Text and Image features, Graph models, Clustering, Indexing and Classification

Akshay Shah
Kshitij Kashettiwar
Nikhil Agarwal
Ronak Tanna
Shivam Dhar
Shreyash Devan
Shubham Verma

Abstract

One of the classic problems in the web domain is to yield the best results as part of a user query that can be in form of multimedia. There are various techniques one can employ for Information Retrieval (IR) from huge sets of multimedia consisting of text, images, audio, video, etc. In this phase, we focus on implementing a search engine by applying the concepts of graphs models, state of the art techniques of clustering, partitioning, indexing and classification - Angular clustering, Single pass iterative clustering, Pagerank, Personalized pagerank, Locality Sensitive Hashing, KNN and Personalized pagerank based classification, and similarity/distance measures for textual and visual descriptors of multimedia on a publicly available dataset comprising of various models to represent text and images.
The search engines may yield varying results based on their model implementation. We strongly focus on the use of proven similarity metrics like Euclidean distance that work well with images and choose the visual models to yield best results in the given time frame.

Keywords

Information retrieval, textual descriptor, visual descriptor, search engine, similarity, graph model, tensor, dimensionality reduction, eigen vectors, clustering/partitioning, indexing, classification.

INTRODUCTION

In this phase, we experiment with the dataset provided by [1] B. et al, with respect to three entities - location, user and images where each user has captured a set of images at certain locations and has tagged them. It also provides two categories of information - some of the features are extracted for text (image tags, title) like TF, DF, TF-IDF scores forming a set of textual descriptors and various models for images like CN, CM, HOG, etc which are extracted from the images and form a set of visual descriptors. These descriptors help in creating a feature vector for each entity in the dataset and enable comparison between them by computing a similarity score using metrics like Euclidean distance.
We primarily work on visual descriptors and graph models. Clustering/partitioning algorithms are explored. Top k ranking algorithms like pagerank and personalized pagerank are also implemented. Search in multidimensional data space is supported well by an index structure, an example of which is Locality sensitive hashing (LSH). It is implemented and query results are evaluated for an input consisting of a combination of visual descriptors given an image. We also evaluate KNN and Personalized pagerank based classification algorithms for the set of images and labels provided.

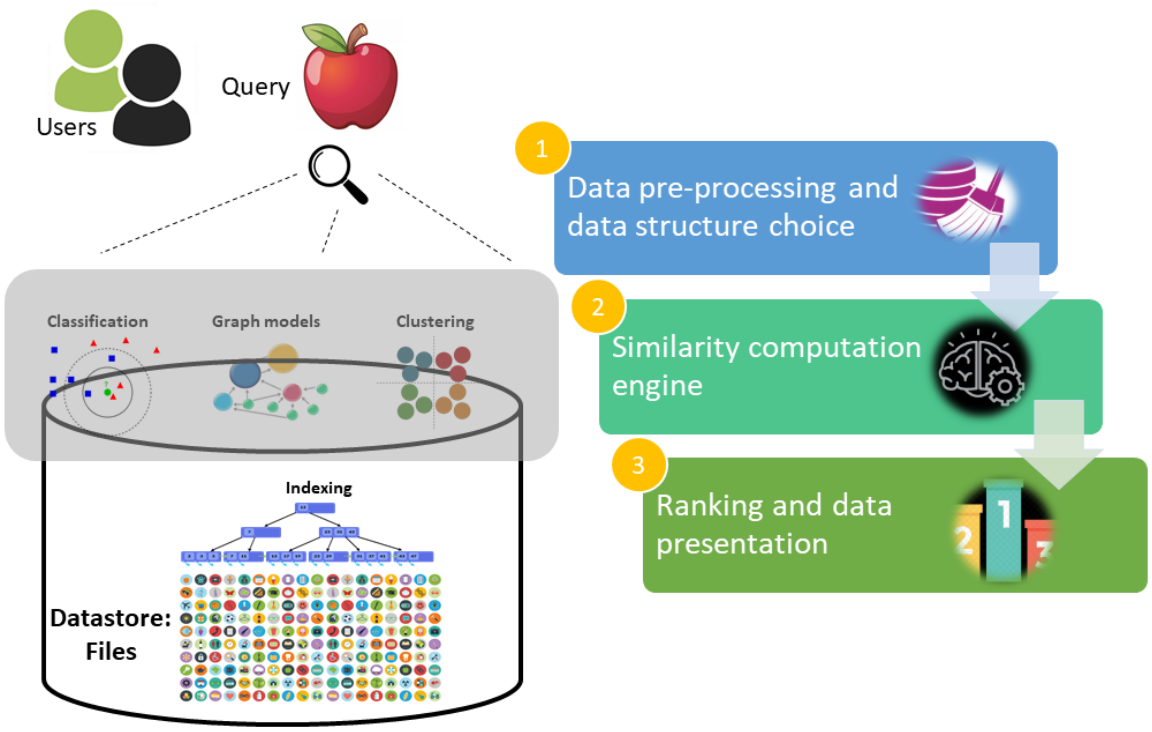


Fig. 1 shows the overview of the project phase.

Terminology

Dataset: The information set consisting of objects and their features.
Object: A particular entity in the dataset. Eg: image, location, user.
Model: A collection of features. Eg: CN (color naming histogram), TF (term frequency).
Feature: A description of object in dataset extracted using Information Retrieval techniques like different colors in the CN, center of moments for CM model.
Descriptor: A collection of models. Eg: textual descriptors - TF, DF.
Similarity: A metric that determines how similar the two entities in the dataset are based on different features associated with them.
Distance: Degree of closeness or separation between the two entities in the dataset. The more the distance, lesser is the similarity.
Vector: A data representation for an object in terms of its features.
Graph: A data structure where objects (nodes) are shown to be related to each other and this relation is defined as an interaction or edge between the nodes of the graph.
Dimensionality reduction: A technique that is employed for reducing the variates under consideration and obtain a set of principal components (or variables) that are hidden in the dataset.
Latent Concepts: Features or concepts that are hidden in the data, and that are identified via a dimensionality reduction technique.
Indexing: A technique that improves search in a database, thereby increasing the rate at which information is retrieved.
Clustering / partitioning: An unsupervised learning algorithm which groups similar objects together based on a similarity metric / distance.
Classification: A supervised learning algorithm which groups similar objects together based on the label set given as input and a similarity / distance metric.

Goal Description

The overall goal of the phase is to get familiarized with the various techniques of performing faster search in a multidimensional space.

- Task 1 considers a complete graph of images across all locations, and aims at finding a reduced graph having the same vertex set but a reduced edgeset i.e. each node has ‘k’ outgoing edges (based on k most similar images).
- Task 2 aims at partitioning the reduced graph generated in Task 1 into ‘c’ clusters using two clustering algorithms - Single pass iterative algorithm using leader approach and angular clustering.
- Task 3 deals with pagerank computation of the reduced graph generated in Task 1. Top ‘k’ images are displayed along with the pagerank score computed.
- Task 4 deals with personalized pagerank score computation of the reduced graph generated in Task 1. Top ‘k’ images are displayed along with the score computed, based on the seeds provided.
- Task 5
    - The first part of the task deals with building an in memory index structure for the image database using the idea of Locality Sensitive Hashing (LSH). The aim is to have efficient query times when there is a query to retrieve similar images from the database for a given image.
    - The second part of the task deals with leveraging the index structure from the first subtask by having the image database as input such that each image is represented by a combined visual model comprising of 256+ dimensions. Once the index is built, given a query image and the integer ‘t’, ‘t’ similar images from the database are retrieved and visualized in the form of a webpage.
- Task 6
  The the task deals with classifying unlabeled images in the data with the labels from the list of small supervised data using
    - kNN and
    - Personalized Pagerank

  based classification method.

Assumptions

- **Preprocessing**
    1. Remove any duplicates for visual descriptor dataset. Eg: If same imageID is present in a model location file twice, we remove one of the instances.
    2. Add missing objects to dataset. The data structure used requires all images in a location to be present across all models in the visual descriptor dataset. Any

instance missing is imputed with mean of the respective feature values in the dataset.

- **Development**
    1. The dataset is a correct representation of facts. All the features extracted across various models will give efficient results for a query search engine. Our similarity engine is based on the assumption that data is noise free and doesn't have outliers.
    2. The input is part of the dataset and the query results include the input instance as well.
    3. Input k to any task given by the user is a positive integer.
    4. Input c - the number of clusters provided by the user is a positive integer.
    5. While computing the image-image similarity Euclidean distance is used so as to preserve the distances between the two, assuming that this similarity metric would be a good measure for their closeness - this holds for task 1, 2, 3 and 4.
    6. Based on the algorithm and the ease of use, we choose to use one of the defined representations of the graph - edge list, adjacency matrix or a list of dictionaries.
    7. We are using clustering quality measures to ensure that resulting clusters are balanced.

    Task 1:
    1. The graph is generated using visual descriptors.
    2. The model used is CN which retains color perspective the best.
    3. The similarity measure considered is Euclidean distance as it works well with CN model - as observed in the previous phases of the project.
    4. The output is a file having edge list with each row having tuple set <node, node, similarity>.

    Task 2:
    1. The reduced graph generated in task 1 is taken as the input.
    2. The single pass clustering algorithm used ensures balanced resultant clusters.
    3. The leader algorithm is used for finding initial cluster heads in case of single pass iterative approach.
    4. Sparse SVD is used for finding singular values in case of similarity based adjacency matrix.

    Task 3:
    1. The reduced graph generated in task 1 is taken as the input.
    2. Pagerank is computed using iterative algorithm.
    3. The pagerank vector is initialized with a uniform probability distribution to ensure no bias / preference for a specific image over others.

4. The damping factor is the conventionally used value of 0.85.
5. The graph data structure chosen is adjacency matrix.

Task 4:

1. The reduced graph generated in Task 1 is taken as the input.
2. Personalized Pagerank is also computed using iterative algorithm.
3. As the seeds are specified, the pagerank vector is initialized 1 for seed entries and rest all entries are zero. This ensures the transportation is restricted to these seeds only.
4. The damping factor is the conventionally used value of 0.85.
5. The adjacency matrix is normalized, so as to make column sum to 1.

Task 5:

1. The assumption is that for a combined visual model, it would be more apt to compare the hidden semantics/topics of the combination rather than the features of the combined model themselves.
2. The distance metric for which the index structure is built & using which the similarity search is done on the reduced search space is Euclidean distance. We assume that the Euclidean distance between two semantic representations of images would incorporate multiple aspects like color, shape/texture of the image.

Task 6a:

1. It is assumed that using a combination of all features will give a better representation of the image and assist us in classification tasks.
2. Euclidean distance measure is used as a similarity basis when comparing various image vectors.

Task 6b:

1. The key assumption here is in case of image having scores of 0 for more than 1 label then, the first label is picked for that image in case of personalized pagerank based classification**.**
2. The damping factor we use is 0.85, which is a key component in the classification algorithm.

DESCRIPTION OF PROPOSED SOLUTION / IMPLEMENTATION

TASK 1

This task deals with creation of a reduced graph using a complete graph of n nodes. Mathematically, given a graph G(V, E) where V is the vertex set and E is the edge set of the graph, we generate a graph G'(V, E') where E' $\subseteq$ E. Also, the edges are weighted with a similarity score computed using Euclidean distance.
The output is stored as a file which can be read as any data structure of choice in succeeding tasks. Also, we show a networkx simulated reduced graph and complete graph of 10 nodes.

TASK 2

We implemented clustering / graph partitioning algorithms - Single pass iterative algorithm (max-a-min) and Angular clustering algorithm to form clusters in the reduced graph generated in Task 1.
Single pass iterative approach employs Leader algorithm for generating the 'c' cluster heads. The algorithm is summarized below:

- Randomly take a node as initial cluster head.
- Find the farthest node (i.e most dissimilar node) from the initial node and mark it as the cluster head for new cluster.
- Continue the above step for the new cluster heads generated till we get 'c' clusters.
- Start assigning clusters to the remaining nodes in the graph based on similarity to the cluster heads i.e find similarity of an node with respect to all cluster heads and choose the maximum one for assigning cluster.
- The algorithm stops until all the nodes are assigned a cluster.
- The basic algorithm would lead to unbalanced clusters, hence we optimised it further to generate balanced clusters by using cluster size as a quality measure while clustering.

Another approach considered is using Angular clustering algorithm, which is summarized below:

- Take similarity based adjacency matrix using the reduced graph found in Task 1.
- Perform SVD on this adjacency matrix and find top k eigenvectors corresponding to largest eigenvalues.
- These k eigenvectors form the clusters.
- Scan each row in these vectors (for each node) to get a maximum value. Assign node to the cluster having maximum value.
- The algorithm stops once all nodes are assigned a cluster.

TASK 3

We implement the iterative version of the pagerank algorithm which is summarized below:

- Initialize a pagerank vector with uniformly distributed probabilities to ensure each node gets a chance of being randomly visited.
- Compute pagerank score for each node which is defined in terms of all the nodes pointing to the current node.
  If node A is pointed by node B and node C, then pagerank for
  node A = (1 - d) + d * ((pagerank(B)/outdegree(B)) + (pagerank(C)/outdegree(C))).
  Here d is damping factor with value 0.85.
- Continue this over for a number of times, default set to 50 or until two consecutive iterations yield same results.
- The final pagerank vector gives pagerank score for each node (image).
- We display and visualize the top k images based on the pagerank score.

TASK 4

We implement the iterative version of the personalized pagerank algorithm which is summarized below:

- Initialize the pagerank vector(vq) with 0 for all it's entries except for 1 for the seed entries.
- The adjacency matrix of the graph is normalized by column so that each column's sum is 1, this is called M.
- A steady state probability vector(uq) is initialized equal to vq.
- The personalized pagerank score for each node is defined as :

  uq = (1 - d) * M * uq + d * vq

  Here d is damping factor with value 0.85.
- Continue this over for a number of times, default set to 50 or until two consecutive iterations yield same results.
- The final pagerank vector gives personalized pagerank score for each node (image) and we display and visualize the top k images based on this score.

TASK 5

We use the idea of Locality Sensitive Hashing and build the index structure for the hash family suited for the Lp norm, $0<p<2$ [8]

The algorithm is comprised of multiple parts and is described with appropriate sub-headings below:

PREPROCESSING:

- We first concatenate all the features from the following visual models in the dataset for every unique image: CM3x3, CN, CSD, LBP3x3, HOG, GLRLM.
    - The choice is because of the simple intuition that we combine 3 models describing the aspect of color in the image (CM3x3, CN, CSD) and 3 models describing the aspect of texture/shape (LBP3x3, HOG, GLRLM) in the image.
- Do a MinMax Scaling of the image vectors.
- We now have 425 features for every image.
- We compute latent semantics of this data using the Singular Value Decomposition (SVD) and retain all the concepts with an eigenvalue >=1. We now have 364 latent semantics of the data that represent a combination of shape/texture and color.
    - The choice of removing all the semantics whose eigenvalues < 1 is because the semantic concepts with eigenvalues < 1 might correspond to noisy data.

INDEX STRUCTURE CONSTRUCTION:

- We construct the index structure for the euclidean distance. Our choice of hash family roots from [8] which works for Lp norms (0<p<=2).
- Any hash function is of the form

  H(v) = floor((r.v + b) / w) ------- (1)

  where

  ‘r’ is a unit random vector whose values are taken from the normal distribution.

  ‘v’ is the vector representation of the combined visual model of any given image.

  ‘w’ is the width of the bucket in any given layer.

  ‘b’ is a random shift sampled uniformly between 0 and w.
- We implement the index structure using ‘L’ hash tables where ‘L’ is the number of layer input by the user. Each hash table/layer comprises of ‘k’ hash functions where ‘k’ is input by the user.
- We concatenate the values generated from ‘k’ hash functions and use that as our key in the hash table. (This represents the conjunction operation of LSH within a single layer).
- Each hash table has the structure such that the key is the hash code generated from ‘k’ hash functions and the value is a list of image IDs mapped to that hash code.
- To differentiate between the values of ‘k’ hash functions that are generated, we use a 12-bit representation of the hash value generated by each hash function.

  For each layer
    - For every image in the preprocessed data:
        - Generate ‘k’ hash values from the hash function defined in (1)
        - Concatenate the values from the 12-bit representations to generate a 12*k bit hash code that serves as the key of the hash table.
        - Append the image id of this image to the list of image IDs mapped to this hash code.

- We now have ‘L’ hash tables, each comprising of ‘k’ hash function mappings of the image. Also, for each hash table we have key-value pairs where each key corresponds to the hash code in which an image falls and the corresponding value for the key represents the list of images mapped to that hash code.
- Finally, w is an parameter that is the characteristic of the data that is inputted for the index construction. For the current setting, the value we have chosen after several trials is 0.2.
    - We observe that for this value, the number of non-empty buckets for each layer is high.

SIMILAR IMAGE SEARCH:

- The user provides with the image ID of the query image and an integer ‘t’ representing the number of similar images desired.
- We access the preprocessed data to retrieve the combined visual model representation of query image.
- We initially have an empty result list, **r** that will be filled with candidate image IDs that are similar to the query image.
- Algorithm description:
    - For each layer/hash table:
        - Generate ‘k’ hash values from the hash function defined in (1)
        - Concatenate the values from the 12-bit representations to generate a 12*k bit hash code in which the query image maps to.
        - The concatenation serves as the conjunction operation performed between ‘k’ hash functions in any given layer.
        - We retrieve the list of image IDs mapped to the hash code as the query and append it to the result list, **r**.
        - Since we do this for each layer, this serves as a disjunction among layers.
    - The result list might comprise of duplicate images, so the size of this list gives the answer to **number of images considered**. The unique images in the list gives the answer to **number of unique images considered**.
    - We now have a reduced search space of candidate image IDs for the similarity search in the form of the result list, **r**.
    - We compute the L2-norm/euclidean distance between the query image and every image in the result list, and sort the results based on the closest to farthest.
    - We return the first ‘t’ images from the list as the ‘t’ most similar images to the user’s query image.

TASK 6

K Nearest Neighbour Classification

The k nearest labelled neighbours are considered for labelling all other images in our dataset.

Here, all visual models are considered for representing an image vector. These image vectors are then normalized using using MinMax Normalizer.

Each image vector is compared with rest of the labelled image vectors and ‘k’ closest image vectors are found. The lowest average score of each label is then considered as the class of the particular image. In case, two labels have equal lowest average scores, both of the labels are considered as labels of the particular image.

Personalized PageRank Classification

We implement the adaption of MultiRankWalk(MRW) algorithm as described in [6] Lin. et al. inspired by [5] Lin. et al. to classify the unlabeled images in the data. The main algorithm is summarized as follows.

- For each label ‘c‘ from the list of labeled image data,
    - Get the seed vector containing images scored to 1 for which label is the c otherwise set the value to 0.
    - Normalize the vector such that sum of the values is 1.
    - Initialize the ranking vector as the seed vector
    - Compute the ranking vector as a iterative formulation of converging technique where on each iteration the ranking vector is computed according to the formula for random walk as stated in [6] Lin. et al.
    - The ranking vector is computed till the scores cannot be improved over the previous computed vector.
- Result of the above step is a matrix of labels and their corresponding ranking vectors containing scores for each image.
    - Using this information, we find the max score and get the indexes which correspond to the labels which will be assigned to the given image in question.
    - Thus by repeating the above step for each instance of the image, we get classification of all the images in the data to one or more label.

INTERFACE SPECIFICATION

COMMON INPUT INTERFACE

The driver program is run to invoke each of the tasks. The choice is between exiting the program and running the tasks. Once the user makes the choice of running the tasks, the user inputs the Task Number (1-6). Subsequently, the inputs specific to each task are provided as described below.

```
Enter your choice:      1) Execute tasks        2) Exit
1
Tasks: 1, 2a(Angular), 2b(Max-a-min), 3(PageRank), 4(PPR), 5(LSH), 6a, 6b
Enter the Task no.: 6a
Enter value of k:2
getting and normalizing data...
reading input labels
Classifying images..
Printing Images.
```

Fig 2. Shows the input interface and the related control flow for an example Task 6a.

COMMON OUTPUT INTERFACE

The output for each task is shown on terminal and the corresponding visualization is built in HTML.

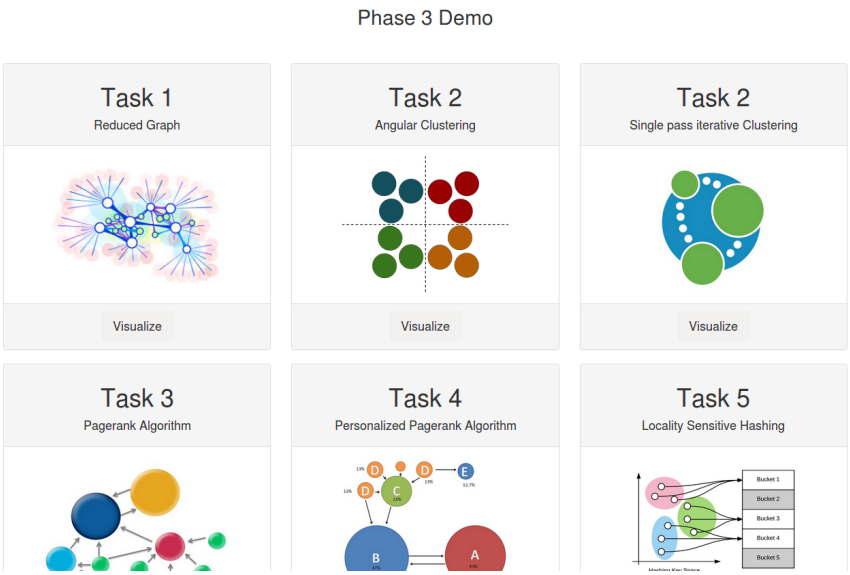


Fig.3 shows the home page to drive individual task visualizations for this phase.

Task 1

INPUT INTERFACE

The driver program is run to take user query for this task. While choosing the option for running the tasks and Task 1, we provide the value of k, which is the number of outgoing edges for each image in the reduced image-image similarity graph.

DATA PRESENTATION

We display the following results -

- Image-image graph with each image having k outgoing edges in descending order of their weights.
- A visualization for first 10 nodes and their links to other nodes using a graphical interface.

Task 2

INPUT INTERFACE
The driver program is run to take user query for this task. While choosing the option for running the tasks and Task 2a or 2b, we provide the value of c, the number of clusters to be formed.

DATA PRESENTATION

We display the following results -

- c clusters having images across dataset assigned to one of the clusters.
- A visualization for these c clusters through a web interface.

Task 3

INPUT INTERFACE

The driver program is run to take user query for this task. While choosing the option for running the tasks and Task 3, we provide the value of k, top images to be displayed based on pagerank score computed.

DATA PRESENTATION

We display the following results -

- Top k images based on pagerank score computation.
- A visualization for these k images through a web interface.

Task 4

INPUT INTERFACE

The driver program is run to take user query for this task. While choosing the option for running the tasks and Task 4, we provide the value of k and three seed images, for personalized pagerank score computation.

DATA PRESENTATION

We display the following results -

- Top k images based on personalized pagerank score computation.
- A visualization for these k images through a web interface.

Task 5

INPUT INTERFACE

The driver program is run to take the inputs of ‘k’ and ‘L’ representing the number of hash functions per layer and number of layers respectively. The user is then made to wait as the index is built for the image database. Once built, the user inputs an image ID of the image for which similar images are to be found and also inputs the value of ‘t’, which represents the number of similar images to be found for the query image.

DATA PRESENTATION

- We display the ‘t’ most similar images in the database by their image IDs in a list.
- We output the total number of images considered and also the number of unique images considered in the reduced search space.
- We also visualize the ‘t’ most similar images in a web interface comprising of t+1 images, where the first image from left to right represents the query image and the next ‘t’ images represent the ‘t’ most similar images for the query image.

Task 6

INPUT INTERFACE

The driver program is used to run the task. The program takes input of file which can be specified in the config file at the time of running.

DATA PRESENTATION

Data is presented in two forms which is as follows.

a) First is the text file containing image_ids sorted by the scores classified under given label for each label present in the input file.
b) We use this output txt file to **visualize** the images via a web interface.

SYSTEM REQUIREMENTS

- Programming Language: Python 3.7
- Libraries
    - NumPy (http://www.numpy.org/)
    - SciPy (https://www.scipy.org/)
    - Pandas (https://pandas.pydata.org/**)**
    - Pickle (https://docs.python.org/3/library/pickle.html)
    - Networkx (https://networkx.github.io/documentation/networkx-1.10/)

These can be installed using the following command -

`pip3 install -lv scipy==1.1.0`

One driver program needs to be run - `driver.py` and this connects to all the tasks (written as modules).

`python3 driver.py` *-> to be run from the project directory submitted*

OBSERVATIONS

1. CLUSTER ANALYSIS FOR ANGULAR CLUSTERING IN TASK 2 (a)

To study the clusters formed after implementing angular clustering on the reduced graph file where each image has k most similar images as nodes attached to it, we visualized the output for a sample input. For each cluster, we have shown below few images.

Fig. 4 shows cluster 1 for task 2a

Fig.5 shows cluster 2 for task 2a

Fig. 6 shows cluster 3 for task 2a

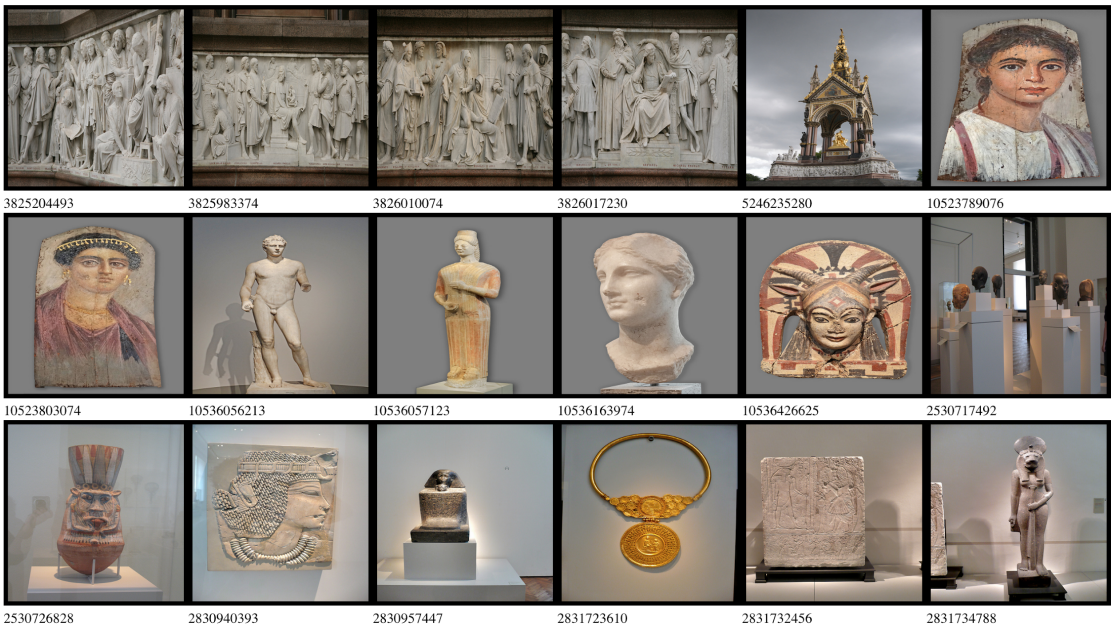
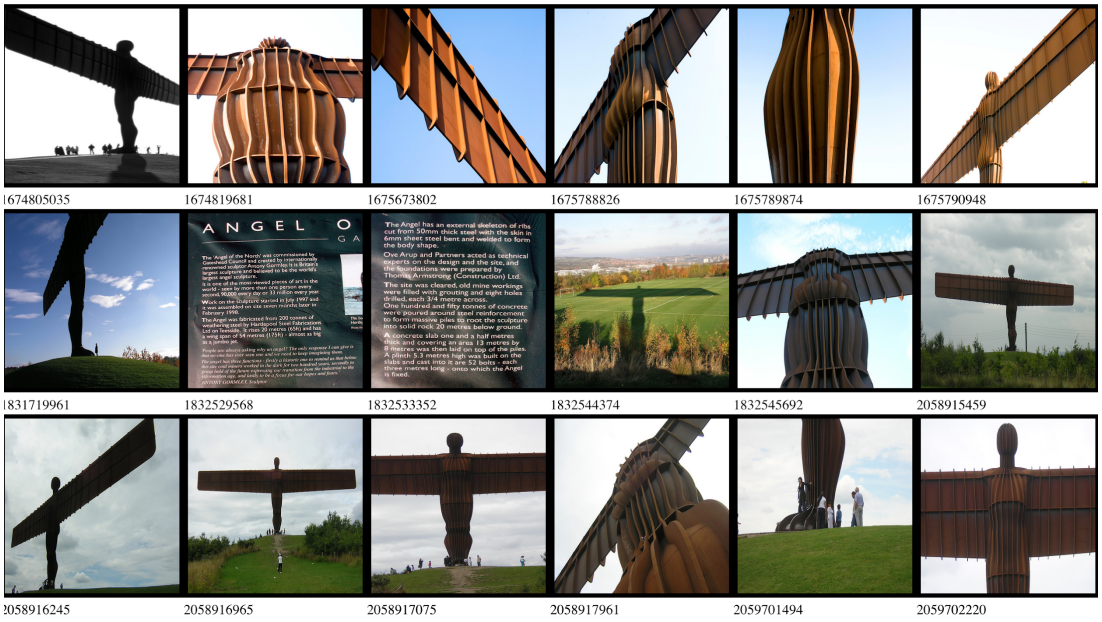


Fig. 7 shows cluster 4 for task 2a



2. CLUSTER ANALYSIS FOR MAX-A-MIN CLUSTERING (SINGLE PASS ITERATIVE ALGORITHM) IN TASK 2 (b)

To study the clusters formed after implementing single pass iterative algorithm on the reduced graph file where each image has k most similar images as nodes attached to it, we visualized the output for a sample input no. of clusters 3. For the cluster, we have shown below few images.

Number of clusters = 3

Fig. 8 shows cluster 1 for task 2b

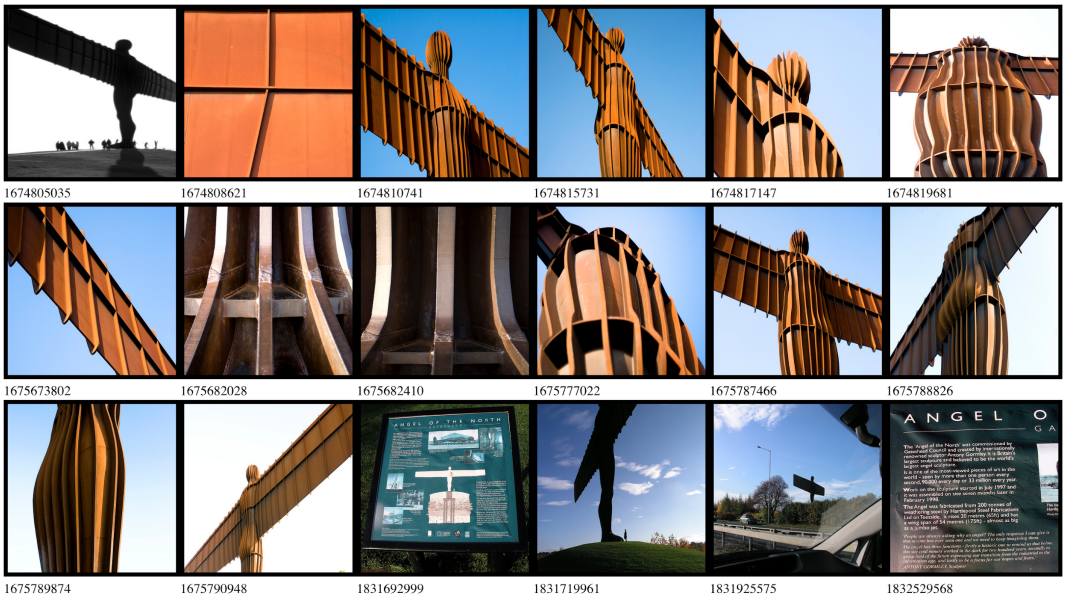


Fig. 9 shows cluster 2 for task 2b

Fig. 10 shows cluster 3 for task 2b

Based on the above image visualisations of the few images within the clusters, we can see that images belonging to the same location have been clubbed together within clusters thus showcasing a good clustering of the images.

3. DOMINANT IMAGE ANALYSIS USING THE PAGERANK ALGORITHM

To study the most dominant images that appear when the PageRank algorithm is implemented on the reduced graph file we get the following outputs. Below, we have visualised the k most dominant images for 2 different values of k.

Fig.11 shows 6 most dominant images based on pagerank

K most dominant images are:

ImageID	In Degree
9203115019	14
169894219	16

6848901676	19
3951806833	16
3415764411	14
5992433646	18

Fig.12 shows 8 most dominant images based on pagerank

K most dominant images are:

9203115019 169894219 6848901676 3951806833 3415764411 5992433646

7698280 369448297

ImageID	In Degree
9203115019	14
169894219	16
6848901676	19
3951806833	16
3415764411	14
5992433646	18
7698280	15
369448297	15

Now, we have visualised the dominant images obtained by considering the K values constant and changing the alpha values. We have considered 0, 0.5, 0.85 and 1.

initial_k = 8

Fig.13 shows top 10 images based on pagerank for alpha = 0.85

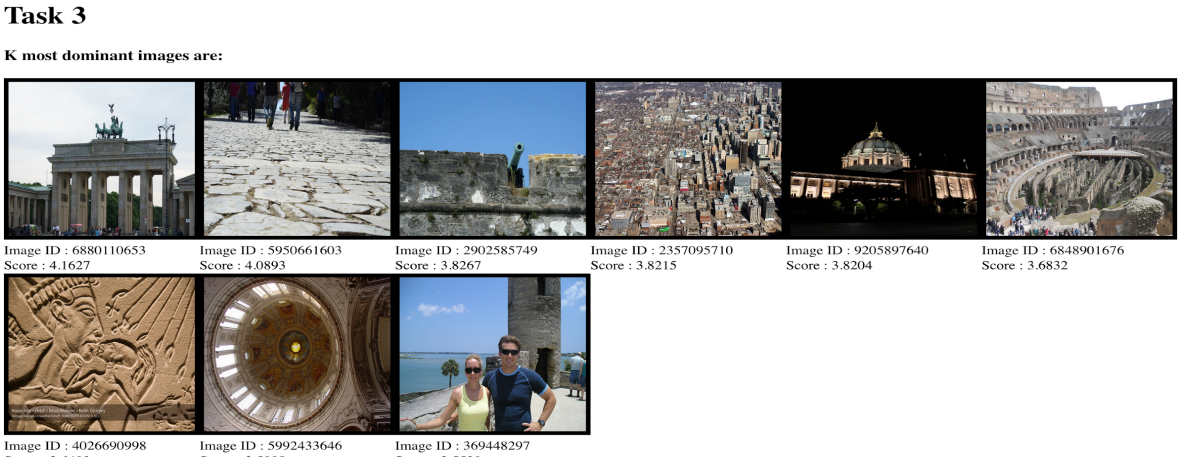


Fig.14 shows top 10 images based on pagerank for alpha = 0.5

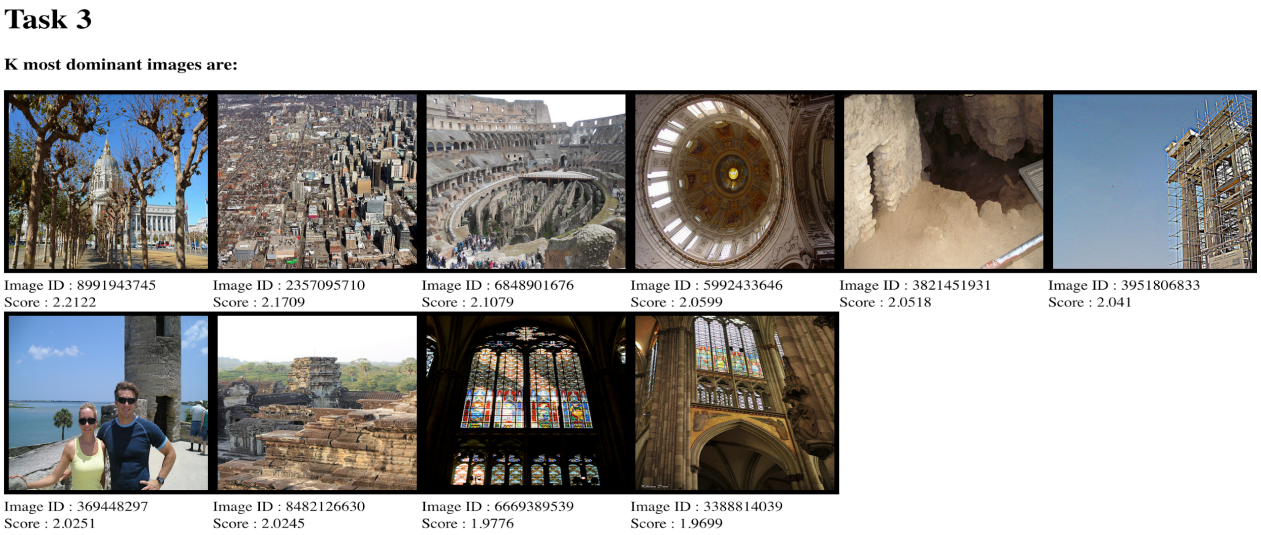


Fig.15 shows top 10 images based on pagerank for alpha = 1.0

Task 3

K most dominant images are:

Image ID : 4026690998
Score : 16.143

Image ID : 4025940717
Score : 15.9031

Image ID : 4025936643
Score : 13.9473

Image ID : 4026694764
Score : 13.7662

Image ID : 3830535342
Score : 12.6141

Image ID : 2419777
Score : 12.6112

Image ID : 421994427
Score : 12.2276

Image ID : 5134037025
Score : 12.1433

Image ID : 9203115019
Score : 11.8907

Image ID : 11285271
Score : 11.833

Fig.16 shows top 10 images based on pagerank for alpha = 0

Task 3

K most dominant images are:

Image ID : 1288780397
Score : 1.0

Image ID : 1674805035
Score : 1.0

Image ID : 1674808621
Score : 1.0

Image ID : 1674810741
Score : 1.0

Image ID : 1674815731
Score : 1.0

Image ID : 1674817147
Score : 1.0

Image ID : 1674819681
Score : 1.0

Image ID : 1675673802
Score : 1.0

Image ID : 1675682028
Score : 1.0

Image ID : 1675682410
Score : 1.0

Alpha = 0

ImageID	In Degree
1288780397	7
1674805035	9
1674808621	4
1674810741	5
1674815731	4
1674817147	10
1674819681	10
1675673802	10
1675682028	11
1675682410	6

Alpha = 0.5

ImageId	In Degree
8991943745	3
2357095710	21
6848901676	1
5992433646	20
3821451931	19
3951806833	18
369448297	18
8482126630	18
6669389539	18
3388814039	19

Alpha = 0.85

ImageID	In Degree
2357095710	21
369448297	18
8991943745	21
3951806833	18
6848901676	21
4049644769	15
5992433646	20
3821451931	19
169894219	17
9203115019	15

Alpha = 1

ImageID	In Degree
4026690998	18
4025940717	14
4025936643	18
4026694764	12
3830535342	14
2419777	14
421994427	14
5134037025	15
9203115019	15
11285271	15

Based on the above tables, we concluded that the in degrees for the images (nodes) are the maximum when we choose alpha as 0.85. The in degree measures the importance of the nodes in a graph. Thus, the alpha value of 0.85 is chosen.

4. DOMINANT IMAGE ANALYSIS USING THE PERSONALISED PAGERANK ALGORITHM

To study the most dominant images that appear when the Personalised PageRank algorithm is implemented on the reduced graph file we get the following outputs. Below, we have visualised the k most dominant images for two different values of k and the seed nodes provided are those provided in the sample input file for Phase 3

k = 6
Seed 1 : 2976144
Seed 2 : 3172496917
Seed 3 : 2614355710

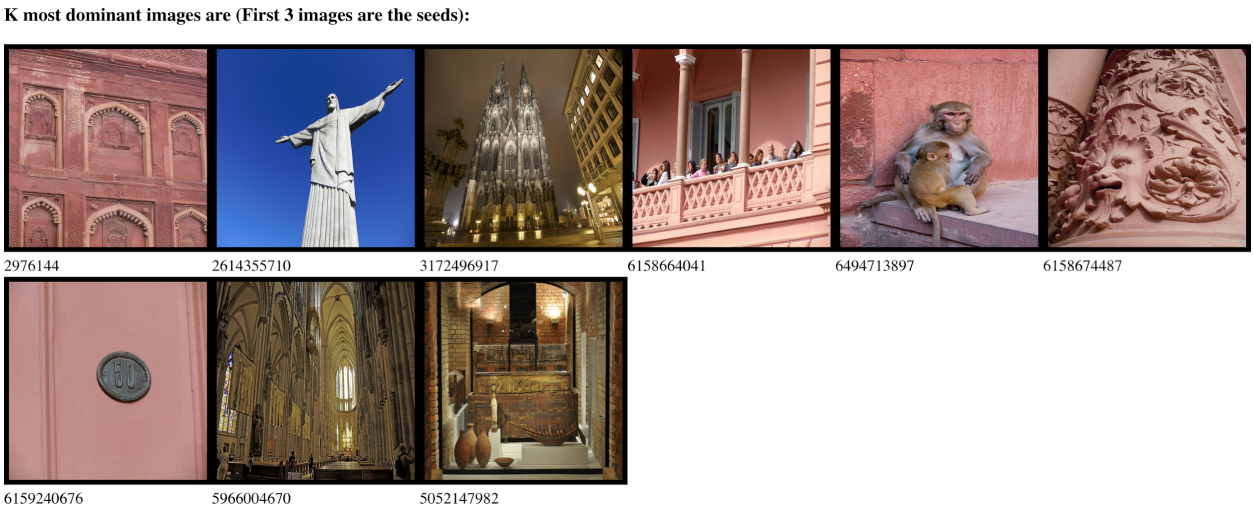


Fig. 17 shows top 6 images based on personalized pagerank for given seeds

Output images	Seeds or images having the output image as its outlink
6158664041	2976144
6494713897	2976144
6158674487	2976144
6159240676	2976144
5966004670	3172496917
5052147982	3172496917

K = 7
Seed 1 : 27483765
Seed 2 : 2492987710
Seed 3 : 487287905

Fig. 18 shows top 7 images based on personalized pagerank for given seeds

Output images	Seeds or images having the output image as its outlink
5258788846	27483765, 4746470290
9792667384	27483765
4746470290	5258788846, 9792727203
9792727203	4746470290
9617662087	3821451931
3821451931	8482554825
8482554825	3821451931

Based on the above image visualisations, it is clear that the Personalised PageRank algorithm has localised the dominant image search with respect to the seeds provided.

5. INDEXING VIA LOCALITY SENSITIVE HASHING

Below are some outputs for obtained for the following parameters:

Run 1:

L=3
k=10
Query Image ID: 339911248
Total images considered: 532
Unique images considered: 502
The T nearest neighbors: [339911248, 8331856290, 3739398765, 8331856100, 4733413255]

Fig. 19 shows T nearest neighbor images for the query image 339911248.

Run 2:

L=3
k=10
Query Image ID: 2482775717
Total images considered: 537
Unique images considered: 513
The T nearest neighbors: [2482775717, 2483578418, 2482789731, 2482764143, 2482756687]

Fig. 20 shows T nearest neighbor images for the given query image 2482775717.

Run 3:

L=3

k=10

Query image ID: 330096935

The T nearest neighbors: [330096935, 330062705, 3739365059, 339911248, 2881963256, 8330800385]

Fig. 21 shows T nearest neighbor images for the given input 330096935.

Total Images considered: 768

Unique Images considered: 728

Run 4:

L=3

k=10

Query Image ID: 2482756687

The T nearest neighbors: [2482756687, 2482775717, 7181455723, 52051413, 2483578418]

Fig. 22 shows T nearest neighbors for the given query image 2482756687

Total images considered: 779

Unique images considered: 753

Key observations:

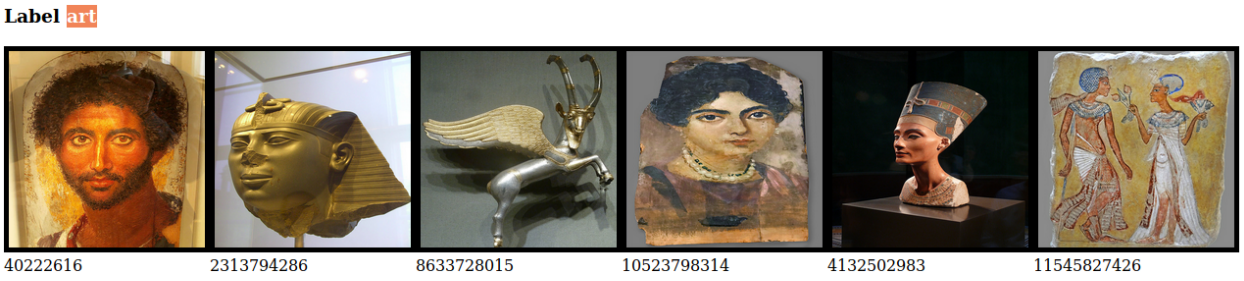
- The search space for similarity search has reduced by quite a bit.
- The “semantic” similarity among the images is getting captured in some form from the outputs that are observed.
- We observe that good quality images that don’t intuitively show multiple possible semantics have very good results.
    - During Run 1, the query image of big ben was of good quality and it led to the output where all the similar images were of big ben itself.
    - During Run 2, the query image of the civic center in San Francisco led to an output where all the similar images where of civic center itself.
- We observe that good quality images that intuitively show multiple possible semantics still have decent results but it comprises of false positives as well as misses. This suggests that this is in fact an approximate index structure and not a perfect one.
    - During Run 3, the query image of big ben returned a miss.
    - During Run 4, the query image of civic center returned a possible false positive and a miss.
        - The false positive was in the form of the image of big ben as one may interpret the semantic similarity between the two images indeed being present -- a rigid structure pointed at the top surrounded by blue sky.
        - The miss was in the form of a band playing that probably has no semantic similarity with the query image other than possibly the color distribution of the image.

To conclude, while the optimal parameter setting for k and L heavily relies on the recall given for different values, which can be obtained from the ground truth, our argument is that using the concept of semantic similarity among the images tuned in with optimal values of k and L, we can achieve good results using the concept of Locality Sensitive Hashing.

6. CLASSIFICATION ANALYSIS

A. Nearest Neighbour Classification

Below are some outputs obtained on some sample input data.



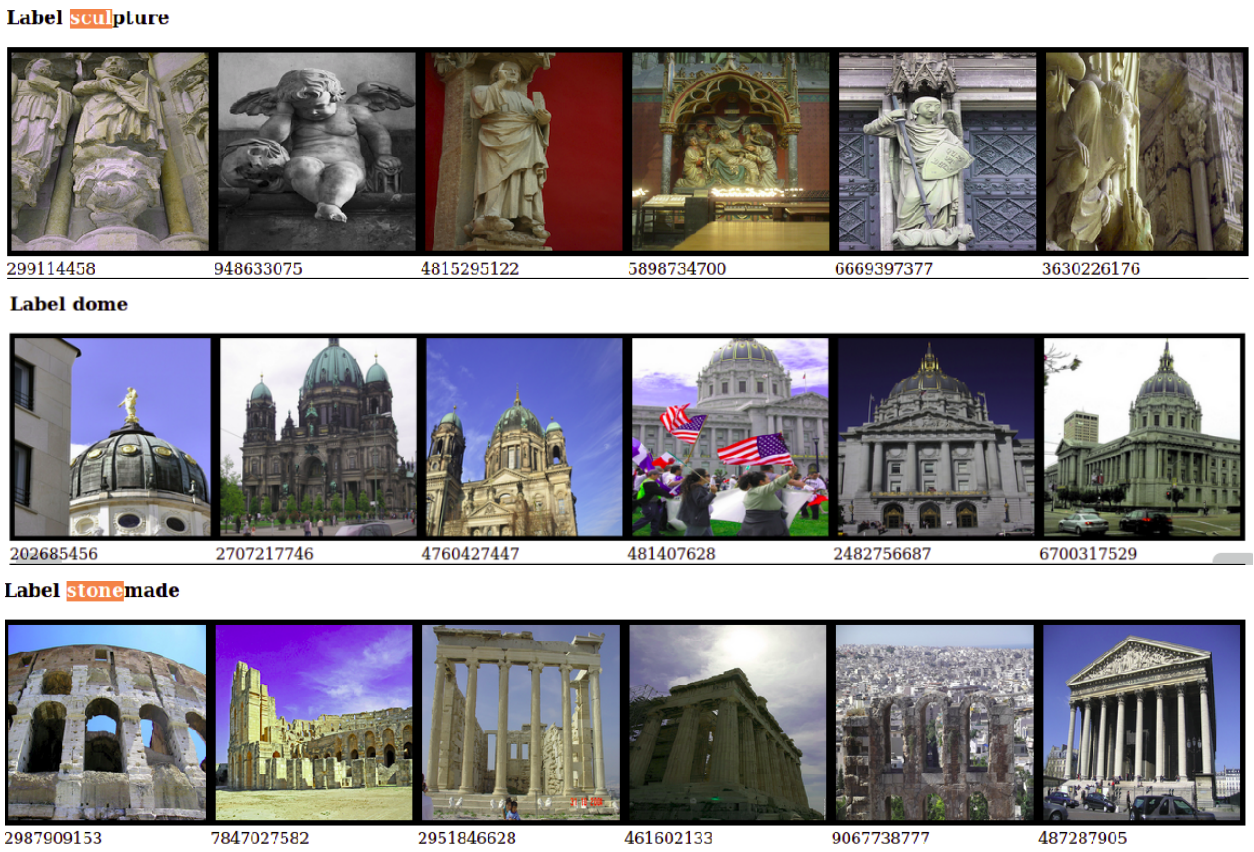


Fig. 23 shows image labels based on kNN classifier

The given grouping of images shows us that our nearest neighbour algorithm is able to group similar image features under similar labels.

B. Personalized PageRank classification

The following are some results for a given set of labels for the images in the dataset:

Art

Stonemade

Label stonemade

Fort

Label fort

4501766904 1806444675 3298433827 4027646409 5001745154 3951834477

Sculpture

Label sculpture

Fig. 24 shows image labels based on personalized pagerank based classifier

It is evident from the classified results that for the given labels (art,stonemade,fort, and sculpture), the images which look very similar conceptually are classified correctly with the respective label. From the overall analysis of the results, there is misclassification of some images as well, but that number is small.

RELATED WORK

This section discusses about past and current work in the direction of similarity based ranking of entities. There has been a lot of activity around the search domain - query time optimisation, relevant query results retrieval using relevance feedback from users and machine learning algorithms, and so on; and we have implemented a similarity based retrieval tool for various entities given an ecosystem. Interpretation of similarity changes with the application - some might say that images having same color based on the colors extracted are more similar than images of the same object in two different filters eg. A grey shaded image of an apple and a mango may be similar in one application whereas a grey shaded apple and a sepia apple would be similar in the other.

User - user search is a classic example of one of the components of collaborative filtering based recommendation systems, where similar users are found based on the number of similar items of interest for both as explained by [2] Melville et al.

Image search moved to a dimension of semantics where image recognition is not the only way to decide similarity between the images as discussed in [3] G. et al. Visual search is used to power content recommendation on Pinterest. [7] Y. et al.

Over the period of time, different models have come up, each model having different features of an image extracted and used for image similarity. Eg. Local binary pattern (LBP) yields good results while comparing image textures whereas CN has proven to be a good model when it comes to color match. Similarly, for text different models have shown their efficiency eg. word2vec is a tool built on the textual descriptors like TF, DF, TF-IDF and is widely used for word embeddings.

CONCLUSION

This was an exploratory phase where we used different algorithms for clustering, indexing and classifying multimedia, find similarity between entities in the graph model and evaluate our search engine implementation and ranking model, augmenting our knowledge base towards mechanics of multimedia and web world.

We used visual descriptors effectively to compute similarity measure, eg. we used Euclidean distance as the similarity metric to compute image - image similarity for CM model. We choose this through our observations made in the previous phases of the project.

At a broader level, the takeaway from the task was to appreciate the nuances of building a similarity ranking based search engine using different techniques which given a input would fetch the top matching or similar images. We were exposed to yet another modeling technique for images i.e using graphs in form of adjacency matrix or edge list, representing different dimensions for a given image.

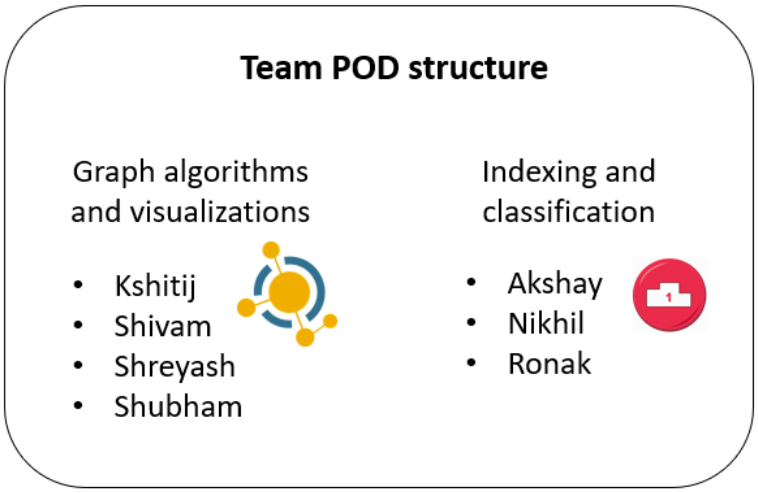
BIBLIOGRAPHY

[1] B. Ionescu, A. Popescu, M. Lupu, A.L. Ginsca, H. Muller, Retrieving Diverse Social Images at MediaEval 2014: Challenge, Dataset and Evaluation, MediaEval Benchmarking Initiative for Multimedia Evaluation, vol. 1263, CEUR-WS.org, ISSN: 1613-0073, October 16-17, Barcelona, Spain, 2014.
[2] Melville P., Sindhwani V. (2011) Recommender Systems. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA.
[3] G. Chechik, V. Sharma, U. Shalit (2010), Large Scale Online Learning of Image Similarity Through Ranking, Journal of Machine Learning Research 11, 1109-1135.
[4] G.W. Furnas, S. Deerwester, S.T. Dumais, T.K. Landauer, R.A. Harshman, L.S. Streeter and K.E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. Original in Proceedings of SIGIR 1998, 469-472
[5] Lin F, Cohen WW. Semi-supervised classification of network data using very few labels. Proceedings of the International Conference on Advances in Social Network Analysis and Mining (ASONAM '10); August 2010; Washington, DC, USA. IEEE Computer Society; pp. 192–199.
[6] Lin, F. (2011). Adaptation of Graph-Based Semi-Supervised Methods to Large-Scale Text Data.
[7] Yushi Jing, David Liu , Dmitry Kislyuk , Andrew Zhai , Jiajing Xu, Jeff DonahueVisual search at Pinterest
[8] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality sensitive hashing scheme based on p-stable distributions. Proceedings of the ACM Symposium on Computational Geometry, 2004

APPENDIX

List of Images:

Individual Contributions:



Kshitij Kashettiwar : Built the presentation layer using javascript and html. Also, implemented personalized pagerank based iterative approach for given seed images. Recorded observations of different parameters on pagerank score computations.

Shivam Dhar : Implemented graph partitioning / clustering algorithms. Evaluated the results for the algorithms without and with quality measures like balance. Added different graph representations - edge list, adjacency matrix and list of maps.

Shreyash Devan : Worked on the networkx visualization of the reduced graph. Implemented pagerank based graph ranking algorithm. Involved in end to end testing of all the tasks in the pod.

Shubham Verma : Implemented reduced graph functionality. Wrote the menu driver having a common user interface for task integrations.

Akshay Shah : Worked on personalized pagerank based classification algorithm. Ensured valid results are returned based on the input set of images and labels provided.

Nikhil Agarwal : Designed and implemented kNN based classification algorithm. Validated the results of classifier for the set of images with labels given as input for different values of k.

Ronak Tanna : Designed and implemented Locality Sensitive Hash index structure for efficient search. Performed required pre-processing on the dataset and evaluated the structure for images with dimensions > 300. Documented the observations and fine tuned the algorithm to yield best results.