Computer Vision (CO462) Assignment 3
Object Classification

Vignesh K (14CO252)

Chosen Categories

The following object categories of the Caltech101 dataset have been chosen to perform the experiments:

- Faces_easy : 435 images
- airplanes : 800 images
- Motorbikes : 798 images

Classifiers Used

- Gaussian Naive Bayes Classifier
- Decision Trees

Methodology

The following steps have been adopted to solve the problem of object classification.

- An appropriate **train_split** value (Fraction of examples in each category chosen for training the classifier) is chosen for training.

- **SIFT features (128-dimensional)** are extracted from the train images and collected in a single matrix of 128-dimensional feature points.

- K-means clustering with a suitable value of **K** is used to cluster these 128-dimensional feature points extracted from the training data.

- Each image's feature points are assigned to one of the K clusters and thus a **histogram of frequency vs cluster index** is obtained for each image. This histogram is normalized by dividing each cluster's frequency by the total number of feature points extracted for that image. The normalized histogram gives us the **bag of words representation** for the image.

- Having obtained the bag of words representation for each image in the train set, we use this set for training the classifier **(Gaussian Naive Bayes or Decision Tree)**.

- The same feature extraction process is followed for the test set. The SIFT features of the test set are **directly converted to the bag of words representation using the K cluster centroids computed for the train set.**

- The trained classifiers are then used to make predictions on the test set given its bag of words representation.

- The results are obtained in the form of **Accuracy** of prediction and the **Confusion Matrix.**

Results

The experiments have been performed by varying two parameters. One being the value of **K used for K-means clustering** and the other being the value of **train_split** which dictates the percentage of samples used for training. Each call to K-means clustering performs **10 attempts** to cluster the data and the set of centroids corresponding to the attempt with the **best compactness** is chosen finally.

Variation with K keeping train-split constant at 0.5

Gaussian Naive Bayes

K	Accuracy (%)	Confusion Matrix
20	79.35	Assignment_3/Results/Varying_K/GNB_confusion_matrix_K_20.png
40	83.87	Assignment_3/Results/Varying_K/GNB_confusion_matrix_K_40.png
70	87.12	Assignment_3/Results/Varying_K/GNB_confusion_matrix_K_70.png
100	86.23	Assignment_3/Results/Varying_K/GNB_confusion_matrix_K_100.png
150	89.68	Assignment_3/Results/Varying_K/GNB_confusion_matrix_K_150.png

Decision Tree

K	Accuracy (%)	Confusion Matrix
20	70.60	Assignment_3/Results/Varying_K/DTree_confusion_matrix_K_20.png
40	72.57	Assignment_3/Results/Varying_K/DTree_confusion_matrix_K_40.png
70	73.65	Assignment_3/Results/Varying_K/DTree_confusion_matrix_K_70.png
100	69.22	Assignment_3/Results/Varying_K/DTree_confusion_matrix_K_100.png
150	75.91	Assignment_3/Results/Varying_K/DTree_confusion_matrix_K_150.png

Variation with train-split keeping K constant at 70

Gaussian Naive Bayes

train_split	Accuracy (%)	Confusion Matrix
0.3	86.45	Assignment_3/Results/Varying_train_size/GNB_confusion_matrix _train_size_03.png
0.4	86.23	Assignment_3/Results/Varying_train_size/GNB_confusion_matrix _train_size_04.png
0.5	87.12	Assignment_3/Results/Varying_train_size/GNB_confusion_matrix _train_size_05.png
0.6	84.64	Assignment_3/Results/Varying_train_size/GNB_confusion_matrix _train_size_06.png
0.7	86.09	Assignment_3/Results/Varying_train_size/GNB_confusion_matrix _train_size_07.png

Decision Tree

train_split	Accuracy (%)	Confusion Matrix
0.3	66.71	Assignment_3/Results/Varying_train_size/DTree_confusion_matrix_train_size_03.png
0.4	68.36	Assignment_3/Results/Varying_train_size/DTree_confusion_matrix_train_size_04.png
0.5	73.65	Assignment_3/Results/Varying_train_size/DTree_confusion_matrix_train_size_05.png
0.6	70.64	Assignment_3/Results/Varying_train_size/DTree_confusion_matrix_train_size_06.png
0.7	71.36	Assignment_3/Results/Varying_train_size/DTree_confusion_matrix_train_size_07.png

Analysis

- A clear observation from the results is that the Gaussian Naive Bayes classifier outperforms the Decision Tree classifier by a significant average of 15% in terms of accuracy.

- A possible explanation for the inferior performance of decision trees could be the shortage of images for training. This is supported by the fact that as we increase the size of **train_split,** the accuracy of the Decision Tree classifier increases by upto **7%**. In contrast, as we already know, one of the major advantages of the Gaussian Naive Bayes classifier is that it requires a small amount of data to train the model. Hence even with train sizes as small as **30% of the total examples**, we get accuracies as high as **86%** with the Gaussian Naive Bayes classifier, where as the Decision Tree classifier is **20%** lower at **66%**. The accuracy of the Gaussian Naive Bayes classifier shows almost no changes **(only 2-3%)** with increase in the train size thus proving its robustness to the availability of data.

- Another reason for Decision Trees failing to perform on par with the Gaussian Naive Bayes classifier is that the bag of words representation of images is a histogram of normalized frequencies in the range 0 to 1. This means that we are dealing with **continuous valued features**. But **Decision Trees are known to work well for discrete valued attributes**.

So clearly, when it comes to classifying continuous valued attributes they fail to perform like the Naive Bayes classifier which is suitable for discrete as well as continuous valued attributes.

- Lastly, the increase in the value of K results in higher accuracies for both the approaches. The **Gaussian Naive Bayes classifier's accuracy increases by almost 10 %** as K varies from **20 to 150.** The **Decision Tree classifier's accuracy increases by 5 %** for the same variation. An intuitive explanation for this variation is that as K increases, the clustering algorithm is able to discern more clusters which are representative of more granular features of the object in question. **Highly granular features are more informative than a high-level summarization of features obtained by smaller values of K.** In short, by choosing smaller values of K we are losing more information about the object. At the same time, **increasing the value of K beyond a certain value may also become detrimental to the performance of the classifier.** Very large values of K tend to give clusters which represent highly localized features thus **losing track of meaningful features which are present at a more global level in the form of an aggregation of these highly localized features.** Another obvious disadvantage of increasing the value of K is the increased runtime for clustering.