Daniel Peterson 109091561

K-Means Clustering

Image segmentation involves separating an image into many different meaningful segments or parts according to some characteristic, such as color. The goal of segmentation is to extract data or objects from the background. Each part that is separated through segmentation is significantly different from each of the other parts relative to the segmentation characteristic. Conversely, each pixel within a given region is similar to all of the other pixels in that given region relative to the segmentation characteristic. Segmentation is often used to find edges and to assist in the analysis of an image by making the parts of an image simpler and more recognizable than in the image prior to segmentation.

A parameter must be determined prior to the start of image segmentation. This is the domain of the segmentation, or the characteristic by which the image will be partitioned. This characteristic could be something such as intensity or color, as long as it is different enough throughout the image to produce useful partitions. For example, if an image was entirely white, such as a blank text page, then there would be no reason to use color for segmentation because there would not be a large enough difference in color throughout the image to warrant segmentation.

The next thing to determine is the segmentation technique. Some techniques include thresholding segmentation, region-based segmentation, and edge-based segmentation. Thresholding takes a threshold value, either determined manually or through an algorithm to determine the mid-point between two categories. The threshold value is applied to every pixel in the image and the pixels are assigned to one group if they are greater than or equal to the threshold value and another group if they are less than the threshold value. Thresholding tends to not perform very well in images with a lot of noise. This is because the pixel values are not entirely well defined and the thresholding becomes less effective.

Region-based segmentation groups pixels together if they have similar characteristic values and splits pixels which have dissimilar characteristic values. Initial points must be “seeded” in order to create a region of similar points. A variance value is applied to each region in order to ensure that the maximum difference of points within a region does not exceed that variance, otherwise the region is invalid. Likewise, if two regions have pixel values whose maximum variance is within the variance value, then those two regions should be combined into one region.

Edge-based segmentation involves applying a label of edge or non-edge to each pixel in an image. Pixels which are labelled as non-edge can be combined in the same category if they form a connected region. Each region of these pixels will be a segmented part of the image. The pixels which are labelled as edge can be labelled as such manually or by an edge-detection filter. Each of these segmentation techniques will perform differently depending on the image. The segmentation technique which will be discussed in this paper is k-means clustering.

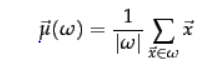
K-means clustering is a technique in which segments an image into *k* different groups of points, called clusters. There are *k* centers (since there are *k* clusters) and each center is determined either randomly or through an algorithm which will be discussed later. After the centers are determined, each pixel is compared to the center of the cluster and the distance between pixel and each cluster center is minimized. The distance is the difference between the pixel and the center of the cluster according to the segmentation characteristic. For example, if the segmentation characteristic is image intensity, then the distance is the difference in image intensity between the given pixel and the intensity of the cluster center. After each cluster center is compared, the pixel is added to the cluster with which it has the smallest difference between. After the pixel is added, the cluster average is recalculated by averaging the intensity (or other segmentation characteristic) of each of the pixels in that cluster. This is performed for each pixel every time the averages are re-calculated, which means that pixels can switch clusters. After a number of iterations, the algorithm will converge (none of the pixels will change clusters so the averages will be final).

The result of k-means clustering is *k* groups of points containing pixels which are the closest to the average of those clusters. This means that throughout the entire image, no pixel in a given group is more similar to the center (average) of another cluster than its own. However, it does not mean that two pixels which are in different groups are not similar to each other. For example, if two cluster centers have intensity values of 5 and 16, pixels which have an intensity of 10 or less will be a part of the 5 cluster and pixels with an intensity of 11 or more will be a part of the 16 cluster. However, this means that two pixels (one with an intensity of 10 and another with an intensity of 11) can be in different clusters, yet be more similar to each other than any of the cluster centers. Additionally, certain solutions are better than others and this is determined by the sum of the squared distance for each pixel. Each pixel has a distance from the cluster center and the total distance for the image is determined by summing the distances for each pixel in the image. Different initial cluster centers result in different summed distances, with the minimum of these being the optimal solution. If cluster centers are chosen randomly then the optimal solution is not likely to be reached. These scenarios can arise due to poor selection of the initial cluster centers or an inadequate number of cluster centers.

Since random selection of cluster centers can lead to sub-optimal results, specialized heuristics and algorithms can be used to determine cluster centers. One example of these is the k-means++ algorithm. This algorithm helps to avoid the sub-optimal cluster centers which can occur if the centers are seeded randomly. The k-means algorithm’s goal is to minimize the average squared distance from points to their cluster centers. The average squared distance is the sum of the distances to the center for all points in the cluster, divided by the number of points in the cluster. The k-means algorithm carefully seeds the initial cluster centers in order to increase the probability that the optimal clustering will occur.

The algorithm starts by selecting a cluster center at random using all of the data points. Then the distance between each of the points and that center point is calculated. After all of the distances have been calculated, the next center is chosen by using the weighted probability of the sum of all distances relative to the point. This means that the distance from this new center to the center is chosen using a weighted probability function to determine the location where the majority of the other data points are located. Ideally, this new center will be placed at a location which minimizes the average squared Euclidean distance to each of these points. There can be many locations which meet this criteria, so the exact location is chosen at random from a selection of these locations. After the new center is chosen, the distance between each of the points and their nearest center is calculated and then a new center is chosen until all *k* centers are selected. Once all of the centers have been determined, k-means clustering can occur to group the points into more optimal clusters. K-means++ also decreases the time to converge as well as the error from the optimal set, meaning it improves both speed and accuracy.

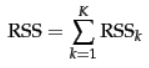
The function to determine a cluster center is as follows:



In this function, *x* is the point, *w* is the cluster, and *u* is the mean centroid of the data points. Ideal clusters are clusters with the centroid as its center of gravity and clusters which do not overlap with one another. The points are assigned to clusters in order to minimize the residual sum of squares (RSS). This RSS can be calculated for each cluster with the following formula:

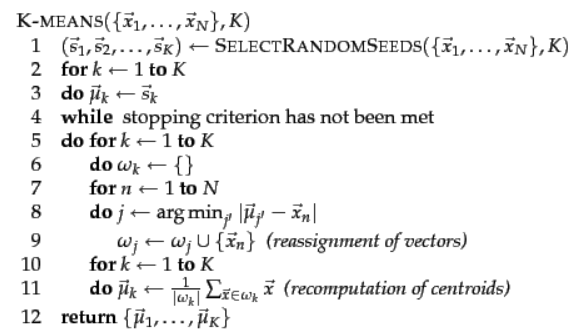


This formula is followed by the following formula to calculate the sum of the RSS values for each cluster:



In these formulas, *k* represents the number of clusters, *x* is the point within the cluster, *w* is the cluster and *u* is the centroid of the cluster. The first formula takes the value of the data point and calculates the squared distance between the point and the centroid of the cluster. This distance is calculated and summed for each point in the cluster to get the total squared distance for a particular cluster. Then in the second formula, the sum of the squared distances is summed for each cluster to get the total sum of squared distances for all points in all clusters.

These two formulas are used to find an optimal RSS (minimum) which means that the data points are located to the best possible cluster center. An example of the k-means clustering algorithm which accomplishes this can be seen below:



The algorithm selects random seeds as opposed to using k-means++ to determine optimal seeds. Then, for each cluster, it assigns each data point to a cluster centroid. If the stopping criterion has not been met, meaning that at least one point changes its assigned cluster in the past iteration, then each cluster is iterated through to find the best cluster for a particular point. Each point is added to its closest cluster and then new cluster centroids are calculated based on the new average distances from all points in that cluster. After the new cluster centers are determined, then the clusters are iterated through again (with the new centroids) to determine which data points should be added to the new clusters. The process will repeat until none of the points change clusters (because the centroids will never change if the points never change). If none of the points change, then the algorithm will end and return the cluster centroids and the points associated with them.

In many cases, setting the stopping criterion as when points do not change clusters can lead to very long runtimes, so some alternative conditions may need to be applied. As an example, the algorithm could stop after a certain number of iterations to guarantee a maximum runtime. Another alternative condition could be when the RSS value is below a certain threshold. Although it may not be optimal or the minimum for the set, the points might need to be within a certain value and this condition allows for that. This is fine, because there is never a guarantee that the minimum value for the set will ever be reached, the only guarantee is that the set will converge at some point, and even that may take a long time.

The time complexity for k-means in general is O(KNM) where *K* is the number of clusters, *N* is the number of points, and *M* is the number of cluster centers a particular point needs its distance measured to. Some applications of k-means clustering are for PET scans to differentiate between different types of tissues, analyzing groups of bacteria for patterns, and MRI scans of the brain to try and find tumors. After using k-means++ to initialize seeds for the clustering, the total time to cluster the data points decreases and the accuracy increases. By applying k-means clustering to images, important distinctions can be made between problematic entities and the rest of the body. K-means, along with other segmentation techniques and other image processing techniques allow medical professionals to analyze data much easier and faster than ever before. The benefits of these techniques extend into many branches of the medical field, including x-rays, neuroimaging, cat scans, ultrasounds, and more. By providing easier analysis of images, medical professionals can provide better, more accurate care for their patients and more accurate diagnoses for problems.