Assignment 2- (K-Means & Auto-Encoder)

Ramesh Pavan Pothamsetty

14th November 2021

1 Assignment Overview

The goal of the assignment is to work with Cifar-10 dataset. In the first part of the assignment, I have implemented k-means clustering by taking ten centroid images and later forming ten clusters of the overall test data by calculating Euclidean distance. In the second part of the assignment, I have used a neural network for the auto-encoder and decoder. After that performed k-means on the encoded image data generated from Cifar-10 train dataset.

2 Dataset

Dataset is taken from http://www.cs.toronto.edu/~kriz/cifar.html.

Downloaded dataset already has images segregated as train and test datasets. Entire dataset contains a total of 60,000 random images from 10 classes, which are divided into 6 batches of 10,000 images each. One batch comes under test dataset and rest fall under train dataset.

Coming to preprocessing of the data, each image in the dataset are color images with size (32*32*3). I have converted them to gray scale images and have performed normalization on the gray images. The resulting sizes after preprocessing in part 1&2 are (10000*1024) and (50000*1024) respectively.

3 Python Editor

I have used Jupiter Notebook IDE for implementation.

4 Part-1

K-Means Clustering

In K-Means clustering the idea is that a machine learns to do a task in an unsupervised way. It tries to learn the inherent patterns in the data which it will use to predict the output class which contrasts with supervised learning, where a machine tries to learn by establishing a relationship between the input and the labeled output.

In the part one of the assignment, I have first extracted the file that is downloaded from the website. After that I have unpickled the data as per the steps provided on the website. Then I have taken the data and labels into two variables. Later I performed the conversion of content of data variable to gray scale images followed by normalization and then reshaped them. The final resulting image data is stored in updated array which is of size 10000*1024.

K-Means Clustering Implementation

In the implementation part I have defined three functions for initializing centroids, calculating distance and assigning the images to clusters and then for updating centroids.

initialize_centroids function: Since this dataset has 10 classes I have randomly taken 10 images as centroids.

assign cluster function: This function calculates the distance between each image and all 10 centroid images and assign it to the cluster which gives least distance between image and the respective cluster centroid.

compute_centroid function: This function updates the randomly selected centroid images to more appropriate ones during model training.

For training the model I have taken termination condition as no.of iterations which is equal to 10.

Results

For calculating the performance of the model based on the quality of the clusters I have calculated Average Silhouette Coefficient and Dunn's Index. The results are as follows:

ASC for test dataset: 0.05406823008462551

DI for test dataset: 0.055636835992773456

5 Part-2

Auto-Encoder

Auto-encoder is also a method in unsupervised learning which is used compress the data and learns how to reconstruct the data from the reduced compressed data. Auto-encoder by design reduces data dimensions by ignoring noise. During the model training auto-encoder uses back propagation to minimize the reconstruction loss.

Implementation

In the part two of the assignment, I have implemented an auto-encoder using neural network from keras library. I have unpickled the train dataset for five batches as per the procedure given on the website. Later I stored all unpickled data of 50000 images in a single 2D array. In total this part of the assignment has 50,000 images from the same 10 classes given to the encoder. The resulting size of the updated data array is after converting to gray scale and normalization is (50000*1024).

In the auto-encoder I have used ReLU as activation function for both encoder and decoder layers. For training the model I have used Adam as optimizer and Mean Squared Error for loss. Then K-Means clustering is performed on the encoder output using the part-1 implementation of the assignment.

Results

Below are the results for ASC which was obtained by performing K-Means on the encoder output.

ASC for test dataset: 0.06839757