5/24/2017

Image Classification

CV Assignment 2

Chinmay Dhawan

I6171015

CONTENTS

OBJECTIVE

The following report discusses image classification techniques using SIFT features as bag of words and compares how various classifiers like KNN, weighed KNN and SVM affects the performance/accuracy by varying their parameters.

BACKGROUND

The SIFT features are extracted for each image. Each feature(descriptor) is of length 128 and each image has different number of features extracted depending on its properties and parameters value set for SIFT. All the features are grouped into one single matrix (descriptor_matrix), and then grouped into clusters by k-means++. A histogram of each of the image is then made according to the cluster each descriptor of the image belongs to. For classification, a histogram of test image is made by similar process and the individual histograms are compared using chi distances. The image is classified according to the labels of k nearest neighbours. KNN with weights as 1/distance is also implemented along with SVM and the results are compared.

IMPLEMENTATION DETAILS

IMPROVING RUNNING TIME BY SAVING TRAINING DATA

Instead of reading training images and extracting descriptors for every run from the disk. The descriptor matrix(descriptor_matrix) is only read for the first run and stored as '.mat' format for more efficiency. Another matrix(descriptor_matrix_info) is saved which contains the label and number of descriptor extracted for each image to help extract information from descriptor matrix.

```
%% save and load training data
% (done only once)
% save('sift_features', 'descriptor_matrix', 'descriptor_matrix_info');
  clear variables; close all;
  load sift_features.mat;
```

K-MEANS IMPLEMENTATION

K-means is a key step for the whole process as it lays the ground for making histograms. K-means ++ is chosen so that the initial points chosen for the centers are as far away from each other as possible and hence, the total energy for clustering can be minimized. Further, the clustering is run multiple times (5 to 20), and the clustering which returns the minimum energy is returned. This is to ensure to get the best clustering as possible and not to deviate too much from the results for multiple runs. The value of k is varied from 50 to 300.

```
%% K-means clustering
num_clusters = 50;
[centres, assignments, energy] = vl_kmeans( double(descriptor_matrix),....
                                          num_clusters, 'verbose', ...
                                          'Initialization', ...
                                          'plusplus', ...
                                          'NumRepetitions', 20 );
```

HISTOGRAM MAPPING:

Histograms are made from both test and training images according to the below function. It is made by calculating Euclidean distance of individual features/descriptors from the centers of the clusters and making a histogram according to the cluster label each feature belongs to.

```
function [histogram_values] = sift_to_histogram_features(image_descriptors,
centers)
    num_clusters = size(centers, 2);
    distances = pdist2(image_descriptors', centers');
    [~, label] = min(distances,[], 2);
    h = histogram(label, (num_clusters) );%, 'Normalization',
'probability');
    histogram_values = h.Values;
end
```

Both normalized and unnormalized histograms are tried as the number of descriptors vary. However, Chi-squared distances on average give better results on histograms not normalized.

K-NEAREST NEIGHBOUR (WEIGHED AND UNWEIGHED)

K-nearest neighbor is implemented, and the results are compared for both weighed and unweighted implementations. The value of k strongly influences the outcome with value of k equal to 1 giving results almost close to random i.e., 20% and value of k = 7 reaching accuracy as high as 80%.

The distance metric used is chi-squared distances.

The weight used is equal to inverse of the distance from the test data, so the data point closer to test data point will have more weight in the voting.

The value of k is varied from 2 to 15.

```
num_labels = 4;
num_test_images = 200;
 for i = 1:num_test_images
     for j = 1:num_labels
         distance_column = sorted_chi_distance(1:5, i);
         matched_label = matched_labels(1:5, i);
         sum_weights(j, i) = sum( 1./distance_column(matched_label == j-1)
);
     end
 end

[~, predictions_weighed] = max(sum_weights, [], 1);
difference = (predictions_weighed - 1)' - actual_labels;
accuracy_1 = length(find(difference == 0))/length(difference);
```

SVM

A linear kernel of SVM using libsvm is implemented as another classifier for comparison to KNN. The results can be seen in the result section. Apparently, linear SVM doesn't give significantly improves the results over KNN. A few other models such as multiple binary SVMs integrated together maybe able to give better results but aren't realized in the scope of this report.

```
model_linear = svmtrain(train_label, train_data, '-t 0');
[predict_label_L, accuracy_L, dec_values_L] = svmpredict(test_label,
test_data, model_linear);
```

ANALYSIS AND RESULTS

NUMBER OF CLUSTERS

Firstly, a plot of varying k, number of clusters for k-means is shown against accuracy of various classifiers.

All other factors (SIFT Parameters and K for Nearest Neighbour) are remained constant.

Number of features = 154,116; K for Nearest Neighbour = 7.

Note: Since K-means results are dependent on its initialization, k-means is run 5 times and the minimum energy is chosen for each value of k, to reduce its probabilistic nature and have better comparisons.

Figure 1 shows the plot of Accuracy against Number of clusters. Blue represents KNN, Red represents weighted KNN and green represents SVM. The plot is for 6 values of k, i.e., 50, 100, 150, 200, 250, and 300.

As can be seen in the figure k = 200, seems to be a good value of k, constraining the given set of parameters for SIFT and KNN.

Another point that can be observed is that the performance of SVM performs worse for smaller values of k than KNN (weighed or Non-Weighed). However, for larger values of k, SVM performs better than KNN. This can be accounted to the fact that since KNN relies on distance-based metric, it's performance usually decrease if number of features are increased by a substantial amount (curse of dimensionality).

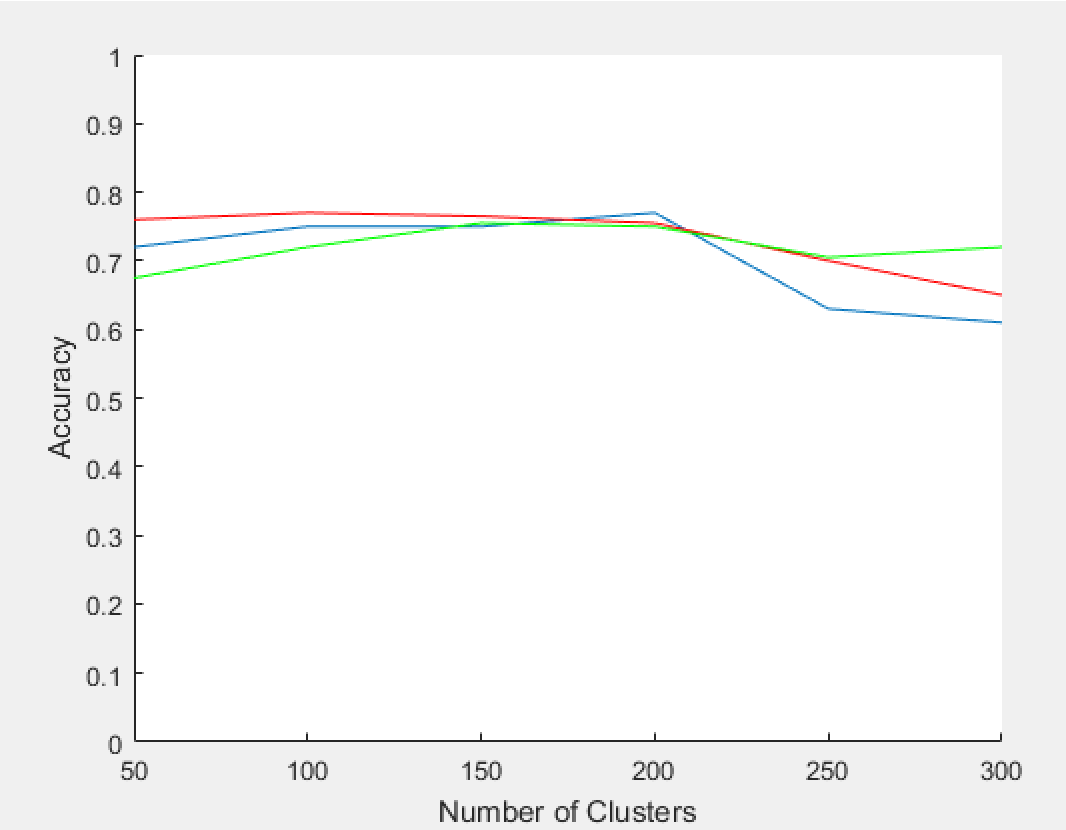

Figure 1: Plot of accuracy vs number of clusters for various classifiers

The best accuracy achieved for the above figure is 77% by KNN at k = 200, and the worst is also by KNN which is 61% at k = 300.

NUMBER OF NEIGHBOURS

Number of clusters are fixed to 200, since from above analysis, that seems to give the better results. The k means is run 7 times to find the lowest energy clustering, following which accuracies are determined for k varying from 2 to 20.

The values of k from 2 to 10 are showed in Table 1, and Figure 2 shows the plot of accuracies weighed and unweighted KNN against values of (2 to 20).

	2	3	4	5	6	7	8	9	10
KNN	69	77	76	72	73	72	70	71	73
Weighed KNN	73	76	79	77	74	72	71	69	70

Table 1

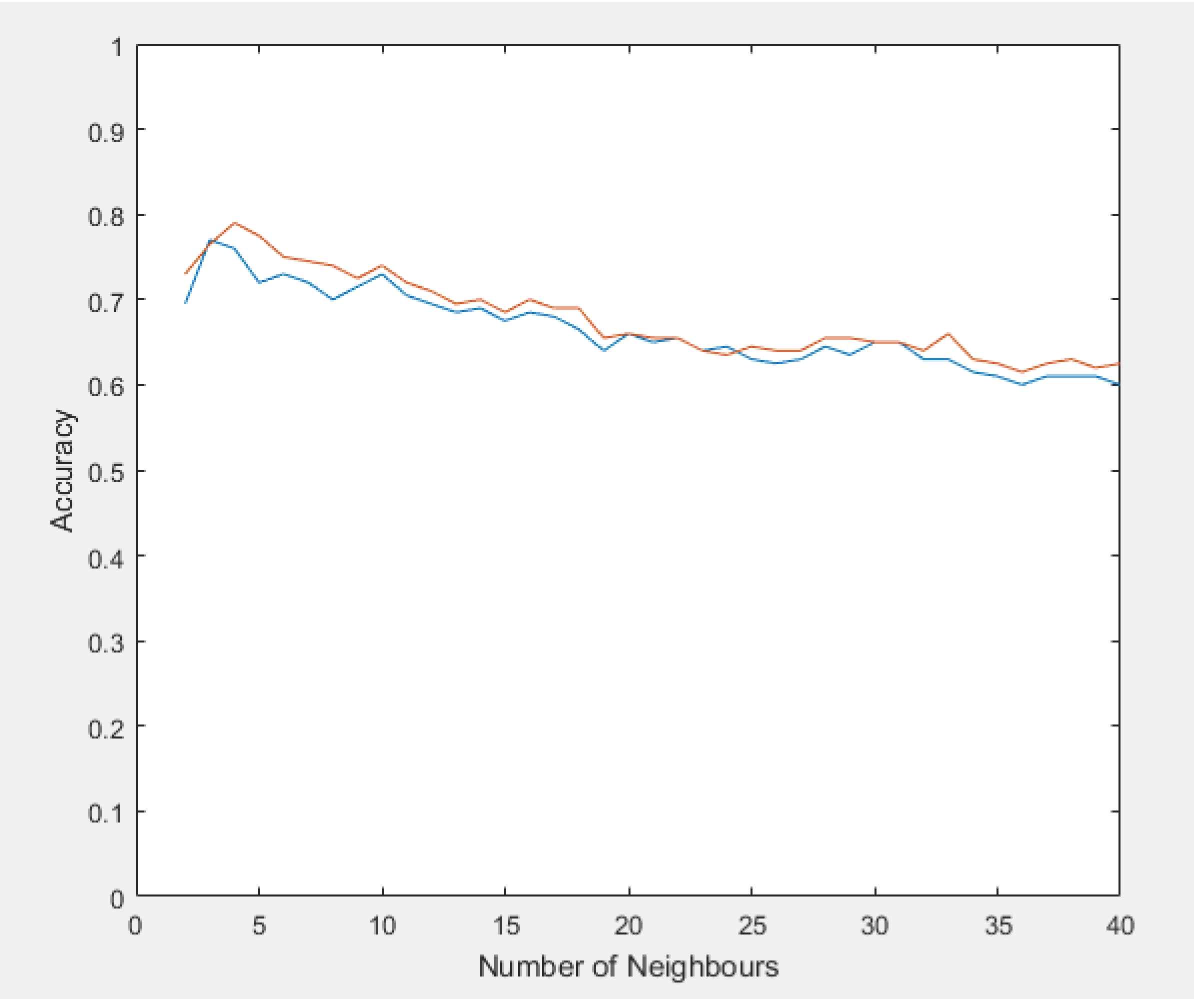

Figure 2: Accuracy vs Number of Neighbours, red and blue curve represent Weighed KNN and KNN respectively.

It can be observed that with weighed KNN, the results are improved in a general scenario. This is because it can overpower the data points with appreciably varying degree of distances from its neighbours. The best accuracy achieved for this plot is 79% for k = 4 for weighed KNN.

The optimum value of k seems to vary from 3 to 6, for dictionary size of 200. The computation time of varying k for KNN isn't observable since the distance of test data from each point is already calculated.

AMOUNT OF TRAINING DATA AND NORMALIZATION

If the number of training images are reduced by half, the accuracy is reduced for all the classifiers, although the training time is improved. This can be said because the more the training data, more information classifier has, until a saturation point is reached where adding more training data doesn't add any more information but noise.

CHANCE PERFORMANCE

A random classifier would give an accuracy of 25% because there are 4 classes. So, the chance of labelling any training image to a correct class is 1/4 = 0.25.

CLASSIFICATION RATE FOR EACH CLASS

Classification rate for individual classes for the best classification result from Figure 2 (Weighed KNN: k = 4, and Number of clusters = 200)

- Test images for cars correctly classified: 46, accuracy = .92
- Test images for planes correctly classified: 40, accuracy = .80
- Test images for bikes correctly classified: 28, accuracy = .56
- Test images for faces correctly classified: 44, accuracy = .88

Bike images are most difficult for KNN to classify while accuracy for car images is highest, i.e., 92%.

EXAMPLES:

MISCLASSIFIED IMAGES AND NEAREST NEIGHBOUR

 

Classified as Incorrectly classified as bike.

Reason: the features such as the road and the body of car matches closely with that of bike and the pavement.

Chi-squared distance = 86.8032

Plane misclassified as Face: Nearest Neighbour is shown below the test image of plane. It's difficult to analyse why these images match as the features are based on gradients calculated using intensity values, and it sometimes it could be hard to analyse just by looking; There can be some smaller features which could be similar to each other not observable by eye.

Bike misclassified as plane. Nearest Neighbour is shown below.

Face incorrectly classified as plane. The pattern of the shirt in first image closely resembles the way windows are aligned and may provide a good match.

CORRECTLY CLASSIFIED IMAGE EXAMPLES:

Images shown below are correctly classified, and their nearest neighbours are also shown with them. All the images shown below have very close match based on chi squared distance.

They are just images taken at same location at different times.

Visibly very similar images

The shapes of certain parts of the vehicle are similar in structure.

Again, images taken at different time instants.