SKETCH BASED IMAGE RETRIEVAL USING DEEP LEARNING

A PROJECT REPORT

Submitted By

DEEPIKA S 312217104033

RAJESH R 312217104121

SAI SEENA P 312217104322

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

SSN COLLEGE OF ENGINEERING

KALAVAKKAM 603110

ANNA UNIVERSITY :: CHENNAI - 600025

April 2021

ANNA UNIVERSITY : CHENNAI 600025

BONAFIDE CERTIFICATE

Certified that this project report titled **"SKETCH BASED IMAGE RETRIEVAL USING DEEP LEARNING"** is the *bonafide* work of "**DEEPIKA S (312217104033)**, **RAJESH R (312217104121)**, and **SAI SEENA P (312217104322)**" who carried out the project work under my supervision.

Dr. Chitra Babu
Head of the Department
Professor,
Department of CSE,
SSN College of Engineering,
Kalavakkam - 603 110

Dr. K. Madheswari
Supervisor
Associate Professor,
Department of CSE,
SSN College of Engineering,
Kalavakkam - 603 110

Place:
Date:

Submitted for the examination held on. . . . . .

Internal Examiner **External Examiner**

ACKNOWLEDGEMENTS

We thank GOD, the almighty, for giving us strength and knowledge to do this project.

We would like to thank and express a deep sense of gratitude to our guide **Dr. K. MADHESWARI**, Associate Professor, Department of Computer Science and Engineering, for her valuable advice and suggestions as well as her continued guidance, patience and support that helped us to shape and refine our work.

Our sincere thanks to **Dr. CHITRA BABU**, Professor and Head of the Department of Computer Science and Engineering, for her words of advice and encouragement and we would like to thank our project Coordinator **Dr. B. BHARATHI**, Associate Professor, Department of Computer Science and Engineering for her valuable suggestions throughout this project.

We express our deep respect to the founder **Dr. SHIV NADAR**, Chairman, SSN Institutions. We also express our appreciation to our Principal, **Dr. V. E. ANNAMALAI** for all the help he has rendered during this course of study.

We would like to extend our sincere thanks to all the teaching and non-teaching staff of our department who have contributed directly and indirectly during the course of our project work. Finally, we would like to thank our parents, siblings and friends for their patience, cooperation and moral support throughout our life.

DEEPIKA S **RAJESH R** **SAI SEENA P**

ABSTRACT

Sketch based image retreival (SBIR) is a sub-domain of Content Based Image Retrieval(CBIR) where the user provides a drawing as an input to obtain i.e retrieve images relevant to the drawing given. The main challenge in SBIR is the subjectivity of the drawings drawn by the user as it entirely relies on the user's ability to express information in hand-drawn form. Since many of the SBIR models created aim at using singular input sketch and retriving photos based on the given single sketch input, our project aims to enable detection and extraction of multiple sketches given together as a single input sketch image. The features are extracted from individual sketches obtained using deep learning architectures such as VGG16 , and classified to its type based on machine learning using Support Vector Machines. Based on the type obtained, photos are retrieved from the database using an opencv library, CVLib , which finds the objects present in a photo image. From the number of components obtained in each photo, a ranking function is performed to rank the retrieved photos, which are then displayed to the user starting from the highest order of ranking upto the least.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

To obtain information about something, it is required to be able to express it unambiguously and with relevance. Verbal descriptions, pictographs and sketches are one of the few ways to describe one's need for relevant information. With the advent of technology, many aim to gain knowledge through information remotely, by using search engines. This explicit description about the object is also used in search engines to get relevant results in the form of images, websites, videos etc. Information retrieval is a domain of interest to many problem solvers where the main difficulty lies in the ability to bridge the gap between information available and the query given by the user. Content based image retrieval is a sub-domain under information retrieval where, to obtain images, the user describes the images to retrieve in text form or as a drawing such. In query by text, the images are mapped to respective tags that are of relevance to the image. In Sketch based image retrieval [1], where a sketch or a drawing is the input given to get images. For example, a drawing of an apple retrieves us images apples. These drawings can be professional, free-hand sketches, symbols and pictures defined by edges. A hand drawn sketch is abstract with roughly drawn shapes and may not contain edges and features as sharply defined in the photos. Thus relevance in the images retrieved depends on how efficiently a model can relate the abstract features of sketches to the photos. The ability to retrieve images heavily depends on the user's subjectivity in expressing them, thus increasing the need to obtain dataset

containing images with both heavy and light levels of abstractions in terms of features.

1.2 Applications

- SBIR is a useful learning tool for autistic kids due to them requiring repetitive teaching to be able to understand and classify objects and shapes.
- Useful in synthesis of faces of humans based on features obtained from the sketches.
- Enhancement of sketches from rough or partially drawn images.
- Creation of 3D models such as furniture from 2D sketch images.
- The retrieval system using sketches can be effective and essential in our day to day life such as Medical diagnosis, digital library, search engines, crime prevention, photo sharing sites, geographical information, and sensing remote systems.
- Creating animations on a sketch based on motion points obtained from objects of motion in a video.

1.3 Problem Statement

This project focuses on extraction of features from the sketch image dataset using pre-trained architectures such as VGG16, Inceptionv3 and training the machine learning classifier such as Support Vector Machines to classify newly given input sketch images. The class of the sketch is then obtained and photos containing the class of the sketch are displayed after a ranking function is performed on those images. The ranking function orders the photos based on the number of classes found in query sketch, starting with the photo containing all classes of the query sketch.

1.4 Introuduction to the Classifier System Created

A classfier or a classification system is one wherein it is trained on a set of images belonging to certain classes and when a test image is produced, the system is able to classify the class of the image correctly. This classification can be done in two ways.

- Machine learning
- Deep learning

1.4.0.1 Machine Learning

Machine learning [20] is done mainly when the user is aware of the features of the dataset, a classifier must learn from. This features can be in the form of Comma

Separated Values (CSV) files or Numpy arrays. Labelling the dataset also changes the way a classifier learns. Based on labelling, it is divided into *Supervised* and *Unsupervised* machine learning.

Supervised Learning

In supervised machine learning the dataset is labelled into the classes.

Eg, a folder of cat images and a folder of dog images.

Supervised learning can be implemented using Support Vector Machines, Naive bayes, Decision trees etc.

Unsupervised Learning

In unsupervised machine learning the dataset is not labelled into the classes. Thus the data can be classified only based on the similarity it obtains between images possibly belonging to the same class.

Unsupervised learning can be implemented using clustering, K-Means clustering, KNN etc.

1.4.0.2 Deep Learning

Deep learning [21] is a subset of supervised machine learning where Convolution Neural Networks (CNN) are used to learn features automatically from images etc. Here the user is not aware of the features that can be explicitly stated to learn from.

The difference in the dataset used for machine learning is that, in case of machine learning the final features from an image are extracted and saved separately which is then used as input for training using the models. In case of deep learning the input is the images themselves rather than the features that are extracted.

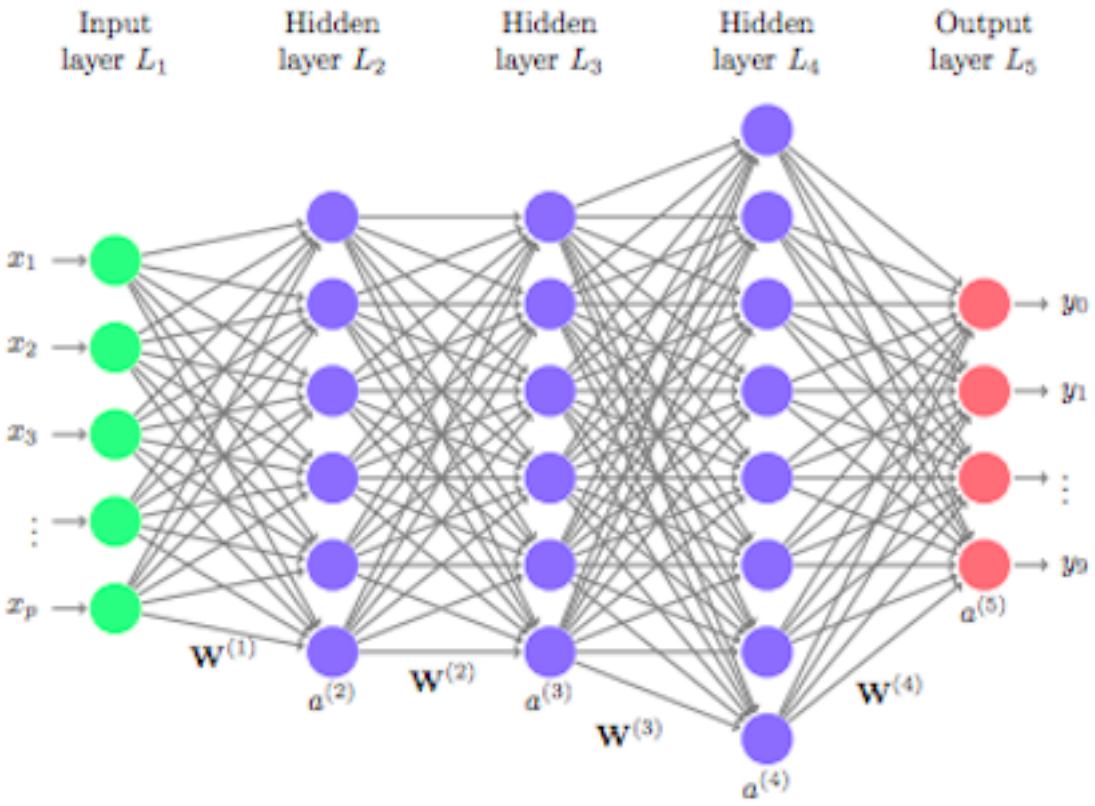

FIGURE 1.1: Simple Convolution Neural Network

Convolution Neural Network

A Convolution Neural Network [22] is analogous to the human brain where artifical neurons are used to learn features from images which then activate the neurons in other layers based on the output from the previous layer. A simple neural network is shown in figure 1.1

A neural network contains an input layer, an output layer and many hidden layers which learn features based on activations from previous layers.

Convolution Neural Network Layers

The various layers in a convolution neural network are as follows:

- **Convolution Layer:** It is used to obtain features from an input image by passing a filtering kernel over it. An image matrix of dimension 5X5 with a kernel of 3X3 will give us convoluted features of size 3X3
- **Pooling Layers:** To further more reduce the features obtained from the convoluted layers, a pooling layer with a fixed kernel sized is given. The types of pooling layers are:
  maxpooling layer, average pooling, minpooling etc.
- **Flatten Layer:** The output obtained from the hidden layers (convolution and pooling layers) are in 2D array form and are converted to a 1D array which becomes the fully connected layer of the neural network
- **Dense Layer:** This is the output layer of the neural network and contains output nodes equal to the number of classes given in the dataset

Activation functions and optimizers:

Activation functions are used to trigger the neurons to produce the desired output to the next layers.

Eg: relu, softmax

Optimizers are used to reduce the error obtained by adjusting the learning rates.

Eg: rmsprop, adam, SGD

1.4.0.3 Pre-Trained Architectures Used

The following pre-trained architectures were used to make a comparison in the performance metrics obtained.

- VGG 16
- InceptionV3

Visual Geometry Group 16 (VGG16) VGG16 [7] is a convolution neural net (CNN) architecture which was used to win ILSVR(Imagenet) competition in 2014. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 fully connected layers (FC) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers as shown in figure 1.2 that have weights. This network is a pretty large network and it has about 138 million (approx) parameters

ADVANTAGES:

1. Easy to change and modify the parameters of a layer

DISADVANTAGES:

1. Slow to train
2. It has so many weight parameters, the models are very heavy, 550 MB + of weight size.

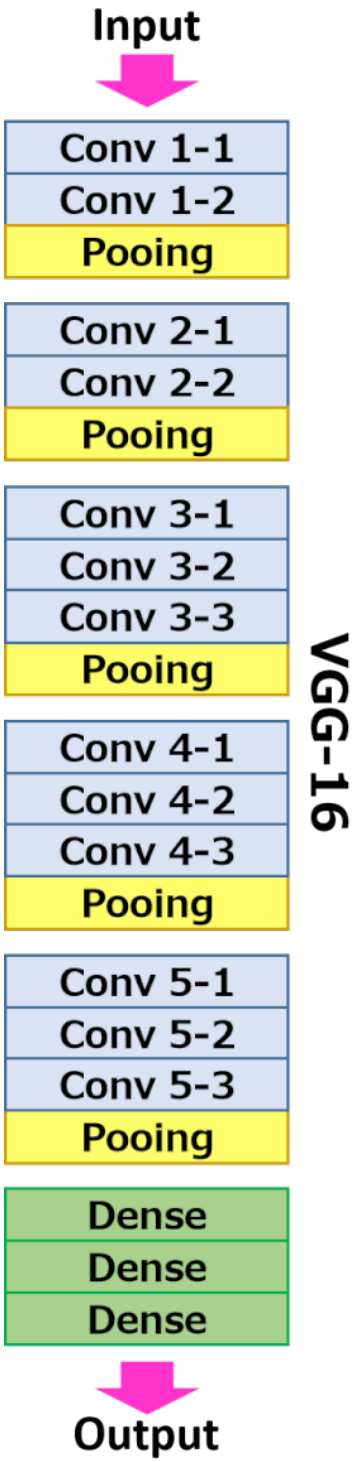

FIGURE 1.2: VGG-16 Architecture

InceptionV3

InceptionV3 [23] is a deep convolutional architecture widely used for classification tasks.The model concept was introduced by Szegedy in the GoogleNet architecture, where InceptionV3 was proposed by updating the inception module.They are mainly known for their parallel execution of different convolution layers with different sized filters and pooling layers. The InceptionV3 network has multiple symmetric and asymmetric building blocks, where each block has several branches of convolutions, average pooling, max-pooling, concatenated, dropouts, and fully-connected layers. This network has 42 total layers and possesses 29.3 million parameters, which means that the computational cost is only about 2.5 higher than GoogleNet.

ADVANTAGES:

1. Computationally more efficient, both in terms of the number of parameters generated by the network and the economical cost incurred (memory and other resources).

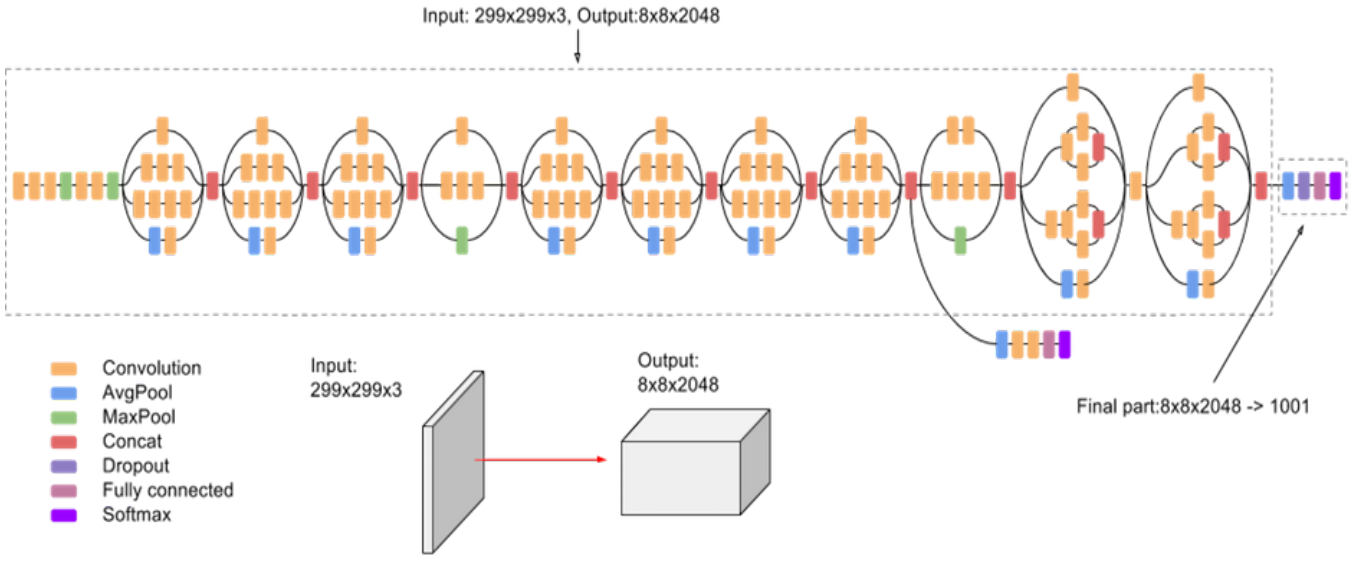

FIGURE 1.3: InceptionV3 Architecture

Support Vector Machine:

A Support Vector Machine (SVM) [16] is a supervised machine learning algorithm that can be employed for both classification and regression purposes. SVMs , figure 1.4 ,are based on the idea of finding a hyperplane that best divides a dataset into two classes or more classes. Support vectors are the data points nearest to the hyperplane, the points of a data set that, if removed, would alter the position of the dividing hyperplane hence becoming critical elements of a data set. A hyperplane is used to divide the dataset into the classes. This classification is done based on the distance (margin) of the support vectors from the hyperplane.

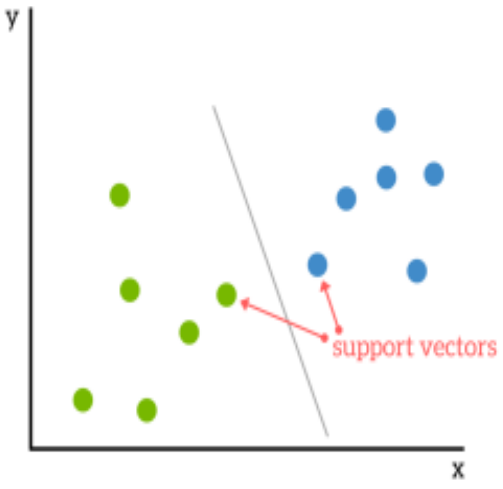

FIGURE 1.4: Support Vector Machine

CHAPTER 2

LITERATURE SURVEY

Ayan Kumar Bhunia, Yongxin Yang, Timothy M. Hospedales et al address the main concern involving the user's ability to draw a complete sketch. A sketch is already a product of a user's subjectivity and ability in expressing a pictographical content. Thus completition of a diagram also depends on such a user's skills in drawing a sketch. Hence the technique used in [2] is a reinforcement learning-based cross-modal retrieval framework that directly optimizes rank of the ground-truth photo over a complete sketch drawing episode and a novel reward scheme that circumvents the problems related to irrelevant sketch strokes, thus providing with a more consistent rank list during the retrieval enabling for faster retrieval of images compared to the time taken by conventional SBIR which involves complete sketching of a query

Most SBIR systems use singular sketches as input and retrieve images. Thus, to retrieve images containing many different classes of sketch, one cannot send a single sketch image containing various sketches belonging to different classes. Hence in [3], Sounak Dey, Anjan Dutta et al addressed this where text and sketches are used for singular sketch input image and a combination of 2 texts and 2 sketches are used to provide as input for multiple classes. Thus a cross-modal deep network architecture is used to learn common embedding between text and images and between sketches and images. An attention model in included to detect the individual multiple input objects in the query on the photo dataset

SBIR models are created by training them with extensive dataset for different classes. It is not possible to always be able to train the model with new classes.

Sasi Kiran Yelamarthi et al [4] suggested a generative approach for the SBIR task by proposing deep conditional generative models that take the sketch as an input and fill the missing information stochastically. This is described as the Zero-Shot framework where a model is able to classify an image from a novel class that was not used during training process giving the name zero for the number of examples used in that class.

Sketch-based image retrieval (SBIR) is a challenging task due to the ambiguity inherent in sketches when compared with photos. A novel convolutional neural network based on Siamese network for SBIR is proposed by Yonggang Qi, Yi-Zhe Song, Honggang Zhang, Jun Liu in [5]. The main idea is to pull output feature vectors closer for input sketch-image pairs that are labeled as similar, and push them away if irrelevant. This is achieved by jointly tuning two convolutional neural networks which linked by one loss function.

Sounak Dey, Pau Riba et al addressed the zero shot learning process by first suggesting a novel ZS-SBIR dataset[6], QuickDraw-Extended, that consists of 330,000 sketches and 204,000 photos spanning across 110 categories was contributed. Highly abstract amateur human sketches are purposefully sourced to maximize the domain gap, instead of ones included in existing datasets that can often be semi-photo realistic. A ZS-SBIR framework was included to jointly model sketches and photos into a common embedding space.

2.1 Literature Survey Summary

TITLE	JOURNAL	METHODOLOGY
Sketch Less for More: On-the-Fly Fine-Grained Sketch Based Image Retrieval	IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2020 [2]	Reinforcement based learning to retrieve photos based on instances of sketch drawn upto then and fixes the photo to fine tune the sketch branch
Learning Cross-Modal Deep Embeddings for Multi-Object Image Retrieval using Text and Sketch	24th International Conference on Pattern Recognition (ICPR) [3]	Text and sketch can be given as input to obtain images based on multiple input
Zero-Shot Framework for Sketch Based Image Retrieval	The European Conference on Computer Vision (ECCV) [4]	Fills the missing class value stochastically for the unseen classes
Sketch-based image retrieval via Siamese convolutional neural network	IEEE International Conference on Image Processing(ICIP) [5]	Creation of a triplet that brings output vector closer for input sketch image pairs that are labelled similar and discarded if irrelevant
Doodle to Search: Practical Zero-Shot Sketch-based Image Retrieval	IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [6]	Creation of a novel dataset containing highly abstract sketches belonging to various classes and a Zero shot framework was suggested to retrieve images.

TABLE 2.1: Literature Survey Summary

CHAPTER 3

PROPOSED METHODOLOGY

Sketch Based Image Retrieval can be done in 2 ways:

- Class based image retrieval - Conventional method
- Sketch based image retrieval

Class based retrieval involves the training the model to classify the input sketch and retrieving the images from class-wise separated photo dataset. In sketch based retrieval, the photo dataset is not classified, rather all images are put together.
Here we use sketch based retrieval of images to obtain the photos from our dataset. The relevant images are retrieved using an object detection library CVlib [8]

3.1 Design overview

3.1.1 Algorithm of Proposed System

INPUT: Sketch containing various drawings
OUTPUT: Photos in ranked order
Step 1: An input sketch containing one ore more drawings is given to the system
Step 2: The various components are identified individually using contours and bounding boxes are drawn around each of them.
Step 3: The individual sketch images are cropped and saved separately

Step 4: Each of the cropped image is sent to a classifier system and its class is identified

Step 5: Using CVLib, bounding boxes and labels the classes obtained from the classifier system are found

Step 6: Based on number of classes contained in each photo, they are ranked.

Step 7: The photos are displayed in order of ranking

3.1.2 Workflow of the Proposed System

The architecture diagram of the system is given in figure 3.1
The modules in the system are explained below:

- **Detection of Individual Sketches:** Here the input image is given and using contours function, the various individual contours are obtained. These contours correspond to individual sketch components. From the contours obtained, left, right and top, bottom coordinates are extracted. Bounding boxes are drawn around each of these sketch components and saved separately

- **Classification of the Sketch:** The separate sketches are loaded individually and their class is predicted.

- **Detection of Component in Image:** Using CVLib, various objects detected in the photos are obtained. This contains list of classes that may also be irrelevant to the query. So the classes of the query sketches are checked and bounding box coordinates of only such labels (query classes) are retained.

- **Ranking:** A photo may contain one or more of the query sketch classes, Thus ranking is performed to order photos based on the number of query sketches found, If a photo contains all query classes, it is given rank 1. Images containing none of the query classes are discarded.

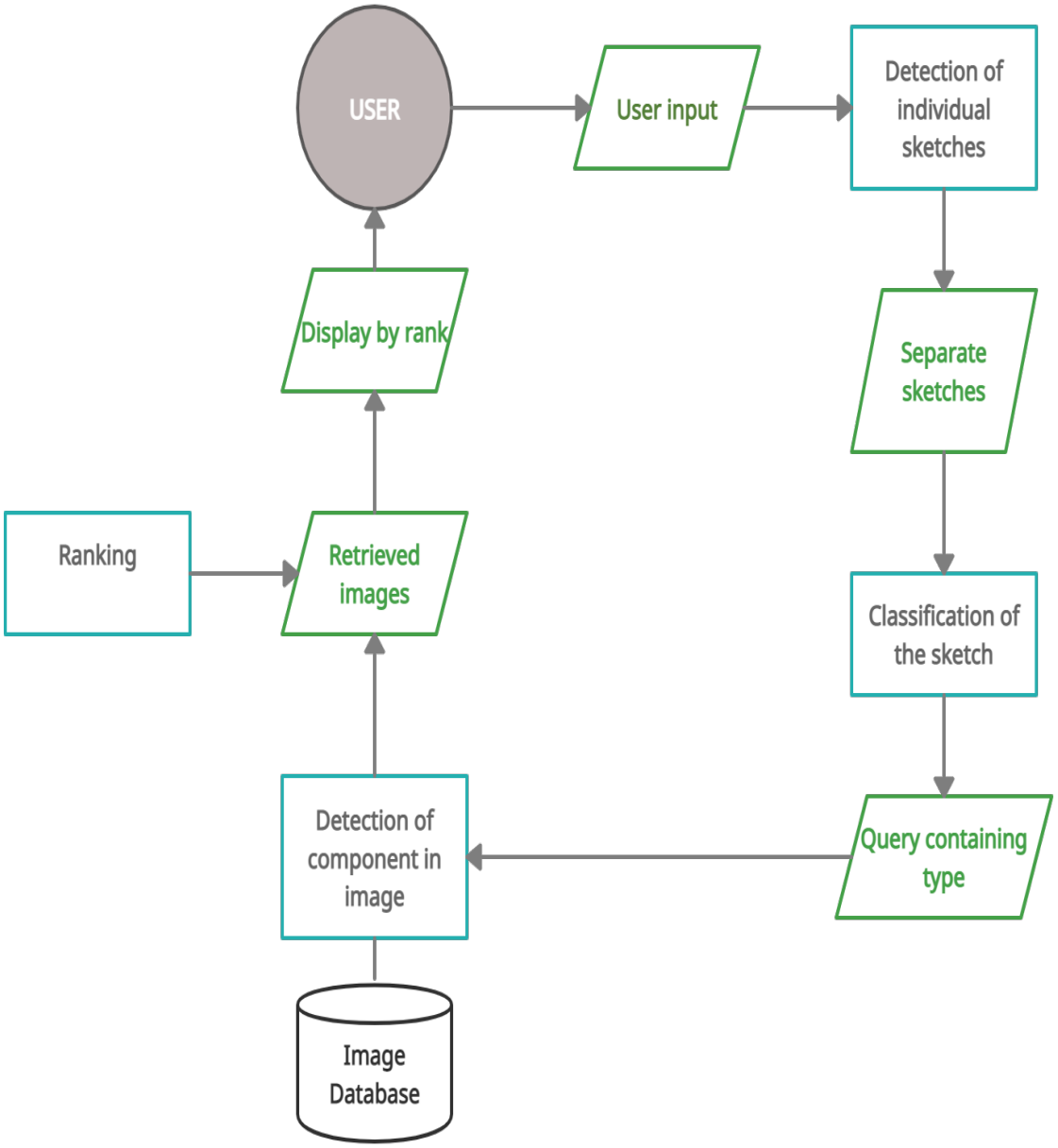

FIGURE 3.1: Architecture Diagram of proposed system

3.2 Dataset Used

The dataset contains two parts.

- Sketches - Drawings of various levels of abstractions
- Photos - A dataset of photos to obtain from based on input query and order by relevance

3.2.1 Sketch Dataset

The sketches are obtained from three sources namely, QuickDraw! [10], TU-Berlin [11] and Sketchy [9]. Each of these sketch datasets contain 110, 251 and 125 classes respectively and the sketches given in each of the dataset are of varying complexities.
The number of sketch images with size of individual image in each of the dataset is given below

SNO	DATASET	IMAGE PER CLASS	IMAGE SIZE (pixels)
1	QuickDraw	3001	256
2	TU-Berlin	80	1111
3	Sketchy	500 to 800	256

TABLE 3.1: Images per class of every dataset

Below are the images in Quickdraw, TU-Berlin and Sketchy dataset for the class apple.

QuickDraw

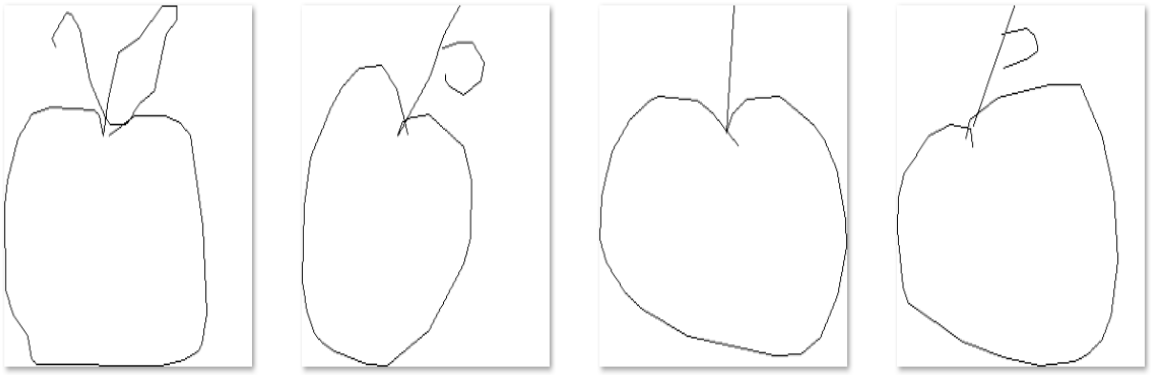

FIGURE 3.2: QuickDraw images for the class apple

Sketchy

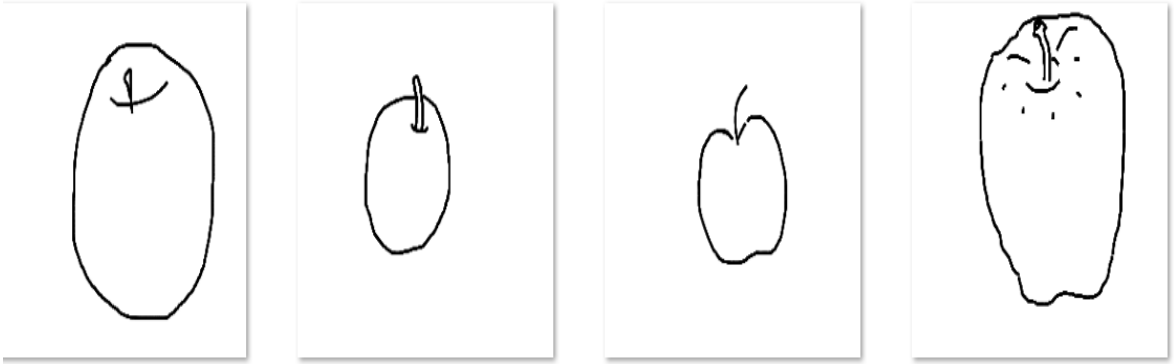

FIGURE 3.3: Sketchy images for the class apple

TU-Berlin

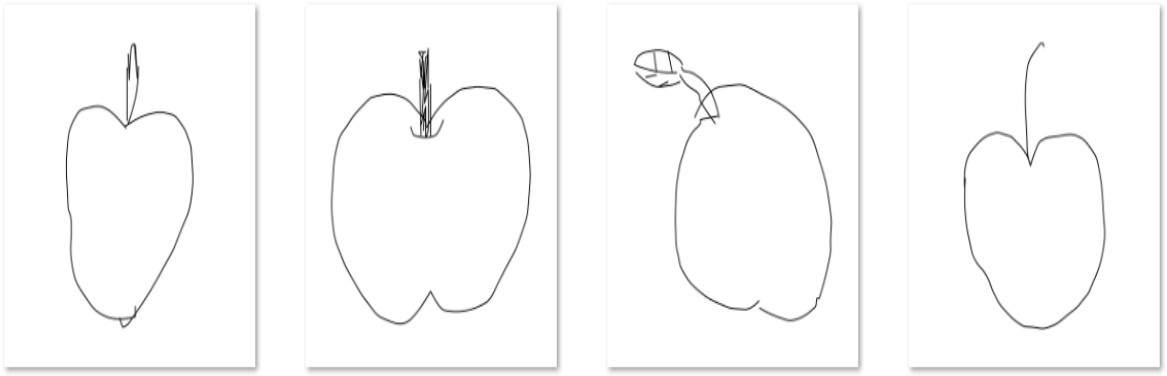

FIGURE 3.4: TU-berlin images for the class apple

3.2.2 Photo Dataset

Sketch dataset Sketchy and QuickDraw comes with photos. To create a simple photo database, 2 images for each class were taken and randomly combined to a single photo. The combinations used are 4 photos, 3 photos and 2 photos. These images are concatenated using python and an online software, Peko-Step[12].

The images combined are given as follows:

4 photos combined to one:

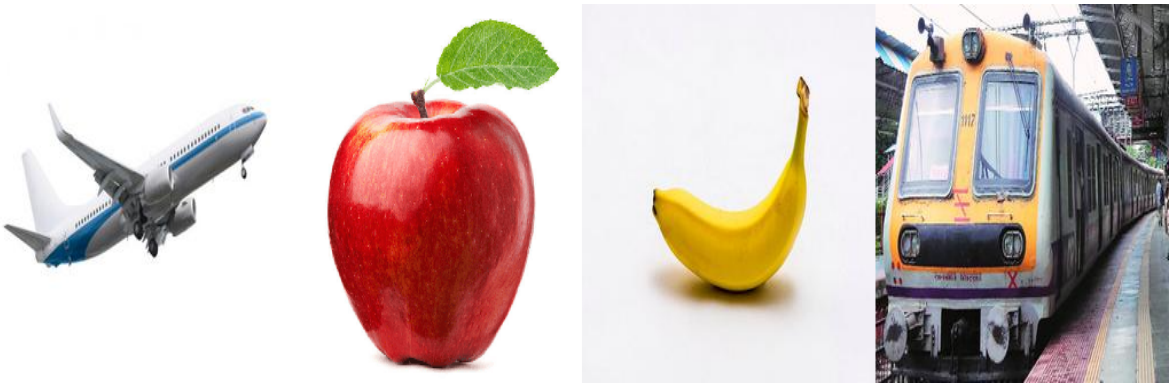

FIGURE 3.5: Photo containing 4 classes - 1

FIGURE 3.6: Photo containing 4 classes - 2

3 photos combined to one:

FIGURE 3.7: Photo containing 3 classes - 1

FIGURE 3.8: Photo containing 3 classes - 2

2 photos combined to one:

FIGURE 3.9: Photo containing 2 classes - 1

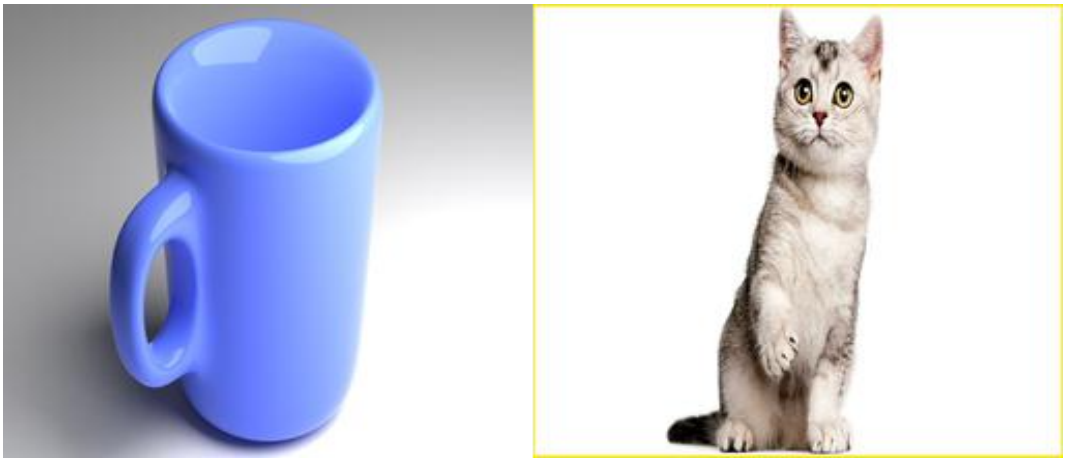

FIGURE 3.10: Photo containing 2 classes - 2

CHAPTER 4

IMPLEMENTATION

This chapter discusses the installation of various python libraries used for implementation of the project and the implementation of the modules with the results obtained. Python 3.8 [13] was installed beforehand to install the required libraries later.

4.1 Libraries Installed

The following libraries were installed using pip in the system:

- Tensorflow - 2.4.1: pip install tensorflow
- Keras - 2.4.3: pip install keras
- MatPlotLib - 3.3.2: pip install matplotlib
- Numpy - 1.19.3: pip install numpy
- OpenCV 2 - 4.5.1: pip install opencv-python
- CVlib - 0.2.5: pip install cvlib
- Sklearn - 0.24.1: pip install sklearn

The codes were compiled on Kaggle [14], Google Colab [15] and Jupyter notebook, which can be installed using the command:
pip install notebook

4.2 Implementation of the SBIR System

4.2.1 Detection of individual sketches

The sketch dataset taken for this project is QuickDraw. It contains a total of 110 classes, of which 15 classes are chosen. The 15 classes are:
Airplane, apple, banana, bicycle, bird, car, cat, chair, cup, elephant, fire hydrant, knife, pizza, teddy bear, train
The sketches in these dataset are in binary format. ie, black and white thus each pixel contains values from 0 to 255 where 0 indicates black, 255 indicates white and the intermediate values denote the greyscale range.

An input image is sent to the system to detect the various individual sketches present in it.

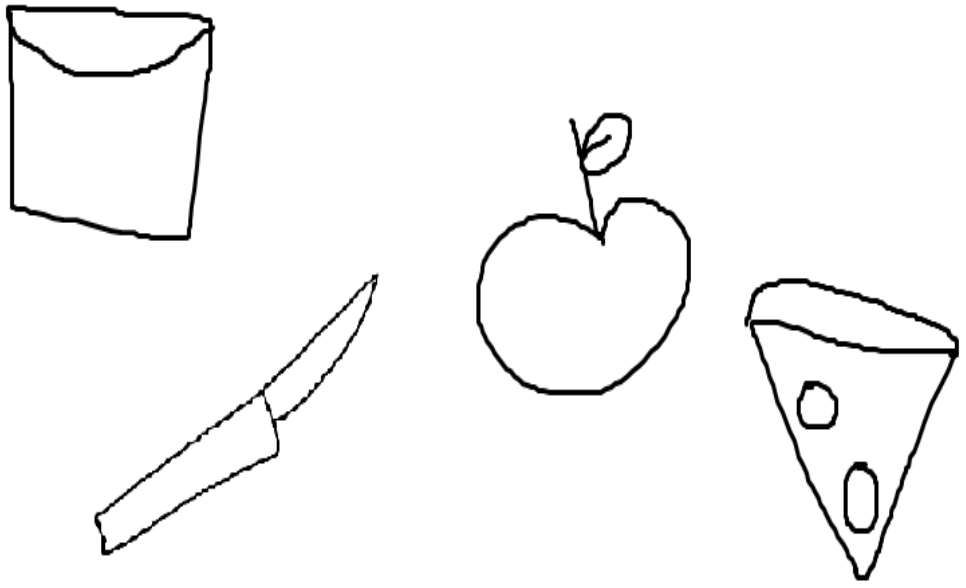

FIGURE 4.1: Input image containing various sketches

Finding individual sketches:

From this input image, the individual sketches are obtained by finding the contours of every single sketch. A contour can be defined as a curve or a line of continuous points containing pixels of the same colour or intensity. These contours can be extracted using various edge detection algorithms such as Canny [17] and Sobel [18].

Here the OpenCV function findContours [19] is used to obtain the contours. To make use of this function, first the images are converted to grayscale and then a threshold function is applied.

This threshold function sets the pixels values above a limit to the said maximum value. The thresholding done here converts pixel values greater than 127 to 255, where 127 is the lowest shade of grey that can be obtained before its transition to white. The thresholded image is inverted and then sent to the findContours

function which returns us arrays containing contours of every individual sketch.

Drawing Bounding Boxes:

To obtain the individual sketch components from the input image, a bounding box is drawn. This can be drawn using the function, cv2.rectangle(), which requires the top left and the bottom right coordinates. These coordinates can be obtained from the contours array by finding the minimum and maximum values in the first column for the left and right coordinates and in the second column for the top and bottom coordinates. Thus from the coordinates a bounding box is drawn, figure 4.2, adding 10 extra pixel values to each coordinates and it is displayed to the user. To crop individual sketches, the coordinates obtained before the addition of the 10 pixel units is taken and individual images are snipped and saved on the local system

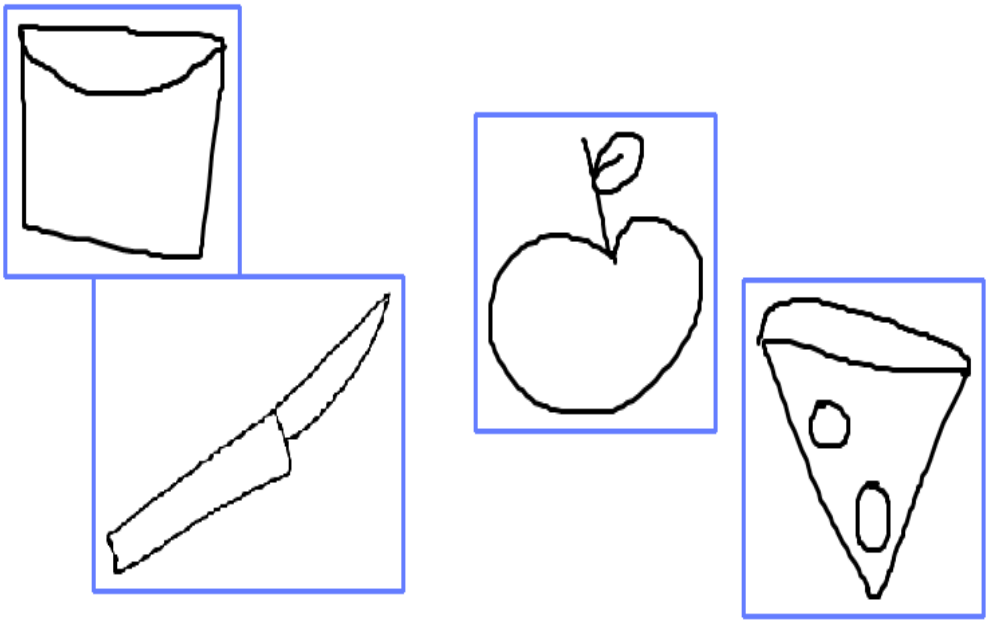

FIGURE 4.2: Individual component detection in the input sketch

The individual sketches are extracted and saved separately as given in figures 4.3 to 4.6

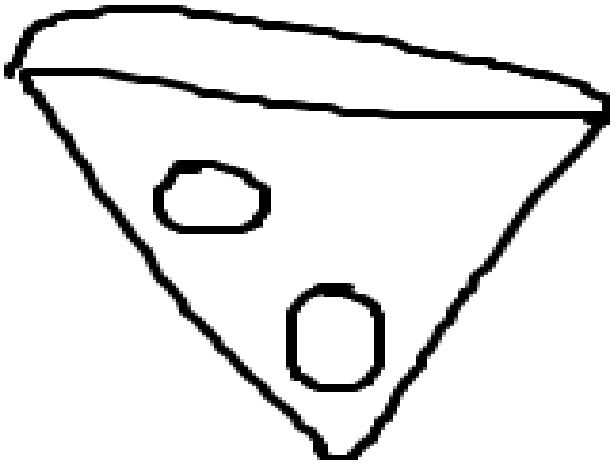

FIGURE 4.3: Individual sketch 1

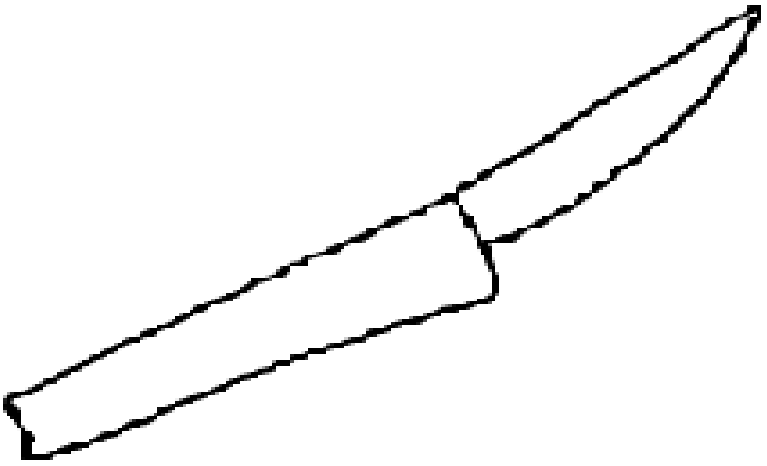

FIGURE 4.4: Individual sketch 2

4.2.2 Creation of a Classification System

4.2.2.1 Model created for the classifier

Deep learning CNN:

The total number of sketch images used in this dataset is 45015 images, which are

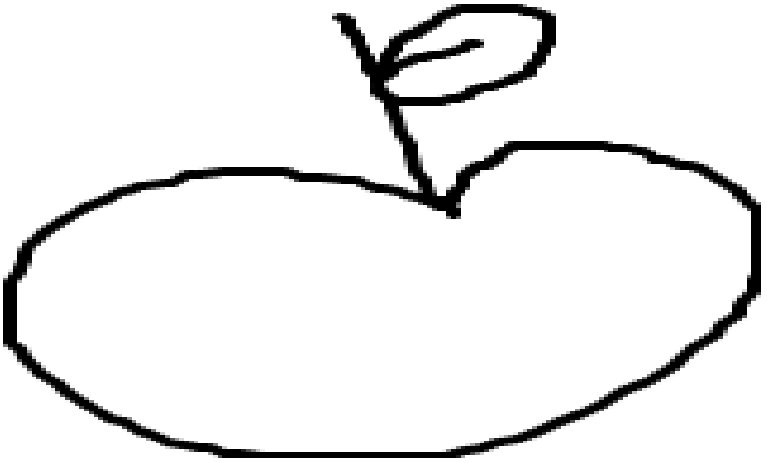

FIGURE 4.5: Individual sketch 3

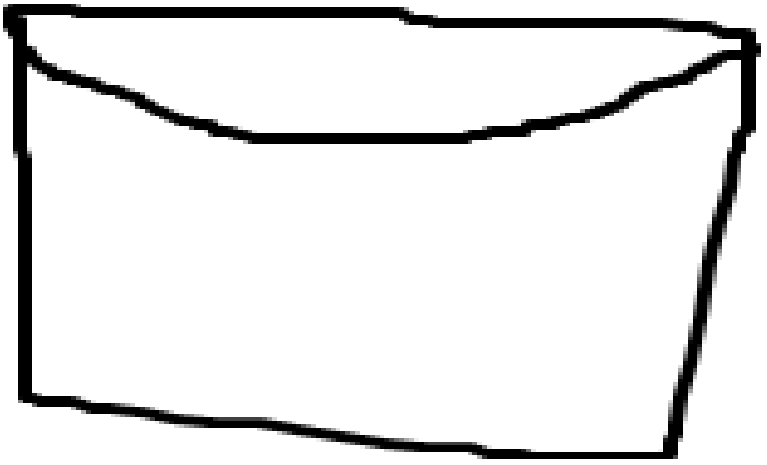

FIGURE 4.6: Individual sketch 4

divided to 38250 training images, 4500 testing images and 2265 validation images. The images are resized to 100*100 for convenience. The model initially created for the classifier incorporated a deep learning approach using Convolution Neural Networks. The architecture of the CNN is given in figure 4.7

The total number of layers in the CNN created is 10. Since the CNN works by reducing the features with every layer, the overall accuracy remains lower. This can also be attributed to the features in the sketch dataset which are very low due to the images containing only black lines on white background.

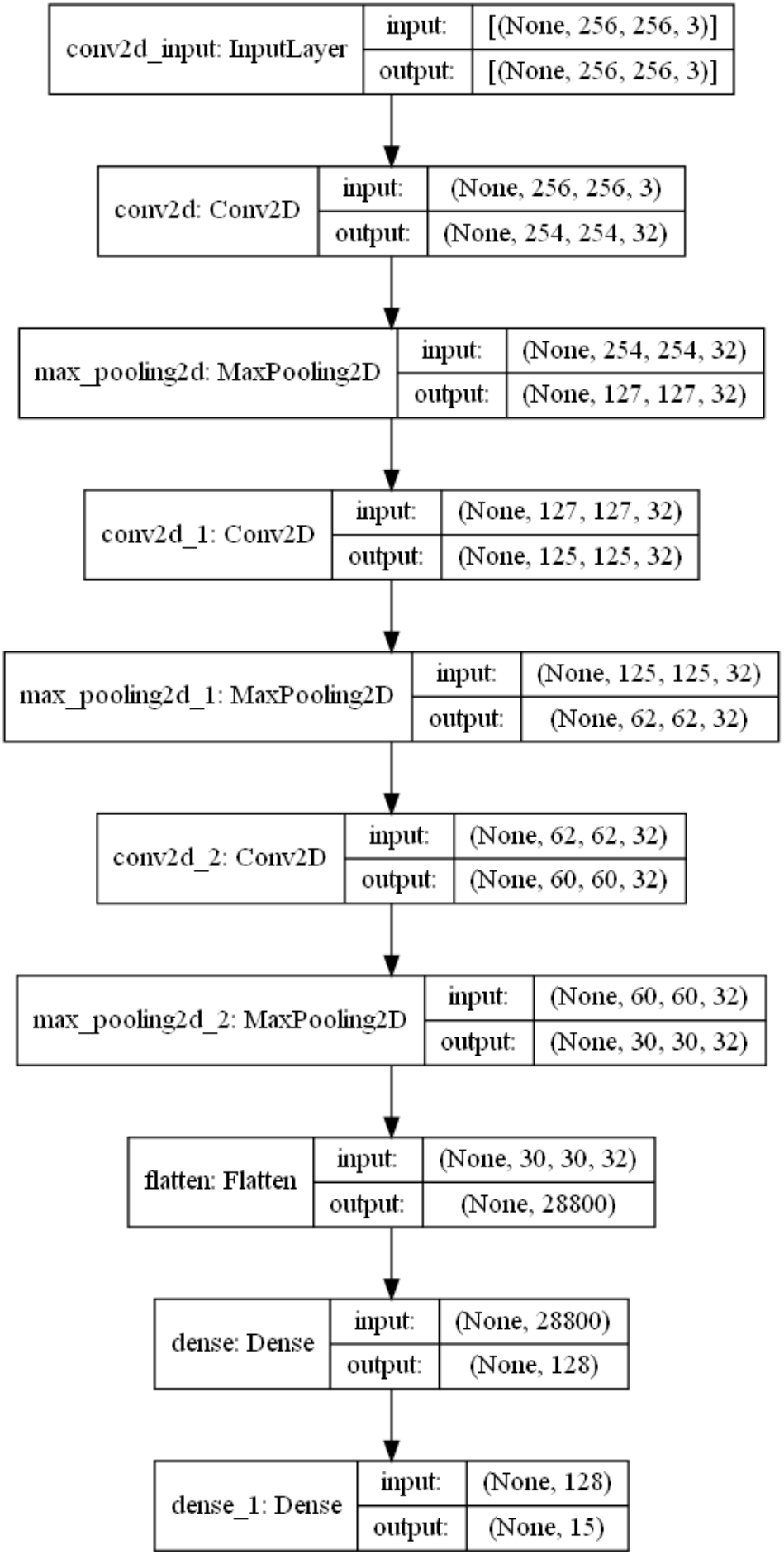

FIGURE 4.7: Architecture of Simple CNN

Performance report of Simple CNN classifier

PRECISION	RECALL	F1-SCORE	ACCURACY
58	44	44	44

TABLE 4.1: Performance analysis of Simple CNN

Deep learning and machine learning combined:

Due to the extreme loss in data features obtained in the convolution neural network, an alternate method was proposed which included the feature extraction from sketches using deep learning architectures such as VGG16 and InceptionV3 and a basic CNN as well.The basic Convolution Neural Network contains 6 layers (figure 4.8). The six layers used are Convolution layer and Maxpooling layer. Here flattening and dense layers are eliminated because of the need for only features and not a classification from the CNN. The feature vectors obtained were then used to train an SVM to classify the various classes. The pretrained models were loaded with imagenet weights. From the classification models created using the pretrained architectures and the simple CNN, it was observed that the sketch classifier is more efficient with neural networks and architectures containing less number of layers. Hence VGG16 proved to work better than InceptionV3.

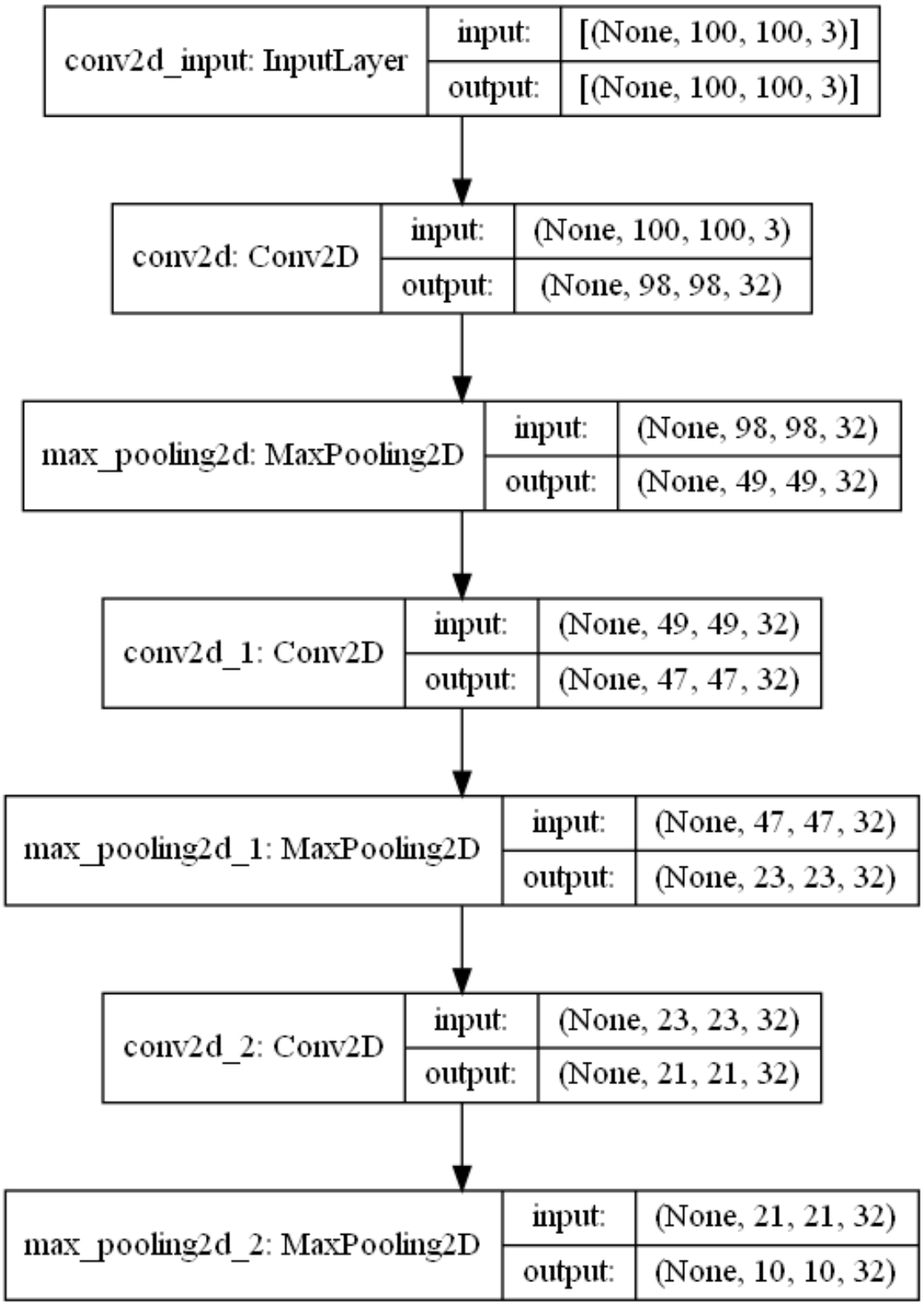

FIGURE 4.8: CNN used for feature extraction

Sample input and output for the classifier system is given in figure 4.9

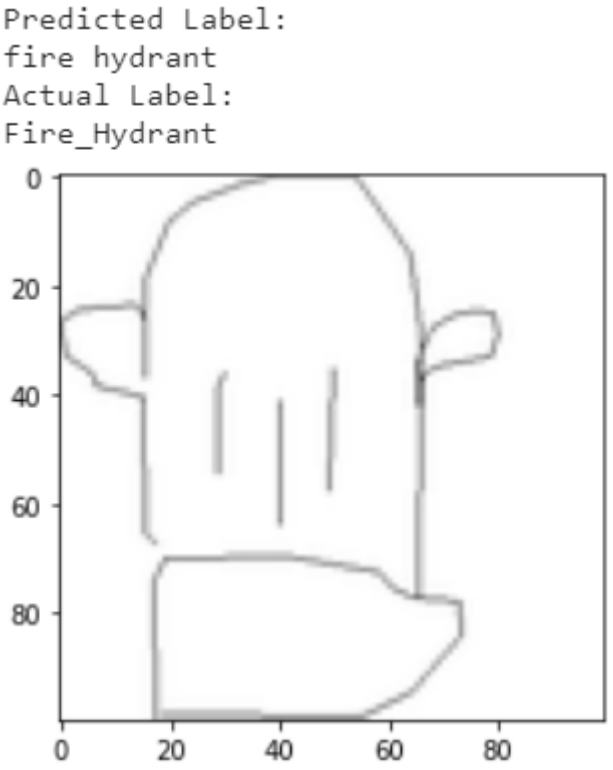

FIGURE 4.9: A sample prediction by SVM Classifier

4.2.3 Object Detection in Photos

The OpenCV library CVlib [8] is used to detect objects in an image. The library is trained on COCO dataset [24] and uses the pre-trained YOLO v3 weights. The dataset contains 80 classes of different objects. CVlib can perform face detection, gender detection and object detection.

Here the object detection function is used to detect multiple objects in a photo. Given a photo, cvlib's Detect Common Object function detects all objects seen in the photo, which includes redundant objects as well. The function returns bounding boxes coordinates, labels and confidence scores of the objects. Hence

we remove the labels, bounding box coordinates and confidence scores of the objects that are not a part of the input query.

Consider an input photo read from the database.

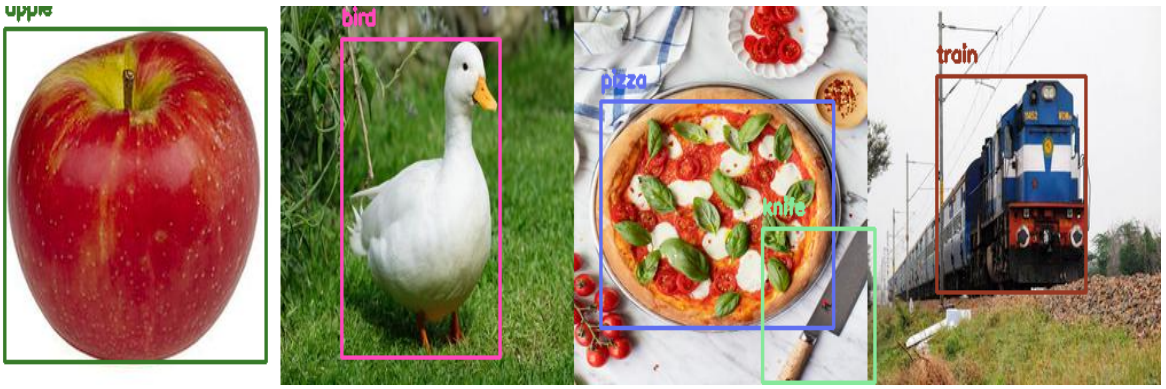

FIGURE 4.10: CVLib object detection performed on a photo

The detect common objects function detected all objects in the photo, figure 4.10.

Consider a query: (knife,pizza)

Since the photo contains detection for objects other than the queries, we eliminate the bounding boxes and labels for those images, thus giving figure 4.11

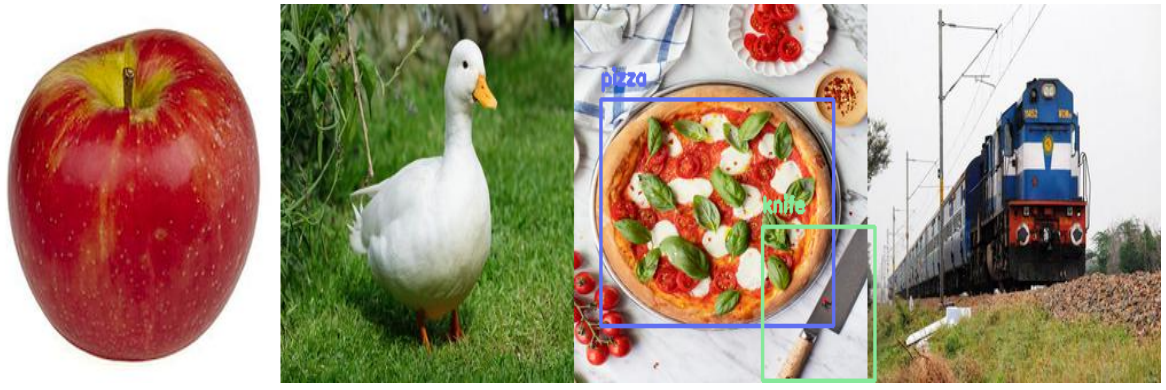

FIGURE 4.11: Photo containing bounding boxes only for query objects

4.2.4 Ranking and Retrieval

After filtering photos containing only components in the query, they must be ranked according to relevance. Relevance, here is a measure that counts the number of query objects found in the photos counting different instances (objects) of the same class as 1. Based on the number of query objects, given in input sketch, found in each of photos, they are given a rank. Rank=1 is given to photos containing all query objects, rank=2 is given to photos missing at-most 1 query object and rank=3 is given for all other images.

NOTE: Images that do not contain any of the query objects are discarded in the retrieval process.

RANKED IMAGES:

```
Sorted dictionary:
 [(7, 2), (0, 3), (1, 3), (3, 3), (4, 3), (8, 3), (10, 3), (11, 3), (12, 3), (1
3, 3), (14, 3), (15, 3), (18, 3), (19, 3), (22, 3), (30, 3), (31, 3), (32, 3),
(34, 3), (35, 3), (36, 3), (37, 3), (39, 3), (40, 3), (43, 3), (44, 3), (45,
3), (49, 3), (50, 3), (54, 3), (57, 3), (58, 3), (62, 3), (63, 3), (66, 3), (6
9, 3), (73, 3), (74, 3), (79, 3), (80, 3), (82, 3), (87, 3), (89, 3), (90, 3),
(91, 3), (92, 3), (93, 3), (94, 3), (97, 3)]
Total number of images retrieved:  49
```

FIGURE 4.12: Images in ranked order

The tuple in the sorted dictionary contains images in the ranked order. Consider a tuple from the dictionary (7,2) in figure 4.12. Here 7 is the photo number and 2 is the rank.

4.3 Performance Analysis

4.3.1 Performance Analysis of 3 models

Performance analysis is made for each model created and compared to find the most efficient feature extractor.

Validation dataset:

MODELS	PRECISION	RECALL	F1-SCORE	ACCURACY
VGG16	0.89	0.89	0.89	0.8922
INCEPTIONV3	0.69	0.69	0.69	0.6866
BASIC CNN - 6 LAYERS	0.65	0.65	0.65	0.76

TABLE 4.2: Performance analysis for validation dataset

Testing dataset:

MODELS	PRECISION	RECALL	F1-SCORE	ACCURACY
VGG16	0.89	0.89	0.89	0.8895
INCEPTIONV3	0.69	0.69	0.69	0.6902
BASIC CNN - 6 LAYERS	0.66	0.66	0.65	0.75

TABLE 4.3: Performance analysis for testing dataset

Training dataset:

MODELS	PRECISION	RECALL	F1-SCORE	ACCURACY
VGG16	0.1	0.1	0.1	0.1
INCEPTIONV3	0.79	0.79	0.79	0.7935
BASIC CNN - 6 LAYERS	0.1	0.1	0.1	0.1

TABLE 4.4: Performance analysis for training dataset

Thus it is shown that the pretrained model VGG-16 works better than inceptionv3 and basic CNN for feature extraction

4.3.2 Performance Analysis VGG-16

The validation and testing classification report for the 15 class dataset in VGG-16 is included below.

VALIDATION CLASSIFICATION REPORT

CLASS	PRECISION	RECALL	F1-SCORE	SUPPPORT
airplane	0.85	0.87	0.86	151
apple	0.97	0.96	0.96	151
banana	0.92	0.93	0.92	151
bicycle	0.92	0.95	0.93	151
bird	0.82	0.81	0.82	151
car	0.88	0.87	0.88	151
cat	0.81	0.85	0.83	151
chair	0.95	0.91	0.93	151
cup	0.93	0.92	0.93	151
elephant	0.84	0.87	0.85	151
fire hydrant	0.81	0.83	0.82	151
knife	0.92	0.87	0.89	151
pizza	0.96	0.97	0.96	151
teddy bear	0.92	0.92	0.92	151
train1	0.90	0.86	0.88	151
Accuracy			0.89	2265
Macro avg	0.89	0.89	0.89	2265
Weighted avg	0.89	0.89	0.89	2265

TABLE 4.5: Classification report for validation set of VGG-16

VALIDATION SET CONFUSION SET

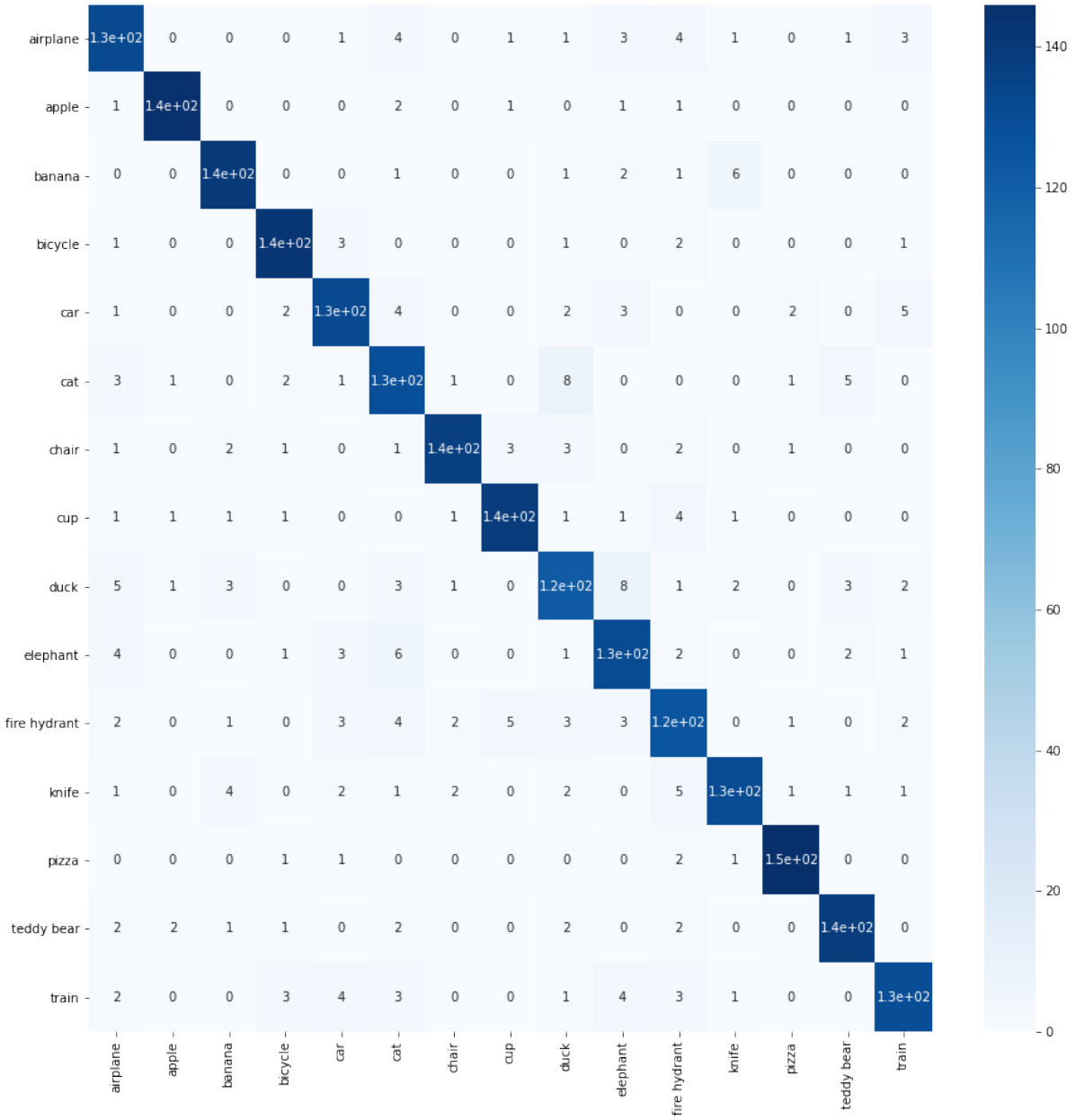

FIGURE 4.13: Validation confusion matrix

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

From the results obtained, it is understood that a model using pure deep learning does not classify images accurately due to the nature of the sketches being binary ie. black and white, when compared to obtaining features using a deep learning model and then training on those features using a support vector machine.The sketch images contain only 0 to 255 as pixel values, which is greatly reduced as we move down the filtering layers from the input convolution layer to the final output dense layer. This is also a possible reason for the lower accuracy of the convolution neural network (CNN). Due to learning features using pretrained architectures like VGG16, the SVM model is able to predict highly detailed images although the training was done only on heavily abstract and light featured sketch images.

5.2 Future Work

Since CVLib is based on the COCO dataset, it is trained to only detect objects belonging to those 80 classes. This limits us to use classes in our dataset common to both the COCO dataset and the sketch dataset. Hence a separate object detection model can be created that allows for usage of all classes taken in the dataset. This object classifier can be created and deployed using Tensorflow's Object Detection API [25] using the guide given in [26].

REFERENCES

1. Hirata, K., Kato. T (1992) 'Query by visual example—content based image retrieval ', International Conference on Extending Database Technology: Advances in Database Technology, pp. 56– 71

2. Ayan Kumar Bhunia, Yongxin Yang, Timothy M. Hospedales, Tao Xiang, Yi-Zhe Song (2020) 'Sketch Less for More: On-the-Fly Fine-Grained Sketch Based Image Retrieval ', IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

3. Sounak Dey, Anjan Dutta, Suman K. Ghosh, Ernest Valveny, Josep Llados,Umapada Pal (2018) 'Learning Cross-Modal Deep Embeddings for Multi-Object Image Retrieval using Text and Sketch', 24th International Conference on Pattern Recognition (ICPR)

4. Sasi Kiran Yelamarthi, Shiva Krishna Reddy, Ashish Mishra, and Anurag Mittal (2018)'A Zero-Shot Framework for Sketch Based Image Retrieval',The European Conference on Computer Vision (ECCV)

5. Yonggang Qi, Yi-Zhe Song, Honggang Zhang, Jun Liu (2016), 'Sketch-based image retrieval via Siamese convolutional neural network', IEEE International Conference on Image Processing(ICIP),

6. Sounak Dey, Pau Riba, Anjan Dutta, Josep Llados, Yi-Zhe Song (2019), 'Doodle to Search: Practical Zero-Shot Sketch-based Image Retrieval', The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

7. *VGG-16,* https://keras.io/api/applications/vgg/

8. *CVLib - object detection library*, https://www.cvlib.net/

9. *Sketchy dataset*, https://sketchy.eye.gatech.edu/

10. *QuickDraw! dataset*, https://github.com/sounakdey/doodle2search

11. *TU-Berlin dataset*, http://cybertron.cg.tu-berlin.de/eitz/projects/classifysketch/

12. *Peko-Step,* https://www.peko-step.com/en/tool/combine-images.html

13. *Python 3.8,* https://www.python.org/downloads/release/python-380/

14. *Kaggle,* https://www.kaggle.com

15. *Google colab,* https://www.https://colab.research.google.com/

16. *Support vector machine,* https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989

17. *Canny,* https://medium.com/sicara/opencv-edge-detection-tutorial-7c3303f10788

18. *Sobel,* http://www.adeveloperdiary.com/data-science/computer-vision/how-to-implement-sobel-edge-detection-using-python-from-scratch/

19. *findContours(),* https://www.thepythoncode.com/article/contour-detection-opencv-python

20. *Machine learning,* https://www.geeksforgeeks.org/machine-learning/

21. *Deep Learning,* https://machinelearningmastery.com/what-is-deep-learning/

22. *Convolution Neural Network,* https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

23. *InceptionV3,* https://keras.io/api/applications/inceptionv3/

24. *COCO Dataset,* http://images.cocodataset.org/zips/val2017.zip

25. *Tensorflow Object Detection API,* https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/

26. *Tensorflow Object Detection API usage*
https://towardsdatascience.com/how-to-train-your-own-object-detector-with-tensorflows-object-detector-api-bec72ecfe1d9