ASSIGNMENT 2

Q2b. Implement Bike vs Horse Classification:

Ans:

Dataset: I got the dataset from the LMS which had two folders bikes and horses. 'bikes' folder contains 80 images of bikes and 'horses' folder contained 81 images of horses.

Procedure: Bag of visual words (BOVW) is a popular image classification technique. It is inspired by NLP's famous bag of words (BOW) technique. Here we use image features as 'words' i.e., unique patterns found in an image. Here we represent an image as a frequency histogram of visual words or image features. Features consists of keypoints and descriptors.

Training

- So, at first we detect keypoints and descriptops from each image using **SIFT or SURF** algorithm.
- Then we run **K-Means** clustering over all the descriptors. Each centroid that we got from the K-Means clustering will work as a bean of the histogram.
- Then for each image we make a **frequency histogram**.
- Then we use image histograms and their respective label as our training data and used **SVM** algo from Sklearn to train.

<u>Testing</u>

- **So for testing when we get an image first we** detect keypoints and descriptops from each image using **SIFT or SURF** algorithm. For each image's each descriptor we assign it a cluster center according to the previously trained K-Means on training data.
- Then for each image we make a **frequency histogram.** Here also use same no of beans or cluster centroids as training images.
- Then we use our SVM model to predict the test image's label.

<u>Implementation:</u>

<u>Training:</u>

- At first, we made two lists named 'images' and 'labels'. 'Images' contains the relative paths of all the images (bikes and horses) & 'labels' contains all images' corresponding label. We encoded 'Bike' as '0' and Horse as '1'.
- Then we randomly shuffled our dataset and did a train test spilt.
- Then we detected keypoints and corresponding descriptors using OpenCV's kpts, des = sift.detectAndCompute(im, None)
- Then we created two lists. First one is 'imgwise_des_list' which is a list of tuples. Each tuple contains relative path of the image and its descriptor

array. And the second one is des_list which contains all the possible descriptors of all the images.

- Then we run Kmeans clustering (taking k as 200) upon des_list and obtained an array of shape (200,128). Its each row represents a cluster centroid of 128 dimension.
- Then we assign each image's every descriptor to some cluster centroid and represent each image as a frequency histogram.
- Finally we get a 'histograms' array of shape (155,200). Whose each row represents histogram of one image.
- Then we perform normalization upon each column of 'histograms' matrix using sklearn's StandardScaler().
- Then we trained sklearn's LinearSVC() throughout 80000 iterations upon normalized 'histograms' matrix and its corresponding labels.

Testing:

- **So for testing when we get an image then we** detect keypoints and descriptops from each image using **SIFT or SURF** algorithm. For each image's each descriptor we assign it a cluster center according to the previously trained K-Means on training data.Then for each image we make a **frequency histogram.** Here also use same no of beans or cluster centroids as training images.Then we use our SVM model to predict the test image's label.

Observation:

We got an accuracy of 95.8333%

- **We implemented above mentioned algo on the cifar –10 dataset as well .**
- **Cifar-10 dataset was divided into 5 batches of training images and one batch of test image. Each batch contained 10000 images and their respective labels.**
- **We appended all images in one list named 'images' and their respective labels into an another list named 'labels'.**
- **Then implemented the algo as we did for our first dataset.**

Observations:

- **While implenting on cifar -10 for some training images no keypoints or descriptors were getting detected by SIFT. So we dropped that images .**

Only 233 images were like that out of 50000 images. So those did not affect our training procedure.

- **We tried different ML models for training:**

Model	**Accuracy On Test Data**

1.LinearSVM with 10000 iterations	**10.08%**
2. LinearSVM with 20000 iterations	**23.05%**
3.XGBClassifier()	**30.01%**
4.Logistic Regression	**12.03%**
5.SVM with C = 1.81 with 50000	**69.83%**
6. SVM with C = 1.81 with 70000	**70.29%**