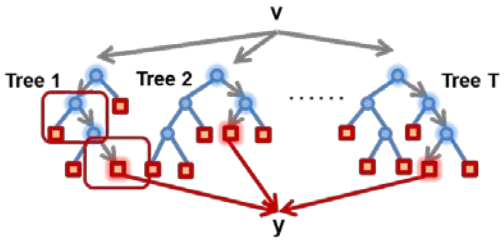
EE462, EE9SO25, EE9CS728

Machine Learning for Computer Vision

Coursework1 on Randomised Decision Forest [50% mark]



Release on 26 Jan, the report due on 16 Mar (midnight)

This course work requires Matlab programming. Use the provided Toy Spiral data and Caltech101 image categorisation dataset. In all questions, as you prefer, you can use the provided code set of Random Forest (it, however, exploits multiple internal functions, and offers not necessarily complete comments for all) or any existing toolbox/library e.g.

https://github.com/karpathy/Random-Forest-Matlab

http://vision.ucsd.edu/~pdollar/toolbox/doc/

http://code.google.com/p/randomforest-matlab/

http://www.mathworks.co.uk/matlabcentral/fileexchange/31036-random-forest

Submission instructions:

One joint report by each pair

Page limit: 4-6 A4 pages per report with 10 font size (use the IEEE standard double column paper format, either in MS word or latex).

http://www.pamitc.org/cvpr16/files/egpaper_for_review.pdf

http://www.pamitc.org/cvpr16/files/cvpr2016AuthorKit.zip

Give insights, discussions, and reasons behind your answers on the scope of lectures, and try to visualise any interesting observations to support your answers. ***Quality and completeness of discussions within the page limit*** will be marked.

Source codes are not mandatory unless specified: optionally they can go to appendix, which does not count for the page limit.

Submit the report **in pdf** through the Blackboard system. No hard copy is needed. Write your full names and CID numbers on the first page.

If you have questions, please write at https://goo.gl/a3fFre, or contact

Dr. Guillermo Garcia-Hernando (g.garcia-hernando@imperial.ac.uk)

Dr. Binod Bhattarai (b.bhattarai@imperial.ac.uk)

In this coursework, we learn the machine learning algorithm called randomised decision forest (RF). Following the lectures on RF, we implement the algorithm, see into the core parts of the algorithm, while training and testing it on the toy data set and Caltech101 image categorisation data set. Try also different parameter values of RF and see their effects. Discuss your implementations, obtained results, advantages and limitations of RF.

In **Q1** and **Q2**, we use the provided Toy Spiral data. In order to access the data, start Matlab, go to the directory where the provided files are. Open the main_guideline.m and do the followings:
Do initialisation.

```
>> init;
```

We provide three toy data sets, each of which is a set of 2D points with their class labels, and the Caltech101 image categorisation data set. Load the Toy Spiral data set here.

```
>> [data_train, data_test] = getData('Toy_Spiral'); % {'Toy_Gaussian',
'Toy_Spiral', 'Toy_Circle', 'Caltech'}
```

The training and testing data are in the format of

```
data_train(:,1:2) : [num_data x dim] Training 2D vectors,
data_train(:,3) : [num_data x 1] Labels of training data, {1,2,3}.

data_test(:,1:2) : [num_data x dim] Testing 2D vectors, 2D points in
the uniform dense grid within the range of [-1.5, 1.5],
data_test(:,3) : N/A.
```

Plot the training and testing data.

```
>> plot_toydata(data_train);
>> scatter(data_test(:,1),data_test(:,2),'.b');
```

Q1. [10] Training Decision Forest

Given the training data set, we first generate multiple data subsets by Bagging (Boostrap Aggregating). Show e.g. four data subsets, and discuss the results, and the way you did bagging e.g. the size of each data subset, whether it was with or without replacement.

Now we take one of the data subsets generated above, and grow a tree by recursively splitting the data.
Split the first (i.e. root) node. We try a few split feature functions randomly chosen (given a split function type e.g. axis-aligned) and thresholds (we can vary the value of ρ, i.e. the degree of randomness parameter), decide the best one in terms of information gain. Show the split functions,

the respective class histograms of the node and its two children nodes, and measure the information gains. Discuss the results.

We now recursively split the nodes in the way above to grow the tree. When the tree growth is done, we save the class distributions in the leaf nodes. Visualise the class distributions of some leaf nodes, and discuss the results. What stopping criteria did you use? Explain the reasons.

Similarly, we grow multiple trees, a tree per a data subset.

Q2. [10] Evaluating Decision Forest on the test data

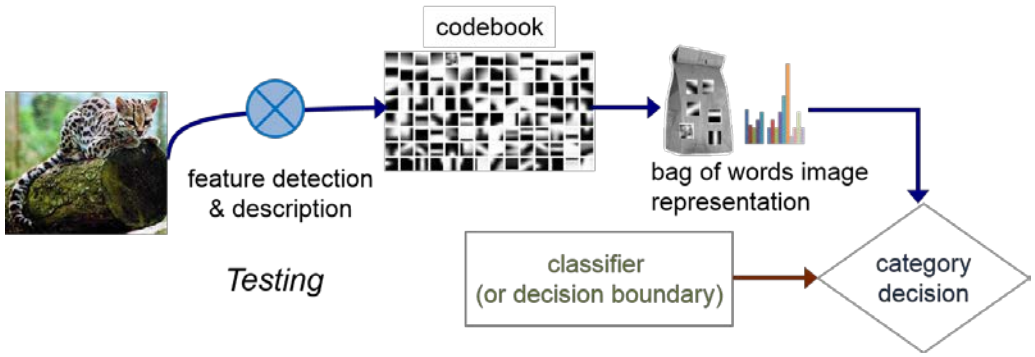
Using the random forest trained above, we evaluate novel data points in the test dataset. Let us grab the few data points (for instance, the points given below) and evaluate them one by one by the learnt RF. Show and discuss the results. Visualise the class distributions of the leaf nodes which the data point arrives at and the averaged class distribution.

```
>> test_point = [-.5 -.7; .4 .3; -.7 .4; .5 -.5];
```

We evaluate all the data points in the dense 2D grid, and visualise their classification results, colour encoded.

Try different parameter values of RF and see the effects. RF has a few important parameters, which need to be set to achieve good results. Here we try changing the number of trees, the depth of trees, and the degree of randomness parameter. Show and discuss the results for different parameter values.

Q3. Experiment with Caltech101 dataset for image categorisation



We apply RF to the subset of Caltech101 dataset for image categorisation. Use the provided Caltech101 dataset. We use 10 classes, 15 images per class, randomly selected, for training, and 15 other images per class, for testing. Feature descriptors **d** are given. They are multi-scaled dense SIFT features, and their dimension is 128 (for details of the descriptor, see http://www.vlfeat.org/matlab/vl_phow.html).

Q3-1. [10] K-means codebook

We randomly select 100k descriptors for K-means clustering for building the visual vocabulary (due to memory issue). Open the main_guideline.m and select/load the dataset.

```
>> [data_train, data_test] = getData('Caltech'); % Select dataset
```

Set 'showImg = 0' in getData.m if you want to stop displaying training and testing images. Complete getData.m by writing your own lines of code to obtain the visual vocabulary and the bag-of-words histograms for both training and testing data. Provide you own lines of code in an appendix of your report. Correctness and efficiency of the code will be marked. You can use any existing code for K-means (note different codes require different memory and computation time). Show, measure and discuss the followings:

- vocabulary size,
- bag-of-words histograms of example training/testing images,
- vector quantisation process.

Q3-2. [10] RF classifier

Train and test Random Forest using the training and testing data set in the form of bag-of-words obtained in **Q3-1**. Change the RF parameters (including the number of trees, the depth of trees, the degree of randomness parameter, the type of weak-learners: e.g. axis-aligned or two-pixel test), and show and discuss the results:

- recognition accuracy, confusion matrix,
- example success/failures,
- time-efficiency of training/testing,
- impact of the vocabulary size on classification accuracy.

Q3-3. [10] RF codebook

In **Q3-1**, replace the K-means with the random forest codebook, i.e. applying RF to 128 dimensional descriptor vectors with their image category labels, and using the RF leaves as the visual vocabulary. With the bag-of-words representations of images obtained by the RF codebook, train and test Random Forest classifier as in **Q3-2**. Try different parameters of the RF codebook and RF classifier, and show/discuss the results in comparison with the results of **Q3-2**, including the vector quantisation complexity.

Optional Further discussion

This part carries no mark, should go to an appendix of your report.

Discuss Boosting (and Boosting cascade) and RF, both as committee-machines in tree-structures. What if we apply Adaboost classifier to the Caltech101 image categorization problem here? What are the pros and cons of RF and Adaboost? No result (or implementation) of Adaboost is needed.