Implementing and Evaluating K-Means Clustering

Christopher Linscott
clinscott@oxy.edu
Occidental College

1 Abstract

The following report dives into an approach to clustering images together: based on purely pixels. After performing both Fuzzy C-Means and K-Means on select color images, it was found that these methods were not useful in creating meaningful clusters, given only pixels as data points.

2 Methods

2.1 Approach/Framework

The general approach to grouping images is the idea of clustering, where the main goal is "to cluster pixels into different image regions, which are regions corresponding to individual objects, surfaces or natural parts of the objects" (Roy). As image pixels are generally unlabeled, the common approach is utilizing a clustering algorithm to group them together. In the context of my COMPS project, being able to separate images means being able to separate them apart based on their differences in objects, shapes, or environment. The main goal is not to recognize the objects, but to recognize meaningful differences or similarities between two images based on their visual similarity. While this is an unsupervised version of machine learning (allowing for mainly testing), there is no response "class", or that the human needs to associate meaning with each cluster it generates.

The two clustering algorithms employed are Fuzzy C-Means and K-Means. With the help of tutorials, I implemented them from scratch and from a library, and tested them in clustering an image where the objects were distinct in color versus where the objects were more similar. To allow the images to be clustered, they were loaded using OpenCV's image read function, creating arrays of data points of size three to represent each value for each channel (red, green, blue). After clustering upon these images using either algorithm, the following images were plotted using matplotlib, coloring pixels of the same cluster with the same color to show the results of the computation.

2.2 Datasets

2.3 Algorithms

A very well-known, partition-type clustering algorithm is K-Means. Partition-based clustering refers to a harder (meaning more strict) form of partitioning where every datapoint can only be in one cluster. The main goal of this algorithm is to create partitions of data points, by creating points of relevance called clusters, which minimize the distance from any given data point to any given cluster. The distance metric utilized by K-Means is Euclidean distance where the distance from a data point x_i to a cluster x_j with d dimensions: $D_{i,j} = \sum_{l=1}^{d} \sqrt{|x_{il}^2 - x_{jl}^2|}$. In my own implementation (for learning, understanding, and documentation), I utilized Euclidean distance for only 2 dimensions via: $D_{i,j} = \sqrt{|x_j - x_i|^2 + |y_j - y_i|^2}$.
In combination with my own implementation, I utilized Lloyd's algorithm to compute and converge on centers of clusters, which is the same as for the library utilized in the tutorial, sklearn.

2.3.1 Pseudocode

Algorithm 1 K-Means: Lloyd's Algorithm

```
1: Initialize k clusters to be at initial positions of (0, ..., 0).
2: Select k number of initial centers (centroids) from datapoints (at random)
3: while Centroid positions don't change do
4:     for n datapoints do
5:         for k clusters do
6: Calculate distance from data point n to cluster k
7:
8:         end for
9: Store closest cluster for each data point n
10:    end for
11: New centroid position = mean of all subsequent data point positions
12: end while
```

The soft (fuzzy) counterpart to the K-Means algorithm is the Fuzzy C-Means (FCM) algorithm. The FCM algorithm is "softer" or "fuzzier" as unlike K-Means, it allows for a datapoint to coexist in several different clusters with different values of membership (i.e. how much it relates to that given cluster). In comparison to the K-Means, the FCM algorithm is more popular for image segmentation because it's more computational efficient (O(n) versus O(knt) (Source)), applicable to multichannel data (i.e. colored images), and has the ability to model uncertainty in the data (Source). In the context of the high number of pixels being grouped upon, as well as the channels of color which apply to colored images, it's clear that an algorithm like Fuzzy C Means is a very good option for image segmentation.

Fuzzy C-Means, in a very similar fashion to K-Means, will utilize a Euclidean distance metric. However, the data points now have an additional set of k membership values, k being the number of clusters. FCM utilizes membership values of the data points, where every datapoint has a set of k membership values adding up to 1, with k being the number of clusters. These membership values will be randomly initialized for each datapoint, and tweaked depending on the distance metric from a datapoint to a cluster. The centers of the cluster will now be tweaked based on the mean of the data points as well as their membership values. In other words, the centroid's datapoint will be most affected by data points which have a membership high for that cluster. Mathematically, this is all represented by first calculating the centroid positions by computing

$$\frac{\sum_i^n (m_{ij})^f * dp_i}{\sum_i^n (m_{ij})^f}$$

with n datapoints dp, a cluster j and a given membership value of a datapoint i for a cluster j m_{ij}. The new f value represents the fuzzifier, or what will convert a hard output into a soft output (i.e. less hard clustering); a value of 1 equates to K-Means, whereas infinity equates to all values being equally in the same cluster. After computing the new centroid values, the data point's membership values will be equal to the euclidean distance from a datapoint i to cluster j, divided by the total euclidean distance to all clusters from the same datapoint. This is calculated as follows:

$$m_{ij} = \frac{1}{\sum_{i=1} k(\frac{|dp_i - c_j|}{|dp_i - c_k|})^{\frac{1}{f-1}}}$$

. In psuedocode, we'd have something like:

2.3.2 Pseudocode

Algorithm 2 Fuzzy C-Means

```
1:
2: $f \leftarrow 2$ Initialize k clusters to be at initial positions of (0, ..., 0).
3: Select k number of initial centers (centroids) from datapoints (at random)
4: while Centroid positions don't change do
5:     for n datapoints do
6:         for k clusters do
7: Calculate distance from data point n to cluster k
8:
9:         end for
10: Store closest cluster for each data point n
11:     end for
12: New centroid position = mean of all subsequent data point positions
13: end while
```

3 Evaluation

Evaluation of these clustering algorithms were performed using "simple" images, or images with objects of few, distinct colors, versus more complex images where the colors of objects often overlapped. The first proposed method for comparing K-Means and Fuzzy C-Means is utilizing a confusion matrix and computing accuracy, precision, and recall to compare on select images. A secondary method of utilizing Intersection of Union (IoU), which is simply the "ratio of the number of common pixels between X and Y to the total number of pixels in X and Y" (Mittal), mathematically calculated by

$$\frac{|X \cap Y|}{|X| + |Y|}$$

To test and obtain these values, we utilized the BLANK dataset as a ground truth to compare the algorithms with each other and the ground truth, the "ground truth" being human-segmented images.

4 Discussion