CSE 515: Multimedia and Web Databases
Phase 3 - Group 10

"Clustering, Classification and Indexing"

[1]Mansi Panpaliya (1217179197), [2]Rekha Yale Gnanaprakash (1217158670),
[3]Harika Kondru (1217032934), [4]Asmi Pattnaik (1217240531),
[5]Anantha Krishnan Kumar (1217112208), [6]Arijit Panda (1217155056)

Abstract: As the data is humungous in today's world and keeps on increasing exponentially day by day, our project team has dealt with large-scale data distribution to index the images by applying a variety of classification and clustering algorithms such as K means, Support Vector Machine and Personalized PageRank algorithm. The project deals with the efficient retrieval of data and indexing the data to carry out similarity among different types of the visual models of their images The project showcases how the required data is retrieved for the necessary operations and how the data has been managed and implemented for certain similarity calculations.

Keywords: Image data, Latent Semantics, Retrieval, Search, Similarity, Euclidean distance, Cosine Similarity, Labels, Classification, SVM, Decision Tree, Personalized Page Ranking, Relevance Feedback, Indexing, LSH, Clustering, k-means, HOG, SVD, LBP, Color Moments.

1. Introduction

A huge amount of data is produced by various means every day in different forms like text, images, videos, etc. Dealing with text data is easier than multimedia data like images, videos, etc.

Multimedia objects have many features that can be represented in the form of a multi-dimensional vector space with each feature as a dimension. As the number of dimensions goes on increasing, the complexity of processing them also increases. Therefore, multimedia database systems use specialized indexing techniques to help speed up the search by pruning the irrelevant portions of the space and focusing on the parts that are likely to satisfy the search predicate. In many multimedia databases, however, we may not have prior knowledge about the explicit features of data. An alternative strategy is to use clustering techniques that do not need explicit features to operate. Unlike indexing and clustering processes that aim to place similar media objects together for efficient access and pruning, the classification process aims to associate media objects into known semantically meaningful categories.

This phase of the project deals with performing the following functions - labeling, classification, indexing, clustering and relevance feedback, on the dataset provided by 'Mahmoud Afifi, "11K Hands: Gender recognition and biometric identification using a large dataset of hand images." Multimedia Tools and Applications, 2019.', which contains 11,076 hand images (1600 x 1200 pixels) of 190 subjects, of varying ages between 18 - 75 years old.

1.1 Terminology

1.1.1 Euclidean Distance

The Euclidean distance between two points a and b is the length of the line segment connecting them, "*Euclidean distance",* Wikipedia, https://en.wikipedia.org/wiki/Euclidean_distance [1].

1.1.2 Color Moments

Color Moments are measures that characterize color distribution in an image in the same way that central moments uniquely describe a probability distribution. "Color Moments", Wikipedia [9].

1.1.3 HOG

Histogram of oriented gradients (HOG) is a feature descriptor used to detect objects in computer vision and image processing. The HOG descriptor technique counts occurrences of gradient orientation in localized portions of an image - detection window, or region of interest (ROI), "Developer reference for Intel Integrated Performance Primitives 2018:1" [2]

1.1.4 Local Binary Patterns (LBP)

"Local Binary Patterns (LBP) is a non-parametric descriptor whose aim is to efficiently summarize the local structures of images(Huang, Shan et al 2017:1, https://hal.archives-ouvertes.fr/hal-01354386/document)."[3]

1.1.5 Singular Value Decomposition(SVD)

SVD is directly applied to the *Object-Feature* matrix itself. $S = PCP^{-1}$, *where S* is an original object-feature matrix, *C* is the diagonal matrix of eigenvalues and *P* is an orthonormal matrix consisting of the eigenvectors of *S*.

1.1.6 K-means Clustering

K means clustering is an NP-hard problem on cluster analysis on the data. It partitions 'n' number of samples into k partitions/ clusters in which each observation stays with the mean of the cluster with the nearest distance to it. In other words, K-means is an iterative method of clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

1.1.7 Classification

Classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. We will be performing 3 techniques(1.SVM, 2. Decision Tree 3.PPR) in this project [8]

1.1.8 Support Vector Machines

Support Vector Machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other. Here we assume that data is linearly separable. [7]

1.1.9 Decision Tree

A decision tree is a hierarchical classification algorithm. There are some cases where SVM doesn't work perfectly even if we use non linear separators and we use decision tree to classify more efficiently. Decision tree uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. We choose the nodes which partition the data homogeneously. Every leaf represents a particular class. But the challenge is that the leaf shouldn't represent only one or few examples/samples as in that case we overfitted the data. (Most probably we have done more partitions than required).There are few measures by which we can achieve homogeneous nodes.

- Entropy
- Gini Index
- Misclassification Error
- Fisher's discriminant ratio

1.1.10 Locality Sensitive Hashing

Locality Sensitive Hashing is used to reduce the dimensionality curse by creating buckets where similar objects with higher probability are mapped to the same 'buckets' using the hash function. It is useful

in finding the nearest neighbors with a good approximation. These are a family of hash functions in each layer. As we do conjunction in each layer the number of false positives decrease much faster although it increases misses a little. To decrease misses we add more layers and do disjunction between them so that the misses decrease much faster although it increases false positives to some extent.

1.1.11 Personalized Page Ranking

Personalized PageRank (topic sensitive page rank) is a modified version of PageRank where we can bias the probability of random jumps according to the topic or query. If we look at the problem, the formulation in the context of a matrix, the teleportation vector or personalization vector can be assigned with more probability for the query topics. Personalized page rank can be calculated as
rank = ((1-d) * M * rank) + (d * p) where d is a damping factor usually set as 0.85 as per many research papers, M is the column stochastic normalized probability matrix, p is personalized teleportation matrix with biased weight for topics.

1.1.12 Relevance feedback

Relevance feedback is a feature of some information retrieval systems. The idea behind relevance feedback is to take the results that are initially returned from a given query, to gather user feedback, and to use information about whether or not those results are relevant to perform a new query.[6]

1.1.13 Cosine Similarity

The cosine similarity between two vectors is the measure of the cosine of the angle between them. It is a measure of orientation and not magnitude.

1.2 Goal Description

This phase of the project has 6 tasks that deal with clustering, indexing, classification and relevance feedback.
The detailed specification of every task is described below:

Task 1

Implement a program which, given a folder with dorsal/palmar labeled images,
– computes k latent semantics (in the feature space) associated with dorsal-hand images,
– computes k latent semantics (in the feature space) associated with palmar-hand images,
and, given a folder with unlabeled images, the system labels them as
– dorsal-hand vs palmar-hand
using only these latent semantics.

Task 2

Implement a program which, given a folder with dorsal/palmar labeled images and for a user-supplied c,
– computes c clusters associated with dorsal-hand images (visualize the resulting image clusters),
– computes c clusters associated with palmar-hand images (visualize the resulting image clusters),
(the graph partitioning/clustering algorithm can be used for this task) and, given a folder with unlabeled images, the system labels them as
– dorsal-hand vs palmar-hand
using only descriptors of these clusters

Task 3

Implement a program which, given a value k, creates an image-image similarity graph, such that from each image, there are k outgoing edges to k most similar/related images to it. Given 3 user-specified image ids on the graph, the program identifies and visualizes K most dominant images using Personalized Pagerank (PPR) for a user-supplied K.

Task 4

Implement a program which, given a folder with dorsal/palmar labeled images,
– creates an SVM classifier,
– creates a decision-tree classifier,
– creates a PPR based classifier,
and, given a folder with unlabeled images, the system labels them as
– dorsal-hand vs palmar-hand
using the classifier selected by the user.

Task 5

5a: Implement a Locality Sensitive Hashing (LSH) tool (for Euclidean distance) which takes as input (a) the number of layers, L, (b) the number of hashes per layer, k, and (c) a set of vectors as input and creates

an in-memory index structure containing the given set of vectors.

5b: Implement a similar image search algorithm using this index structure and a visual model function of your choice (the combined visual model must have at least 256 dimensions): for a given query image and integer t, visualizes the t most similar images (also outputs the numbers of the unique and overall number of images considered).

Task 6

Let us consider the label set "Relevant (R)" and "Irrelevant (I)". Implement
– an SVM based relevance feedback system,
– a decision-tree based relevance feedback system,
– a PPR-based relevance feedback system,
– a probabilistic relevance feedback system
which enable the user to label some of the results returned by 5b as relevant or irrelevant and then return a new set of ranked results, relying on the feedback system selected by the user, either by revising the query or by re-ordering the existing results.

1.3 Assumptions

1. Prerequisite knowledge of Python programming
2. Data provided is pre-computed and we are supposed to use as-is.
3. Data provided is consistent in its form (example column 1 represents image id)
4. For Task 2, the number of clusters should be greater than or equal to 2.(Being 1 simply means we are not performing any clustering mechanism).
5. For Task 6, the relevant and irrelevant feedback should have greater than 0 values.

2. Problem Solution

Task 1

Task 1 requires labeling of a folder of images as dorsal or palmar, using the latent semantics computed from the images of the labelled folder.
Approach :

1. First, the features of the images of dorsal and palmar hands in the labelled folder are extracted and stored separately. HOG is used in extracting the features of these images..
2. Next, the dimensions of the features are reduced using SVD and we store the latent semantics in the feature space of both the dorsal and palmar image features separately.
3. Then, for each image in the unlabelled folder, features are extracted using HOG feature model.
4. The dot product of the transpose of this feature matrix and the latent semantics in the feature space of dorsal images is computed to get a k*1 matrix called result_dorsal, where k is a user given value for the number of dimensions.
5. Similarly, the dot product of the transpose of the unlabelled image's feature matrix and the latent semantics in the feature space of palmar images is computed to get a matrix of the dimension k*1, called result_palmar.
6. The distance between the elements of the matrix result_dorsal of the unlabelled image and technique_result_dorsal is found. Similarly, the details of the matrix result_palmar and the technique_result_palmar is calculated. If the distance of dorsal elements is less, the image is labeled as dorsal if not; it is labeled as palmar.
7. If the sum of the elements of result_dorsal is greater than the sum of the elements of result_palmar, the image is labelled as dorsal, otherwise it is labelled as palmar.

Task 2

Task 2 focuses on using the labeled images(Dorsal/Palmar) and forming c number of clusters which is user input. We are using the K Means clustering technique to achieve the goal. K means work well with huge datasets. Also, the number of clusters is user input. It is a simple algorithm and can be made efficient with the right optimization. Hence we choose K-means clustering for accomplishing this task[4].
Approach:

1. Input: Given a folder of labeled images and the number of clusters as c
2. The feature extraction is performed using the HOG Method. This technique was chosen above Color moments and histogram of

Oriented Gradients as it gave maximum accuracy

3. The data for dorsal and palmar labeled images are separated into different data sets
4. Initial Cluster Formation: The General form of k means clustering picks random points as initial centroids. Then all points in the data set are assigned to the clusters based on the distance to the centroids. The point is assigned to a particular cluster if its distance is minimum to the centroid as compared to other centroids. Though it is the simplest method it takes additional time to converge
5. To tackle this problem we used the max_a_min algorithm to find c farthest points in the cluster. These c points are used as the initial centroids. This ensures an approximately equal distribution of points in the cluster. By using this technique to find initial centroids, our accuracy for prediction increased
6. Cluster Refinement: K means clustering is an iterative method. Once all c clusters are initialized, k-means tries to refine clusters iteratively by updating the centroids of each cluster and then re-mapping the dataset points to those clusters. This is repeated until the centroids in the previous pass are the same as the currently calculated centroids.

Accuracy achieved using our implementation for k means clustering is 76%.

Task 3

In task 3 we have been given a folder of images. The expectation is to find the K most dominant images given 3 user-specified Images using Personalized PageRank Algorithm.

Approach:

1. First, we extract the Features of all the images in the folder. In this task, we are using Histogram of Gradients as the visual model.
2. After that, we reduce the dimensionality of the given matrix using singular value decomposition with dimensions to reduce being 30.
3. Then we create a Image-Image graph(we represent the graph using matrix) matrix using the most k similar images of every image. We get the K similar images using the cosine similarity. Cosine similarity produced better results when compared to euclidean distance. The matrix created is the Random walk matrix(T).
4. After that, we use the 3 images specified by the user and we create a seed vector(s).
5. Seed vector is created by giving a value of ⅓(since given images are 3 all of those are of equal importance) to all the Images specified by the user. The seed vector is the Teleportation matrix.
6. A seed vector is used to control how much importance we give for a particular node/Image.
7. We consider **"alpha"** which is random walk probability as 0.85 and teleportation probability which is **"1-alpha"** as 0.15.
8. We then calculate an eigenvector using this formula

$$\vec{\prod} = [I - \alpha T]^{-1}(1-\alpha)\overline{S}$$

9. After the eigenvector is calculated we get the K most dominant images based on the values of the eigenvector. We use the highest K values and their corresponding Images to get the K most dominant Images

Task 4

Given a classifier selected by the user and a folder of Dorsal and Palmar Images. We need to classify an unlabeled folder using the Support Vector Machines, Decision Tree and Personalized PageRank Algorithms.

SVM:

Approach :

1. First we extract the features of all labelled and unlabelled images by using HOG as the feature descriptor and then use SVD to extract top 20 features.
2. We have added the labels in the extracted labelled image's latent semantics to use it as train data.
3. We classify if an image belongs to dorsal or palmar using linear SVM with the help of gradient descent for optimization.
4. The classification of hypothesis is considered 1 if sigmoid(z) is greater than or equal to 0.5 and 0 if it is less than 0.5.

5. We want z (= Theta transpose * X) > 0 for y = 1 and z < 0 for y = 0. (1= Dorsal, 0=Palmar)
6. The cost for such classification without margins is hence:

$$\text{cost} = \sum_{i=0}^{m} \{y_i(-\log(h_\Theta(x_i)) + (1 - y_i)(-\log(h_\Theta(x_i))$$

$$h_\Theta(x) = \frac{1}{1 + \exp(-\Theta^T x)}$$

7. It can be inferred that the cost is zero when z>=1 for y=1 and increases linearly for z<1. Similarly, the cost is zero when z<=-1 for y=0 and increases linearly for z>-1. Maximising margins is the main intuition behind the functioning of support vector machines.
8. For gradient descent, we used learning rate as 0.05 with 600 number of iterations.

Tried and not implemented :

1. Since, we were getting varied level of accuracy (>70% for labeled set 1 and labeled set 2 and less than <50% for the query result), we tried to implement cross validation but that did not improve the results and hence kept the number of validation as 1 to reduce execution time.

Decision Tree:

Approach :

1. Here we extract the features of all labelled and unlabelled images by using Color Moments as the visual model.
2. We have added the labels in the extracted labelled image features to use it as train data.
3. For each feature we take each value and calculate how homogenous node it would be if we select that value as a node.
4. We measure homogeneity by the help of gini index and information gain. Whichever value has the highest information gain we select that node and create a corresponding node in the tree . If the maximum information gain is 0 we don't create a node in the tree.
5. In higher dimensions the overfitting is a bigger issue. As discussed in the class we gave importance to the support of each leaf nodes by not considering/creating the nodes which have information gain less than 0.1. (To avoid overfitting so that the leaf can represent more samples/examples in the node) . By doing this our accuracy increased significantly.
6. The benefit of decision tree is unlike SVM it doesn't assume that data is linearly separable. It can get more complex partitions than SVM.

Tried and not implemented :

1. We tried taking the latent semantics and built our decision tree . Although we got good results, our results were not consistent when we varied our data.
2. We also tried to measure homogeneity by measuring entropy but ended with a little less accuracy than we are getting now.

Personalized PageRank:

Approach :

1. In this, we first combine all the images both labeled and unlabeled.
2. Then we extract the Features of all the images both labeled and unlabeled. In this task, we are using Local Binary Patterns as the visual model.
3. Then we create a Image-Image similarity graph(we represent the graph using matrix) using "80" similar images of every image. We tested with various similar image values and 80 gave better results over other values of similar images. Here also we use the cosine similarity matrix to find the similarity matrix. The matrix created is the Random walk matrix(T).
4. After this, we create a seed vector using all the dorsal images so the images which are dorsal are given a value of 1/length of number of dorsal images in the labeled folder(since given images are all of those are of equal importance) and the rest of them to zero. The seed vector is the Teleportation matrix.
5. A seed vector is used to control how much importance we give for a particular node/Image.
6. We consider "alpha" which is random walk probability as 0.85 and teleportation probability which is "1-alpha" as 0.15 as it worked better for us.

7. We then calculate an eigenvector with dorsal images as our subject of importance using this formula

$$\vec{\prod} = [I - \alpha T]^{-1}(1 - \alpha)\overline{S}$$

8. We repeat this entire process of computing the seed vector again by making palmar images as the subject of importance.
9. We get another eigenvector for the palmar images too.
10. We then sorted the values in descending order for both the vectors.
11. Based on the ranking of both the vectors whichever has the lowest ranking we assign the labels for the unlabeled vector.

12. Time taken for execution given that LBP features is already computed is about 45 seconds.

Task 5

In this task we first are given with the 11k hand images. We are given the number of layers and the number of hashes per layer.

Approach:

1. First We create a random vector of size l x h x size of each feature vector. Where l is the number of layers , h is the number of hashes per layer, we reduce the feature vector to a size of 256 using the dimensionality reduction technique of SVD. So the size of random vector would be lxhx256.
2. We take the dot product between each layer random vector and the data semantics of every image.
3. After this we get a list of values. If the value is less than 0, we replace it by 0 and if the value is greater than 0 we replace it by 1. Then we concatenate all the values in the list to create the hash key.
4. So we get a hash value corresponding to an image. We do this for all the images and then we repeat this same process for all the layers.
5. Then we take the query image and extract the features of it. Then we take the dot product between the feature semantics of the original set of images and the features of the new query image.
6. Now we get the hash values for the new query image. After that, we see if the hash values of the query are in the hash values of the original set of images. We get all those images that are in that bucket of hash values of the query image. Once found we found the distance of the query image and all the images that are in the list of hash values.
7. We then query the top T similar images based on the distance measure.

Task 6

Task 6 comprises of relevance feedback mechanism implementation using

- Support vector machine classifier
- Decision tree classifier
- Personalized page rank
- Probabilistic approach

In this task we used the output from task 5 which is the T most similar images for a query image obtained using LSH. User can give list of relevant and irrelevant images out of these T images and the results will be revised based on user feedback.

Approach:

1. Apply LSH and find T most similar images for the given query image Q
2. Get user feedback in terms of relevant and irrelevant images upto 7 each
3. Based on the feedback:
    a. For classifiers(SVM,Decision Tree)
        i. Train the model using relevant and irrelevant feedback images
        ii. Use this model to classify all the unique images found using LSH for the query image Q
    b. For PPR
        i. Apply PPR with higher weights for relevant images. The input to PPR is the dataset of unique images found using LSH for the given query image Q
        ii. This should automatically give higher rank to all the relevant images
    c. For Probabilistic approach:
        i. The Dataset used for this approach is feature vectors of unique images found using

LSH for the given query image Q

ii. The first step would be converting feature vectors to binary space by assigning values zero if they are less than the mean value for the feature and one otherwise.

iii. Based on the user feedback calculate[11] for each feature:
$$\frac{p(o_i|rel)}{p(o_i|\overline{rel})}$$

iv. Compute similarity score by multiplying the above ratio with the value for each feature in the image descriptor and adding the values for every feature of an image

v. This will give a ranked list of images based on similarity

All the approaches use data of the unique images given by LSH. We are choosing this approach as we want to prune as much as possible using LSH as a first filter. Because using entire data would have given us worse results as test data is huge as compared to train data. The relevance feedback techniques help to prune them further and give better results.

2.1 Sample Queries and Outputs

Task 1

Query 1 :

k : 30
Folder: Labelled/Set 1
Classify : Unlabelled/Set 1

python3 task1.py -k 30 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set1/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set1.csv" -l 1 -u 1

Output : Refer the doc Phase3:Output
Execution time: 42 seconds
Accuracy: 56%

Query 2 :

k: 30
Folder : Labelled/Set 1
Classify : Unlabelled/Set 2

python3 task1.py -k 30 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set1/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set1.csv" -l 1 -u 2

Output : Refer the doc Phase3:Output
Execution: 62 seconds
Accuracy: 49%

Query 3 :

k : 30
Folder : Labelled/Set 2
Classify : Unlabelled/Set 1

python3 task1.py -k 30 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 1

Output : Refer the doc Phase3:Output
Execution time:67sec
Accuracy: 51%

Query 4 :

k: 30
Folder : Labelled/Set 2
Classify : Unlabelled/Set 2

python3 task1.py -k 30 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 2

Output : Refer the doc Phase3:Output
Execution time: 44sec
Accuracy: 54%

Task 2

Query 1 :

c: 5
Folder : Labelled/Set 2
Classify : Unlabelled/Set 1
python3 task2_kmeans.py -c 5 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 1

Output : Refer the doc Phase3:Output
Execution : 48sec
Accuracy: 70%

Query 2 :

c : 10
Folder: Labelled/Set 2
Classify : Unlabelled/Set 1

python3 task2_kmeans.py -c 10 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 1

Output : Refer the doc Phase3:Output
Execution : 70sec
Accuracy: 60%

Query 3 :

c: 5
Folder : Labelled/Set 2
Classify : Unlabelled/Set 2

python3 task2_kmeans.py -c 10 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 1

Output : Refer the doc Phase3:Output
Execution Time :82sec
Accuracy: 76%

Query 4 :

c : 10
Folder : Labelled/Set 2
Classify : Unlabelled/Set 2

python3 task2_kmeans.py -c 10 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 2

Output : Refer the doc Phase3:Output
Execution Time : 57secs
Accuracy: 70%

Task 3

Query 1 :

k: 5
K : 10
Folder : Labelled/Set 2
Image IDs: Hand_0008333.jpg, Hand_0006183.jpg, Hand_0000074.jpg

python3 task3.py -k 5 -K 10 -f D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/ -q "Hand_0008333.jpg,Hand_0006183.jpg,Hand_0000074.jpg"

Output : Refer the doc Phase3:Output
Execution time : 5 sec

Query 2 :

k: 5
K : 10
Folder : Labelled/Set 2
Image IDs: Hand_0003457.jpg, Hand_0000074.jpg, Hand_0005661.jpg

python3 task3.py -k 5 -K 10 -f D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/ -q "Hand_0003457.jpg,Hand_0000074.jpg,Hand_0005661.jpg"

Output : Refer the doc Phase3:Output
Execution time : 5 sec

Task 4

Query 1 :

Classifier : SVM
Labeled Image Folder : Labelled/Set 2
Classify : Unlabelled/Set 2

python3 task4.py -c svm -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 2

Output : Refer the doc Phase3:Output
Execution time : 147 sec
Accuracy: 45%

Query 2 :

Classifier : Decision Tree
Labeled Image Folder : Labelled/Set 2
Classify : Unlabelled/Set 2

python3 task4.py -c dcsn -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 2

Output : Refer the doc Phase3:Output
Execution time : 88 sec
Accuracy: 85%

Query 3 :

Classifier : PPR
Labeled Image Folder : Labelled/Set 2
Classify : Unlabelled/Set 2

python3 task4.py -c PPR -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 2

Output : Refer the doc Phase3:Output
Execution time : 120 sec
Accuracy: 61%

Task 5

Query 1 : (***)

L: 10
k : 10
t : 20
Query Image: Hand_0000674.jpg

python3 task5.py -l 10 -k 10 -t 20 -q Hand_0000674.jpg -folder-path "Hands/"

Output : Refer the doc Phase3:Output
Execution time : 2.42 sec

Query 2 :

L: 10
k : 13
t : 20
Query Image: Hand_0000674.jpg

python3 task5.py -l 10 -k 13 -t 20 -q Hand_0000674.jpg -folder-path "Hands/"
Output : Refer the doc Phase3:Output
Execution time : 2.57 sec

Query 3 :

L: 5
k : 10
t : 20
Query Image: Hand_0000674.jpg

python3 task5.py -l 5 -k 10 -t 20 -q Hand_0000674.jpg -folder-path "Hands/"

Output : Refer the doc Phase3:Output
Execution time : 1.68 sec

Task 6

Query 1 :

Feedback System: SVM based
Use results from query 1 of task 5 (marked with ***)
Select upto 7 relevant , upto 7 irrelevant according “as per the expected results”

python3 task6_relevance_feedback_1pass.py -t svm -f /home/mansi/PycharmProjects/mwdb_phase3/Hands/

Output : Refer the doc Phase3:Output
Execution time : 121 sec

Query 2 :

Feedback System: Decision Tree based
Use results from query 1 of task 5 (marked with ***)
Select upto 7 relevant , upto 7 irrelevant according “as per the expected results”

python3 task6_relevance_feedback_1pass.py -t decision_tree -f /home/mansi/PycharmProjects/mwdb_phase3/Hands/
Output : Refer the doc Phase3:Output
Execution time : 26 sec

Query 3 :

Feedback System: PPR based
Use results from query 1 of task 5 (marked with ***)
Select upto 7 relevant , upto 7 irrelevant according “as per the expected results”

python3 task6_relevance_feedback_1pass.py -t ppr -f /home/mansi/PycharmProjects/mwdb_phase3/Hands/
Output : Refer the doc Phase3:Output
Execution time : 15 sec

Query 4 :

Feedback System: Probabilistic Relevance based
Use results from query 1 of task 5 (marked with ***)
Select upto 7 relevant , upto 7 irrelevant according “as per the expected results”

python3 task6_relevance_feedback_1pass.py -t probabilistic -f /home/mansi/PycharmProjects/mwdb_phase3/Hands/

Output : Refer the doc Phase3:Output
Execution time : 16 sec

3. Interface Specification

Command Line Interface is used to run all the python files in this project.

The syntax is as follows:
python3 filename.py [-arg 1] [-arg 2] … [-arg n]

Refer the Readme file more information

4. System Requirements and Installations

The software and tools mentioned below were used throughout this phase of the project. The readme file contains instructions about the installation and execution of the program.

4.1 Operating System:

Windows 10, Mac OS and Ubuntu

4.2 Python:

- Download Link: https://www.python.org/downloads/windows/
- For installing Python 3.7.0, instructions are provided at https://www.ics.uci.edu/~pattis/common/handouts/pythoneclipsejava/python.html

Python Libraries:

S.No	Library	Version
1	pillow	6.2.0
2	numpy	1.17.2
3	opencv-python	3.4.2.16
4	opencv-contrib-python	3.4.2.16
5	scipy	1.3.1
6	scikit-learn	0.21.3
7	pandas	0.25.1
8	scikit-image	0.15.0

4.3 Steps for execution:

- Create a Python virtual environment
    - Install virtualenv
      $ sudo apt-get install python3-venv
    - Go to the desired directory and create a virtual environment
      $python3 -m venv venv
    - Activate virtual environment
      $ source venv/bin/activate

- Install Python Libraries
  Note: All commands mentioned below are being run in a python virtual environment
    - Install the necessary libraries:
      $ pip3 install -r
      <path_to_source>/requirements.txt

Note: It is assumed that you are in the python virtual environment

Below are the steps to run the project for various tasks:

Task 1 :

Command :

> python3 task1.py -lf <labelled dataset path> -uf <unlabelled dataset path> -lm <labelled dataset metadata path> -l <labelled set number> -u <Unlabelled set number> -k <latent semantic>

Example :

python3 task1.py -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set1/" -uf "D:/Masters/FALL 2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set1.csv" -l 1 -u 1 -k 30

Task 2 :

Command :

> python3 task2_kmeans.py -c <Number of clusters> -lf <labelled dataset path> -uf <unlabelled dataset path> -lm <labelled dataset metadata path> -l <labelled set number> -u <Unlabelled set number>

Example :

python3 task2_kmeans.py -c 5 -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set 1/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 1

Task 3 :

Command :

python3 task3.py -k <Outgoing Edges> -K <Most Dominant Images> -f<labelled Folder Path> -q <query images>

Example :

>python3 task3.py -k 5 -K 10 -f D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/ -q "Hand_0008333.jpg,Hand_0006183.jpg,Hand_0000074.jpg"

Task 4 :

Command :

 > python task4.py -c <Type of Classifier> -lf <labelled folder> -uf <unlabelled folder> -lm <labelled images metadata> -l <labelled set number> -u <unlabelled set number>

Example :

> python task4.py -c svm -lf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Labelled/Set2/" -uf "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/Unlabelled/Set2/" -lm "D:/Masters/FALL-2019/MWDB/Phase3/phase3_sample_data/labelled_set2.csv" -l 2 -u 2

Task 5 :

Command :

> python3 task5.py -l<number of layers> -k <number of hashes per layer> -t<Set of vectors> -q <Query Image> -folder-path <path of images folder>

Example :

> python3 task5.py -l 10 -k 10 -t 20 -q Hand_0000674.jpg -folder-path "Hands/"

Task 6 :

Command :

> python3 task6_relevance_feedback_1pass.py -t <relevance feedback type> -f <Folder of 11k images>

<relevance feedback type> : Allowed values - probabilistic/decision_tree/ppr/svm

Example :

> python3 task6_relevance_feedback_1pass.py -t decision_tree -f /home/mansi/PycharmProjects/mwdb_phase3/Hands/

5. Conclusion

We have successfully implemented all the 6 tasks as per problem specification of this phase with good understanding of index structures and image similarity concepts. All the tasks were perfectly designed which was practical implementation of concepts taught in class. This helped us to get deep insight of the topics such as Personalized PageRank algorithm, K means clustering, Support Vector Machine, Decision Tree, and Locality Sensitive Hashing. Dealing with such a huge data was challenging given the system constraints, thus code optimization and properly handling memory allocation was crucial part. We learnt that how the image graph is formed for given set of images and features to index and cluster them using the appropriate algorithms. We saw the application side by the practical implementation of the theoretical concepts in class on how to deal with indexing, clustering and classification provided the given challenges. Also, convergence was an issue for iterative algorithms and error rate had to be balanced to get the convergence.

Bibliography:

[1] https://en.wikipedia.org
[2]https://software.intel.com/en-us/ipp-dev-reference-histogram-of-oriented-gradients- hog-descriptor
[3] Di Huang, Caifeng Shan, Mohsen Ardabilian, Yunhong Wang, Liming Chen. Local Binary Patterns and Its Application to Facial Image Analysis: A Survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Institute of Electrical and Electronics Engineers, 2011, 4, 41, pp.1-17. ff10.1109/TSMCC.2011.2118750ff. Ffhal-01354386f
[4]https://hdbscan.readthedocs.io/en/latest/comparing_clustering_algorithms.html
[5] "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions" (by Alexandr Andoni and Piotr Indyk). Communications of the ACM, vol. 51, no. 1, 2008, pp. 117-122.
[6]https://en.wikipedia.org/wiki/Relevance_feedback
[7]https://en.wikipedia.org/wiki/Support-vector_machine
[8]https://en.wikipedia.org/wiki/Statistical_classification
[9]https://en.wikipedia.org/wiki/Color_moments
[10]SVM implementation inspired from : https://github.com/AbhinavThukral97/LinearSVMClassification
[11]Candan, K.S. and Sapino, M.L., 2010. *Data management for multimedia retrieval*. Cambridge University Press.
[12] "J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In KDD, pages 653658, 2004"
[13] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science. 41, pp. 288-297, 1990.

Appendix:

For the successful completion of this phase of the project, we worked as a team to implement all the given tasks, everyone sharing their conceptual and implementation ideas and helping each other understand every task. On a broad basis, the table below demonstrates how we have shared this phase of the project between ourselves :

S.no	Task	Team Members
1.	Task1	Rekha, Harika
2.	Task2	Mansi
3.	Task3	Ananth
4.	Task4	Asmi,Arijit, Ananth
5.	Task 5	Ananth,Mansi

6.	Task 6	Mansi, Ananth, Arijit
7.	Query results	All
8.	Project Report	All
9.	Readme	All