COMPUTER VISION AND IMAGE PROCESSING (CS676A)

SCALABLE RECOGNITION WITH A VOCABULARY TREE

GROUP 6

PRAMOD CHUNDURI - 13221

KRITI JOSHI - 13358

INTRODUCTION:

The method is used to find best matches of the queried image in a huge database of images. The tree approach makes the search faster and scalable. A subset of UKY data set (1000 images) was used for training and testing. The detailed explanation of each step and the performance results obtained are mentioned below. To run the codes, use following **commands:**

g++ mainReport.cpp train.cpp queryReport.cpp features.cpp -o main `pkg-config --cflags --libs opencv`

./main

EXECUTION:

STEP 1: Feature Extraction And Tree Formation

SIFT features were first extracted from the given database images and then by using k-means clustering algorithm in a hierarchical manner, vocabulary tree was constructed. A 128-bit SIFT feature was stored at each node till this point. The tree size was decided by branch factor(k) and depth of tree (level)

STEP 2: Populate Tree

The extracted features were then assigned appropriate leaf nodes, by tree traversal using Euclidean distance as similarity measure. At each node, its weight was calculated and a reverse mapping to the images whose features belong to the subtree was stored.

$Weight\ of\ node\ = \frac{N}{N_i}$ *where* $N = 1000$ *and* N_i *is number of images passing through the node i*

STEP 3: Query And Performance Evaluation

The images used for training were one by one queried and performace was evaluated.

Two performance measures were used, namely:

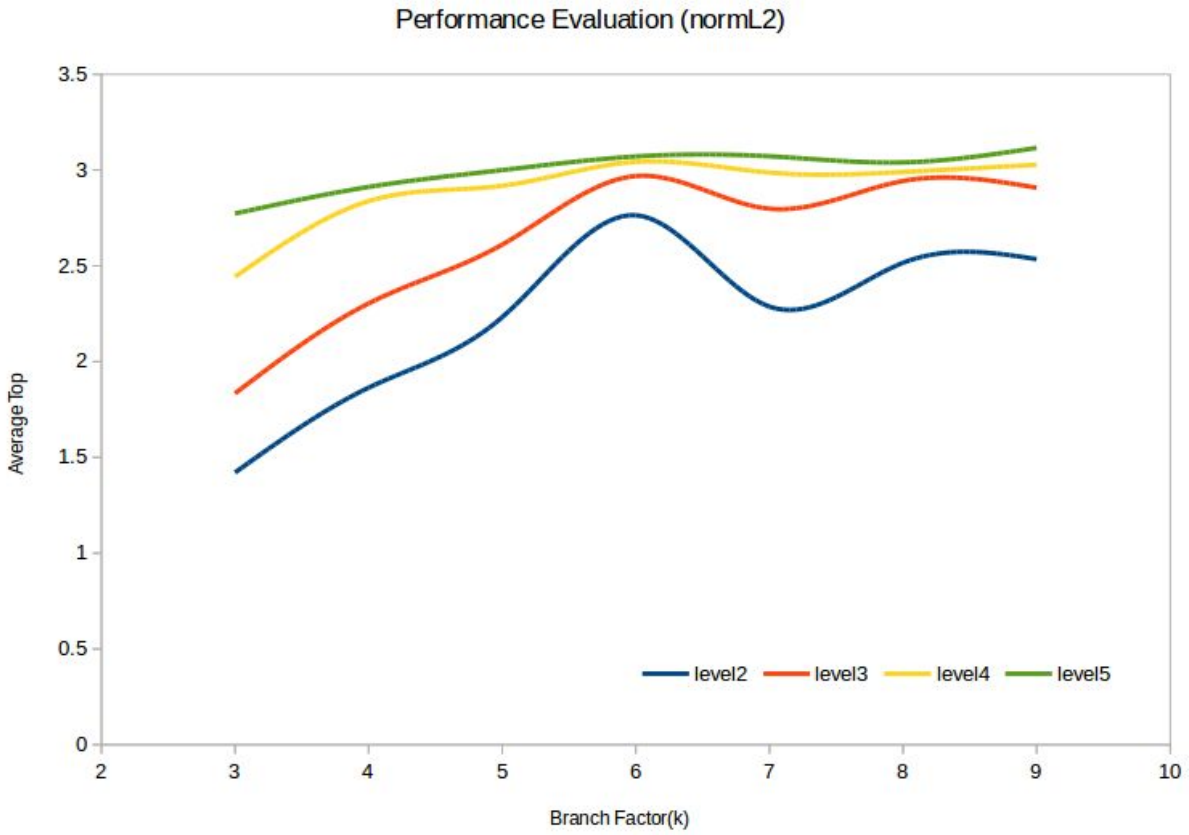
- Average number of similar images in the top 4 best matches found by the method (Range:0-4) *Refer: http://vis.uky.edu/~stewe/ukbench/*
- Percentage of images with all the similar matches present in some top x%. *Refer: http://www-inst.eecs.berkeley.edu/~cs294-6/fa06/papers/nister_stewenius_cvpr2006.pdf*

RESULTS:

Performance Evaluation Using Average Top Matches:

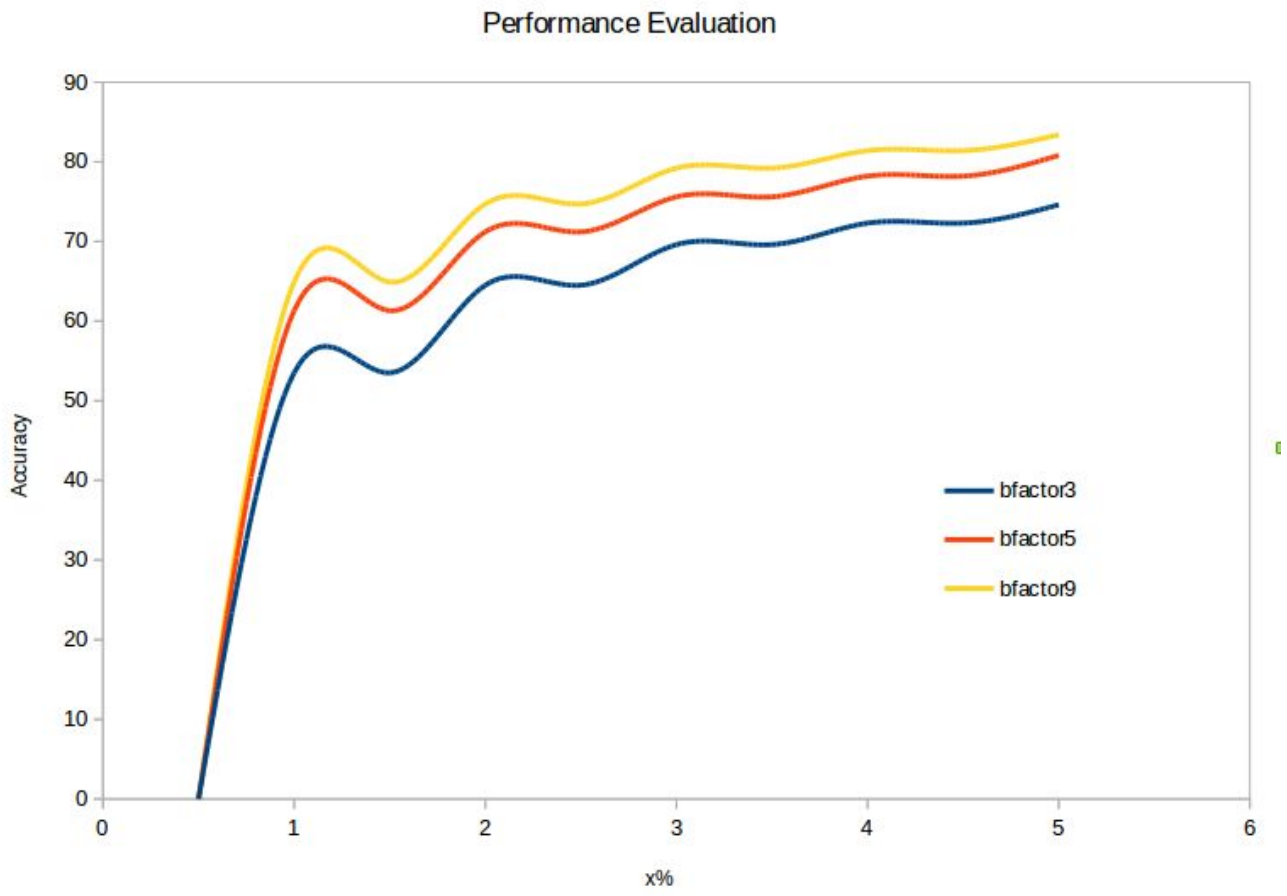
Branch factor(k)	Depth factor(L)	Accuracy (L1 norm)	Accuracy(L2 norm)
3	2	1.446	1.421
3	3	2.036	1.834
3	4	2.690	2.443
3	5	2.897	2.773
3	6	2.627	2.863
4	2	1.944	1.864
4	3	2.554	2.304
4	4	2.944	2.838
4	5	2.408	2.912
4	6	1.369	2.981
5	2	2.341	2.232
5	3	2.826	2.611
5	4	2.648	2.918
5	5	1.424	3.000
5	6	1.129	3.055
6	2	2.457	2.224
6	3	2.915	2.764

6	4	2.106	2.969
6	5	1.216	3.043
6	6	1.115	3.071
7	2	2.548	2.287
7	3	2.773	2.798
7	4	1.492	2.987
7	5	1.134	3.072
8	2	2.724	2.516
8	3	2.716	2.943
8	4	1.361	2.99
8	5	1.107	3.04
9	2	2.758	2.535
9	3	2.479	2.908
9	4	1.271	3.028
9	5	1.1	**3.116**



Performance Evaluation Using Average number of Images having all other three similar Images in top x%:

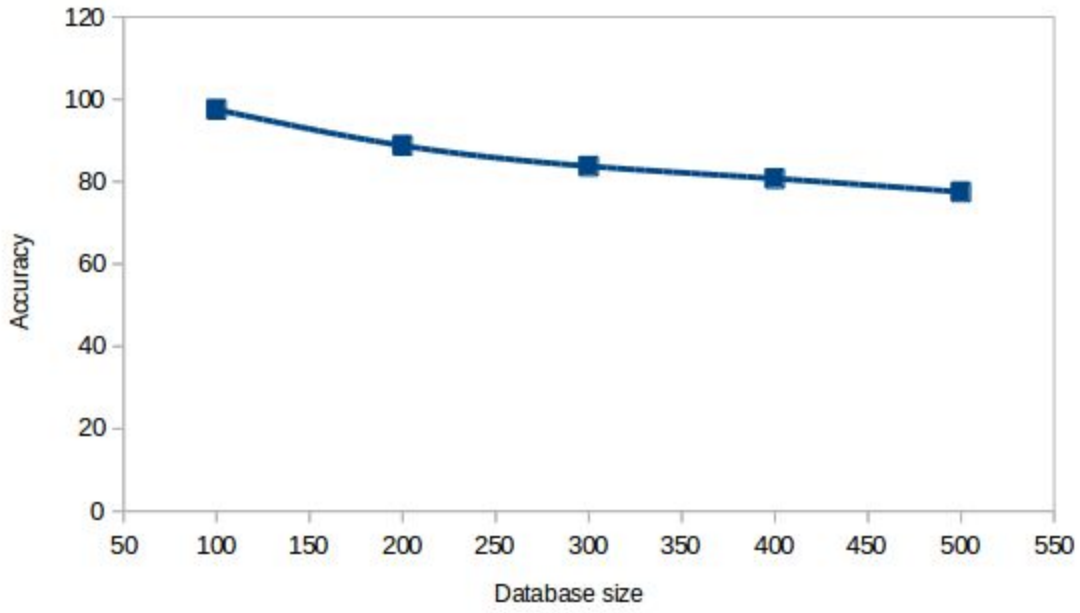
x%	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
Branch Factor =3 and Depth Factor =5										
Accuracy	0.0	53.5	53.5	64.5	64.5	69.6	69.6	72.3	72.3	74.6
Branch Factor =5 and Depth Factor =5										
Accuracy	0.0	61.3	61.3	71.2	71.2	75.6	75.6	78.2	78.2	80.8
Branch Factor =9 and Depth Factor =5										
Accuracy	0.0	64.9	64.9	74.7	74.7	79.2	79.2	81.4	81.4	83.4



Accuracy and database size

database size	100	200	300	400	500
Accuracy	97.5	88.75	83.75	80.75	77.5

Above data is for level=4 and k=5



CONCLUSIONS:

- Increasing branch factor and level gives better results but increases computation time and requires lot of memory. Hence the tradeoff between accuracy and computation has to be decided.
- It was also seen that on increasing size of database, accuracy decreased as more number of images are present which can get matched with the query image and get higher score than the true matches.