CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Every day, we human beings see and notice a million different objects around us. While we are familiar with some objects, we might not be familiar with some other objects. However, one might not be equipped with an image capturing device at every point in time. Hence, we human beings tend to remember or capture an image of those objects in our minds, which can be reproduced or put on a paper in the form of a sketch. Often, however, a user may not have a digital photograph of an object for which he or she desires to perform a search. In this case, it is desirable that the user be able to perform the search using a rough sketch of the desired object. Although image-based search is a valuable tool that is closely tied to the computer vision problem of object recognition and classification, recently, the ability to recognize real world objects using rough hand-drawn sketches has been of particular interest.Sketch recognition is the automated recognition of object diagrams, by a computer. It focuses on matching the sketches of the objects with the image database present. Based on the result obtained we classify them and retrieve the appropriate images.In a system that performs image searches, it is desirable for both to find close image matches and to classify an input image. However, the main focus here is on the specific problem of matching sketched query/input images to a database of sketches which are obtained by converting the digital images to their respective sketches and also classifying those query images. Recognition algorithms usually are gesture-based, appearance-based, geometry-based, or a combination thereof. Research in sketch recognition lies at the crossroads of Artificial Intelligence and Human Computer Interaction. The idea behind the sketch recognition project is to bring in human-computer interaction. Human–computer interaction (HCI) researches the design and use of computer technology, focused on the interfaces between people (users) and computers. Researchers in the field of HCI both observe the ways in which humans interact with computers and design technologies that let humans interact with computers in novel ways. As a field of research, human–computer interaction is situated at the intersection of computer science, behavioral sciences, design, media studies, and several other fields of study. The work is meaningful because it demonstrates the potential of the computer to classify and compare hand drawn sketches, which are imperfect and sparsely relative to actual photos. The first challenge in building a sketch-based system is recognizing the meaningful patterns implied by a user’s pen stroke. This task is not trivial because pattern matching must be flexible enough to allow some tolerance in sketch recognition, but sufficiently constrained not to accept incorrect patterns. Moreover, free hand sketching is an inherently imprecise process and varies greatly from person to person.

A subfield of computer science and artificial intelligence (AI) that focuses on the design of systems that can learn from and make decisions and predictions based on data is Machine Learning. Machine learning enables computers to act and make data-driven decisions rather than being explicitly programmed to carry out a certain task. Machine learning works by finding a function, or a relationship, from input X to output Y. The high level and most commonly accepted definition is: machine learning is the ability for computers to learn and act without being explicitly programmed. The system has to be trained to learn the low level features of object sketches and the database images for performing the recognition, classification and searching tasks. Image recognition, in the context of machine vision, is the ability of software to identify objects, places, people, writing and actions in images. Computers can use machine vision technologies in combination with a camera and artificial intelligence software to achieve image recognition. Image recognition is used to perform a large number of machine-based visual tasks, such as labeling the content of images with meta-tags, performing image content search and also forms the basis for sketch recognition systems.

1.2 MOTIVATION

People use sketches to express and record their ideas. Sketching is one of the primary methods people use to communicate visual information. Therefore, a sketch is the basic form of human-human interaction. The idea behind the sketch recognition project is to bring in human-computer interaction. The main motive behind this project is to make the process of searching for an object easier and user friendly. A user may not always have a digital photograph of an object for which he or she desires to perform a search. Therefore one should be able to search, classify or retrieve information using a sketch.

1.3 PROBLEM STATEMENT

“To develop an approach, to find the similarities between the natural images and sketch images, classify and retrieve them with efficient features.”

The approach employed here focuses on the sketches of objects. Here, the user interface asks for a sketch input from the user on which the search has to be performed. Then, the low level features of the sketch such as the boundary lines, outlines etc are extracted. The images from the database are also processed in the same way and the extracted details are stored in a database. Once the database is made the classification algorithms are applied to produce appropriate results. Here, some image processing and classification algorithms are used to match the input sketch and the training example sketches present in the database. Once the match is found, the related information and images is retrieved.

1.4 OBJECTIVES

- The main objective of the Sketch Recognition project is to bring in Human-Computer interaction.
- One should be able to draw any object, upload it to the recognition system and recognize and classify the objects easily.
- To improve retrieval accuracy through sketch-to-sketch comparison.

1.5 SUMMARY

Currently, there are techniques/systems existing, where in, a text input is given, appropriate search is performed and the results are obtained. Some systems accept audio input which is recognized and understood by the computer and a search is performed. Recently, there have been various approaches where images are also recognized and used for searching. This project implements a new searching technique which performs the search using a sketch. Sketch recognition is the automated recognition of object diagrams, by a computer. This approach takes a hand drawn or sketch pad drawn sketches, extracts its low level features and matches with the low level features of the images obtained from the database. This approach or project will prove to be very helpful in searching and retrieving information about objects whose images aren’t available with the user but one knows how the object looks or appears. Further, the recognition and classification of facial sketches can prove to be helpful in forensics and various other departments.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

Extraction of discriminative features from input images is one of the most challenging tasks in object sketch recognition systems. Much effort has aimed at determining optimal feature sets for a specific task, based on the attributes of objects to be recognized and classifiers to be used. Many of these features produced very promising results. However, due to the ambiguity and lack of general task-independent rules for optimal feature selection, the process of data classification has been recently dominated by various approaches using neural networks. The important advantage of these neural network approaches is that during the training process the network self-determines the optimal set of features from the data. The disadvantage is that large training data sets may be required and thus the training process could be very lengthy. Neural networks have been shown to provide excellent performance in multiple image classification benchmarks, ranging from simple feature datasets such as MNIST to complicated challenges such as ImageNet. However, more recent research in computer vision has demonstrated the dominating power of a neural network methodology known as deep learning. The design and training of deep learning structures today is also a big challenge. This is particularly a challenge when the performance of the design is very sensitive to the implementation details, which is often the case. Fortunately, using an implementation of already trained structures is quite straightforward and if the power of the network depth can be applied to other classification scenarios, then this offers great advantages.

2.2 LITERATURE SURVEY

Sketch recognition is the automated recognition of object diagrams, by a computer. It focuses on matching the sketches of the objects with the image database present. Based on the result obtained we classify them and retrieve the appropriate images.

The main objective of [2] is to extract a concise and simple sketch from a single image. The feature extraction from a sketch is mainly based on the histogram of orientations approach and sketches are typically represented by global or local descriptors. [Histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection.]

Since sketch generation from natural images is an essential part of sketch based image retrieval (SBIR) a saliency detection method is employed, which is based on the content of the improved absorbing Markov Chain.

The results show that this method can produce a clean pseudo-sketch from a single image. In addition, this technique obtains better map retrieval results and shows high efficiency for SBIR. Figure 2.2.1 is an overview of the complete processing.

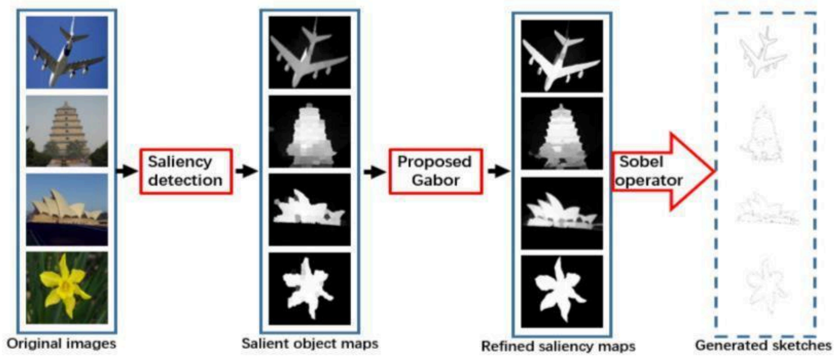


Figure 2.2.1

The sketches are stored in a database and compared with the input sketch obtained from the user interface. The system proposed in [5] focuses on the specific problems of matching sketched query images to a database of sketches and classifying those query images. A user interface is provided for creating sketches and the related images are searched accordingly. The dataset consisted of a group of 20,000 human sketched images by Eitz et al.2012 collected from Amazon Mechanical Turk. The dataset included 250 categories of objects ranging from simple(e.g. apple, sun) to complex (e.g. clock etc).Each category has 80 images, and the images contain only the sketch lines of the object (i.e., no other scene context is contained within the sketches).The implementation consists of matching HOG features of a query image to a dictionary of code words. This dictionary is generated by clustering the histograms of gradients computed from the dataset. Using the trained dictionary, each sketch can be described by a bag of words (BoW) histogram. The histogram is calculated using aGaussian kernel as in [Eitz et al. 2012a]. Each feature descriptor “f” from the input sketch is compared with each codeword “d” in the dictionary using the equation -

$$S_{f,d} = e^{-|f-d|2/2\delta 2} \quad (1)$$

where δ is the length scale of the Gaussian kernel (we use δ = 0.1). The histogram of feature f is then givenby the n dimensional vector

$$F = (S_{f,d1}, S_{f,d2}, ..., S_{f,dn}) \ (2)$$

in which d1, d2, ..., dn are n visual words in the dictionary. The Gaussian kernel allows a smoothed distribution of a feature in the feature space spanned by the n visual words.

The system was tested by sketching a variety of different objects. A subset of the sketches along with the top ten closest matches from the dataset and the top object class is given in Figure 2.2.2.

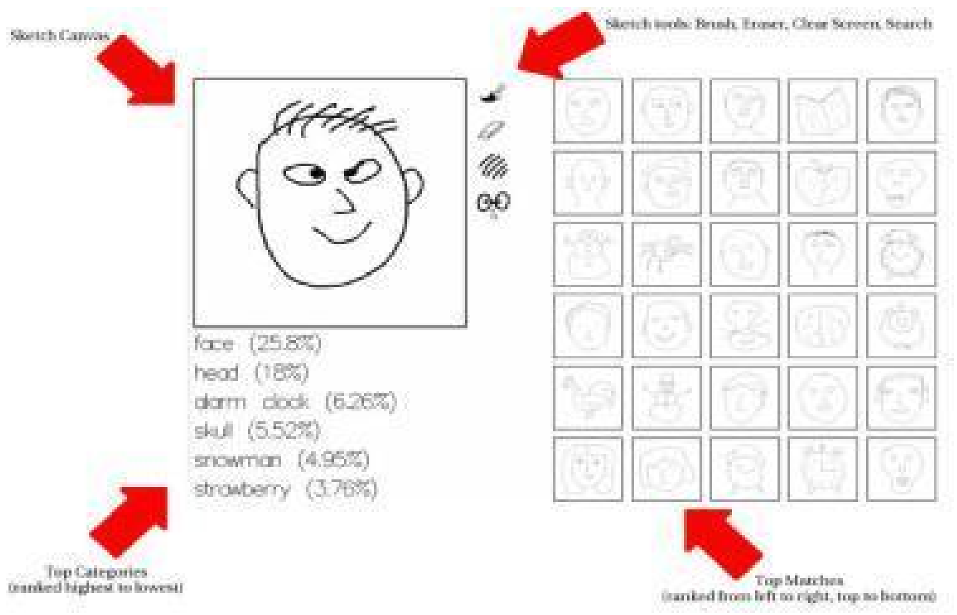

Sketch Canvas
Sketch tools: Brush, Eraser, Clear Screen, Search
face (25.8%)
head (18%)
alarm clock (6.26%)
skull (5.52%)
snowman (4.95%)
strawberry (3.76%)
Top Categories (ranked highest to lowest)
Top Matches (ranked from left to right, top to bottom)

Figure 2.2.2

According to [3], Deep neural networks (DNNs) are trained on large data sets that have been able to capture high-quality features describing image data. Numerous studies have proposed various ways to transfer DNN structures trained on large data sets to perform classification tasks represented by relatively small data sets. Due to the limitations of these proposals, it is not well known how to effectively adapt the pre-trained model into the new task. Typically, the transfer process uses a combination of fine-tuning and training of adaptation layers; however, both tasks are susceptible to problems with data shortage and high computational complexity. To overcome this limitation, an improvement to the well-known AlexNet feature extraction technique is explained in [3]. The proposed approach applies a recursive neural network structure on features extracted by a deep convolutional neural network pre-trained on a large data set. Object recognition experiments conducted on the Washington RGBD image data set have shown that the proposed method has the advantages of structural simplicity combined with the ability to provide higher recognition accuracy at a low computational cost compared with other relevant methods. The new approach requires no training at the feature extraction phase, and can be performed very efficiently as the output features are compact and highly discriminative, and can be used with a simple classifier in object recognition settings.

In this work, several low-level layers of the AlexNet trained on the ImageNet dataset were selected, and each of these layers were examined as a black-box feature extractor. The full structure of the AlexNet is shown in the top half of Fig. The network was trained using fixed size

RGB images from ImageNet. The network structure consists of 8 layers, where the rest 5 layers (conv1, conv2, conv3, conv4, conv5) are convolutional and the remaining 3 layers (fc6, fc7, fc8) are fully connected. The last fully-connected layer (fc8) has the form of a Softmax classifier to categorize an input image into one of the classes used in training. The proposed new object recognition structure, with part of AlexNet embedded as the feature extractor, is illustrated in the bottom half of Fig.

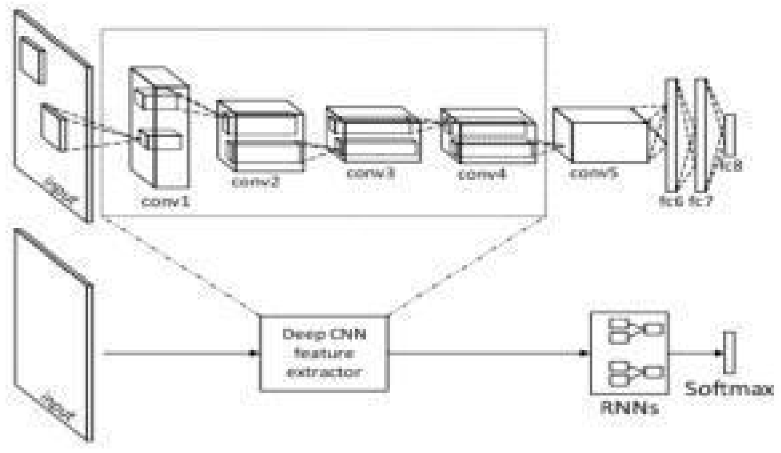


Figure 2.2.3

Further, the query image must be compared with the images stored in the database. This is explained in[1] which describes a parsing technique for recognition systems. Once integrated into a sketch editor, the parser incrementally and unobtrusively interprets the sketches as they are created. Thus, the proposed approach is especially tailored to interactive recognition environments, because the system is aware at any point of possible future interpretations of an incomplete sketch. Moreover, the use of well known visual language parsing methodologies allows us to benefit from many of the results reached in this field, such as flexible semantic interpretation and code generation. Starting from a survey on existing recognizers, Mankoff et.al. have presented a set of ambiguity resolution strategies, which are concerned with how ambiguity should be presented to the user and how the user should indicate his or her intention to the software. This work highlights a number of critical issues that demand consideration for a better interaction strategy between the end user and the software.

Figure 2.2.3 shows the architecture of the proposed framework. As the strokes are sketched into the editor, a pattern recognizer tries to match the stroke with the elementary shapes (registered in a pattern database).The recognizer produces the result of the pattern matching as a list of classifications and probabilities ordered by probability.

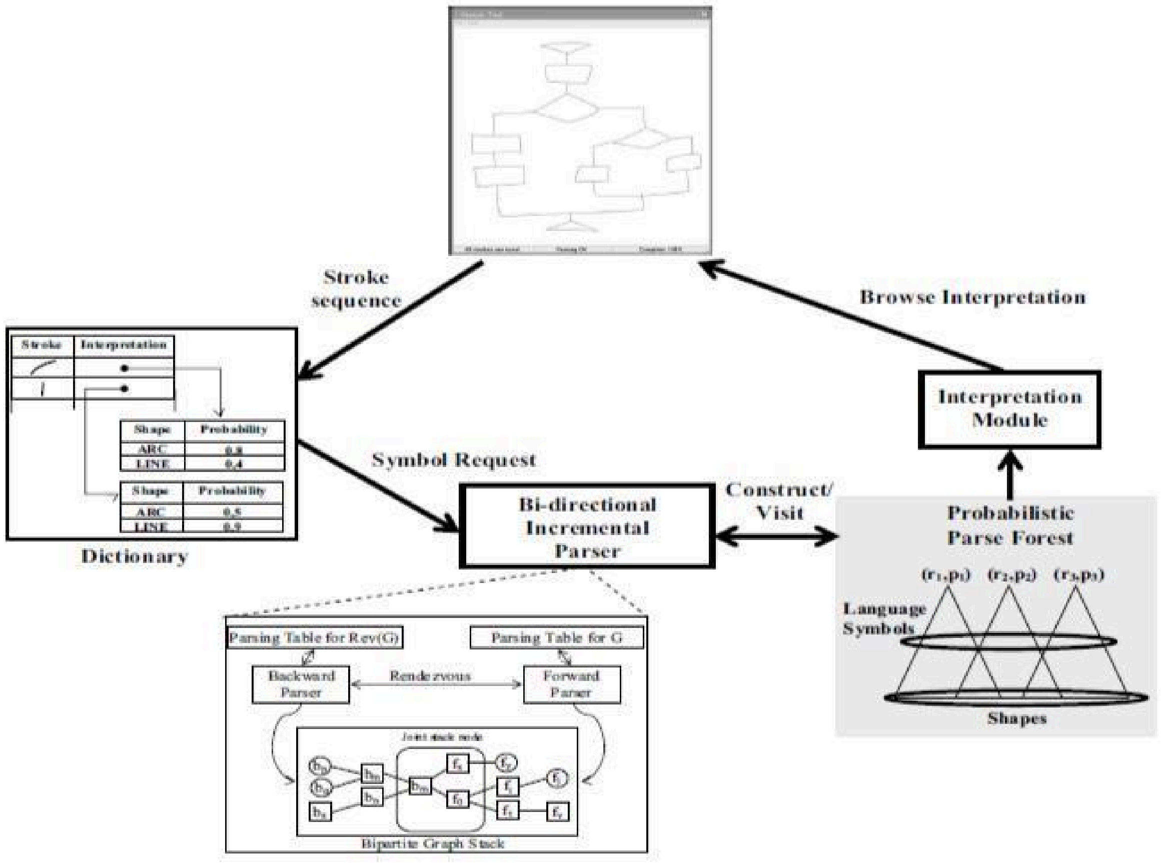


Figure 2.2.4

In this work illustrated in [4], a Sketch2Tag system is introduced for hand-drawn sketch recognition. Sketch2Tag is a general sketch recognition system, towards recognizing any semantically meaningful object that a child can recognize. This system enables a user to draw a sketch on the query panel, and then provides real-time recognition results. To increase the recognition coverage, a web-scale clipart image collection leverages the knowledge base of the recognition system. Better understanding a user’s drawing will be of great value to a variety of applications, such as, improving the sketch-based image search by combining the recognition results as textual queries. It allows the user to draw a sketch in the query panel in the main page, as shown in Fig. 2.2.4, and click the ‘search’ button to start the recognition. Then, the result page will come out, in which the recommended tags with corresponding probabilities to be the object name of the sketch, will be listed in the left part, as shown in Fig. 2.2.5. Meanwhile, the sketch-based image search results are also shown, which are retrieved by combining the sketch query and the top one recognition result. We also zoom in the recognition results for a better view in Fig. 2.2.5. We can see that, besides the correct recognition result 'horse', we can also provide some tags of animals with similar shapes to the drawing, such as 'dog' and 'goat' with smaller probabilities.

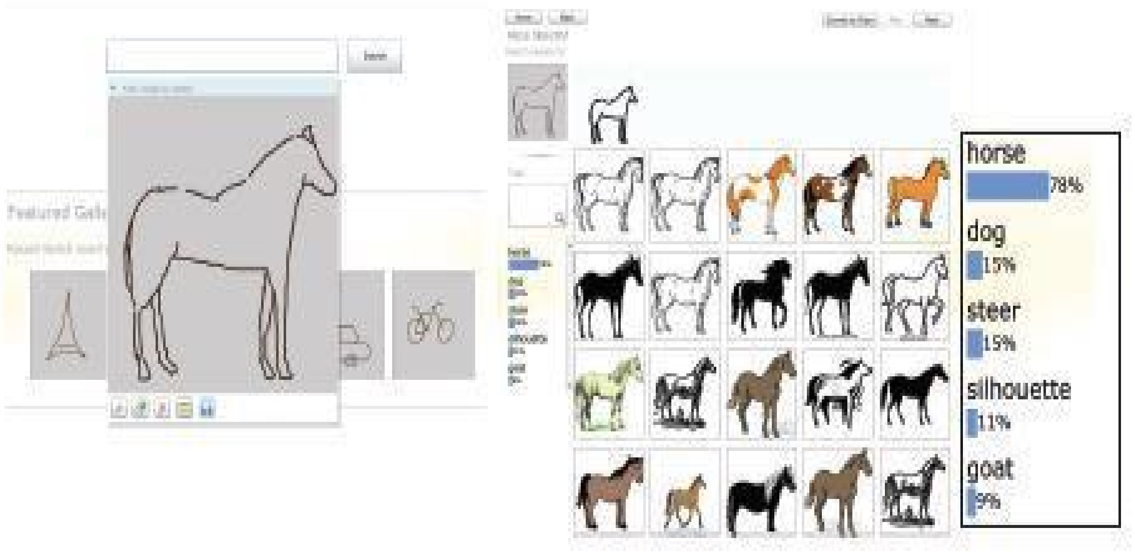


Figure 2.2.5

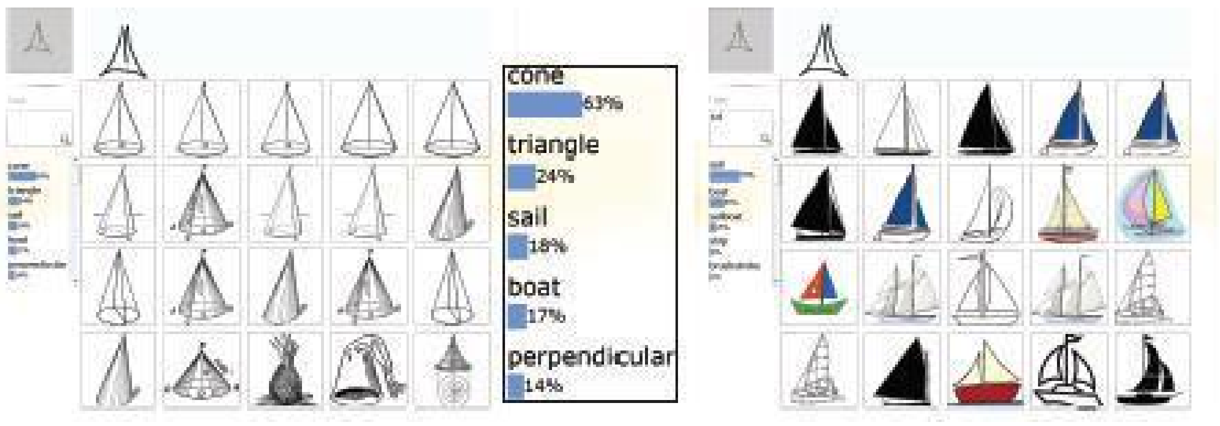


Figure 2.2.6

In [6] they introduce a benchmark for evaluating the performance of large scale sketch-based image retrieval systems. The necessary data is acquired in a controlled user study where subjects rate how well given sketch/image pairs match. FOR most image databases, browsing as a means of retrieval is impractical, and query based searching is required. Queries are often expressed as keywords (or by other means than the images themselves), requiring the images to be tagged. Here, it is said that outline sketches are typically easier and faster to generate than a complete color description of the scene. Several approaches for SBIR have been suggested. However, to achieve interactive query response, it is impossible to compare the sketch to all images in the database directly. Instead, descriptors are extracted in a pre-process and stored in a data structure for fast access. Very commonly, the descriptors are interpreted as points in a high-dimensional space and finding close matches means searching for nearest neighbors in this space.

The choice of query sketches essentially defines the difficulty of the resulting benchmark. If the sketches would contain too much abstraction, current systems would perform very badly and an evaluation would have to deal with more noise. Sketches should depict shape rather than symbols; they should depict non-fictional objects; if any perspective was used it should be reasonably realistic. They generated preliminary sketch/image pairs by using a subset of the best-performing SBIR systems from Eitz et al. the benchmark cannot only be used to evaluate existing systems but can also be used to optimize new systems by giving a means for measuring the influence of various retrieval system parameters on retrieval performance. Here they propose using a bag-of-features approach for SBIR employing small local descriptors. This allows basing the search on a standard inverted index data structure from text-retrieval. We discuss the four main components of our retrieval system:

a) definition and representation of local features,

b) sampling strategies defining the coordinates in image space of the features to be extracted,

c) codebook definition and

d) the actual search algorithm based on an inverted index.

While we rely on existing methods proposed in the literature for sampling, learning a codebook and defining the search algorithm, our work focuses on feature representation, as this is the part where SBIR systems differ strongly from classical example-based retrieval systems.

2.3 METHODOLOGIES

Machine learning is a subfield of artificial intelligence, which presents human intelligence by machines. Deep learning as a subset of machine learning has become the most popular research hotspot currently. It employs an artificial neural network (ANN) that is an information process machine modeled on the structure and action of a biological neural network in the brain. ANN is flexible and self-adaptive to solve complex problems that are difficult described by mathematical models, such as pattern recognition and classification, function approximation and control. Recent years the increasing interest in deep neural networks (DNNs), which employs many layers, has heightened the need for ANN in both industrial and academic areas. Deep neural networks learn experience from data to approximate any nonlinear relations between the input information and the final output. A well-trained deep neural network has the ability to capture abstract features over the entire data set.

The purpose of this project is to recognize sketches by applying deep neural networks. The work is meaningful because it demonstrates the potential of the computer to classify and compare hand drawn sketches, which are imperfect and sparsely relative to actual photos. The first challenge in building a sketch-based system is recognizing the meaningful patterns implied by a user's pen stroke. The problem of detecting a match in the low level features of a sketch diagram and the image features can be regarded as a classification problem. This task is not trivial because pattern matching must be flexible enough to allow some tolerance in sketch recognition, but sufficiently constrained not to accept incorrect patterns. Moreover, free hand sketching is an inherently imprecise process and varies greatly from person to person.

2.4 SUMMARY

While classification and information retrieval are classical problems in the field of Natural Language Processing (NLP) and Machine Learning (ML), most of the proposed algorithms work on large corpora. We extend these ideas to our application to demonstrate how deep learning methodologies can be used to create a system that classifies user-supplied sketch images appropriately, and retrieves information as well as related images based on closeness in low level features, with a reasonable degree of accuracy.

CHAPTER 3

3.1 INTRODUCTION

A Software Requirements Specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe interactions between Users and the System. Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function. Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. The software requirements specification document enlists enough and necessary requirements that are required for the project development. To derive the requirements, a clear and thorough understanding of the products to be developed or being developed is required. This is achieved and refined with detailed and continuous communications with the project team and customer till the completion of the software. Software applications he used could be off-the-shelf applications modified to suit the project or they may be bespoke applications already available within the company. The overall purpose of the system specification documentation is to lay down exactly how the system is made up. It is the process of determining user expectations for a new or modified product. These features, called requirements, must be quantifiable, relevant and detailed. Requirement analysis is critical to the success of a system of software projects. Conceptually, requirements analysis includes three types of activities:

● Eliciting Requirements for business process, documentation and stakeholder interviews. This is sometimes also called requirements gathering.

● Analyzing Requirements for determining whether the stated requirements are clear, complete, consistent and unambiguous and resolving any apparent conflict.

● Recording Requirements for requirements to be documented in various forms, usually including a summary list and may include natural-language documents, use cases, or process specifications.

3.2 Hardware and Software Requirements

Hardware and software requirements together are known as system requirements. For a system to be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as system requirements and are often used as a guideline as opposed to an absolute rule. Most Software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software. System requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements. A second meaning of the term of System requirements is a generalization of this first definition, giving the requirements to be met in the design of a system or subsystem. Typically, an organization starts with a set of Business requirements and then derives the System requirements from there.

Types of system requirements are:

1. **HARDWARE REQUIREMENTS:**
    - System : Pentium IV/V 2.4 GHz.
    - Hard Disk : 500 GB.
    - Ram : 4 GB
    - Any desktop / Laptop system with above configuration or higher level
2. **SOFTWARE REQUIREMENTS:**
    - Operating system : Windows 7/8/MAC
    - Coding Language : Python
    - Web Technology : HTML, CSS/BOOTSTRAP
    - Database : My-SQL
3. **FUNCTIONAL REQUIREMENTS:**
    - This system has two Actors- Admin and End user.
    - Admin is a main user who can grant access permission and overall layout.
    - End users can use the website/input panel.
    - End users can upload images and upload to the system where all the classification and retrieval takes place.
    - End user can then view all the query results which contain all images similar to that of the query image along with the description.

    This system must use the most effective feature extraction method and compare that with the data-set sample and use optimal matching method to classify them accordingly. Therefore, provide the most effective image classification and retrieval platform.
4. **Non-Functional Requirements:**
    - **Usability**

      Simple is the key here. The system must be simple enough that people like to use it, but not so complex that people avoid using it. The user must be familiar with the user interfaces and should not have problems in migrating to a new system with a new environment. The menus, buttons and dialog boxes should be named in a manner that they provide clear understanding of the functionality. Several users are going to use the system simultaneously, so the usability of the system should not get affected with respect to individual users.

- **Reliability**
  The system should be trustworthy and reliable in providing the functionalities. The overall retrieval and classification procedure should be to near accuracy. The user must get the expected results and all other similar results.
- **Performance**
  The system is going to be used by many users one at a time or simultaneously. Since the system will be hosted on a single web server with a single database server in the background, performance becomes a major concern. The system should not succumb when many users use it simultaneously. It should allow fast accessibility to all of its users and also better retrieval results.
- **Scalability**
  The system should be scalable enough to add new functionalities at a later stage. There should be a common channel, which can accommodate the new functionalities.
- **Maintainability**
  The system monitoring and maintenance should be simple and objective in its approach. There should not be too many processes running on different machines such that it gets difficult to monitor whether the jobs are running without errors.
- **Portability**
  The system should be easily portable to another system. This is required when the web server, which is hosting the system gets stuck due to some problems, which requires the system to be taken to another system.
- **Reusability**
  The system should be divided into such modules that it could be used as a part of another system without requiring much work.
- **Flexibility**
  The system should be flexible enough to allow modifications at any point of time.

3.3 Summary

For any system to work accordingly to give the expected result, all the system requirements including hardware and software requirements must be fulfilled. The major functionality of the system or model will be dependent on the overall systems requirements preferred. Python the programming language being used here. It's preferred because of its easy to use and optimal performance functionality. Python has effective libraries that cut down huge code and program lines into fewer ones, making it the most preferred programming language among all while creating models related to machine language and artificial intelligence. Codes with less lines and more effectiveness if preferred worldwide.

CHAPTER 4

4.1 INTRODUCTION

Every day, we human beings see and notice a million different objects around us. While we are familiar with some objects, we might not be familiar with some other objects. However, one might not be equipped with an image capturing device at every point in time. Hence, we human beings tend to remember or capture an image of those objects in our minds, which can be reproduced or put on a paper in the form of a sketch. Often, however, a user may not have a digital photograph of an object for which he or she desires to perform a search. In this case, it is desirable that the user be able to perform the search using a rough sketch of the desired object.
Image-based search is a valuable tool that is attached to computer vision problems in object recognition and also classification. Recently, the ability to recognize real world objects using rough hand drawn sketches has been of particular interest.

Sketch recognition mainly focuses on matching the sketches of the objects with the image database present. Based on the result obtained we classify them and retrieve the appropriate images. In a system that performs image searches, it is desirable for both to find close image matches and to classify an input image.

But extraction of distinct features from input query is quite challenging in object sketch recognition systems. More effort is needed to determine optimal features for certain set, based on attributes of the object that needs to be recognized and classifiers to be used.
However, due to the ambiguity and limitations the procedure of object classification has been recently over-powered by various approaches on neural networks. The main advantage of deep neural network methodologies is that the training process self-determines the best features from the data and comparative data set. The disadvantage is that the training procedure could be very time consuming. Neural networks have provided promising performance in numerous image classification.

However, recent research has demonstrated the imperious power of a neural network method known as deep learning. Sketch recognition is the automated recognition of object diagrams, by a computer. It focuses on matching the sketches of the objects with the image database present. Based on the result obtained we classify them and retrieve the appropriate images.

The main objective is to extract a compressed and straight forward sketch from a query image. The feature extraction from a sketch is mainly established on the histogram of orientations accession and sketches are typically represented by global or local captions.

Since sketch creation from natural images is a crucial part of sketch-based image retrieval (SBIR) a saliency detection method is employed, which is based on the content of the enhanced Markov Chain.

The sketches are stored in a database and compared with the input sketch obtained from the user interface. The system proposed focuses on the specific problems of matching sketched query images to a database of sketches and classifying those query images.

A user interface is provided for creating sketches and the related images are searched accordingly. The dataset consisted of a group of 20,000 human sketched image by Eitz et al.2012 collected from Amazon Mechanical Turk. The dataset included 250 categories of objects ranging from simple (e.g. apple, sun) to complex (e.g. clock etc). Each category has 80 images, and the images contain only the sketch lines of the object (i.e., no other scene context is contained within the sketches).

CHAPTER 5

INTRODUCTION

5.1 Introduction

There are a wide range of methodologies that can be used in order to design a system. The most important aspects when it comes to selecting a methodology for a project are:

- The accuracy
- The complexity
- The implementability

These are the aspects that need to be taken into consideration while selecting an ideal way in which a project can be implemented.

The design of the system must be done in such a way that all the above-mentioned conditions are taken into account and are used in the best possible way.

This section discusses the various algorithms that are used, and the reason behind using them.

5.2 System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed. Put another way, the physical portion of system design can generally be broken down into three sub-tasks:

- User interface
- Data design
- Process design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the system design phase, documentation describing the three subtasks is produced and made available for use in the next phase.

There are multiple solutions to a problem. The objective is to find the best and the most feasible one. Some of the methods or algorithms used for the design of the system are: Feature extraction, KNN algorithm, chi-squared distance, Scale Invariant Feature Transform (SIFT).

5.2.1 ALGORITHMS/METHODS USED:

Feature extraction:

In machine learning, pattern recognition and in image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is a dimensionality reduction process, where an initial set of raw variables is reduced to more manageable groups (features) for processing, while still accurately and completely describing the original data set.

When the input data to an algorithm is too large to be processed and it is suspected to be redundant (e.g. the repetitiveness of images presented as pixels), then it can be transformed into a reduced set of features, also named a feature vector. Determining a subset of the initial features is called feature selection. The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.

❖ **Edge detection:**

Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. It has been widely applied in various computer vision systems. Canny has found that the requirements for the application of edge detection on diverse vision systems are relatively similar.

cv2.Canny(img, minVal, maxVal)

img: Path of the image,

minVal: intensity gradient value, below which represents non-edges,

maxVal: intensity gradient value, above which represents edges.

The process of ***Canny edge detection algorithm*** can be broken down into following steps:

➢ **Apply Gaussian filter to smooth the image in order to remove the noise:**

Since all edge detection results are easily affected by image noise, it is essential to filter out the noise to prevent false detection caused by noise. To smooth the image, a Gaussian filter is applied to convolve with the image. This step will slightly smooth the image to reduce the effects of obvious noise on the edge detector. The equation for a Gaussian filter kernel of size ***(2k+1)×(2k+1)*** is given by -

$$H_{ij} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2}\right); 1 \le i, j \le (2k+1)$$ ***.... Equation 5.1***

➢ **Finding the intensity gradients of the image:**

An edge in an image may point in a variety of directions, so the Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the blurred image. The edge detection operator (such as Roberts, Prewitt, or Sobel) returns a value for the first derivative in the horizontal direction (G_x) and the vertical direction (G_y). From this the edge gradient and direction can be determined:

$$\mathbf{G} = \sqrt{{\mathbf{G}_x}^2 + {\mathbf{G}_y}^2}$$

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x),$$ ***.... Equation 5.2***

➢ **Track edge by hysteresis:**

Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges, as shown in Fig. 5.1. To track the edge connection, blob analysis is applied by looking at a weak edge pixel and its 8-connected neighborhood pixels. As long as there is one strong edge pixel that is involved in the blob, that weak edge point can be identified as one that should be preserved.

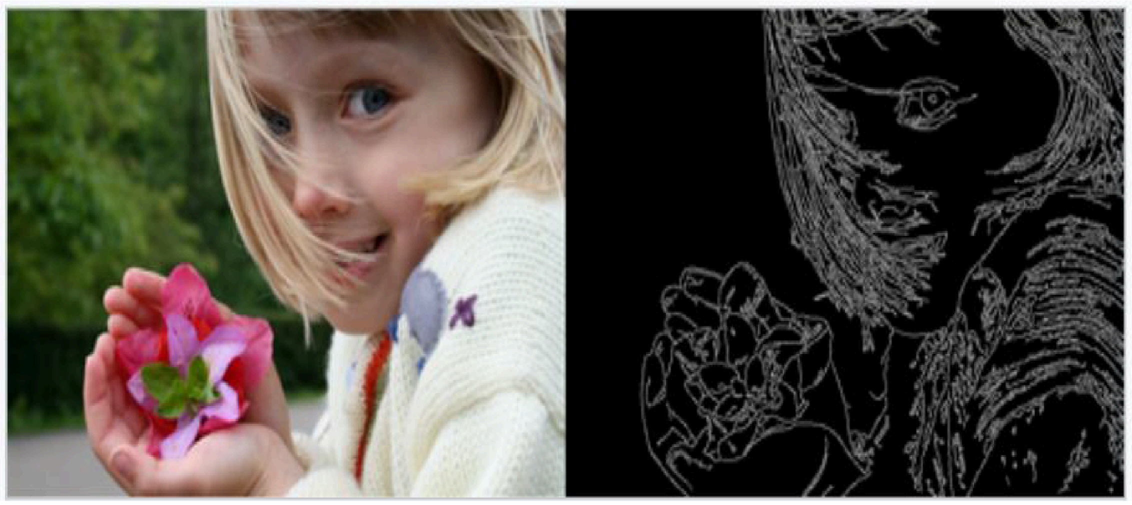


Figure 5.1 Canny edge detection applied to a photograph

❖ **Color histogram:** A color histogram of an image represents the distribution of the composition of colors in the image. It shows different types of colors and the number of pixels in each type of the colors appeared. The relation between a color histogram and a luminance histogram is that a color histogram can be also expressed as “Three Luminance Histograms”, each of which shows the brightness distribution of each individual Red/Green/Blue color channel.

cv2.CalcHist(img, hist, accumulate=0, mask=None)

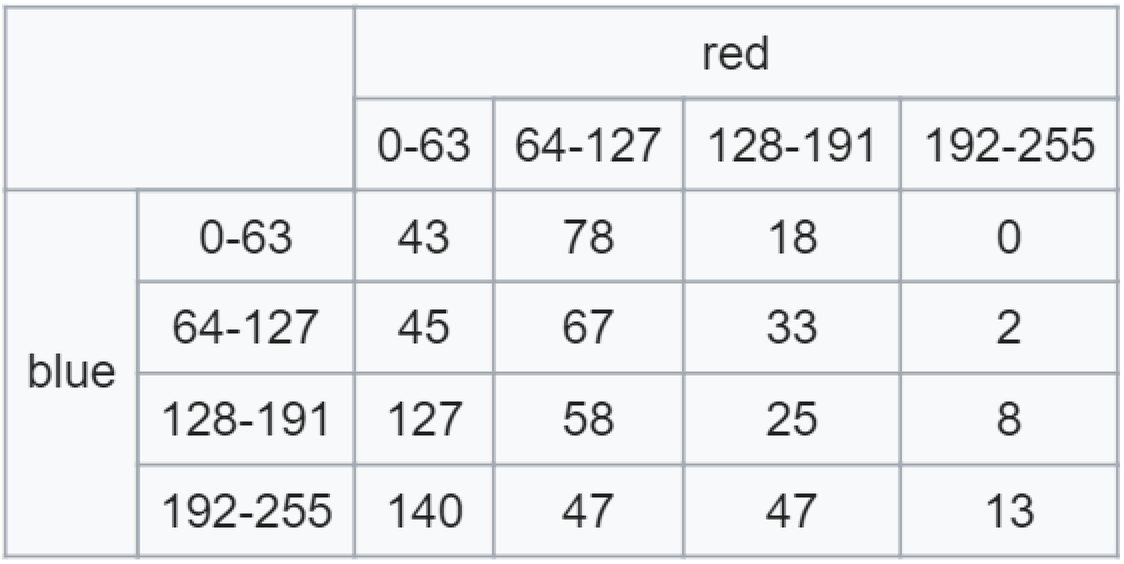
img: Source arrays. They all should have the same depth, and same size,

hist: Output histogram,

accumulate: If it is set, the histogram is not cleared in the beginning when it is allocated.

mask – Optional mask. If the matrix is not empty, it must be an 8-bit array of the same size as images[i]. The non-zero mask elements mark the array elements counted in the histogram

The color histogram can be built for any kind of color space, although the term is more often used for three-dimensional spaces like RGB or HSV. For monochromatic images, the term intensity histogram may be used instead. For multi-spectral images, where each pixel is represented by an arbitrary number of measurements (for example, beyond the three measurements in RGB), the color histogram is N-dimensional, with N being the number of measurements taken. If the set of possible color values is sufficiently small, each of those colors may be placed on a range by itself; then the histogram is merely the count of pixels that have each possible color. Most often, the space is divided into an appropriate number of ranges, often arranged as a regular grid, each containing many similar color values. The color histogram may also be represented and displayed as a smooth function defined over the color space that approximates the pixel counts. Like other kinds of histograms, the color histogram is a statistic that can be viewed as an approximation of an underlying continuous distribution of colors values. Color histograms are flexible constructs that can be built from images in various color spaces, whether RGB, rg chromaticity or any other color space of any dimension. A histogram of an image is produced first by discretization of the colors in the image into a number of bins, and counting the number of image pixels in each bin. For example, a Red–Blue chromaticity histogram can be formed by first normalizing color pixel values by dividing RGB values by R+G+B, then quantizing the normalized R and B coordinates into N bins each. A two-dimensional histogram of Red-Blue chromaticity divided into four bins (N=4) might yield a histogram that looks like this table:



		red			
		0-63	64-127	128-191	192-255
blue	0-63	43	78	18	0
	64-127	45	67	33	2
	128-191	127	58	25	8
	192-255	140	47	47	13

Table 5.1 Example of Red-Blue chromaticity histogram

❖ **Contours:**

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having the same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.

➢ For better accuracy, use binary images. So before finding contours, apply threshold or canny edge detection.

➢ ***findContours()*** function modifies the source image. So if you want source image even after finding contours, store it to some other variables.

➢ In OpenCV, finding contours is like finding white object from black background. So, object to be found should be white and the background should be black.

cv2.findContours(img,retrieval_mode,approx)

img: first one is source image,

retrieval_mode: second is contour retrieval mode,

approx: third is the contour approximation method. And it outputs the contours and hierarchy.

To draw the contours, ***cv2.drawContours()*** function is used. It can also be used to draw any shape provided you have its boundary points. Its first argument is source image, second argument is the contours which should be passed as a Python list, third argument is index of contours (useful when drawing individual contour. To draw all contours, pass -1) and remaining arguments are color, thickness etc.

KNN Algorithm:

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression:

• In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If ***k = 1***, then the object is simply assigned to the class of that single nearest neighbor.

• In K-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples.

A commonly used distance metric for continuous variables is Euclidean distance. For discrete variables, such as for text classification, another metric can be used, such as the overlap metric (or Hamming distance). In the context of gene expression microarray data, for example, k-NN has been employed with correlation coefficients, such as Pearson and Spearman, as a metric. Often, the classification accuracy of k-NN can be improved significantly if the distance metric is learned with specialized algorithms such as Large Margin Nearest Neighbor or Neighbourhood components analysis.

A drawback of the basic "majority voting" classification occurs when the class distribution is skewed. That is, examples of a more frequent class tend to dominate the prediction of the new example, because they tend to be common among the k nearest neighbors due to their large number. One way to overcome this problem is to weight the classification, taking into account the distance from the test point to each of its k nearest neighbors. The class (or value, in regression problems) of each of the k nearest points is multiplied by a weight proportional to the inverse of the distance from that point to the test point. Another way to overcome skew is by abstraction in data representation. For example, in a self-organizing map (SOM), each node is a representative (a center) of a cluster of similar points, regardless of their density in the original training data. K-NN can then be applied to the SOM. Fig. 5.2 is an example of 2 classes, represented graphically using K-NN.

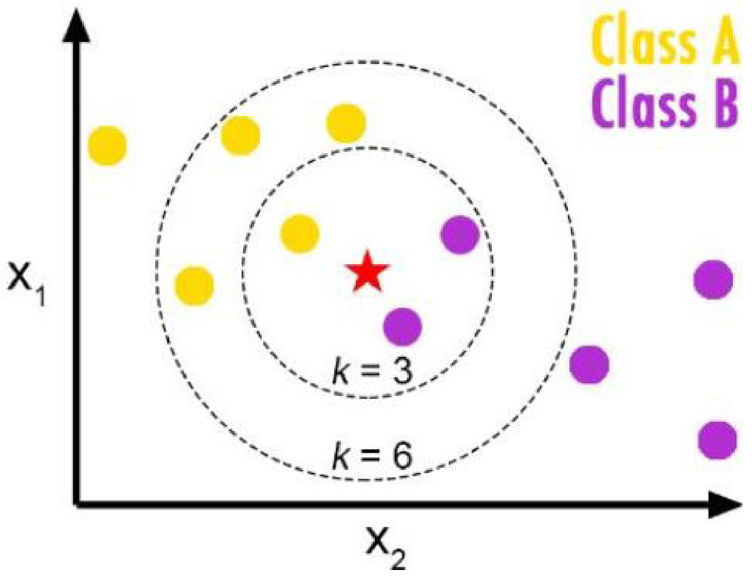


Fig 5.2: Graphical representation of KNN Algorithm

Scale Invariant Feature Transform (SIFT):

The scale-invariant feature transform (SIFT) is a feature detection algorithm in computer vision to detect and describe local features in images. It was patented in Canada by the University of British Columbia and published by David Lowe in 1999. Applications include object recognition, robotic mapping and navigation, image stitching, 3D modeling, gesture recognition, video tracking, individual identification of wildlife and match moving.

SIFT keypoints of objects are first extracted from a set of reference images and stored in a database. An object is recognized in a new image by individually comparing each feature from the new image to this database and finding candidate matching features based on Euclidean distance of their feature vectors. From the full set of matches, subsets of keypoints that agree on the object and its location, scale, and orientation in the new image are identified to filter out good matches. The determination of consistent clusters is performed rapidly by using an efficient hash table implementation of the generalised Hough transform. Each cluster of 3 or more features that agree on an object and its pose is then subject to further detailed model verification and subsequently outliers are discarded. Finally, the probability that a particular set of features indicates the presence of an object is computed, given the accuracy of fit and number of probable false matches. Object matches that pass all these tests can be identified as correct with high confidence.

• ***Scale-invariant feature detection:*** Lowe's method for image feature generation transforms an image into a large collection of feature vectors, each of which is invariant to image translation, scaling, and rotation, partially invariant to illumination changes and robust to local geometric distortion.

• ***Feature matching and indexing:*** Indexing consists of storing SIFT keys and identifying matching keys from the new image. Lowe used a modification of the k-d tree algorithm called the best-bin-first search method that can identify the nearest neighbors with high probability using only a limited amount of computation. Each of the SIFT keypoints specifies 2D location, scale, and orientation, and each matched keypoint in the database has a record of its parameters relative to the training image in which it was found. The similarity transform implied by these 4 parameters is only an approximation to the full 6 degree-of-freedom pose space for a 3D object and also does not account for any non-rigid deformations.

As shown in Fig 5.3, after scale space extrema are detected (their location being shown in the uppermost image) the SIFT algorithm discards low-contrast keypoints (remaining points are shown in the middle image) and then filters out those located on edges. Resulting set of keypoints is shown in the last image.

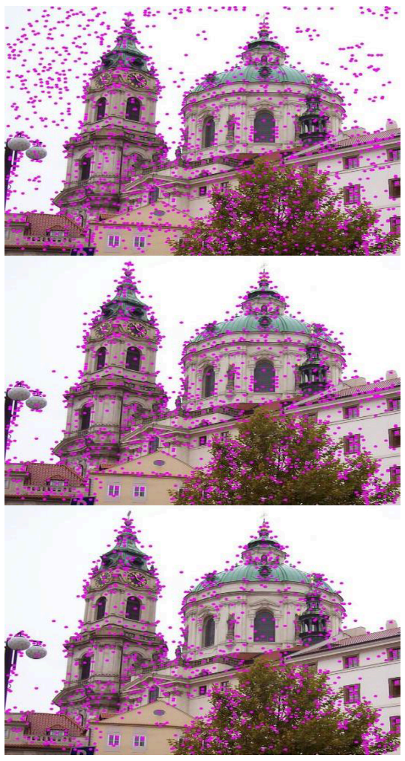


Fig 5.3 SIFT key points

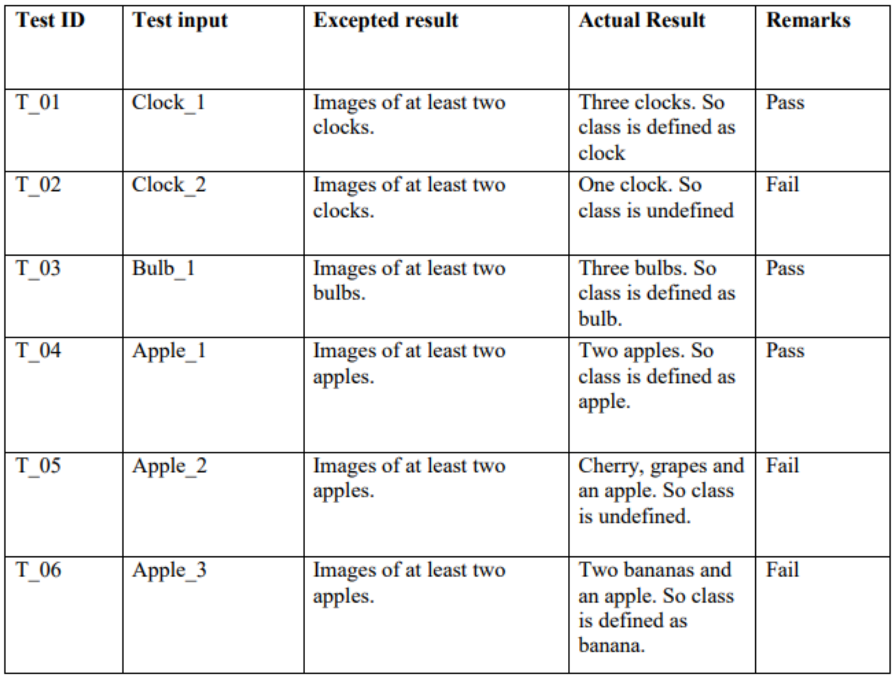
5.3 Summary

The features of the various images stored in the dataset can be extracted using the various feature extraction techniques mentioned here. Once the feature vector is obtained, the mean values are extracted and stored in the .csv file, along with the path where the image is stored in the system. In the same way, the mean feature values of the input image are also extracted. Once the .csv file is created, K-NN algorithm can be applied to detect the class to which the input image belongs to. If K=3, the top 3 matches are displayed and if K=5 top 5 matches are displayed and so on. The SIFT algorithm directly extracts features as keypoints and descriptors and stores it. It matches these keypoints with the input image to show appropriate output images.

CHAPTER 6

RESULT ANALYSIS AND CONCLUSION

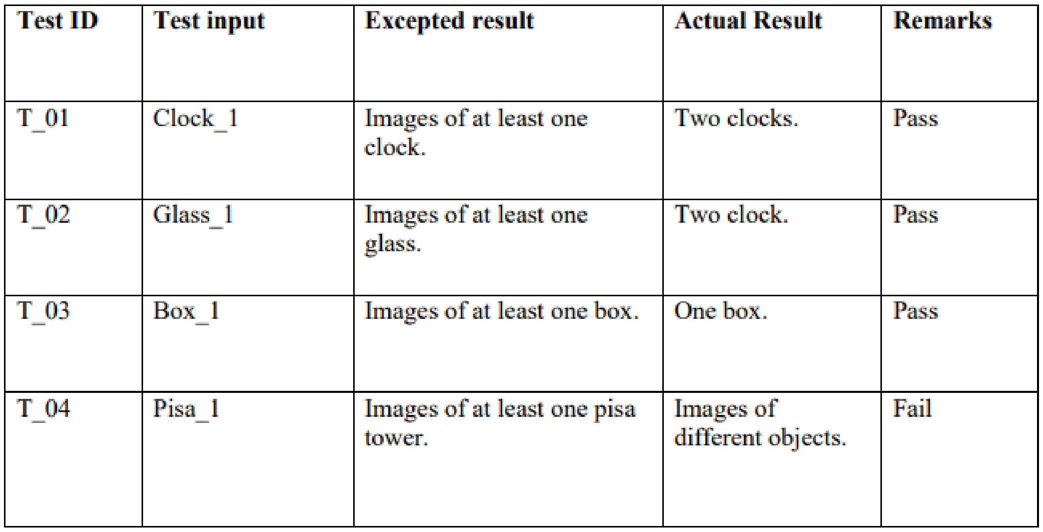
6.1 KNN



Test ID	Test input	Excepted result	Actual Result	Remarks
T_01	Clock_1	Images of at least two clocks.	Three clocks. So class is defined as clock	Pass
T_02	Clock_2	Images of at least two clocks.	One clock. So class is undefined	Fail
T_03	Bulb_1	Images of at least two bulbs.	Three bulbs. So class is defined as bulb.	Pass
T_04	Apple_1	Images of at least two apples.	Two apples. So class is defined as apple.	Pass
T_05	Apple_2	Images of at least two apples.	Cherry, grapes and an apple. So class is undefined.	Fail
T_06	Apple_3	Images of at least two apples.	Two bananas and an apple. So class is defined as banana.	Fail

Table 6.1 Predictions from KNN

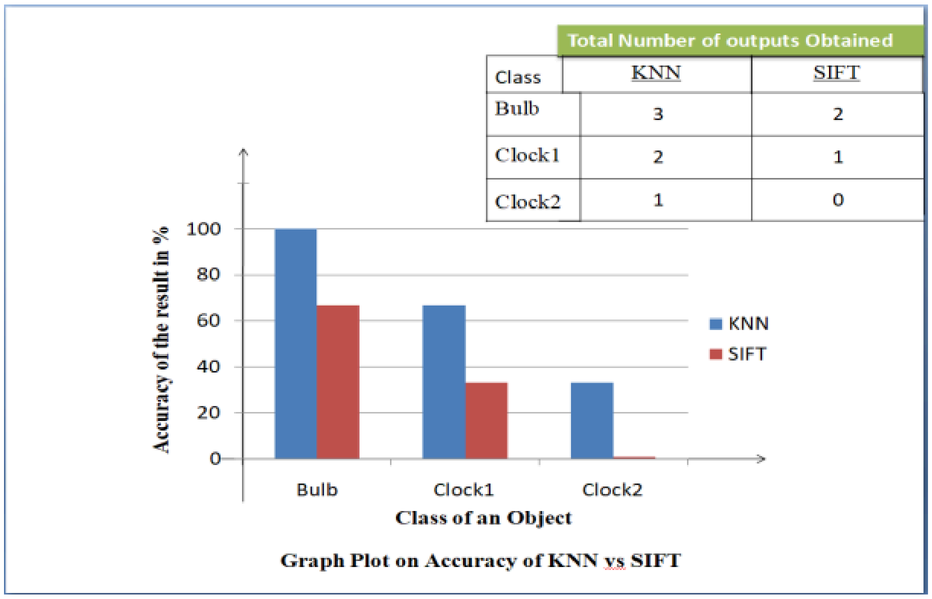
6.2 SIFT



Test ID	Test input	Excepted result	Actual Result	Remarks
T_01	Clock_1	Images of at least one clock.	Two clocks.	Pass
T_02	Glass_1	Images of at least one glass.	Two clock.	Pass
T_03	Box_1	Images of at least one box.	One box.	Pass
T_04	Pisa_1	Images of at least one pisa tower.	Images of different objects.	Fail

Table 6.2 Predictions from SIFT

6.3 PERFORMANCE EVALUATION AND COMPARISON



Total Number of outputs Obtained

Class	KNN	SIFT
Bulb	3	2
Clock1	2	1
Clock2	1	0

Figure 6.1 Visualize Predictions from both KNN and SIFT

The performance evaluation metrics show that the KNN algorithm has an accuracy rate of 77% for the considered dataset while SIFT has an accuracy of 51%. Hence KNN performs better, with higher accuracy and speed compared to the SIFT algorithm. Moreover the KNN algorithm predicts the class to which a particular object might belong, but SIFT just provides the names of each individual result making the object recognition task difficult for the user.

6.4 CONCLUSION

The project, Sketch Recognition for Image Classification and Retrieval, is developed with the objective of recognizing hand drawn sketches and retrieving related images of that hand drawn object or sketch. The main motive behind the project is to improve human computer interaction by developing an approach to help computers identify or classify a simple sketch drawn by a human being. A good user interface is provided to the user to upload a sketch input image and receive related images. At the back end of the project various python codes are run in order to extract the low-level features of the query image as well as the training images which are maintained in the system. The two main algorithms employed for classification of images are KNN and SIFT. The performance evaluation conducted after the completion of the project, for the respective algorithms, show that the KNN algorithm works best for classification. It has an accuracy rate of 100% for most of the images among all the dataset images. This helps in concluding that the project employees' two methods or algorithms to serve the purpose of image classification, although it works best when the KNN algorithm is chosen.

REFERENCES

[1] A Parsing Technique for Sketch Recognition Systems by Gennaro Costagliola, Vincenzo Deufemia, Giuseppe Polese and Michele Risi Dipartimento di Matematica e Informatica Università di Salerno 84084 Fisciano(SA), Italy

[2] Photo-to-Sketch Transformation in a Complex Background by XIANLIN ZHANG1, XUEMING LI2, SHUXIN OUYANG1, AND YANG LIU31 Institute of Information and Communication, Beijing University of Posts and Telecommunications, Beijing 100089, China

[3] Object Recognition Using Deep Convolutional Features Transformed by a Recursive Network Structure by HIEU MINH BUI1,2, MARGARET LECH1, EVA CHENG1, (Member, IEEE), KATRINA NEVILLE1, (Member, IEEE), AND IAN S. BURNETT3, (Senior Member, IEEE)

[4] Sketch2Tag: Automatic Hand-Drawn Sketch Recognition by Zhenbang Sun1∗, Changhu Wang2, Liqing Zhang1, Lei Zhang2

[5] Sketch-based Object Recognition by Bo Zhu∗ Stanford University Ed Quigley† Stanford University

[6] Sketch-Based Image Retrieval: Benchmark and Bag-of-Features Descriptors by Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur and Marc Alexa