CSISE: CLOUD-BASED SEMANTIC IMAGE SEARCH ENGINE

A THESIS IN
Computer Science

Presented to the Faculty of the University
Of Missouri Kansas City In partial fulfillment
Of the requirements for the degree

MASTER OF SCIENCE

By
VIJAY WALUNJ

B.E. UNIVERSITY OF MUMBAI, INDIA 2008

Kansas City, Missouri

2013

CSISE: CLOUD-BASED SEMANTIC IMAGE SEARCH ENGINE

Vijay Walunj, Candidate for the Master of Science Degree

University of Missouri – Kansas City, 2013

ABSTRACT

Due to rapid exponential growth in data, a couple of challenges we face today are how to handle big data and analyze large data sets. An IBM study showed the amount of data created in the last two years alone is 90% of the data in the world today. We have especially seen the exponential growth of images on the Web, e.g., more than 6 billion in Flickr, 1.5 billion in Google image engine, and more than 1 billon images in Instagram [1]. Since big data are not only a matter of a size, but are also heterogeneous types and sources of data, image searching with big data may not be scalable in practical settings. We envision Cloud computing as a new way to transform the big data challenge into a great opportunity.

In this thesis, we intend to perform an efficient and accurate classification of a large collection of images using Cloud computing, which in turn supports semantic image searching. A novel approach with enhanced accuracy has been proposed to utilize semantic technology to classify images by analyzing both metadata and image data types. A two-level classification model was designed (i) semantic classification was performed on a metadata of images using TF-IDF, and (ii) image classification was performed using a hybrid image processing model combined with Euclidean distance and SURF FLANN measurements.

A Cloud-based Semantic Image Search Engine (CSISE) is also developed to search an image using the proposed semantic model with the dynamic image repository by connecting online image search engines that include Google Image Search, Flickr, and Picasa. A series

of experiments have been performed in a large-scale Hadoop environment using IBM's cloud on over half a million logo images of 76 types. The experimental results show that the performance of the CSISE engine (based on the proposed method) is comparable to the popular online image search engines as well as accurate with a higher rate (average precision of 71%) than existing approaches.

APPROVAL

The faculty listed below, appointed by the Dean of School of Computing and Engineering, have examined a thesis titled “CSISE: Cloud-Based Semantic Image Search Engine” presented by Vijay Walunj, candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Yugyung Lee, Ph.D., Chair
School of Computing and Engineering

Ghulam Chaudhry, Ph.D.
School of Computing and Engineering

Praveen Rao, Ph.D.
School of Computing and Engineering

CONTENTS

ILLUSTRATIONS

Figure Page

TABLES

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank the following people who have directly or indirectly helped me in academic achievements. Firstly, I would like to thank Dr. Yugyung Lee, my mentor and advisor, for her continuous support and guidance throughout my master's program in computer science. I sincerely thank Dr. Ghulam Chaudhry and Dr. Praveen Rao for accepting to be a part of my thesis committee and making time for me off their busy schedule. Finally, I would like to thank my family members and friends for all their encouragement and support.

The views and conclusions contained herein are those of the author's and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the University of Missouri - Kansas City.

CHAPTER 1

INTRODUCTION

1.1 Research Motivation

The exponential increase in data is posing a great challenge for researchers in handling big data, which is a major contributor in making different software's and websites. Researchers calculate that digital data has grown to 2.75 zeta bytes and is expected to reach nearly 8 zeta bytes by 2015 [1]. Creating, replicating, saving, mining and analyzing this huge data has become a big challenge and the way we handle this challenge will drive our next generation applications.

In a recent study conducted by IBM on creation of data, it has been observed that amount of data created is alone 90% in last 2 years and a similar trend is expected in next years to follow. With data, images are showing a similar increasing trend with the same rate. As of now, Flickr has more than 6 billion images, Google has 1.5 billion and Instagram also contains more than 1 billion images. Private image stores such as Facebook have been dealing with more than several billion images in recent months. These are just the typical visible sources of images but there are other invisible heterogeneous sources of images making it more laborious to search images.

Cloud computing is booming in its own way to transform big data challenge into a great opportunity. It is also interesting to find how this technology will help in image searching techniques.

In this thesis, we look forward to improve search image efficiency over text based searching legacy tools. Earlier, text based legacy systems ruled searching over Internet which is improvised by effective text search based techniques. Since searching images using text based methods is advanced, it can be applied for image based searching.

We are looking forward to bring up performance as it takes a while to perform image processing on loads of images. Images are entities that hold loads of information in terms of contents, text, objects, metadata, etc. Images have been key interest in recent times to use as searching option.

The application described in Figure 1.1 is the key motivation for us.

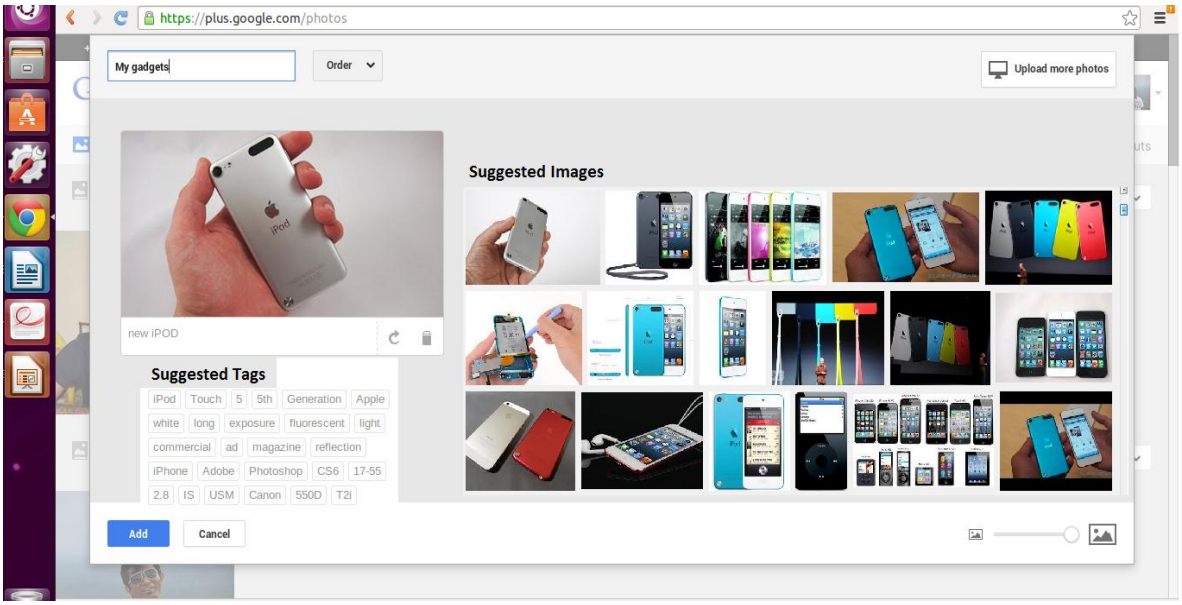

Figure 1.1 Proposed Application View [3][4]

We are proposing suggestion based technique which works on listing users with range of suggestions for images uploaded by user over social networks. As shown in Figure 1.1, consider a case where user buys apple iPod generation 5 and he/she uploads pictures from it over social network. By using Cloud-Based Semantic Image Search Engine (CSISE), a user will be notified with suggested images and keywords for uploaded images in CSISE

environment. From the Figure 1.1, a user is provided with relevant pictures and keywords of apple iPod generation 5. The CSISE system will help us in searching images in CSISE environment and annotate whatever user has uploaded on the CSISE.

1.2 Problem Statement

With potentially thousands of features per image, hundreds to millions of images and their millions of keywords to search, how image searching can be improved?

Considering the growing number of images, we look forward to develop a system which can be relevant to the query image uploaded by user. We need a system which can have means to make use of image features for image searching. As efficient text searching is already available, we can use it to support image searching. Having said about image and text searching, we need a system to have efficient and fast image storage, which make use of latest cloud technologies such as Hadoop. Also, we can take advantage of various online image stores such as Picasa, Google Image, etc. which can be inter-connected.

1.3 Thesis Outline

In chapter 2, we discuss about related works that were performed to find out key aspects of image searching. In chapter 3, we will explore our CSISE model and how the components interact with each other's. Chapter 4 will emphasize on the implementations of CSISE. Experimental results and evaluation will be discussed in chapter 5. Finally, chapter 6 will state the conclusion and about future work.

CHAPTER 2

RELATED WORK

In this chapter we will review several image matching techniques, which will help in coming up with a new architecture to address the problem statement. While dealing with image searching, fundamentals such as retrieving and matching an image are need to be stated. We will explore image matching related works in following sections.

2.1 Image Retrieval and Annotation

2.1.1 Image Retrieval

Image Retrieval allows browsing, searching and retrieving images from large image store. Image store is a large database of digital images from multiple sources. Traditional image retrieval makes use of different metadata such as captions, keywords, tags and descriptions. Images retrieval is performed over the annotations.

There are different types of image retrieval approaches.

- QUERY-BY-TEXT [5][6][7]
- QUERY-BY-CATEGORY [8][9][10]
- QUERY-BY-FEATURE [11][12][13][14][15][16][17][18][19]
- QUERY-BY-EXAMPLE [20][21]

QUERY-BY-TEXT is used only for keywords which are used in searching the image store for annotation purpose. These kinds of systems use keywords to retrieve and sort results based on matching. Logic can be setup to specify the extent of matching (partial or exact).

QUERY-BY-CATEGORY is used for accessing images which are categorized for facilitating fast search on the category based storage which acts as a layer for image searching over a large database.

QUERY-BY-FEATURE is used for images having letters, objects, shapes and keypoints. Searching operation is done by using this metadata which allows us to limit the search across image store.

QUERY-BY-EXAMPLE is the used when a query image is passed as input. It makes use of query image to recognize the objects/text/features. Also, it searches image store for similar images.

All systems can have their own logic to identify query image and search across.

Figure 2.1 Image Retrieval

2.1.2 Image Annotation [22][23]

Image Annotation is a process by which we can search manually/automatically words which describe images. It makes use of all metadata in terms of captions/tags to a digital image [22]. It is normally used for image retrieval techniques in the field of computer vision

for organization and location of image interests. Image Annotation has advantages in terms of performance, when we compare image annotation based image searching methods with the ones which makes use of content-based-image search techniques [23].

Multi-class image classification is one of the popular methods of image annotation which works on the huge vocabulary. Typically, annotation systems make use of machine learning techniques which generate keywords for images in image store.

Consider, following example where a FedEx truck has following tags.

Figure 2.2 Image Annotation

2.2 Text Based Models

2.2.1 Query-By-Text

Query-By-Text works on text in query and the image store [5]. Logic is setup for weighing for each tag, which contributes towards selection of specific images which can be matched [6] [7]. Bag of words is a popular method which makes use of query by text.

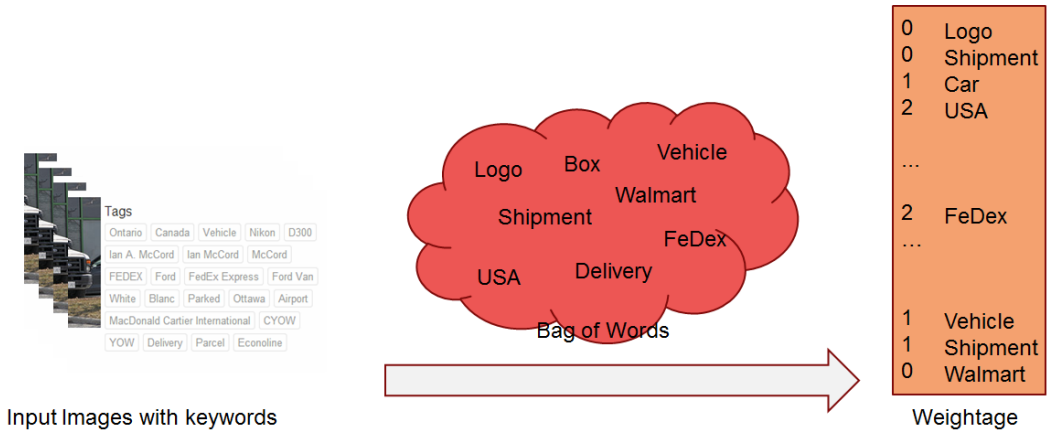

Figure 2.3 Query-By-Text

As shown in Figure 2.3, the images with their tags are shown in image store. The bag of words allows the system to assign weightages to different tags. The weightages determine possibility of image selection based on its weightage.

2.2.2 Query-By-Model

Query-By-Model is querying a particular image from image store by using a model that organizes image tags [8]. Using this model, we can generate weightages and assign accordingly to retrieve images.

Vector space model is an example of Query-By-Model which uses algebraic model for representing tags associated with images as vector of identifiers. It is normally used for tags filtering, tags retrieval, indexing and relevancy rankings. Term Frequency-Inverse Document Frequency (TF-IDF) is a popular scheme based on Vector space model [9] [10].

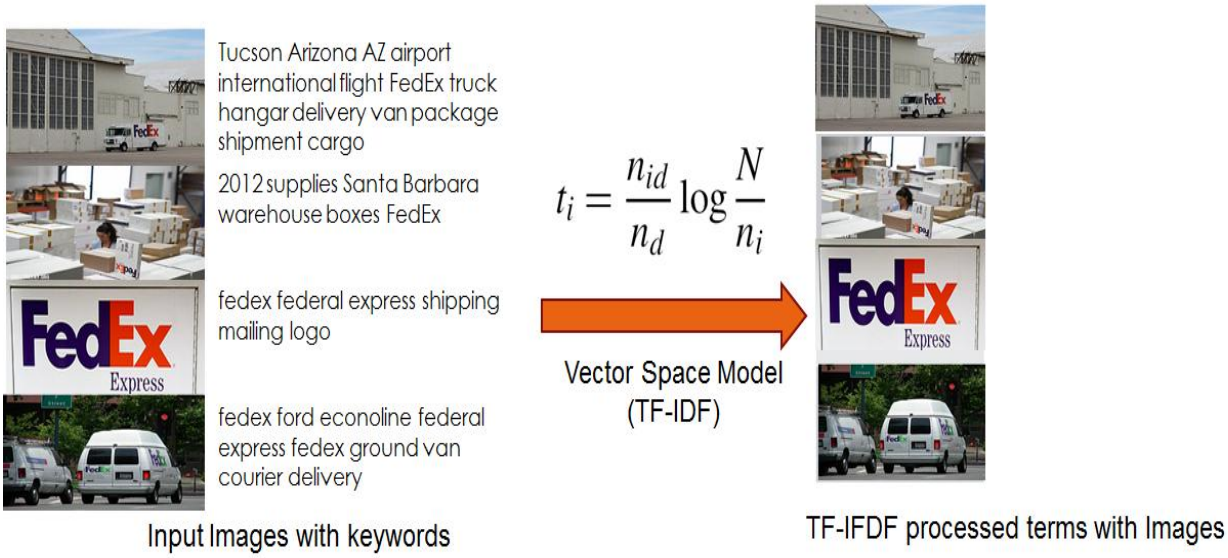

Figure 2.4 Query-By-Model

As shown in Figure 2.4, images have their respective tags as documents and they are collected from image store. We process them for a specific category of images and then use Vector Space model to assign specific weightages to the terms. The weighted terms document is used in filtering non-important tags from these images in image store and felicitate a better search.

2.3 Image Matching Based Models

Image matching has been prominent driver for developing CSISE. There are five different models in image matching which are discussed below.

2.3.1 Laplacian of Gaussian

Laplacian of Gaussian is the premier algorithm that allows selecting edges.

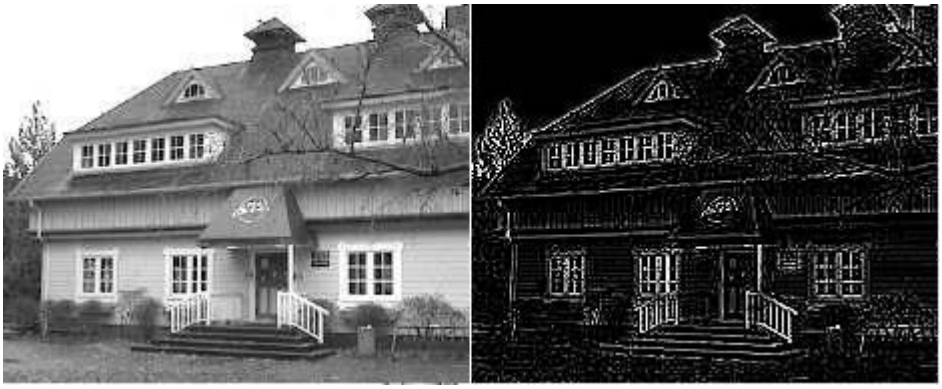

Figure 2.5 Laplacian of Gaussian

As shown in above Figure 2.5, Gaussian filter can be applied on input image to get the edges in input image. Similarly, image store contains information about edges present in all images. [11].The Laplacian filter is known for its sensitivity to noise which makes it hard to recognize objects with definite edges [12].

Examples of Laplacian of Gaussian filters are shown in Figure 2.6 below,

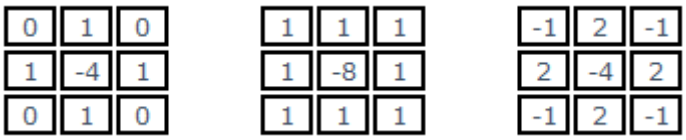

Figure 2.6 Example of Laplacian of Gaussian Filter

2.3.2 Difference of Gaussian (SIFT)

Difference of Gaussian is a famous method for feature enhancement that involves subtraction of one blurred version of original image from another, less blurred version of original. The blurred images are obtained by using convolution of gray scale images with Gaussian kernels having different standard deviations [13] [14]. In other terms, difference of Gaussian is a band-pass filter that allows discarding plenty of spatial frequencies that are present in original image.

Consider the example in Figure 2.7 below,

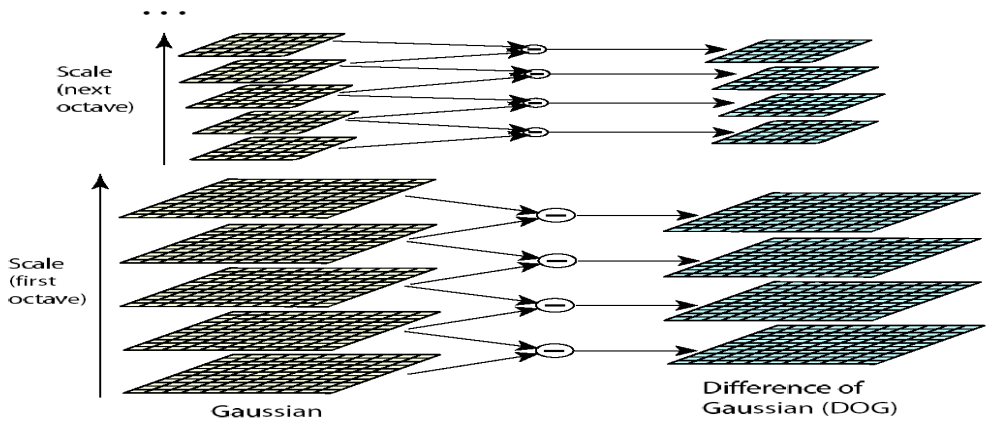

Figure 0.7 Difference of Gaussian (SIFT) Working 1

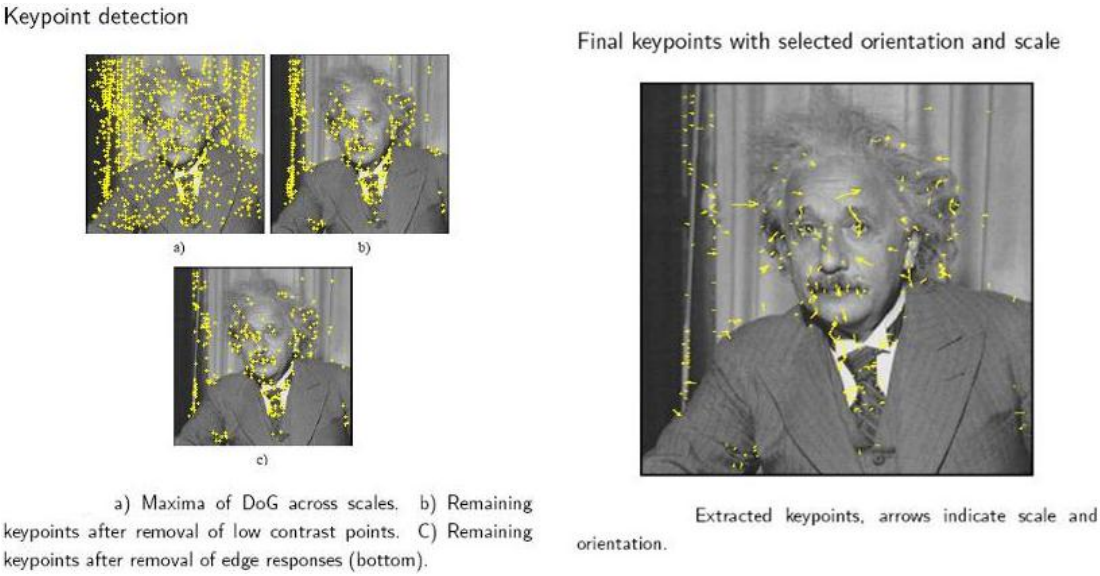

Figure 2.8 Difference of Gaussian (SIFT) Working 2

From above Figure 2.8, convoluted versions are subtracted to obtain difference of Gaussian fed to different systems for processing is shown. It also involves key steps of detection and scaling the extracted keypoints and orientation.

We observed that Difference of Gaussian performs better than Laplacian of Gaussian.

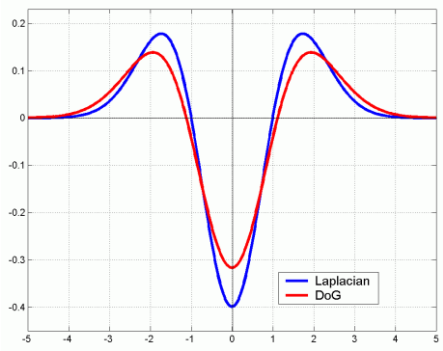

Figure 2.9 Laplacian of Gaussian vs. Difference of Gaussian (SIFT)

2.3.3 Compressed histogram of gradients

Compressed histogram of gradients has been popular on image matching which uses histogram and techniques for image processing [15].

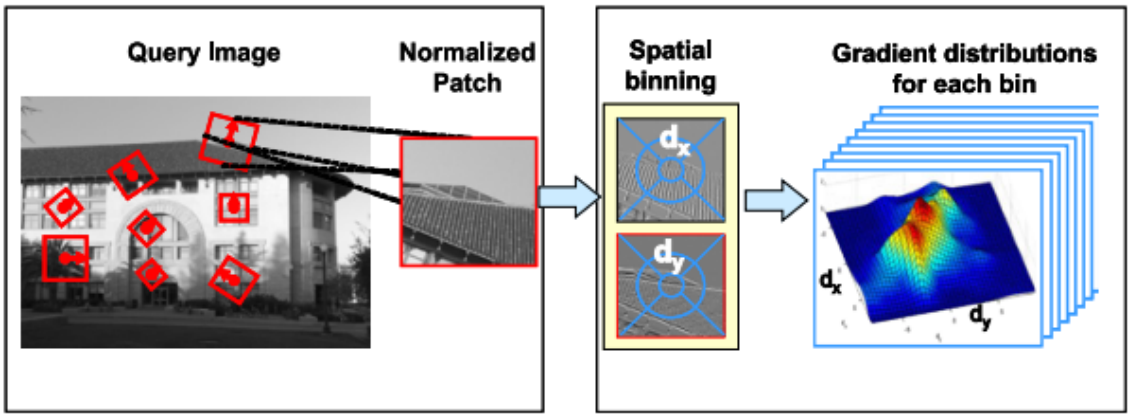

Figure 2.10 Compressed Histogram of Gradients

From Figure 2.10, patches are initially recognized, normalized for performing spatial binning. The compression of histogram makes processing easier.

2.3.4 Euclidean Distance

Euclidean distance is the distance between 2 points that one would measure using ruler and it works on basic geometry principles that allow pixel to pixel matching. It compares two images by matching distances of keypoints between them [16] [17].

- The dimensionality of vector = k (= w*h)

- An input vector x = (x1, x2, …, xk)
- A codeword yi = (yi1, yi2, …, yik)
- The Euclidean distance between x and yi

$$d(x, y_i) = \|x - y_i\|^2 = \sum_{j=1}^{k} (x_j - y_{ij})^2$$

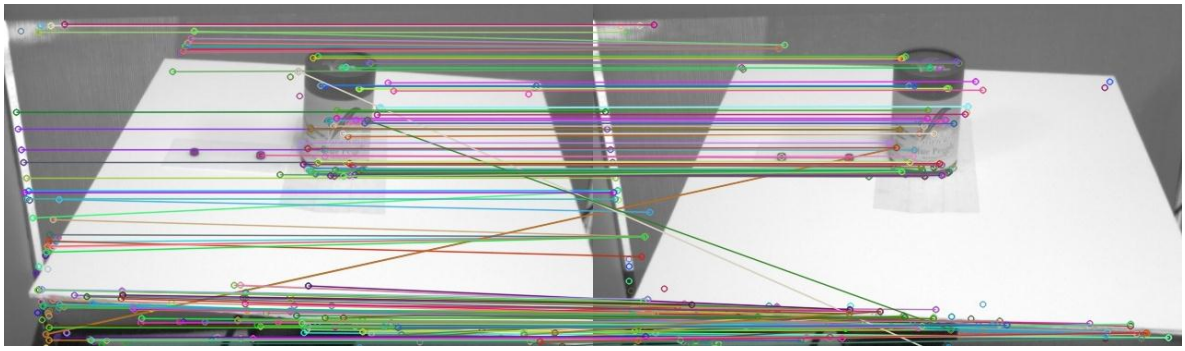

Figure 2.11 Euclidean Distance

We mapped the points to images and then compared images in above Figure 2.11.

2.3.5 Speeded Up Robust Features (SURF)

SURF is one of the key content based images searching available today, which performs several operations over data to generate key points and match the points one by one for image comparison [18].

SURF makes use of 3 steps such as,

1. Keypoints detection
2. Keypoints description
3. Keypoints matching

Keypoints detection is a process of selecting points in an image that is considered to have ‘good’ features, in terms of image quality. Previous studies on content based image searching like SIFT provides keypoints with ‘good’ features have been a key aspect and SURF returns quality ‘good’ feature points.

Keypoints description deals with extraction of descriptors for keypoints which encode properties of the features like contrast with neighbors. Keypoints matching works on comparing points from both images and it will find best points to fit image points. We can make use of Fast Approximate Nearest Neighbor Search Library (FLANN) matcher for this purpose.

SURF processes, recognizes keypoints in an image and takes care of its edges, where intensity of the points changes. Points are categorized internally to work on critical points pertaining to images. We perform keypoints matching using matcher algorithm which tries to read SURF vector points and perform matching based on algorithm for matcher [19].

Figure 2.12 Speeded Up Robust Features (SURF)

2.4 CSISE Hybrid Model: Image Matching

From several image matching algorithms, layered filtering on images is observed to serve better. Euclidean is performed on set of images and then SURF. By using this hybrid approach we can achieve the precision by reducing the comparisons needed.

Euclidean is primitive to represent images as points in high dimensional space. Fast processing of Euclidean makes it unique among others. Euclidean plays a significant role in filtering out some candidates with low similarity to query image in Euclidean system with image store.

SURF is well known for content based image searching. We concluded that performance of SURF is better than other content based image searching algorithm in terms of accuracy. With respect to speed, it is observed that SURF is 3 times faster than SIFT with similar accuracy. SURF can handle images with blurring or rotation while, it cannot handle images with viewpoints.

2.5 Applications: QUERY-BY-EXAMPLE

QUERY-By-Example is used in following applications.

2.5.1 Like.com

Like.com has logic to identify objects within image and then search objects in image store where respective metadata information is available [20].

Figure 2.13 Query-By-Example Like.com

Like.com is providing efficient selection of specific objects in cropped image among noisy surroundings. Usage of search is based on object using machine vision techniques makes it unique of all other algorithms.

2.5.2 Google Image Searching

Google image searching is the best image based search engine, which is currently available. It allows you to perform query by the image and or keywords [21].

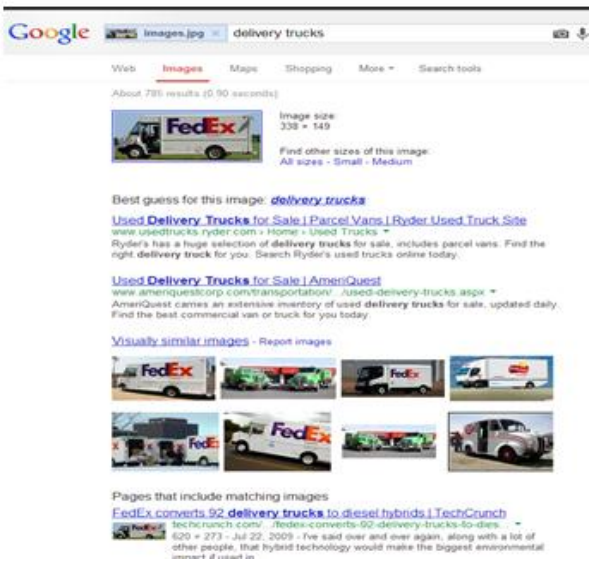

Figure 2.14 Query-By-Example Google Image Search

Figure 2.14 show processing image and generation of keywords from image. The keywords are then passed to keywords based searching to get results based on extracted keywords.

CHAPTER 3

CLOUD-BASED SEMANTIC IMAGE SEARCH ENGINE MODEL

3.1 Introduction

CSISE is a mix of several models like text searching and image processing. Image searching is a key ingredient, which is evolved in hybrid image search engine with advanced text search handling. Several approaches have been discussed in following sections which provide understanding of image searching and text searching as individual ideas. The CSISE model learns and adapts to changing features and data values.

3.2 CSISE System Architecture

The CSISE architecture is based on image matching, storage and image processing methodologies that we studied and implemented accordingly as shown in Figure 3.1. The below diagram provides several ways to perform search over the CSISE system.

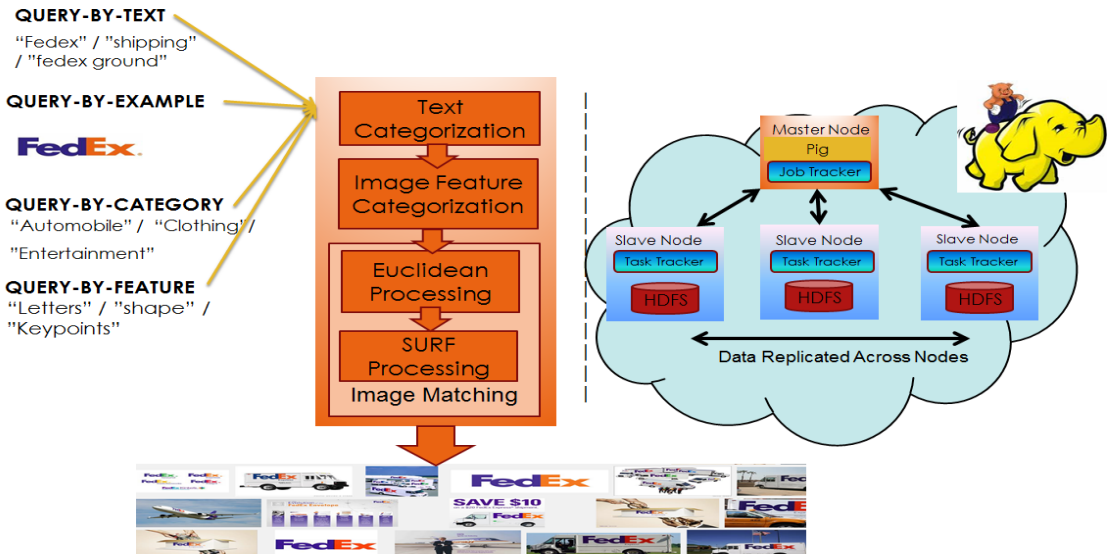

Figure 3.1 CSISE Architecture

QUERY-BY-TEXT –Users can search by text, which they use to search keywords associated with images in system. Vector space modeling [24] is performed on keywords, which makes results relevant to query text input given by users.

QUERY-BY-EXAMPLE – Users are allowed to pass an example image to the system, which searches across image store by using a layered approach.

QUERY-BY-CATEGORY – Category for storage of system is used in providing option for user to specify which he/she can narrow image search.

QUERY-BY-FEATURE - User can specify if an image has features such as letters, objects, etc., this information is assumed as metadata and searched accordingly.

These phases are briefly explained in the following sections. Also, Hadoop is used for storing images and its platform is used for processing.

3.2.1 Categorization Model

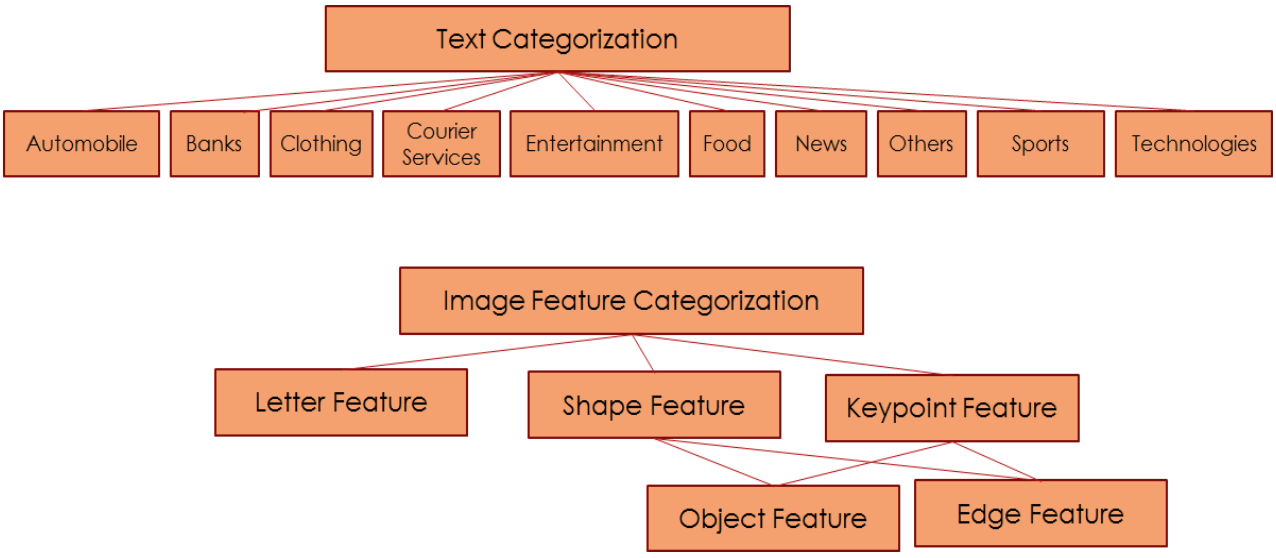

Figure 3.2 Categorization Model

Images are categorized with various companies associated with particular industries, followed by their image features like letter, shape, and keypoints. As shown in Figure 3.2, root categorization has several categories such as Automobile, Banks, etc. It is later divided into letter, shape and keypoints features. Shape and Keypoints have object and edge as features. In CSISE, we currently worked on only top level features as shown in Figure 3.2. The categorized levels help to restrict image matching that is performed by significantly less numbers.

3.2.2 Phase 1: Text Categorization

Text Categorization phase focuses on handling text portion of CSISE. As shown in Figure 3.3, we will explain 3 steps in following sections.

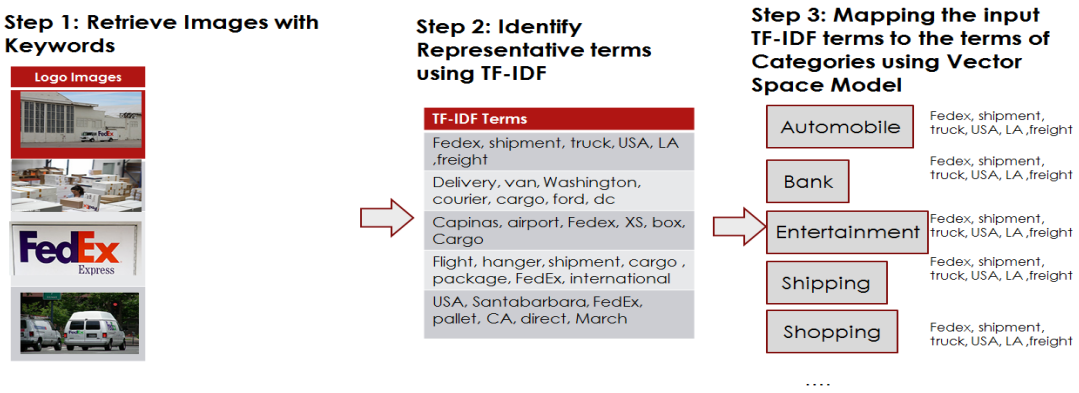

Figure 3.3 Phase 1: Text Categorization

Step 1: Retrieve Images with Keywords

In order to map input text to different sets of categories and their images, we have used text processing which understands incoming text and matches it with text data sets. Term Frequency–Inverse Document Frequency (TF-IDF) value increases proportionally to the number of times a word appears in the document, but is offset by frequency of the word in corpus, which helps to control the fact that some words are generally more common than

others. TF-IDF is used for text matching [24]. It is frequently used as a weighting factor in information retrieval and text mining. Following operations are being performed on:

- Parsing input text so that it will be supplied to comparison logic
- Comparison of parsed input text to TF-IDF terms which have been previously calculated for various categories as representative.
- Generation of categories as a result which gets ready for hybrid model

Input text is passed from input text to the parsing module. The parsing module performs basic text validation. It also checks for size of the text, existence of special characters and counts of words. As texting strategy is based on TF-IDF terms we kept restrictions on the number depending on a configurable size that we supply to the system.

Step 2: Identify Representative terms using TF-IDF

TF-IDF stands for term frequency-inverse document frequency, and TF-IDF weight is often used in information retrieval and text mining. The weight is a statistical measure used to evaluate importance of word to document in a collection or corpus. Frequency of a word appears in document as offset in corpus. TF-IDF implementation is incorporated to improve keywords filtering for screening high level categories.TF-IDF can be successfully used for text filtering in categories subject to keywords which does text summarization and classification. In Figure 3.4, we have shown formulas that we have used.

(Normalized) term frequency of term k_i in Document d_j

$$tf(i,j) = \frac{freq(i,j)}{\max_{k \in T} freq(k,j)}$$

Inverse document frequency of term k_i

$$idf(i) = \log(\frac{n}{n_i}) \in [0, n]$$

n_i number of documents in which term k_i occurs

Term weight $w_{ij} = tf(i,j)\ idf(i)$

Figure 3.4 TF-IDF Formulae

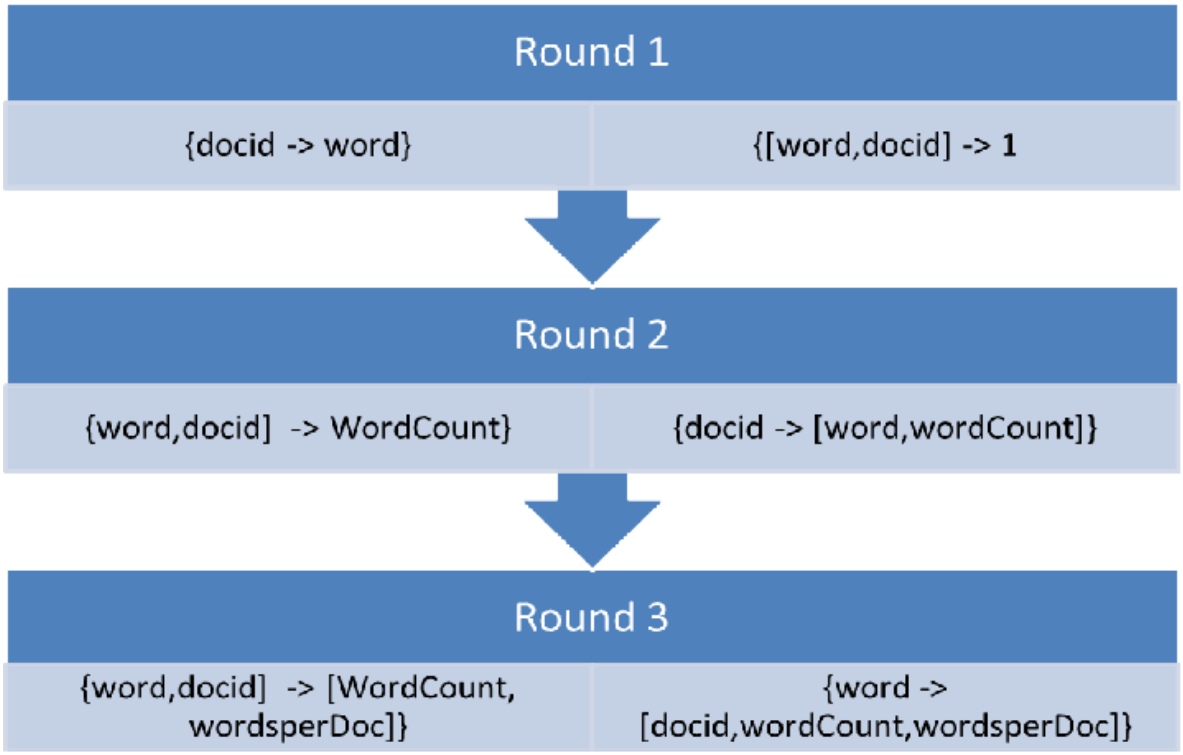

Figure 3.5 Phase 1: TF-IDF Process Steps

In Figure 3.5, the process of calculating TF-IDF terms for CSISE is shown. In round 1 , we collect all terms of respective images as documents from image store and flat files which

have terms. In round 2 ,words in a particular set of documents over the category are counted. We formulate words per document as RF-IDF in round 3.

Step 3: Mapping the IF-IDF Terms

Matching input text with TF-IDF terms results in generation of selected categories. The categories can have their respective TF-IDF terms which are calculated initially and updated model will undergo changes as we have more images added to the system. Increase in TF-IDF terms improves the overall performance. Selection of more categories at this stage can probably reduce comparison for next stage, but can affect accuracy. Selection of single category can work well on accuracy and can turn out impact resulting sets.

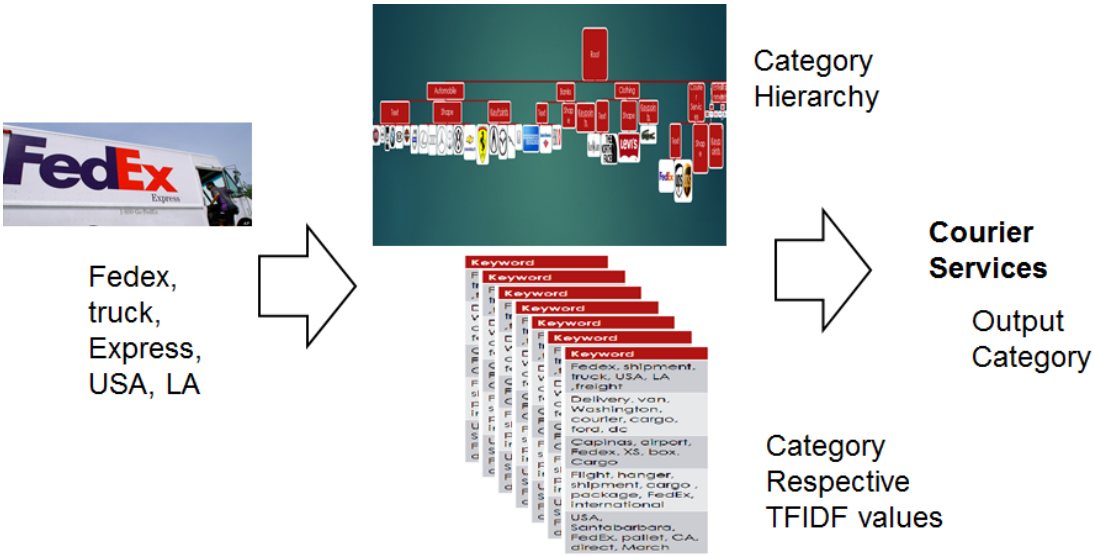

Figure 3.6 Keywords Matching: Example

A query image is fed to the Text categorization module which has Keyword Categorization. The keywords associated with image are checked against the TF-IDF values for specific categories. Input image is checked with other TF-IDF terms, representing each category. As shown in Figure 3.6, “Courier Services” will be selected based on the input image and will be used by Feature Categorization system.

3.2.3 Phase 2: Image Feature Categorization

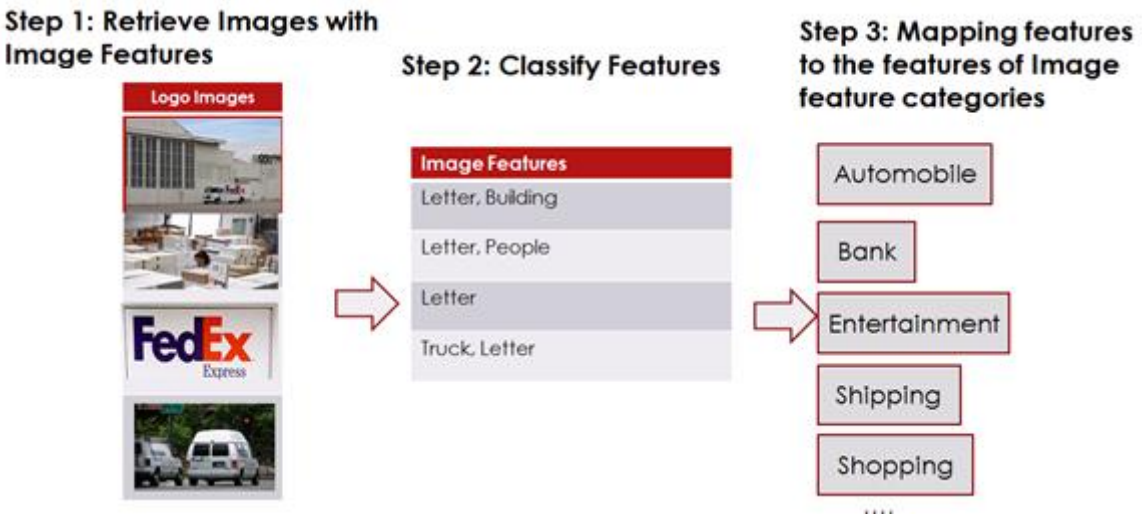

Figure 3.7 Phase 2: Image Feature Categorization

Step 1: Retrieve images with image features

In Figure 3.7, the process of image feature categorization is shown. All images are collected with features as metadata; and clubbed them based on logo details. The collected metadata is used to extract key features based on image size. In our current CSISE, we manually assign the metadata to image and compile the mapping features.

Step 2: Retrieve images with image features

Images are classified and assigned the image features based on the industry. The industry separation will allow us to map images to individual feature level hierarchy. Also, Industry has associated feature levels. And the above set will be used while searching.

Step 3: Mapping features to feature categories

CSISE places the images to different image features bin which are attached to particular industry which are at highest level. The industry level bins will eventually have the real logo set and the logos will be categorized and organized based on the features.

Consider following example where the input category is searched for the text.

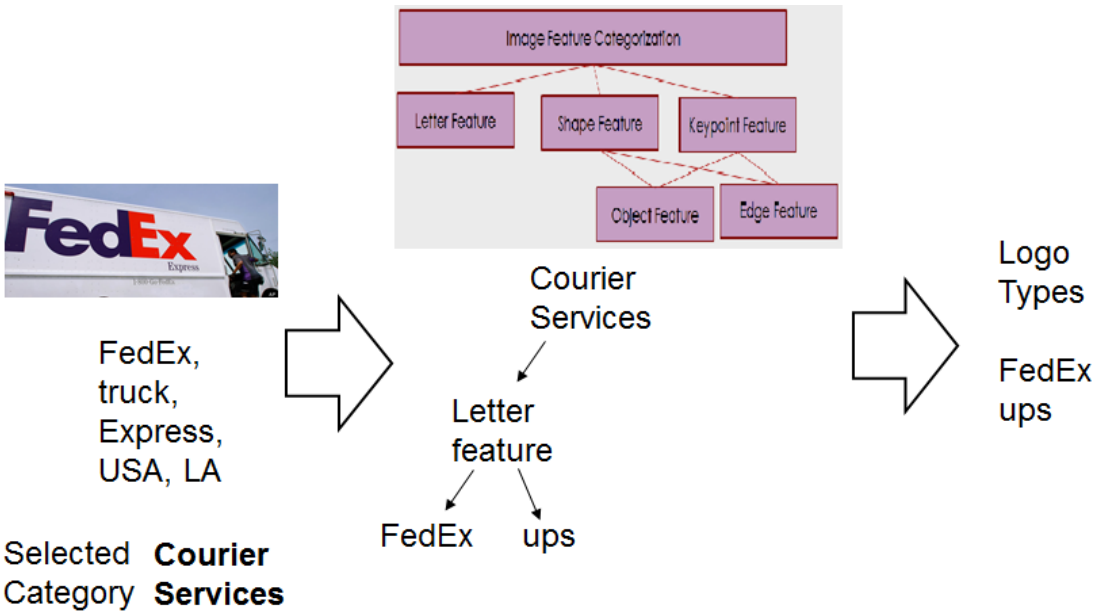

Figure 3.8 Image Feature Matching Example

In Figure 3.8, selected category and image feature hierarchies are checked for accurate category which needs to be searched. If a particular category selected was “Courier Services”, it utilizes an image feature hierarchy. Courier Services has only Letter feature, which has 2 entries and both entries are to be selected to perform image processing operations on it.

3.2.4 Phase 3: Hybrid Image Matchmaking

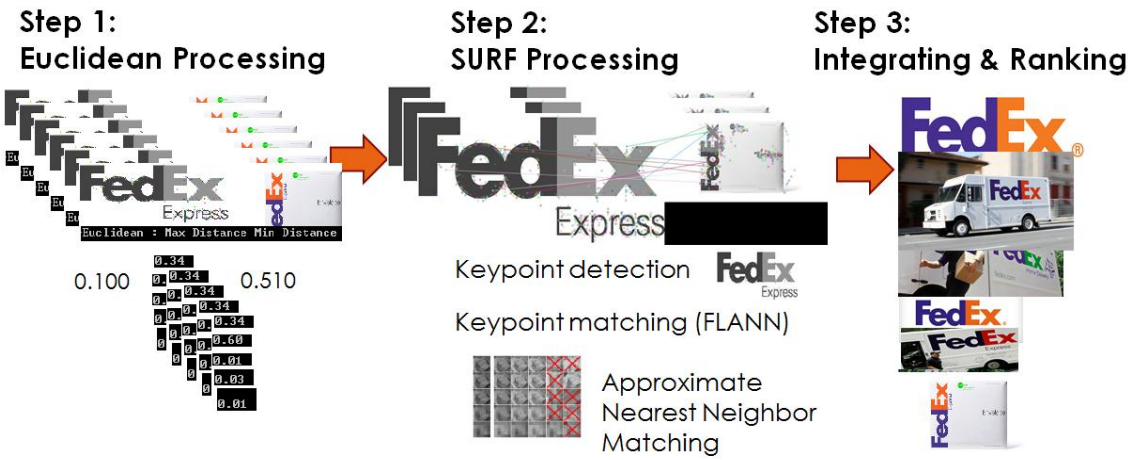

Figure 3.9: Hybrid Image Matchmaking

Hybrid Image Matchmaking is one of the approaches discussed here .It helps us to handle the image aspects of our database and giving more accurate results for searching. In Figure 3.9, we have the steps; we need to perform to achieve Hybrid Image Matchmaking. We take two layered approach, where first Euclidean image processing is applied followed by SURF image processing [18]. In following sections, we describe the process for each of the image processing types. List of categories can be obtained from the previous stages, which will lead to selection of appropriate categories and images. Selection of images is also based on the keywords matched.

Step 1: Performing Euclidean Image Processing

Distance mapping is frequently used in picture processing. Usually, it is based on one of the metrics. We used following formulas to develop, Euclidean distance processing algorithm. Also, we explain their processing in this section. d (i,j) is the "city block distance", whereas the other distance is called "chessboard distance"

$$d \quad ((i , j) , (h , k))= [i -- hi + IJ – kl]$$

$$ds((i,j),(h,k)) = \max(I/- h I, [j - kl)$$

Based on a two-component descriptor, a distance label for each point is shown. Euclidean distance maps can be generated by effective sequential algorithms. The map indicates for each pixel in the objects (or the background) of the original binary picture is the shortest distance to the nearest pixel in the background (or the objects). A map with negligible errors can be produced in two picture scans which include forward and backward movement for each line. Thus, for expanding/shrinking purposes it may complete very successfully with iterative parallel propagation in the binary picture itself.

Figure 3.10 Phase 3: Euclidean Image Processing

The image set in Figure 3.11 is being checked against each other to find out the similarity score.

The images having similar features have,

Minimum Euclidean distance less than 0.100

Maximum Euclidean distance is less than 0.510

Consider the Figures 3.11 and 3.12,

Figure 3.11 Phase 3: Euclidean Image Processing Input

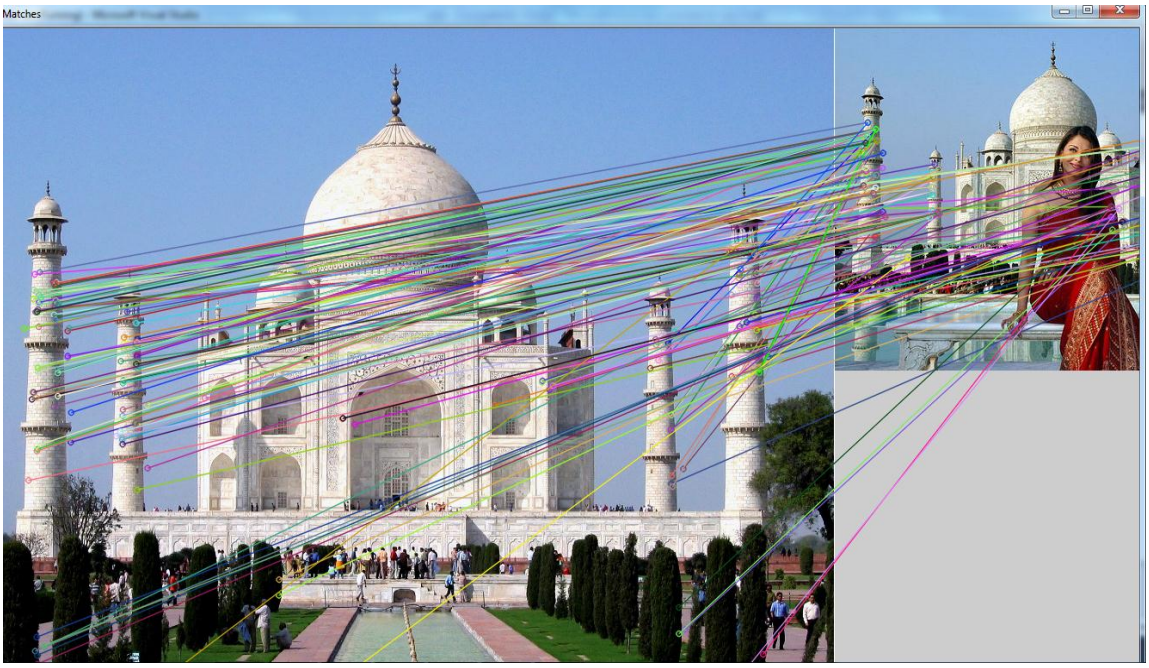

Figure 3.12 Phase 3: Euclidean Image Processing Output I

As shown in the Figure 3.11 and 3.12, two images have been compared with each other using Euclidean matching technique. First the image distances are calculated and then they are matched with each other to understand the similarity between the points.

The similar points are being mapped and the image is created which have mapping of the points. If one image is matched to same image, the mapping is same and minimum and maximum distances are 0. If images are of a same size, then minimum distance and maximum distance are in the range of (0-0.51) and (0-0.1). Several operations are performed on Euclidean distance, and it has been concluded that if Euclidean distance is same, images are exactly the same. But, if images are matched in intermediate state, range is used to determine approximate similarity.

If images are of variant sizes, the range changes and it gets difficult to keep track of image that represents a cluster of images. The system works well for images that are independent of sizes .This distance change can be observed below.

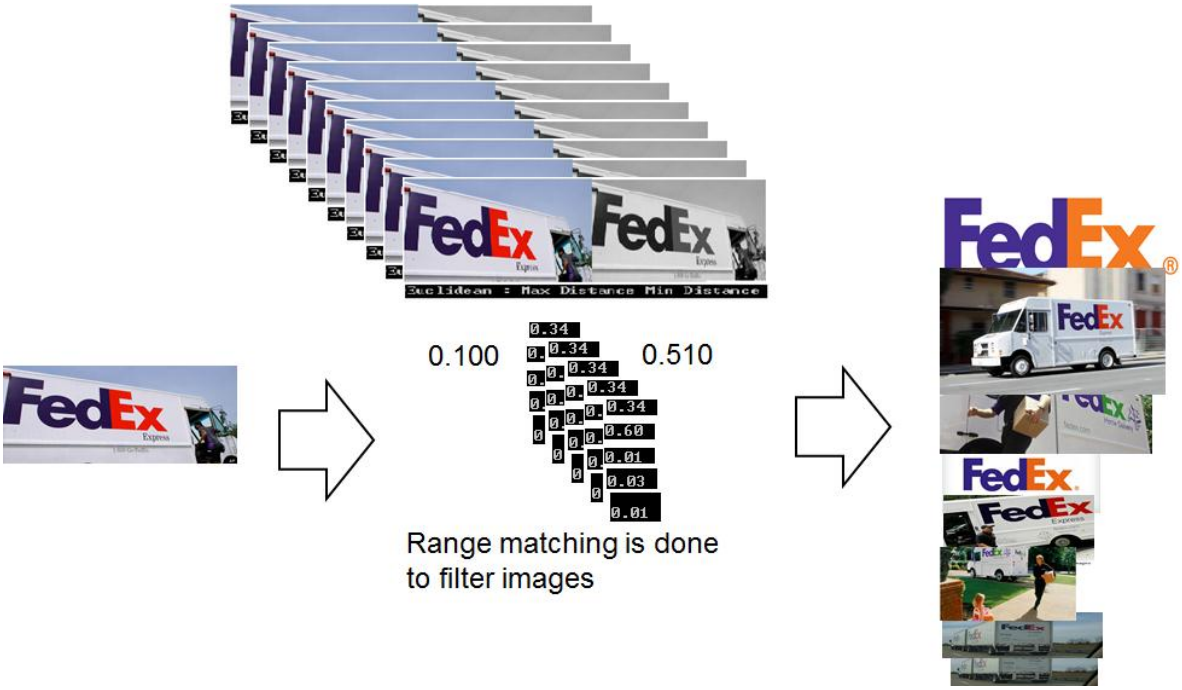

Figure 3.13 Phase 3: Euclidean Image Processing Output II

As shown in Figure 3.13, input image is checked against several images in image store and then retrieves the list of images with some similarity which falls in the range, which we derived.

Step 2: Performing SURF Image processing

This is a sample of matching descriptors detected on one image to descriptors detected in the image set. So, one query image and several train images are used. For each key point descriptor of query image, the nearest train descriptor is found on the entire collection of train images. To visualize the result of matching, images are saved and each of which combines query and train image with matches between them (if they exist). Match is a line drawn between corresponding points. Count of all matches is equal to count of query key points, so we have the same count of lines in all sets of resulting images (but not for each result).

FLANN Based- Fast Approximate Nearest Neighbor

Step 1: Detect the key points using SURF Detector

Step 2: Calculate descriptors (feature vectors)

Step 3: Matching descriptor vectors using FLANN matcher

Step 4: Quick calculation of max and min distances between key points

Step 5: Draw only "good" matches

Step 6: Draw only "good" matches

Figure 3.14 Phase 3: SURF Image Processing Input

Figure 3.15 Phase 3: SURF Image Processing Output

As shown in Figure 3.14 and 3.15, the images from image store are being searched for the logo and results are shown as well. Figure 3.14 has sets of input images along with log has to be searched. After performing the SURF operation, results are recordsed in Figure 3.15.

Now, the FLANN based approach is appropriate to get the matching images where the comparison is with small objects such as logos, pen, balls, mobile, etc. Performance also varies with the number of different points under consideration. Matching will improve the objects search i.e., very compact mean the exact size and the dimensions

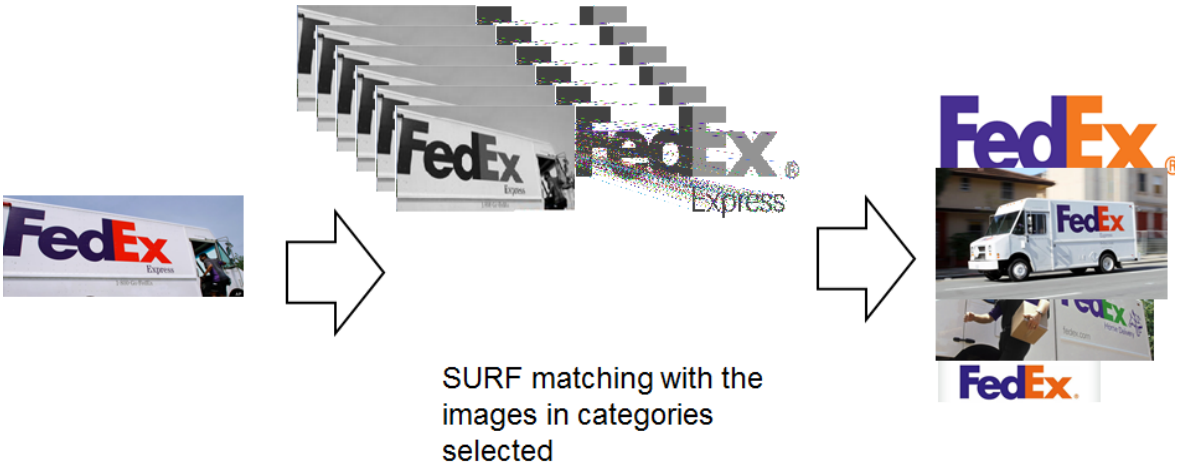

Figure 3.16 Image Processing Example: SURF Module

As shown in Figure 3.16, it is observed that the query image is gray scaled and fed to the SURF processing. SURF vectors of image store are matched against SURF vectors of the input image. Filtered images are processed against FLANN.

Step 3: Integration and Ranking

Images are recorded and count of matching points to dataset in matrix form. The matrix has its own properties which are being understood by interactive modules. In the last part of the hybrid process sorting results is performed based on matched points. We forward the list of images and their locations to the Merging module. Once we get the list of points matched, we sort it and the list is given a ranking. The highest points matched logo is being put on top.

CHAPTER 4

CLOUD-BASED SEMANTIC IMAGE SEARCH ENGINE (CSISE) IMPLEMENTATION

4.1 Introduction

The CSISE architecture is divided into several modules. Images are stored on HDFS of Hadoop system. Pig system is used for processing images. The keywords are stored on flat files which we read when needed.

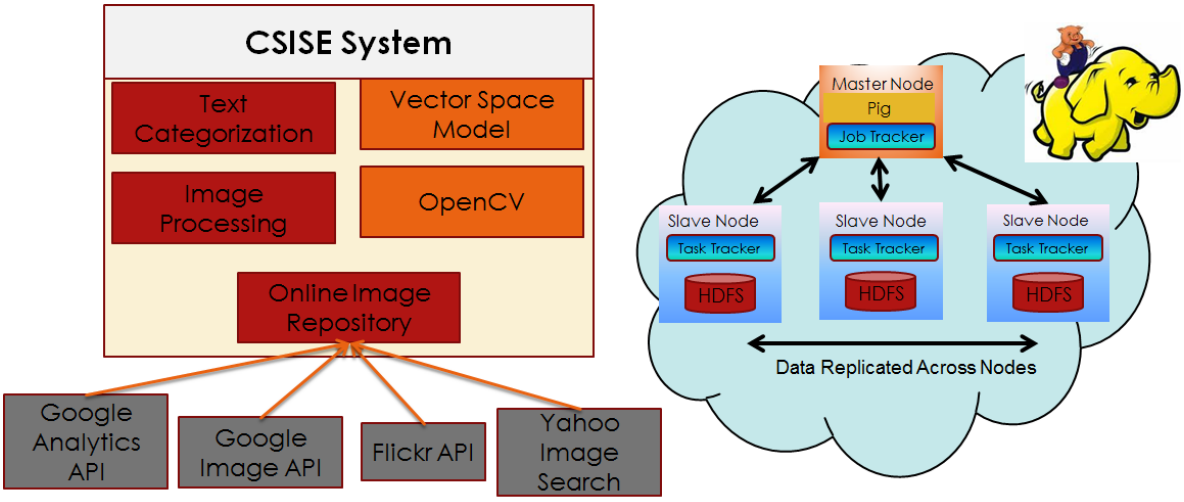

Figure 4.1 CSISE System: Platform and Architecture

As shown in above Figure 4.1, the CSISE system is divided into modules, each module interacts with other for searching. Text Categorization and Image Processing are the key modules which work with Vector space model and OpenCV. Online Image repository is developed from various sources as described in following sections. The Hadoop configuration shown in the diagram has Pig installed on master node, where the data is accessed on slave nodes.

4.2 CSISE Configurations

Use of IBM Smartcloud platform is made while working on CSISE system. The virtual instances provided by IBM were used in Windows and Linux based environment. CSISE

code is deployed in the Red Hat Enterprise Linux environment having server configuration of Copper - 64 bit (CPU: 2, RAM: 4 GB, Disk: 60 GB). The Hadoop systems had 2, 3, 4 node configuration with master/Slaves settings. HDFS is used for file storage and ran Pig to process the OpenCV code written mostly using Python/C++. Online Image Repository has stored images from Google Image, Yahoo and Flickr. Vector Space Model implementation was done in Python. Visual Studio, Eclipse and Notepad++ were used while working on the development of CSISE. Flat files were used for database operations.

4.3 CSISE Architecture

This section describes the thesis implementation for developing framework for Semantic Image Search Engine. The CSISE system is implemented in 2 ways. First, on Windows platform, where OpenCV libraries are used and C/C++ based desktop based applications. The second implementation is done on Hadoop framework using Pig, Python and OpenCV libraries. Thesis is presently running on Windows machine and several Hadoop instances. The interface for the thesis is currently hardcoded in the configuration file. Division of the CSISE architecture into several modules is shown in Figure 4.1.

4.2.1 Text Categorization

As shown in Figure 4.1, Text Categorization is one of the processing module which focuses on processing the text part of input query. Particular piece of text is being processed in two level hierarchies which allow us to filter out the key ingredients effectively. Text Categorization process is divided into two levels: Keyword level and Category level. At keyword level, it checks if the input keyword exists in the keywords list in the TF-IDF for

particular category type. Particular type of category is selected against image feature. Text Categorization module has 2 compartments: Keywords Matching and Image Feature Matching.

4.2.1.1 Keywords Matching

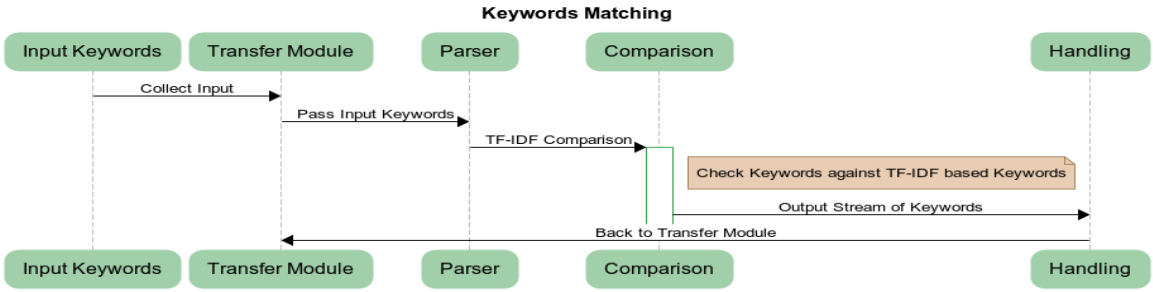

Figure 4.3 Sequence Diagram: Keywords Matching

The list of keywords is passed as an input to system and parsing of keyword is performed which takes care of simple text setups and allow to check the number of strings, the length of input strings. The parsing logic provides limits of text processing.

The parsed text is being checked against comparison function which works with TF-IDF stacks of data and allows specific logo types selection. This selection is based on TF-IDF that select a level of category types. Different TF-IDF logic ranging from 4 to 7 TF-IDF keywords are setup for whole system.

Handling module checks for keywords recognized in comparison function and arranges them in the matching order, matched logo types are checked against the image features which are being treated as metadata.

4.2.1.2 Image Feature Matching

Once, text categorization on input image is performed and its keywords are generated, category hierarchy process is applied.

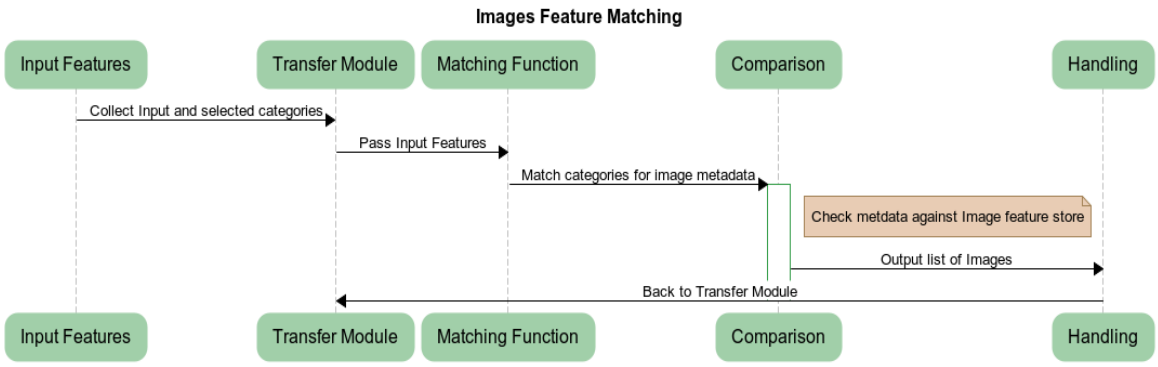

Figure 4.5 Sequence Diagram: Image Feature Matching

In Figure 4.5, image feature data is processed using the Image Feature Matching module.

Selected categories from the previous keywords matching module are used. Image selection is restricted to few feature leveled sub-categories. Data is divided into several internal feature based sub-categories which form a base for an extra level of searching.

Matching function will perform category level mapping to data which organized into categories using image features. Image features can be letters, shapes and keypoints. Letters category has logos with more stress on text within, whereas the shapes category will have more emphasis on shapes. And rest of the logo types we put into keypoints.

Handling module plays the role of sorting the logo types, which selected and can undergo the image processing module for logo types.

4.2.2 Image Processing

Image processing is the core of the whole CSISE system and developed by using several models. Two levels of image processing are kept, one works on Euclidean processing model and other on SURF processing.

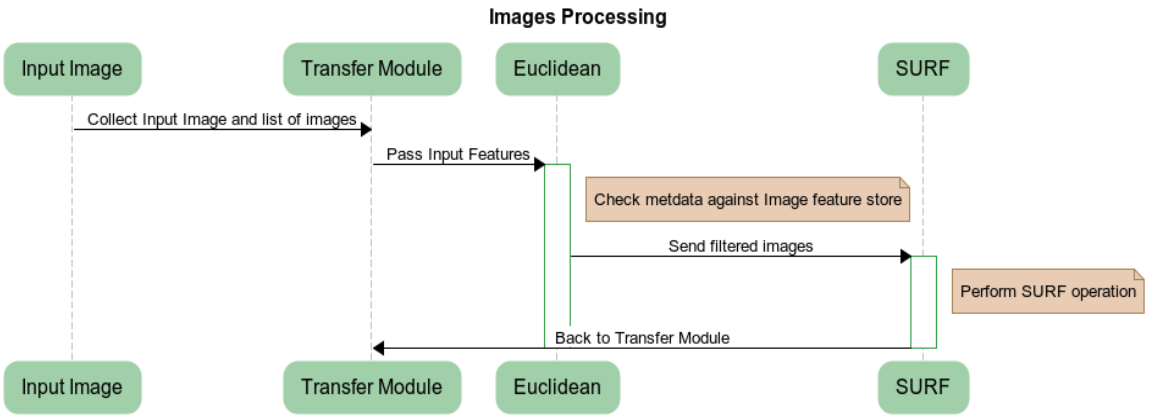

Figure 4.7 Sequence Diagram: Image Processing

As shown in the Figure 4.7, it is observed that the categories selected in the text categorization process are being processed for the image processing operations. Query on the cloud location is performed where all the images are stored and are organized as per the categorization hierarchy. CSISE first processes images with Euclidean followed by SURF model. Euclidean model have set specific ranges for which has some level of similarity and which we try to compare and get the list of matching images. The resulted image list is being sent to SURF model.

At SURF model, filtered images by Euclidean are checked against SURF logic. Here, FLANN matcher is used which allows matching the SURF vectors and generating the similar images list. Matching points are set to 5. Having several experiments done we set matching point's threshold to produce new approach of filtering images.

4.3 Online Image Repository

Use of various online resources is made when to make an Image repository for CSISE. Google Image Searching, Flickr, Yahoo is key resources while working on the collection of images. Also, we have developed several applications for collecting these images which also consumes the overall timing along with application development.

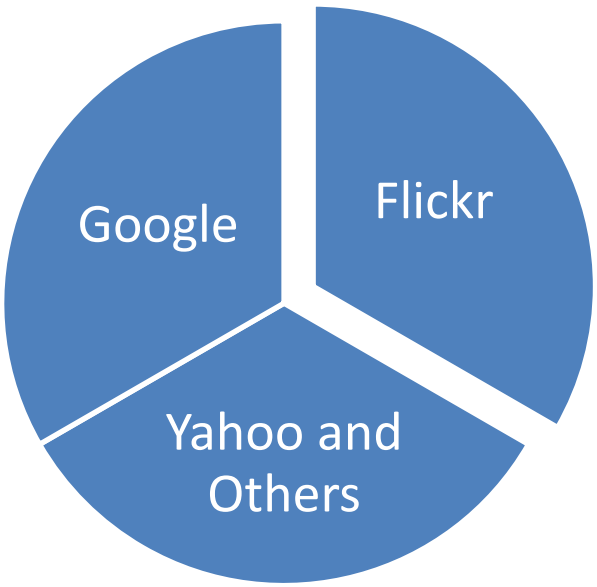

Figure 4.10 Online Image Repository

By making use of developers module provided by the search engines, download of several images is done. Search engines like Flickr has tremendous amount of text keywords and along with images. Also, we have added images and keywords manually.

4.4 Hadoop Architecture

Using IBM SmartCloud, CSISE made use of Hadoop architecture where we changed number of master/slaves in the system to perform variations while working on CSISE

system. Each slave node is connected to master node. Once, we submit a job to master node, it is passed to slave node by job tracker. Job tracker divides the job across the number of available slave. And it will handle combined result of its own.

We replicated image data in all slave nodes so that data will be available at any time and can fasten the system. Pig usage allows us to make use of any high level language such as Java and Python. Pig translates the code from C++/Java to map-reduce program and passes it to the Job tracker.

EXPERIMENTAL RESULTS AND EVALUATION

5.1 Data and Categorization

In CSISE, two datasets with sizes 726183 and 53,200 (700 images for 76 Logo types) Logo images are used. As described in previous chapter, we have collected images from various sources. As shown in below Table 1, count images per logo type were copied for "Automobile" category.

Table 1 Categorized Data

Logo	Type	Count Of Images
Harley Davidson	Automobile	9850
Chevrolet	Automobile	9892
Volkswagen	Automobile	9996
Ferrari	Automobile	9822
Mazda	Automobile	9906
KIA	Automobile	9530
VOLVO	Automobile	9678
Lexus	Automobile	9825
Nissan	Automobile	9685
Honda	Automobile	9959
Audi	Automobile	9976
BMW	Automobile	9375

5.2 Experimental Setup

Several tests are done over CSISE and other competitive solutions from which 4 were mainly considered. One with Large Scale Image Data, second with Small Image Matching

data, third with other search engines and the last one was with different parallel processing logic.

In Large Scale Image matching methods, a large image set was tested with three methods namely SURF, Hybrid and CSISE. TF-IDF categorization is performed from 3 to 6 in which Accuracy (Precision) and Runtime Performance was measured in seconds.

In Small Scale Image matching methods, Euclidean, SURF, Hybrid and CSISE methods are used to test large image set. TF-IDF categorization is done using 5 TF-IDF terms. Hadoop configuration used was of 4, 1 master and 3 slaves. We measured Accuracy (Precision) and Runtime Performance in seconds.

In searching with other Search engines methods, small image set is tested with CSISE, Google Image Search, Flickr, Yahoo, and Bing. TF-IDF categorization is performed using 5 TF-IDF terms. Hadoop configuration used was of 4, 1 master and 3 slaves. We measured Accuracy (Precision) and Runtime Performance in milliseconds.

In classification with parallel processing methods, large image set is tested with three methods namely Euclidean, SURF, Hybrid and CSISE. TF-IDF categorization is performed using 5 TF-IDF terms. Hadoop configuration used was 2PE, 3PE, 4PE and runtime performance is measured in seconds.

5.3 User Study

A user study is conducted, which focusses on understanding user's perspective on image searching and how they think about image searching as a whole. About 80 % users

selected for survey are from IT industry and aged 25-40 years, among which 60% of them were female and rests are male. They were requested to rank 5 logos according to their higher searching capability in decreasing order.

5.4 Vector Space Model – Accuracy (Precision)

In below Figure 5.1, categories are arranged across accuracy (precision) which was achieved while testing large data over CSISE. From Figure 5.1, it can be observed that performance of CSISE improves when number of TFDIF terms is 5/6. In few categories such as Sports and Technologies good performance in the trend is observed. In other categories, a good spike is observed but due to less or same number of keywords across whole category. The performance of each category varies as per number of TF-IDF terms.

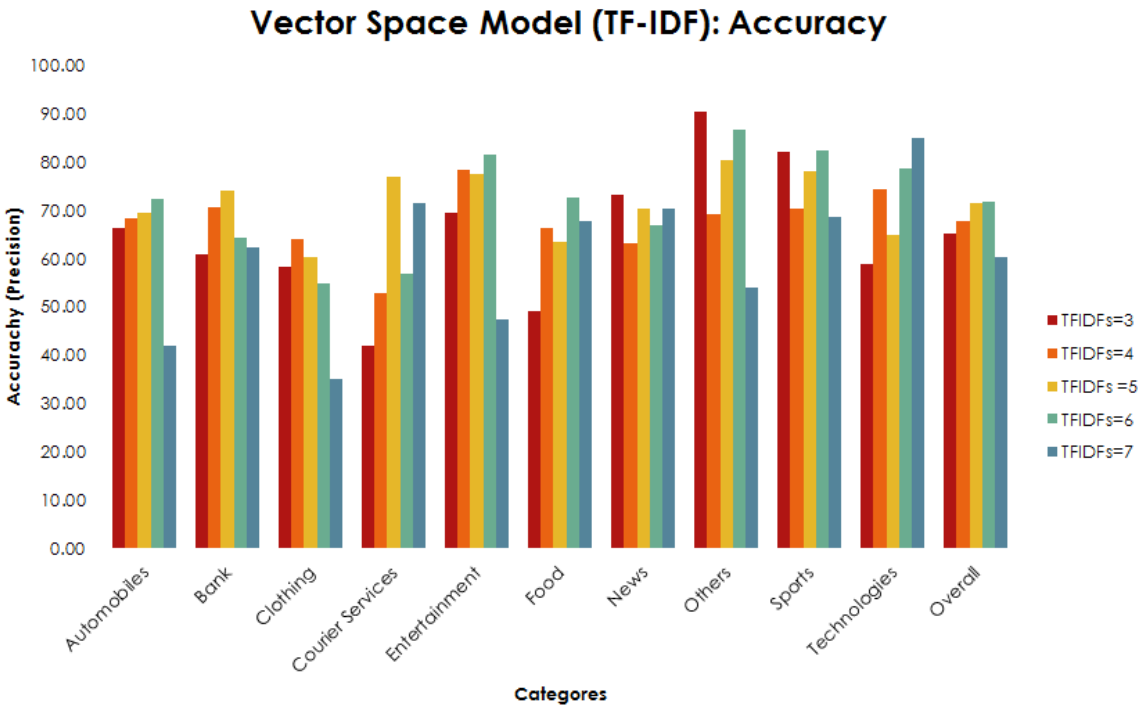

Figure 5.1 Vector Space Model (TF-IDF): Accuracy

5.5 Large Scale Searching Evaluation: Accuracy (Precision)

In Table 2 and Figure 5.2, performance of Euclidean, SURF, Hybrid and CSISE for different categories along with number of images and types of logos in each category is written in section 5.1.

Various approaches of image searching are analyzed as listed in table and saw that overall performance of CSISE is around 71% in terms of accuracy, as shown in Table 2. Number of logo types within each category varies from 2 to 19, making use of larger image database. Performance of Sports, Entertainment categories have been better than other categories due to the presence of letters and objects in the logos. The performance of Euclidean and SURF, has been poorer than the CSISE model.

Table 2: Large Scale Searching Evaluation

Category	Type #	Image#	Euclidean	SURF	Hybrid	CSISE
Automobile	18	174360	13%	39%	47%	70%
Bank	3	27911	14%	43%	58%	74%
Clothing	4	38122	13%	46%	48%	60%
Courier Services	2	18819	13%	43%	52%	77%
Entertainment	4	39717	19%	44%	47%	77%
Food	7	65538	14%	41%	40%	63%
News	3	29383	13%	40%	55%	70%
Others	8	76396	16%	40%	49%	80%
Sports	8	76868	18%	43%	47%	78%
Technologies	19	179069	18%	41%	47%	65%
Overall	76	7261983	16%	41%	49%	71%

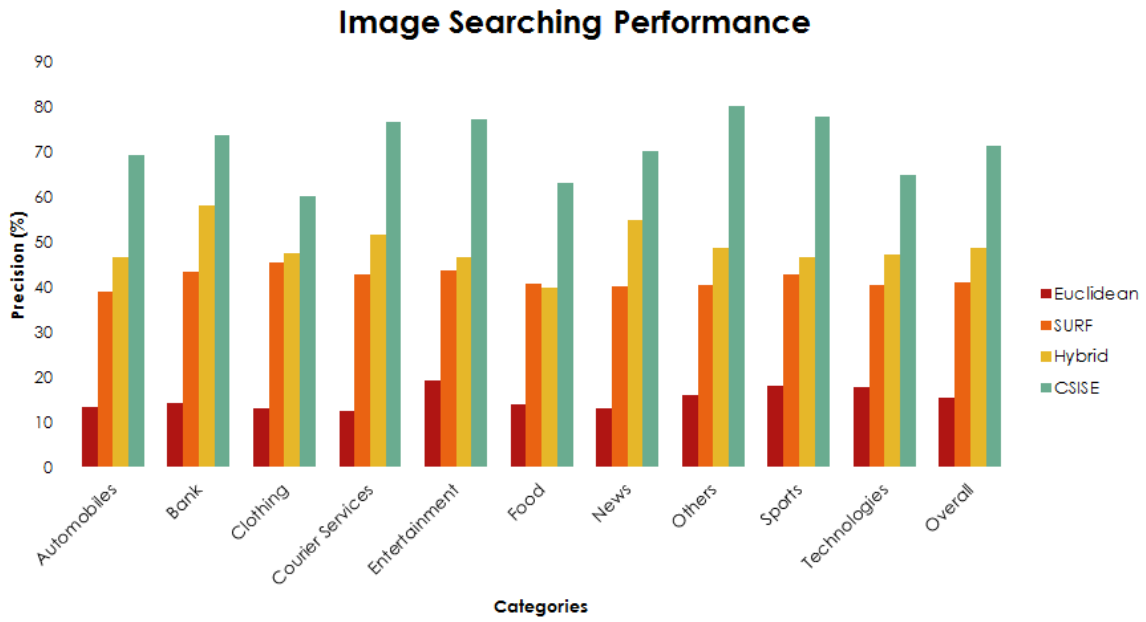

Figure 5.2 Image Searching Performance

Several operations are performed on CSISE system by changing configuration of Hadoop Master/Slaves setting. As the number of nodes increase, the performance improves as shown in Figure 5.3, Figure 5.4 and Figure 5.5. In Figure 5.3, experiment number is indicated across time in milliseconds in Hadoop 2PL environment. Time needed for Hybrid approach is less than Euclidean and SURF. CSISE is better than other 3, as filtered approach will save the number of comparisons needed at few steps.

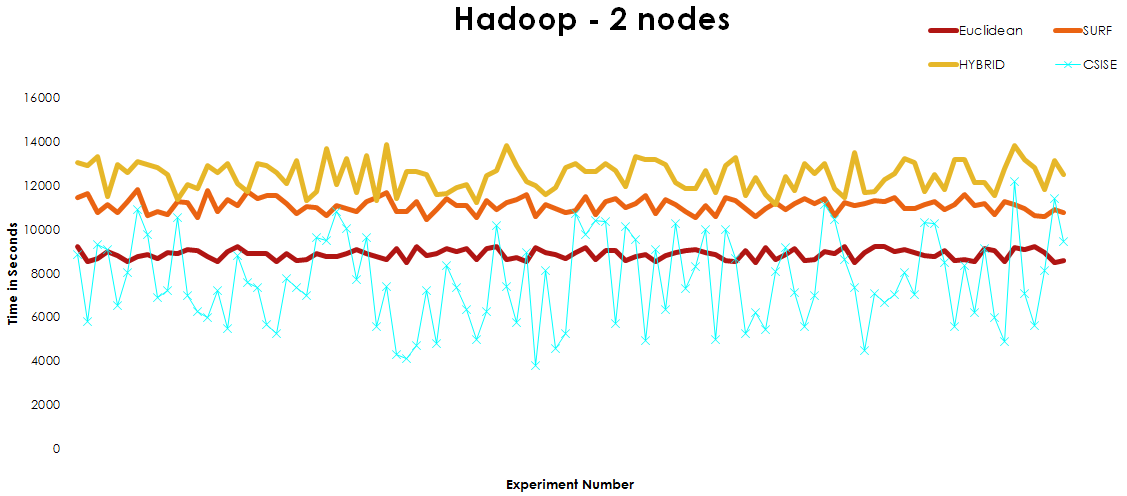

Figure 5.3 Hadoop – 2 Nodes Performance

In the below Figure 5.4, Experiment number and times in milliseconds share the X and Y axes respectively in Hadoop 3PL environment. Time needed for the hybrid approach is less than Euclidean and SURF by a better margin than 2PL approach. CSISE is better than other 3, as the filtered approach will save the number of comparisons needed at few steps.

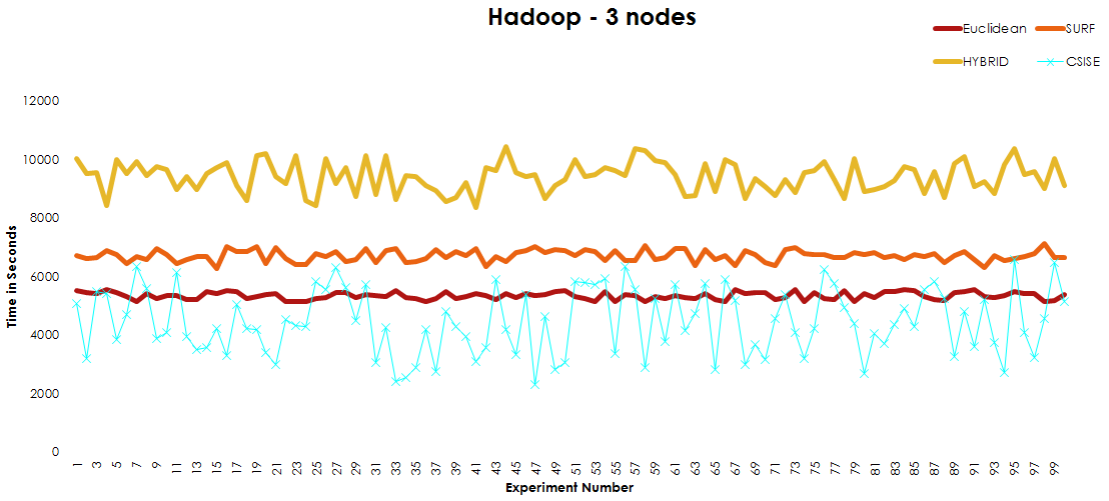

Figure 5.4 Hadoop – 3 Nodes Performance

In this Figure 5.5, time needed for the hybrid approach is less than Euclidean and SURF by a smaller margin than the 4PL approach. 3PL gave the optimum performance in other approaches.

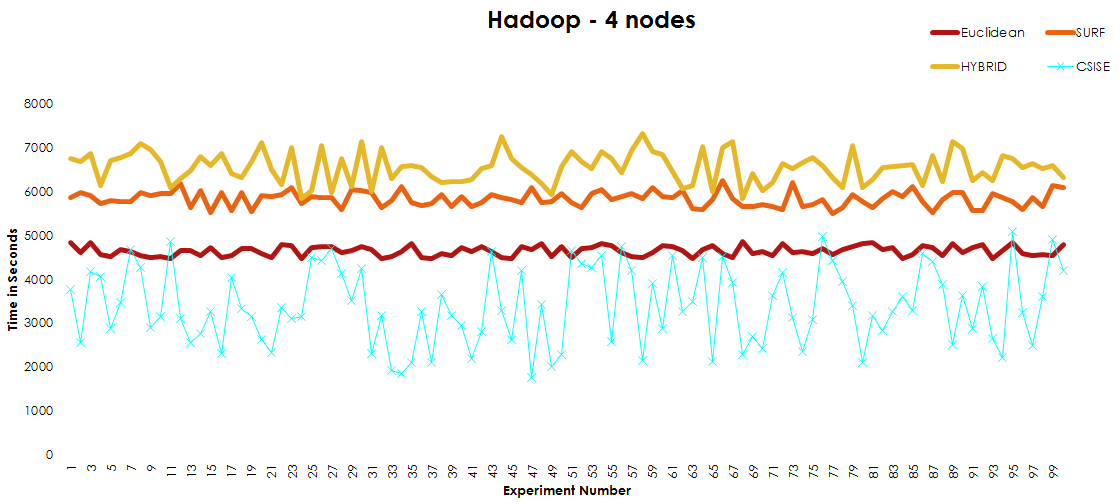

Figure 5.5 Hadoop – 4 Nodes Performance

Figure 5.6 shows CSISE performance over changes in the Hadoop configurations. It is observed that the performance is lower for two nodes and improves as we increase the number of nodes. We can see the variations of performance from the below Figure 5.6.

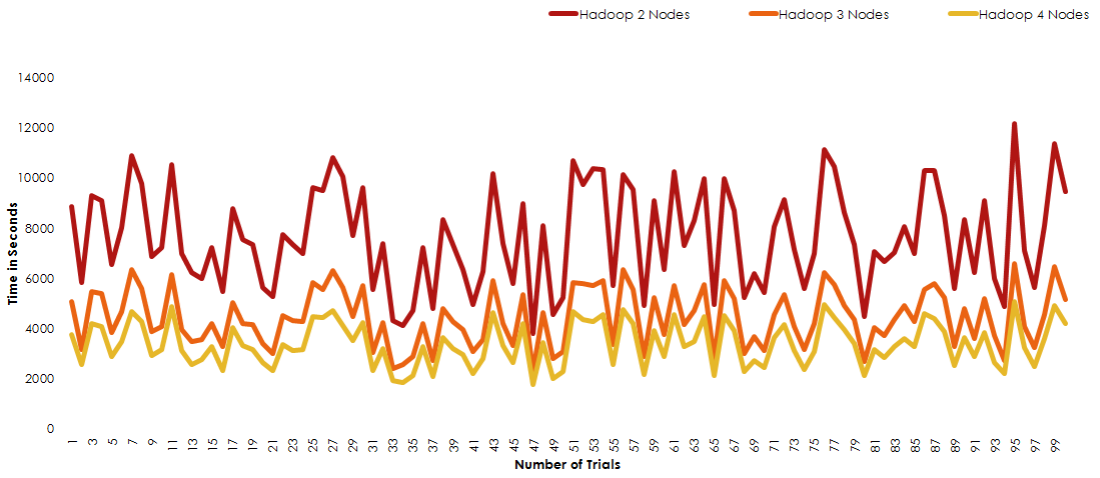

Figure 5.6 CSISE Performance for different Hadoop Configurations

We recorded the number of comparisons we do for each methodology and found that comparisons are reduced in case of CSISE approach, shown in Figure 5.7. The number of

comparisons has to be optimum for Hybrid where we perform Euclidean over SURF using all selected images through text categorization. Using CSISE approach we see performance is better than others.

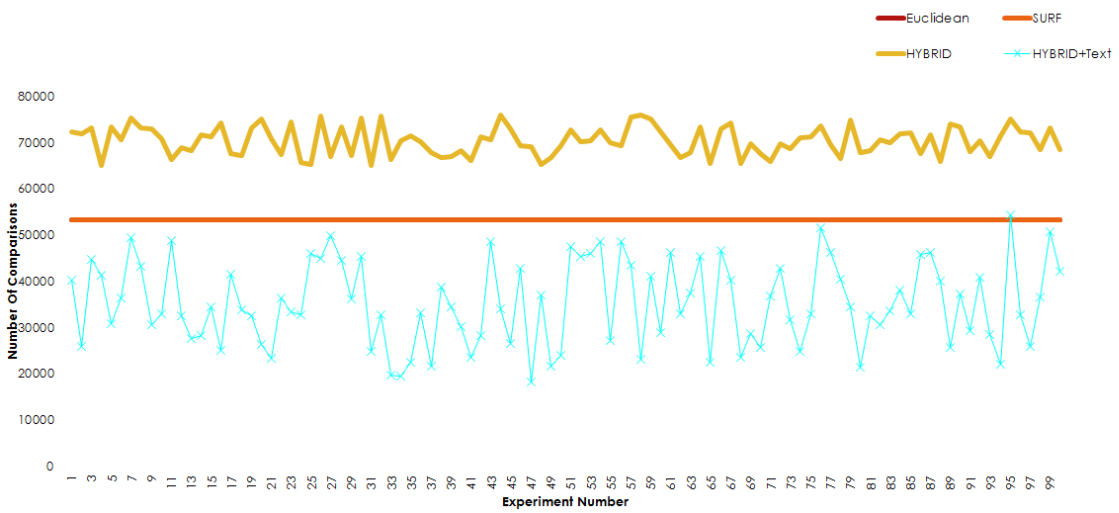

Figure 5.7 Methodology Performance

5.6 Small Scale Searching Evaluation: Accuracy (Precision)

A small set is also used to perform more diverse operations on CSISE and other image searching. Categorization of test environment helped us in generating different types of testing environments.

11 experiments have been done, where the performance of query-by-text using names and keywords is compared and the results are recorded. Performance of CSISE has been head-to-head with most of the search engines. The terms used for searching involved logo type names.

Table 3: Query-By-Text (Name) Performance

Search Engine	#Image Retrieved	#Image Relevant	Runtime (msec)	Precision
Bing	729	401	406	0.55
Flicker	1069	758	315	0.709
Yahoo	507	283	330	0.558
Google	1196	861	295	0.72
CSISE	389	225	4127	0.58

In the Query-By-Text performance of CSISE has been comparative with most of the search engines in terms of accuracy and at the same time poor compared to runtime performance. The terms used for searching general keywords were the ones which we saw on image keywords collection.

Table 4: Query-By-Text (Keywords) Performance

Logo	#Images Retrieved	#Images Relevant	Logo Type #	Runtime (msec)
Bing	715	367	NA	370
Flicker	784	517	NA	330
Yahoo	549	328	NA	310
Google	1238	866	NA	316
CSISE	376	214	76	4265

We compared CSISE with four, competitive solutions and recorded its size and precision. The performance of different comparative systems was similar to the performance of CSISE. Most of the systems were adhered to specific sets of image database, at the same time we have generated the image set by using our own methods.

Table 5: CSISE Comparison with Competitive Solutions

	Li et al. [25]	Michal et al. [26]	Lin et al. [27]	VLDBM [28]	[CSISE]
Focus	descriptors	local features	fast features	video processing	Semantic Image Processing
Image Size	3000	104	19200	5000	726,283
Precision	0.8	0.78	0.8	1	0.71

Similar experiments were performed by Query-By-Example using input images, performance of CSISE has been better than Google image search engine in terms of accuracy and at the same time poor compared to runtime performance.

Table 6: Query-By-Example Comparisons

Logo	#Image Retrieved	#Image Relevant	Runtime (msec)	Precision
Google Search by Image	1073	665	280	0.619
CSISE	337	213	4265	0.632

We performed 100 experiments with Euclidean, SURF, Hybrid and CSISE approaches. Variation in terms of Hadoop configurations is also done. In addition to that we added runtime, precision, recall and F-measure as comparisons parameters.

Table 7 : Overall F-Measure

	Searching Performance						Searching Accuracy		
	2PE		3PE		4PE		Precision	Recall	F-measure
	#Match	Runtime (msec)	#Match	Runtime (msec)	#Match	Runtime (msec)			
Euclidean	53200	8852	53200	5325	53200	4658	0.21	0.11	0.14
SURF	53200	11092	53200	6668	53200	5784	0.35	0.18	0.24
Hybrid	70210	12534	70210	9439	70210	6578	0.41	0.26	0.32
CSISE	35440	7695	35440	4468	35440	3415	0.53	0.41	0.46

CSISE has been able to perform better overall, but the runtime performance of CSISE has been poor compared to other comparative solutions. At the same time precision achieved is better with its performance reaching 0.46 and which is better than others precisions of 0.32, 0.24, and 0.14.

5.7 Performance In Terms of Logo Searching Capability

Two approaches have been studied along with user case study and found logo pattern affects overall searching paradigm. Table 8 shows the results of Hybrid, CSISE and User study (Human). We see that objects and letters in the logo improved precision on the system.

As per the users study we can say that users have voted to the logos which have more clear letters. For CSIS, we see logos which have letters and shapes combination, performed better than other logo types.

Table 8 : Performance : Logo Searching Capability

	HYBRID ALONE	CSISE	HUMAN
1st			
2nd			
3rd			
4th			
5th			

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

We performed a cloud-based novel approach based on classifying images, analyzing metadata and handling it on big data level. Text categorization and image processing were key ingredients of our CSISE approach. Usage of Vector support model for text classification was critical as text based approach plays big role while returning list of images.

Image processing is handled by using Euclidean distance model and SURF FLANN measurements. Level 2 filtering has reduced overall number of comparisons and hence overall time as well. CSISE has also implemented image repository which has its image set from various online image search engines.

Large-scale Hadoop IBM's SmartCloud experiments uses over 70,000 images of 76 types. The performance is compared to popular online image search engines as well as accurate with a higher rate (precision 71%) than existing approaches.

6.2 Future Work

CSISE is primarily focused on accurate image searching and fast processing. We can work on improving accuracy by having improvised image searching with low quality images such as blurred and distorted images. The Hadoop configuration can be customized by having its own logic of job distribution, which in turn can help in getting maximum throughput from CSISE.

Usage of Pig can slow down the performance, we need direct implementation of map-reduce technology which can speed up Hadoop jobs. Indexing techniques to represent image and metadata can also be useful for improvising CSISE. We are trying to implement feature-based image classification and connect it to existing modules.

Better evaluation plan with standard image set can also allow us to compare with most of the comparative solutions

REFERENCES

[1] Where is your Data URL:

http://www.smartercomputingblog.com/wp-content/uploads/2012/06/where-is-your-data.png

[2] Eaton, C., Derous, D.,Deutsch, T.,Lapis, G. and Zikopoulos, P. *Understanding Big Data* , IBM (2011)

[3] Picasa Image Sharing Site. http://www.picasa.com

[4] Google+. http://www.plus.google.com

[5] Jegou, H., Douze, M. and Schmid, C. Improving bag-of-features for large scale image search. *International Journal of Computer Vision* (2010): 1-6

[6] Deng, J.,Dong W., Socher, R., Li, L. and Li,K. Imagenet: A large-scale hierarchical image database. *Computer Vision IEEE* (2009): 1-8

[7] Sun, A., Grishman, R. and Sekine, S. Semi-supervised Relation Extraction with Large-scale Word Clustering. *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1* (2011): 1-9

[8] Zhang, Y., Jia, J. and Chen, T. Image retrieval with geometry-preserving visual phrases. *Computer Vision and Pattern Recognition* (CVPR), IEEE Conference on (2011): 6-8

[9] Zhang, W., Yoshida, T. and Tang, X. A comparative study of TF* IDF, LSI and multi-words for text classification. *Expert Systems with Applications* (2011): 4-6

[10] Jegou, H., Douze, M. and Schmid, C. Recent advances in large scale image search. *Emerging Trends in Visual Computing* (2009): 6-19

[11] Gao,S., Tsang, I.W., and Chia, L.T. Local features are not lonely–Laplacian sparse coding for image classification. *Computer Vision and Pattern Recognition* (CVPR), IEEE Conference on (2010): 1-2

[12] Kyprianidis, J.E. and Kang, H. Image and Video abstraction by coherence-enhancing filtering. *Computer Graphics Forum* (2011): 7-10

[13] H Jégou, M Douze, C Schmid. Aggregating local descriptors into a compact image representation. *Computer Vision and Pattern Recognition* (CVPR), IEEE Conference on (2010): 1-3

[14] Wu, Z., Ke, Q., Isard, M. and Sun, J. Bundling features for large scale partial-duplicate web image search. *ACM Transactions on Multimedia Computing, Communications, and Applications* (TOMCCAP) (2009): 2-3

[15] Chandrasekhar, V., Takacs, G. and Chen, D. CHoG: Compressed histogram of gradients a low bit-rate feature descriptor. *Computer Vision and Pattern Recognition* (CVPR), IEEE Conference on (2009): 4-6

[16] Jegou, H., Douze, M. and Schmid, C. Exploiting descriptor distances for precise image search. *HAL Forum* (2011): 4-6

[17] Felzenszwalb, P.F. and Huttenlocher, D.P. Distance transforms of sampled functions. *Theory of Computing* (2012): 1-15

[18] Zhou, W., Lu, Y., Li, H., Song, Y. and Tian, Q. Spatial coding for large scale partial-duplicate web image search. *International Journal of Computer Vision* (2010): 1-2

[19] Kumar, A. and Batra, A. Image Retrieval using SURF Features. *DSPACE* (2011): 48-54

[20] Gokturk, S.B., Anguelov, D., Vanhoucke, V. and Lee, K.C. System and method for enabling the use of captured images through recognition. *US Patent No US20060251339* A1 (2009): 6-16

[21] Choi, H., Varian, H. Predicting the present with google trends. Economic Record (2012): 1-6

[22] Nowak, S. and Ruger, S. How reliable are annotations via crowdsourcing: a study about inter-annotator agreement for multi-label image annotation. *Proceedings of the international conference on Multimedia Information Retrieval* (2010): 8-10

[23] Alham, NK., Li, M. and Hammoud, S. Evaluating machine learning techniques for automatic image. *Fuzzy Systems* (2009): 1-3

[24] Mikulik, A., Chum and O., Matas, J. Image retrieval for online browsing in large image collections. *6th International Conference, SISAP* , (2013): 3-8

[25] Li, KW., Chen, SY., Su, S., Duh, DJ., Zhang, H. and Li,S. Logo detection with extendibility and discrimination. *Multimed Tools Application* (2010): 2-5

[26] Kottman, M. Planar Object detection using local feature descriptors. *Applied Informatics* (2011): 3-5

[27] Lee, G. and Lin, K. Development of a fast logo recognition system. *UTAR* (2010): 85-94

[28] Bai, H., Hu, W., Wang, W., Feng Tong, X., Liu, C. and Zhang, Y. A Novel Sports Video Logo Detector Based on Motion Analysis. *International conference on Neural Information Processing (*2012): 8-10

VITA

Vijay Walunj was born on October 07, 1986, in Mumbai, India. He completed his schooling in Mumbai and graduated from high school in 2004. He then completed his Bachelor's degree in Computer Engineering from University of Mumbai in 2008. He worked with several companies while working on his start-up in India. He worked with Indian IT-giant Infosys and product based company Miles software in India.

In fall 2011, he came to US for his masters and took several courses towards Software Engineering. He also did work with various startups like U.S.Sourcelink, Innovating Solutions and CSA as an intern from August 2011 to May 2012. From August 2012 to till date, he is working with Ericsson on next generation monitoring tools for wireless networks. He also will be joining Ericsson as full time employee after graduation.