but this is also to be expected as well, as each iteration requires we reclassify a mean each time we finish all reclassifications. This can occasionally results in a large reclassification as a cluster center is sudden moved just far away from a cluster of pixels for them to all require reclassification. This makes sense as in an image, large number of pixels will have the same value (i.e. a large, symmetric patch of green on a picture will all have the same pixels), so the reclassification of a single pixel value may cause a large number of individual pixels with said value to reclassify.

3.2.1 Visual Metrics

For visual metrics, the following are all the images from the KMeans iterations:

Figure 4: KMeans 256 Clusters

Figure 5: KMeans 128 Clusters

Figure 6: KMeans 64 Clusters

Figure 7: KMeans 32 Clusters

As expected there is a general "pixelation" affect over iterations as the number of clusters decreases. The overall saturation of the image increases as well. The colors seem to darken or lighten dramatically, in particular with the yellow and red colors. In the final 32nd cluster we see the most dramatic shift, as it is dominated by the yellow and indigo colors. We also notice these colors are not the same shade as the most dominate colors in the images, namely the light yellow and purple. We also see that similar colors, such as red, may be completely overtaken by their most similar counter part, in this case indigo. This is indicative of our expectations as we are classifying solely based upon the mean of the data. As such, the actual distribution of the data is not taken into account so it is entirely possible to get colors which, while are the correct color, are far off in shading as only relative proximity to a general cluster is accounted for. This is also why their is a general degradation of color as the number of clusters decreases, their is less room for any sort of variance, and our images reflect all of these notions.

3.3 Winner Take All

3.2.1 Empirical Metrics

As with KMeans, for each iteration, the number of reclassifications per clustering was tallied. This was also done alongside the tally of the total number of epoch necessary for convergence of the space shown in figures 8 and 9.

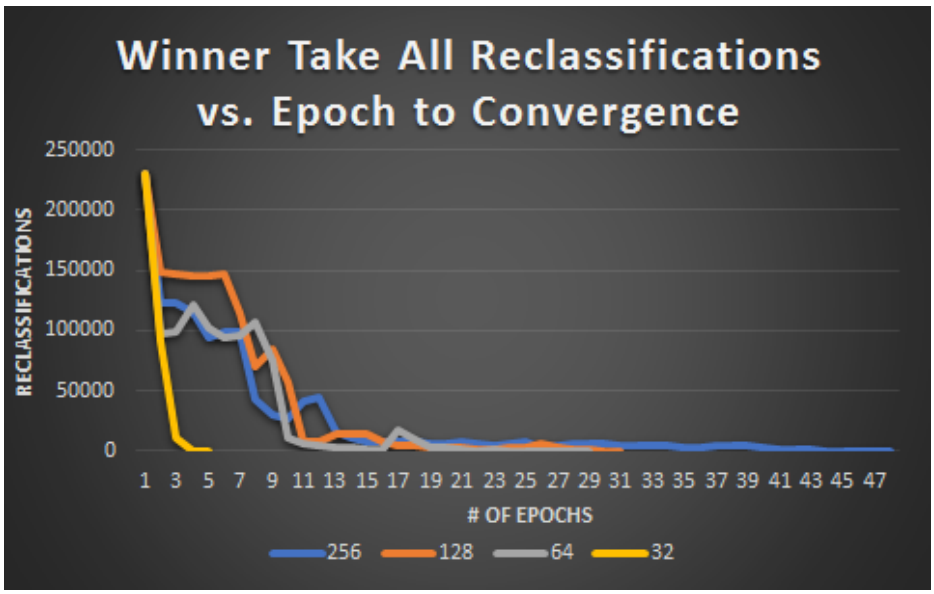

Figure 8: WTA reclassifications vs number of Clusters

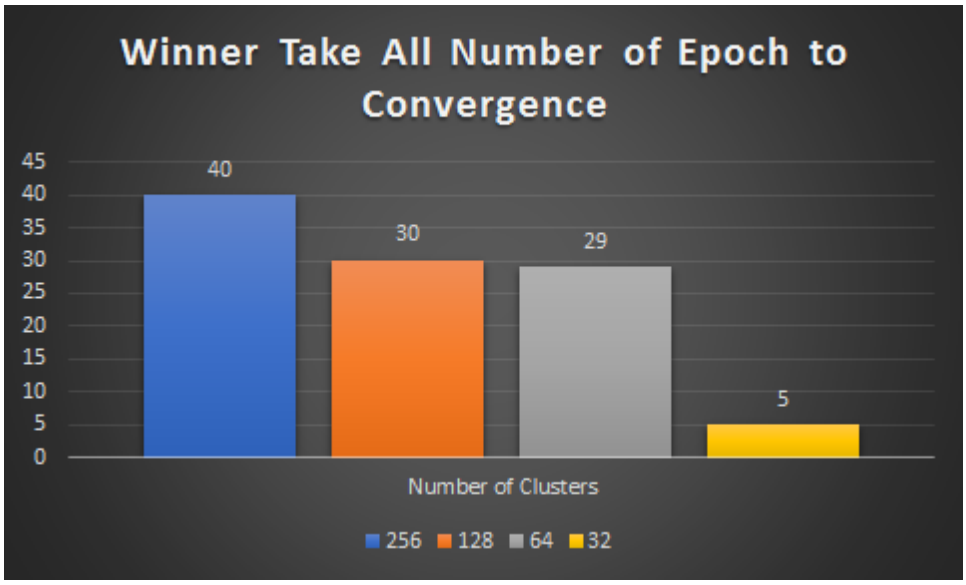

Figure 9: WTA Number of Clusters vs. Epochs to Convergence

Similarly to KMeans, we notice an overall trend downward as the number of clusters decreases in the time it take for convergence. We also notice performance for the 128 and 32 clusters is actually better for WTA than it was for KMeans. Also of note is how there is more of a generalized leveling off of the ability of WTA to progress to lower numbers of classifiers. That makes sense with our data as we would expect this sort of behavior to occur as your learning rate

changed. As our learning rate became to large, the number of reclassifications would increase or level off. Once the learning rate had be decreased by a necessary amount the overall number of reclassifications would indeed decrease, as is the behavior we see. The general trend toward WTA talking less iterations as well makes sense as there is the additional movement of the cluster center for each reclassification, allowing for faster convergence. On a whole, while each iteration will take more time proportional to how many clusters are utilized, this method will converge more rapidly than KMeans, and our data reinforces this assertion.

3.2.1 Visual Metrics

For visual metrics, the following are the images from the Winner Take All Iterations:

Figure 10: WTA 256 Clusters

Figure 11: WTA 128 Clusters

Figure 12: WTA 65 Clusters

Figure 13: WTA 32 Clusters

Similar to KMeans we see a general degrading and pixelation effect on the image. Interestingly, it seems to retain more of it's individual colors, even down to the lower clustering. If we consider the algorithm, this behavior makes sense. WTA reconfigures the cluster center each time a value is classified. Because of this, it will take an ultimate value much more representative of the overall data set, because it is accounting for the variance of clusters, instead of just the mean value. From this, while we should still see degradation, colors should not be as easily overpowered as we would see in KMeans with an overall more clear image, and this is indeed the result.

4. Summary

In summation, utilizing the agglomerative clustering methods we may create a form of classification useful when only limited, the number of classes, information about the data set is known. Though both methods have their strengths and weaknesses. The KMeans method is faster to complete a round, although it generally takes more epochs total and only accounts for the mean values of the data centers, not the data variance so more information may be lost. WTA takes longer to complete each epoch but generally does complete in fewer epoch and is capable of accounting for variance of pixels in a cluster, allowing for better data mapping. In a way, this is similar to our previous data reduction methods, PCA and FLD. Like FLD, KMeans sought more to simply classify the data as well as possible with out accounting unduly for how each class of data was scattered. Like PCA, WTA sought to more best represent the data with as little loss as possible and did account for the variances of data within classes when representing said

data.

Creation of this project was not overtly difficult, and allowed me to review my skills in image manipulation. This software is also quite useful as it serves as a form of image reduction, with a message that may be used to display an image on a less powerful machine when necessary with minimal reduction in the image. Considering the extreme prevalent of legacy software, in particular in modern Cyber Physical Systems such as the Electrical Grid SCADA which utilizes HMI and Visualization Workstations to display data, this is a very useful skill. In broader terms we now have a generalized method of classification when limited knowledge about the data set is known. As this is most commonly the case when gathering many forms of data in the world, this has extreme power and may be used in a myriad of industries.

5. References

[1] Fritzke, B. (1994). Growing cell structures—a self-organizing network for unsupervised and supervised learning. *Neural networks*, *7*(9), 1441-1460.

[2] Kauth, R. J., Pentland, A. P., & Thomas, G. S. (1977). Blob: An unsupervised clustering approach to spatial preprocessing of mss imagery.

[3] Maass, W. (2000). On the computational power of winner-take-all. *Neural computation*, *12*(11), 2519-2535.

[4] Ray, S., & Turi, R. H. (1999, December). Determination of number of clusters in k-means clustering and application in colour image segmentation. In *Proceedings of the 4th international conference on advances in pattern recognition and digital techniques* (pp. 137-143).