IMAGE CLASSIFICATION OF DOGS VS CATS USING CNN

A Project report submitted in

Partial fulfillment of the requirement for the award of the Degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED

By

R.SAMPADA **19671A0541**

Under the esteemed guidance of

Mrs.S.GAYATHRI DEVI

ASSISTANT PROFESSOR

Department of Computer Science and Engineering

J.B. Institute of Engineering & Technology

UGC AUTONOMOUS

(Accredited by NAAC & NBA, Approved by AICTE & Permanently affiliated by JNTUH)

Yenkapally, Moinabad mandal, R.R. Dist-75 (TG)

2019-2023

J.B.INSTITUTE OF ENGINEERING & TECHNOLOGY

UGC AUTONOMOUS

(Accredited by NAAC & NBA, Approved by AICTE & Permanently affiliated by JNTUH)

Yenkapally, Moinabad mandal, R.R. Dist-75 (TG)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project report entitled "IMAGE CLASSIFICATION OF DOGS VS CATS USING CNN" submitted to the Department of Computer Science and Engineering, J.B Institute of Engineering & Technology, in accordance with Jawaharlal Nehru Technological University regulations as partial fulfillment required for successful completion of Bachelor of Technology is a record of bonafide work carried out during the academic year 2019-20 by,

R.SAMPADA **19671A0541**

Internal Guide
Mrs.S.GAYATHRI DEVI
ASSISTANT PROFESSOR

Head of the Department
Dr. P. SRINIVASA RAO
PROFESSOR

External Examiner

J.B.INSTITUTE OF ENGINEERING & TECHNOLOGY

UGC AUTONOMOUS

(Accredited by NAAC & NBA, Approved by AICTE & Permanently affiliated by JNTUH)

Yenkapally, Moinabad mandal, R.R. Dist-75 (TG)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We hereby certify that the Main Project report entitled **"IMAGE CLASSIFICATION OF DOGS VS CATS USING CNN"** carried out under the guidance of**, Mrs.S.GAYATHRI DEVI, Assistant Professor** is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **computer science and engineering.** This is a record of bonafide work carried out by us and the results embodied in this project report have not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

Date: 14/02/2022

R.SAMPADA **19671A0541**

ACKNOWLEDGEMENT

At outset we express our gratitude to almighty lord for showering his grace and blessings upon us to complete this Main Project. Although our name appears on the cover of this book, many people had contributed in some form or the other to this project Development. We could not have done this Project without the assistance or support of each of the following.

First of all we are highly indebted to **Dr. S SUDHAKARA REDDY**, Principal for giving us the permission to carry out this Main Project.

We would like to thank **Dr. P. SRINIVASA RAO,** Professor & Head of the Department of COMPUTER SCIENCE AND ENGINEERING**,** for being moral support throughout the period of the study in the Department.

We are grateful to **Mrs.S.GAYATHRI DEVI,** Assistant Professor COMPUTER SCIENCE AND ENGINEERING, for his valuable suggestions and guidance given by him during the execution of this Project work.

We would like to thank Teaching and Non-Teaching Staff of Department of Computer Science and Engineering for sharing their knowledge with us.

R.SAMPADA **19671A0541**

ABSTRACT

Many problems in computer vision were saturating on their accuracy before a decade. However, with the rise of deep learning techniques, the accuracy of these problems drastically improved. One of the major problem was that of image classification, which is defined as predicting the class of the image. Cat and Dog image classification is one such example of where the images of cat and dog are classified. This paper aims to incorporate state-of-art technique for object detection with the goal of achieving high accuracy. A convolutional neural network is been build for the image classification task. Image classification is a fundamental problem in computer vision. Deep learning provides successful results for machine learning problems. Many algorithms like minimum distance algorithm, K-Nearest neighbour algorithm, Nearest Clustering algorithm, Fuzzy C - Means algorithm, Maximum likelihood algorithm are used for the purpose of image classificstion. Generally convolutional neural network uses GPU technology because of huge number of computations but, in proposed method we are building a very small network which can work on CPU as well. The network is trained using a subset of Kaggle Dog-Cat dataset. This trained classifier can classify the given image into either cat or dog. The same network can trained with any other dataset and classify the images into one of the two predefined class.The Dogs vs. Cats image classification has been around for a long time now. The Dogs vs. Cats competition from Kaggle is trying to solve the CAPTCHA challenge, which relies on the problem of distinguishing images of dogs and cats. It is easy for humans, but evidence suggests that cats and dogs are particularly difficult to tell apart automatically. Many people has worked or are working on constructing machine learning classifiers to address this problem. A classifier based on color features got 56.9% accuracy on the Asirra dataset. An accuracy of 82.7% was achieved from a SVM classifier based on a combination of color and texture features.

TABLE OF CONTENTS

1. INTRODUCTION	**01-03**
1.1 Existing System	02
1.2 Disadvantages	02
1.3 Proposed System	02
1.4 Advantages	03
1.5 System Scope	03
2. LITERATURE SURVEY	**04-05**
2.1 Previous work	04
2.2 Contributions	05
3. PROJECT ARCHITECTURE	**06**
4. SYSTEM ANALYSIS	**07-09**
4.1 Feasibility study	07
5. SYSTEM REQUIRMENT SPECIFICATIONS	**10-13**
5.1 Introduction	10
5.2 Purpose	11
5.3 Functional Requirements	12
5.4 Non-Functional Requirements	13
6. SYSTEM DESIGN	**12-19**
6.1Introduction	12
6.2CNN Architecture and Background	14
A. Convolution 3D Layer	
B. Max Pooling 3D Layer	

C . Global/Fully Connected Layer

D. Dense

6.3 Different parameters of CNN 17

A. Activation

B. Loss

C. Optimizer

D. Matrices

7. IMPLEMENTATION **20 -22**

7.1 Introduction 20

7.2 Technology Description 21

8. UML DIAGRAMS **23-30**

8.1 Introduction to UML Diagrams 23

8.2 Types of UML Diagrams 23

9. MODULES **31-32**

9.1 Preparing Training Data 31

9.2 Build and Train CNN Model 31

A. Convolution

B. Pooling

C. Fully Connected

9.3 Make Predictions 32

10. TESTING **33-34**

11. TESTING **35-37**

11.1 Testing Methodologies 35

11.2 Integration Testing 36

11.3 Acceptance Testing 37

12. SCREENSHOTS 38

12.1 Input Screenshots 38

12.2 Output Screenshots 38

13. EXAMPLES 39

14. CONCLUSION AND FUTURE REMARKS 40-42

14.1 Future works 41

14.2 Concluding Remaks 42

15. REFERENCES 43

16. RECOMMENDATIONS 44

17. SELF-ASSESSMENT 45

18.APPENDIX 46

LIST OF FIGURES

Figure 1.1 A Simple Neural network 01

Figure 2.1.1 Error Rates on ImageNet challenge 04

Figure 3.1 Feature Extractor and classifier 06

Figure 6.3.1 CNN Architecture 19

Figure 7.2.1 Python3 The Standard Type Hierarcy 22

Figure 8.2.1 Deployment Diagram 24

Figure 8.2.2 Activity Diagram 25

Figure 8.2.3 E-R Diagram 26

Figure 8.2.4 Flowchart Diagram 27

Figure 8.2.5 Use Case Diagram 28

Figure 8.2.6 State Chart Diagram 29

Figure 8.2.7 Class Diagram 30

Figure 8.2.8 Component Diagram 30

Figure 8.2.7 Convolution 31

Figure 8.2.7 Pooling 31

Figure 8.2.7 Fully Connected 32

Figure 12.1 Input Screenshot 38

Figure 12.2 Output Screenshot 38

Figure 13.1 Example 1 39

Figure 13.1 Example 2 39

1. INTRODUCTION

Image classification is one of the fundamental problems in computer vision. It forms basis for many other computer vision tasks such as object recognition, image segmentation and object detection. The task of categorizing images into one of several predefined classes is called image classification. Though the task of classifying images is easy for human beings, it is very difficult for an automated system. By using machine learning techniques, images can be classified. These machine learning algorithms falls under the category of deep learning. Deep learning is a type of neural network algorithms in which each layer is responsible for extracting one or more features of the image. A neural network is a computational model that is similar to a human brain. It is collection of nodes called as neurons. These nodes are organized into layers where each neuron in the one layer takes some input processes it and passes the output to the neuron in the next layer. Different layers may perform different kinds of transformations. Data transfers from the input layer (first layer) to the output layer (last layer) by traversing various hidden layers. One of the most popular techniques used for improving the accuracy of image classification is Convolutional Neural Networks (CNN). [1]. Figure 1 shows the structure of an artificial neural network.

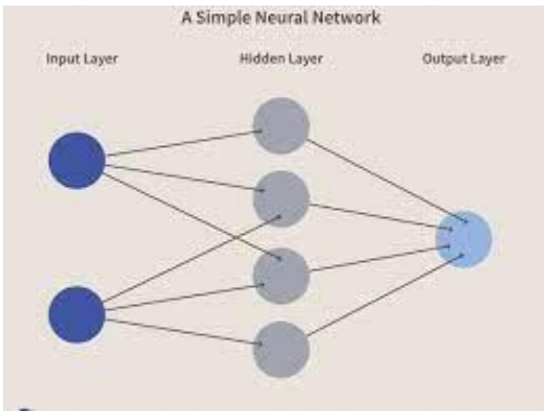


FIG 1.1

Image classification can be done using both supervised classification algorithms and unsupervised classification algorithms. Supervised classification uses training data along with human intervention whereas in unsupervised classification human intervention is not required as it is fully computer operated. The supervised classification has two phases namely training phase and classification phase. In training phase the classifier is given information about classes. This is the phase where learning of a model takes place. In classification phase it uses the information provided by the training data and classifies the image into one of the predefined classes. Various algorithms such as minimum distance algorithm, K-Nearest neighbour algorithm, Nearest Clustering algorithm, Fuzzy C - Means algorithm, Maximum likelihood algorithm and so on are used for the purpose of classification of images. Ever since Alex Krizhevsky, Geoff Hinton and Ilya Sutskevar won ImageNet in 2012, Convolutional Neural Networks (CNNs) have become the standard for image classification. [2] In this paper, we would like to build a convolutional neural network based image classifier which will identify and separate images of dogs from that of cats. Traditional neural networks like AlexNet, Inception which are very good at doing image classification need GPU and take few hours of training time. Therefore, a very small CNN of six layers is build which can work on CPU as well. However, the objective of this paper is to show how to build a real-world convolutional neural network using Tensorflow framework.

1.1 EXISTING SYSTEMS

- The problem of detection and classification of objects in real- time started after major developments , processing hardware, and deep learning model.
- Very littlework has been done in this field.
- In previous field reports were based on imaging techniques like millimeter – wave and infrared imaging.

1.2 DISADVANTAGES

- They were costly to use in many locations because they need to be coupled with X-ray scanners so objects were not accurate.
- Economic cost and health risks limited the practical implementation of such methods**.**

1.3 PROPOSED SYSTEMS

- Deep learning is a branch of machine learning inspired by the functionality and structureof the human brain also called an Artificial Neural Network.
- The methodology adopted in this work features the state of art deep learning, especially the Convolutional Neural Networks due to their exceptional performance in this field.
- These mentioned techniques are used for both the classification as well as localizing the specific object in a frame so both the object classification and detection algorithms were used.

- To achieve high precision, increase number of frames per seconds and improve localization, we moved to the object detection and region proposal methods

1.4 ADVANTAGES

- High accuracy.
- Classification and detection problem as both have a separate requirement for performing the tasks.
- Mean average precision as well as frame per second for the real-time implementation.

1.5 SYSTEM SCOPE

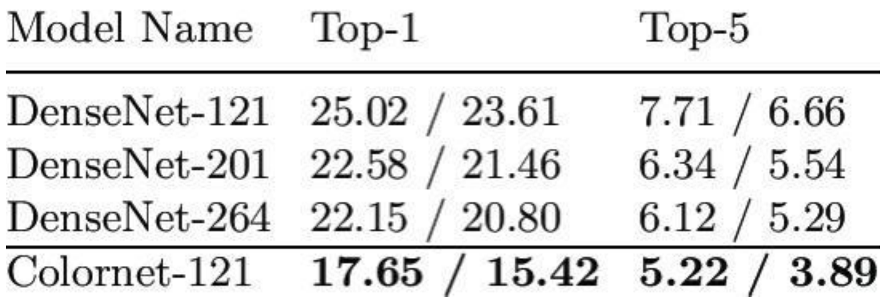
Machine learning is a vast field and in order for this study to be completed within the given time span, the scope of the study needs to be limited to some subset of machine learning. The theoretical scope of the study is therefore limited to neural networks, with a focus on convolutional neural networks. The practical scope will be limited to creating and training three equivalent neural networks in both TensorFlow and CNTK for three already prepared and well known datasets in order to evaluate the frameworks' performance on the same neural network. Gathering and preparing training data is therefore beyond the scope of this study.

2. LITERATURE SURVEY

In my project I am going to build a convolutional neural network to solve the problem and achieve higher performance and better results. In my project instead of using the Kaggle data set comprising of total 25000 images, I would be working on subset of these images. My dataset would be comprising of total 10000 images. Keras would used for model building and all the code would be implemented on JUPYTER NOTEBOOK.

2.1 PREVIOUS WORK

State-of-the-art CNN models for image classification are judged by their performance on the ImageNet challenge. They have proceeded incrementally since (Krizhevsky et al., 2012) introduced AlexNet in 2012, with modifications to the architecture that have im- proved performance. In 2014, (Szegedy et al., 2015) introduced GoogLeNet, which was an improvement on AlexNet, mainly through greatly reducing the num- ber of parameters involved. Also, in 2014, (Simonyan & Zisserman, 2014) introduced the VGGNet, which achieved good performance because of the depth of the network. Most recently, in 2015, (He et al., 2015b) introduced the ResNet, which utilizes “skip connec- tions” and batch normalization. The performance of these models on ImageNet is shown in Table 1.



Model Name	Top-1	Top-5
DenseNet-121	25.02 / 23.61	7.71 / 6.66
DenseNet-201	22.58 / 21.46	6.34 / 5.54
DenseNet-264	22.15 / 20.80	6.12 / 5.29
Colornet-121	**17.65 / 15.42**	**5.22 / 3.89**

FIG 2.1.1

The image captioning problem has also seen a lot of work in recent years. In the last year, (Karpathy & Fei-Fei, 2015) introduced a method that combines a pre-trained CNN (VGGNet) as a feature extractor, a Markov Random Field (MRF) model for alignment, and an RNN for generating text descriptions. The approach of (Donahue et al., 2015), similarly, uses a pre-trained VGGNet as a feature extractor, and di- rectly inputs these feature vectors and the word em- bedding vectors for sentences at each time step of a Long Short-Term Memory (LSTM) model, which was introduced by (Hochreiter & Schmidhuber, 1997). In (Vinyals et al., 2015), Vinyals et al. improve on this approach by only inputting the image vector at the first time step of the LSTM, which they found to im- prove the results.

2.1 CONTRIBUTIONS

In this paper, we explore both the image classification problem and the image captioning problem.

For the image classification problem, we explore and motivate methods for improving the classification ac- curacy of CNN models in a bottom-up manner. Along the way, we show the impact that these modifications have on model performance. We have implemented all the models we tested (plus more popular models), in Python using both the Keras and Lasagne deep learn-ing libraries. This “model zoo” is available for the community to use and improve.

For image captioning, we have implemented a method in the style of Vinyals et al. The method compares to the performance of the original implementation. How- ever, we found that with smaller amounts of training data, using Gated Recurrent Units (GRUs) (Cho et al., 2014) instead of LSTMs improve the performance of the model.

3. PROJECT ARCHITECTURE

1) OBJECT RECOGNITION

As the name suggests, it is the process of predicting the real class or category of an image to which it belongs by making probability high only for that particular class. CNNs are used to efficiently perform this process. Many states of the art Classification and Detection algorithms uses CNN as a backend to perform their tasks

2) IMAGE CLASSIFICATION

The classification model takes an image and slide the kernel/filter over the whole image to get the feature maps. From the feature extracted, it then predicts the label based on the probability.

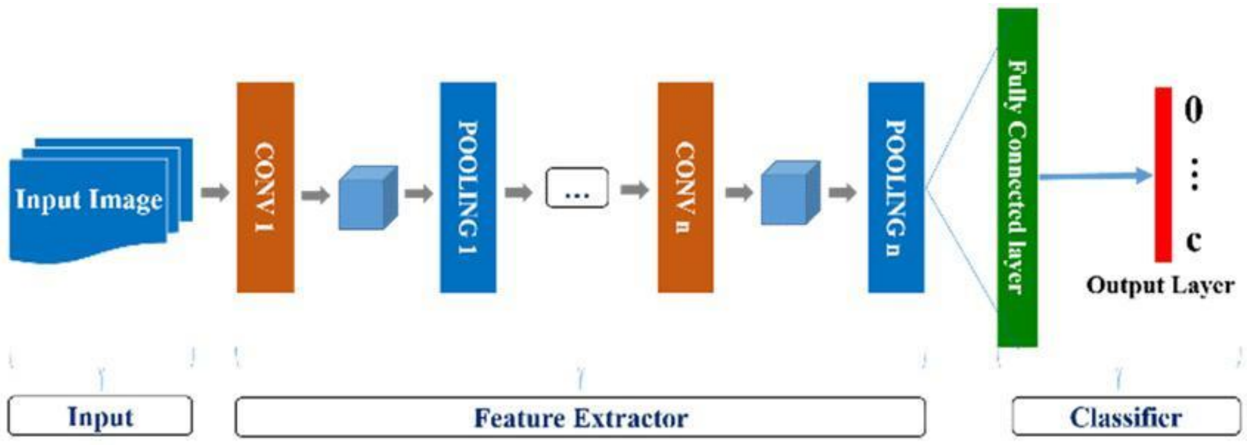


FIG 3.1

4. SYSTEM ANALYSIS

5.1 Feasibility Study

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

A. Economic Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, there is nominal expenditure and economic feasibility for certain.

B. Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following:

- Is there sufficient support for the management from the users?

- Will the system be used and work properly if it is being developed and implemented?

- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So, there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

C. Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?

- Do the proposed equipment's have the technical capacity to hold the data required to use the new system?

- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?

- Can the system be upgraded if developed?

- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web-based user interface for audit workflow at NIC-CSD. Thus, it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source.

5. SYSTEM REQUIREMENT SPECIFICATIONS

5.1 Introduction

A Software Requirements Specification (SRS) – a requirements specification for a software system – is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development life cycle domain, typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- Business requirements describe in business terms what must be delivered or accomplished to provide value.

- Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)

- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify specific methodologies that must be followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, a maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement); a requirement that the product be maintainable (a Product requirement) often is addressed by imposing requirements to follow particular development style

6.1 Purpose

A systems engineering, a requirement can be a description of what a system must do, referred to as a Functional Requirement. This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called non-functional requirements, or 'performance requirements' or 'quality of service requirements.' Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability. A collection of requirements defines the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying how the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality (e.g., of user interfaces) with other owned products. In software engineering, the same meanings of requirements apply, except that the focus of interest is the software itself.

6.2 Functional Requirements

A. Software Requirements

Language	:	Python, Deep Learning.
Frontend	:	Deep Learning.
Backend	:	Python
IDE	:	Jupyter Notebook

OS	:	Windows 8, 10.

B.Hardware Requirements

Processor	:	Intel i3 or Above Hard
Disk	:	100GB or Above
RAM	:	4GB
OS Bit	:	32,64 bits.

6.3 Non-Functional Requirements

A.Usability

The system is designed with completely automated process hence there is no or less user intervention.

B.Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using python, deep learning is more reliable.

C.Performance

This system is developing in the high-level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

D.Supportability

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having python, built into the system.

6. SYSTEM DESIGN

6.1 Introduction

The evolution of convolutional neural network played a vital role in recent times in real-time applications which have been improved a lot in the neural networks and significantly deployed all the features in convolutional network which is having enormous benefits in real-world applications and many sectors utilized the convolutional network in their operating modes such as image classification and video recognition and in section 3.2 it overviews and examines the development of convolutional architectural background and varied networks which were performed and used in deep learning methods which is the important step of the convolutional network for classification and recognition techniques and performed in real-time environment which clearly explained the interaction process of a human brain by activities performed using the weapon activity recognition which is computer-based activity and briefly explained the layers of convolutional network that is used to determine the dimensional procedure and various operations performed by the CNN network. In section 3.3 it briefly explained the parameterized functions which are used and how they trained the model by testing and validating the output by evaluating the validation results and analyzed the various functions by estimating the bias and the weights of the layers in a determined manner. In section 3.4 it described all the recent researches and the real-time applications applied in the analysis of image and video recognition and various sectors.

6.2 CNN Architecture and Background

Convolutional neural network is also referred as shift invariant which is used to analyze visual imagery (Zhang, 2019). Convolutional neural networks possess all characteristic features which are present in neural networks. Convolutional neural network is a neural network mainly used for image classification, natural language processing, image and video recognition, medical image analysis, financial time series, image segmentation, brain-computer interfaces, and image classification (Wang *et al.* 2020). It comprises of one or more neural networks. It consists of Multi-Layer Perceptron's (MLP). CNN has an output layer, input layer, and various hidden layers. Convolutional neural network comes under the branch of deep learning which is used to recognize the biological human brain activity. It is used in computer vision and also used for text classification in natural language processing (Pouyanfar *et al.* 2018).

Convolutional neural network has 20 or 30 layers and there is a special layer in CNN known as convolutional layer. CNN comprises of a lot of convolutional layers which are stacked on the top of each other. CNNs are capable of identifying more sophisticated shapes. It is possible to recognize handwritten digits with three or four convolutional layers and it is possible to distinguish human faces with 25 convolutional layers. The convolutional layers in CNNs reflects the structure of a human visual cortex which has a series of layers which helps in processing the incoming image and in identifying complex features. Convolutional neural network architecture is used to represent semantic images at different levels to extract the better-quality features of CNNs. It has been stated that this method develops a new attribute extractor by fine tuning the CNNs which are used in a large set of natural image data processing (Cengil *et al.* 2017). The convolutional neural networks have been producing extreme improvements in the image processing. By using the CNN architecture, the relationship between the distant pixels can be identified (Hosaka, 2019). The layers present in the convolutional neural network architecture is used to to transform the input volume to an output volume for neuron activation (Voulodimos *et al.* 2018). CNN is a multi-layered neural network which comprises of many hidden layers sequentially stacked on the top of each other. This sequential design helps the convolutional neural networks to learn hierarchical features. The hidden layers in the CNNs are followed by activation layers and pooling layers. A convolutional neural architecture is used to gain an output volume from the input volume by using a differentiable function.

Sequential Layer

Sequential layer consists of a multiple layer. The image is visualized as pixels by the computer. It is used to build a model in kernels is shown in Fig. 7. 1 In the above figure the image is read by the convolutional layer. The convolutional layer scans the image and the features of the image required for the processing is detected by this layer. The required features of the image will be output of the convolutional layer. In polling layer, the pixels of the image are transferred into dimensions. In the flattening layer these dimensions are converted into 1-dimensional array. This array is taken as the input for the dense layer. After performing the dense operations, the required output will be generated. The functioning of the various layers present in the CNN architecture are discussed below.

A. Convolution 3D Layer

It is considered as an important layer in convolutional neural network. The convolutional layer contains a set of kernels in which a small receptive field is present which is extended to the total volume of an input dimensions. The convolutional layer of a CNN basically identifies the fundamental features of an image such as edges and corners. The convolutional layer can be visualized as a small square template which look for patterns in an image. This layer performs the convolutional operations on weights of an image.

B. Max Pooling 3D Layer

This is the second layer present in the convolutional neural network. This operation is carried out on the image after the convolutional operation is performed. By using this operation, the high dimensional input is gradually reduced to low dimensional space by maximizing or averaging. This layer is used to reduce the dimensions of an image. In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling. The Pooling Layer usually serves as a bridge between the Convolutional Layer and the Fully Connected Layer (FC).

C. Global/Fully Connected Layer

Fully connected layer represents the behavior of artificial neural network. It is present in between the convolution layer and dense layer. In this layer each neuron is connected to another neuron of the preceding layer. The fully connected layer or flatten layer serves as a bridge between the convolution and dense layers. This layer is used for converting the data into a 1-dimensional array and this array is provided as the input to the next layer. By connecting multiple convolutional layers and flatten layers the image will be processed for feature extraction.

D.Dense

It is a standard type of layer that is used in many cases for neural networks. It is used in the output layer. It is the most commonly and frequently used layer. It is a fully connected in which all the neurons in this layer are connected to those which are in the next layer. This layer provides the learning characteristics which are obtained from the combinations of the characteristics of the previous layer.

6.3 Different parameters of CNN

CNN parameters are the varied parameters which are used in training and testing the models to evaluate the estimated values of the convolutional network parameters. So, it can be used in the validation process of the model and transfer the data to the training model and these are initialized with random distribution values and parameters to calculate the observed values from the models of the network which are performed in a pre-determined way. The main aim of the parameters performed is to evaluate and separate the data into the training and trained model for the testing and validation processing technique and the evaluated result is applied to different layers of the models used in convolutional network. The estimation process is applied to those classifiers and models used in trained model which is used in separation of the data from the set and applied with parameters for the estimation and examine the weight and biases of the of the convolutional layer. Parameters are mainly used in training the models to examine the particular layer with their sum biases and connection layers which are connected in the convolutional layer which will determine the parameters by testing and training the data. The parameters of the convolutional network can examine and determine the various layers with the input and output calculation (Soon *et al.* 2018).

A.Activation

The activation function which is used in the convolutional network is used to interact with the input and output nodes of the activated function which produces the various activated functions used in the convolutional network. These are used to determine the bias estimated activated function of the nodes and layers of the convolutional network and the main aim of the activation function is to check the non-linearity acceptance of the input and output and used to solve the complicated task and examines the errors occurred in the hidden layer. The Rectified linear unit activation function is the determined function which is comparatively cost effective and can be used for the variant of the activation function and the concept of activate function is used to check whether the layer can be activate or can be used in linearity transformations (Kakuda *et al.* 2019). The automated system of rectified linear unit is the default function mainly used and depends on the forecasted models and the activation function will be same and applied for all the layers of the network. They are different activation function used and the most important and default function is the RELU function.

B.Loss

The loss function is the vital function used in the convolutional network which is used to estimate and determine the values and examine the weights of the layer to estimate the loss value of the function. They are different loss functions used in the estimation and calculation purpose and the functions are mean squared error, binary cross entropy and categorical entropy and sparse categorical entropy which are used in various operating modes and examines the varied estimations. The various loss functions which are used in the operations are performed in regression complex task and also used for binary operating grouping for multi-tasking process. The loss function is used to evaluate the performance of the model which is trained and validated in a pre-defined manner and produced the best possible outcomes.

C.Optimizer

The optimizer is the determined method which is used in training the model into a definite validate model using optimization method by using the algorithm methods for estimation of biases which are used to minimize the loss functions by algorithm methods. There are two optimizing functions used in the optimization process which are used by the gradient function which are used in the data sets for training the model and minimize the errors by performing the optimization process in an effective manner. The gradient descent algorithm is performed with the model parameters for the estimation and determining the errors to train and validate the model to examine the best results.

D. Matrices

A machine can detect only computerized method which represents in mathematical language so it is easily used in detecting the operations of the layers. The convolutional network which is the representation of the matrix used to evaluate and examine the layers by its matrix' operations depending on the input and output representation and convolutional network is the layered representation of the neural network. The matrices operation involved in convolutional layer is the weight matrix used to estimate and derive the extracted information from the operating modes and the weights are determined in prediction models. The kernel is the extractor which is used to examine the information from the input and kernel is a matrix used to multiply the output in a determined manner and dimensional matrix operations are performed using the kernel filters.

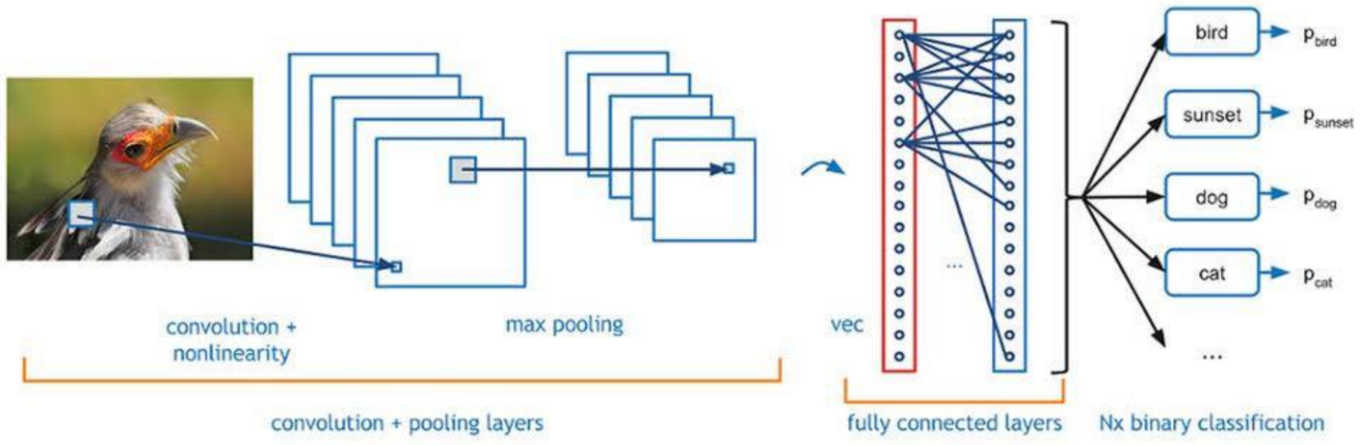


FIG.8.3.1

7.TECHNOLOGY DESCRIPTION AND IMPLEMENTATION

7.1 Introduction

Implementation is the stage where the theoretical design is turned in to working system. The most crucial stage is achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively. The system can be implemented only after through testing is done and if it found to work according to the specification. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover and an evaluation of change over methods a part from planning. Two major tasks of preparing the implementation are education and training of the users and testing of the system. The more complex the system being implemented, the more involved will be the systems analysis and design effort required just for implementation. The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The System may require some hardware and software acquisition is carried out.

The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued. Implementation is the process of having systems personnel check out and put new equipment in to use, train users, install the new application, and construct any files of data needed to it. Depending on the size of the organization that will be involved in using the application and the risk associated with its use, system developers may choose to test the operation in only one area of the firm, say in one department or with only one or two persons. Sometimes theywill run the old and new systems together to compare the results. In still other situations, developers will stop using the old system one-day and begin using the new one the next. As we will see, each implementation strategy has its merits, depending on the business situation inwhich it is considered. Regardless of the implementation strategy used, developers strive to ensure that the system’s initial use in trouble-free.

7.2 Technology Description

About the Google Collab Workspace

Google is quite aggressive in AI research. Over many years, Google developed AI framework called TensorFlow and a development tool called Collaboratory. Today TensorFlow is open-sourced and since 2017, Google made Collaboratory free for public use. Collaboratory is now known as Google Collab or simply Collab. Another attractive feature that Googles offers to the developers is the use of GPU. Collab supports GPU and it is totally free. The reasons for making it free for public could be to make its software a standard in the academics for teaching machine learning and data science. It may also have a long-term perspective of building a customer base for Google Cloud APIs which are sold per-use basis. Irrespective of the reasons, the introduction of Collab has eased the learning and development of machine learning applications.

About the Python Technology

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects (Kuhlman, 2012) Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0 (Rossum, 2009). Python

2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting. Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020 (Peterson, 2020) Python consistently ranks as one of the most popular programming languages. The complete hierarchical framework is shown in Fig .8.1. It consists of different types of datatypes, sequences, set types, mappings, callable and modules are detailly shown in the hierarchical.

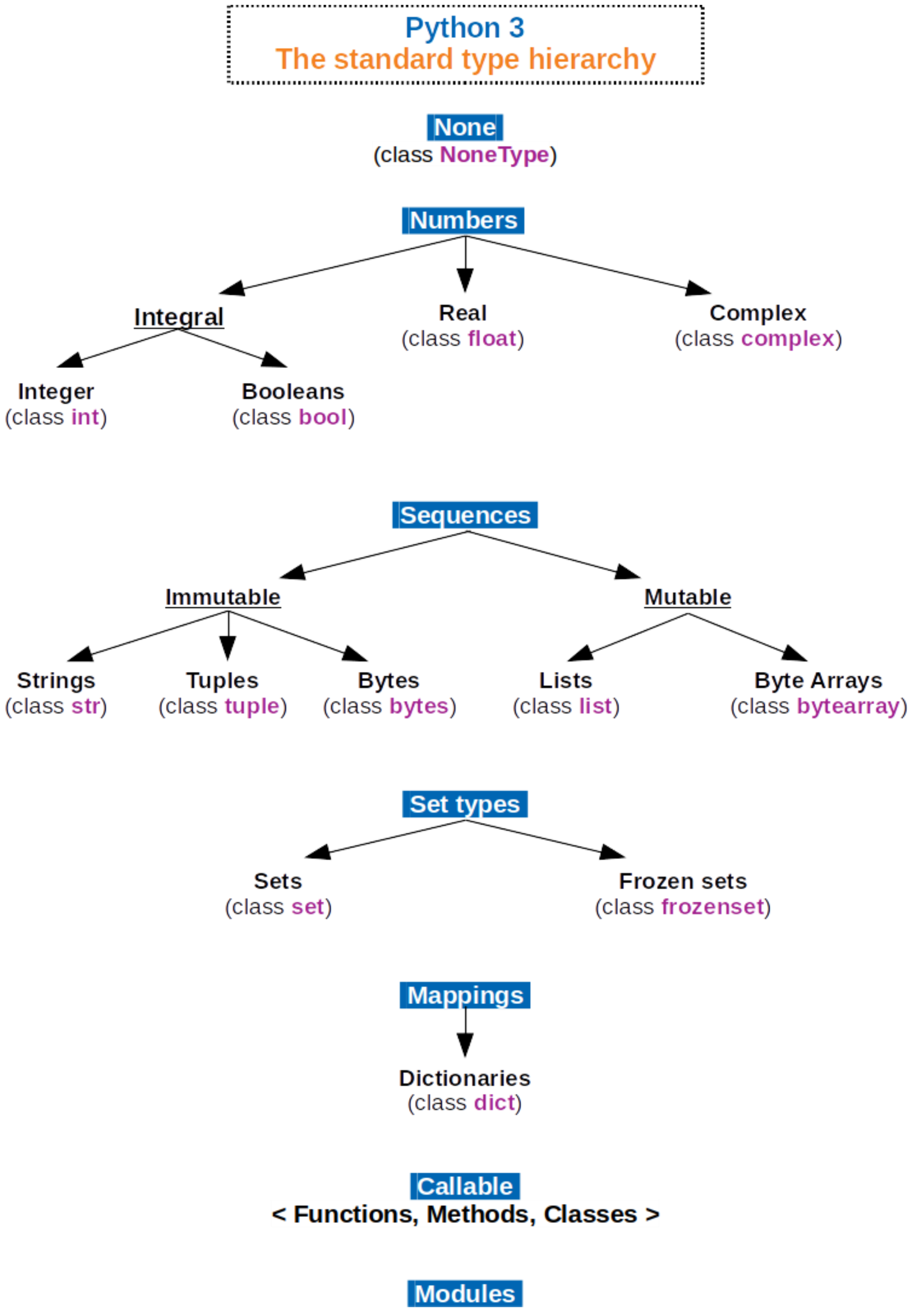


FIG 7.2.1

8. UML DIAGRAMS

8.1 Introduction to UML

The unified Modeling Language (UML) is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct and document the artifacts of software-intensive system. The goal of UML is to provide a standard notation that can be used by all object - oriented methods and to select and integrate the best elements. UML is itself does not prescribe or advice on how to use that notation in a software development process or as part of an object - design methodology. The UML is more than just bunch of graphical symbols. Rather, behind each symbol in the UML notation is well-defined semantics.

The system development focuses on three different models of the system.

- Functional model
- Object model
- Dynamic model

Functional model in UML is represented with use case diagrams, describing the functionality of the system from user point of view.

Object model in UML is represented with class diagrams, describing the structure of the system in terms of objects, attributes, associations and operations.

Dynamic model in UML is represented with sequence diagrams, start chart diagrams and activity diagrams describing the internal behavior of the system.

8.2 Types of UML diagrams

The different types of UML diagrams are shown below.

DEPLOYEMENT DIAGRAM:

In UML, deployment diagrams model the physical architecture of a system. Deployment diagrams show the relationships between the software and hardware components in the system and the physical distribution of the processing. Communication paths and deploy relationships model the connections in the system is shown in Fig. 9.1. The main purpose of the deployment diagram is to represent how software is installed on the hardware component. It depicts in what manner a software interacts with hardware to perform its execution. The parts of deployment are listed below.

- Nodes: A node represents any hardware component.
- Components: A component represents software.
- Dependencies: The reliability of one component with that of another is depicted by dependencies.
- Links: To tie up tow nodes, the links are utilized.

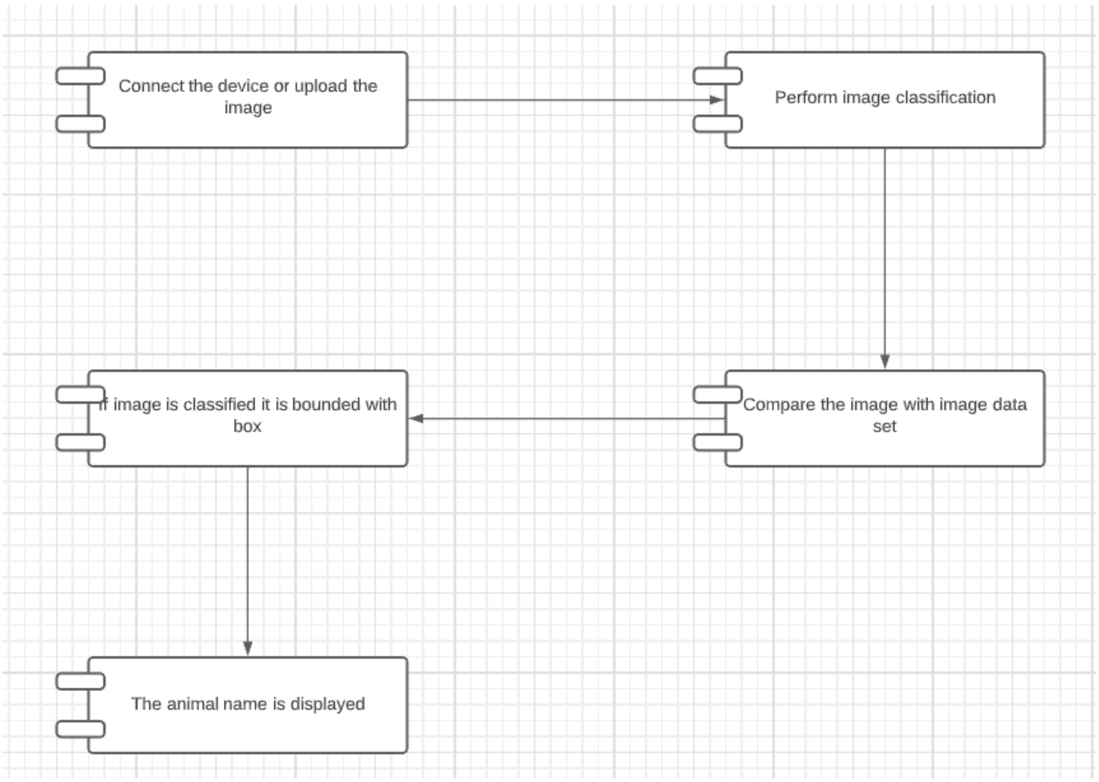

Connect the device or upload the image
Perform image classification
Compare the image with image data set
If image is classified it is bounded with box
The animal name is displayed

FIG 8.2.1

ACTIVITY DIAGRAM:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system is shown in Fig.

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent.

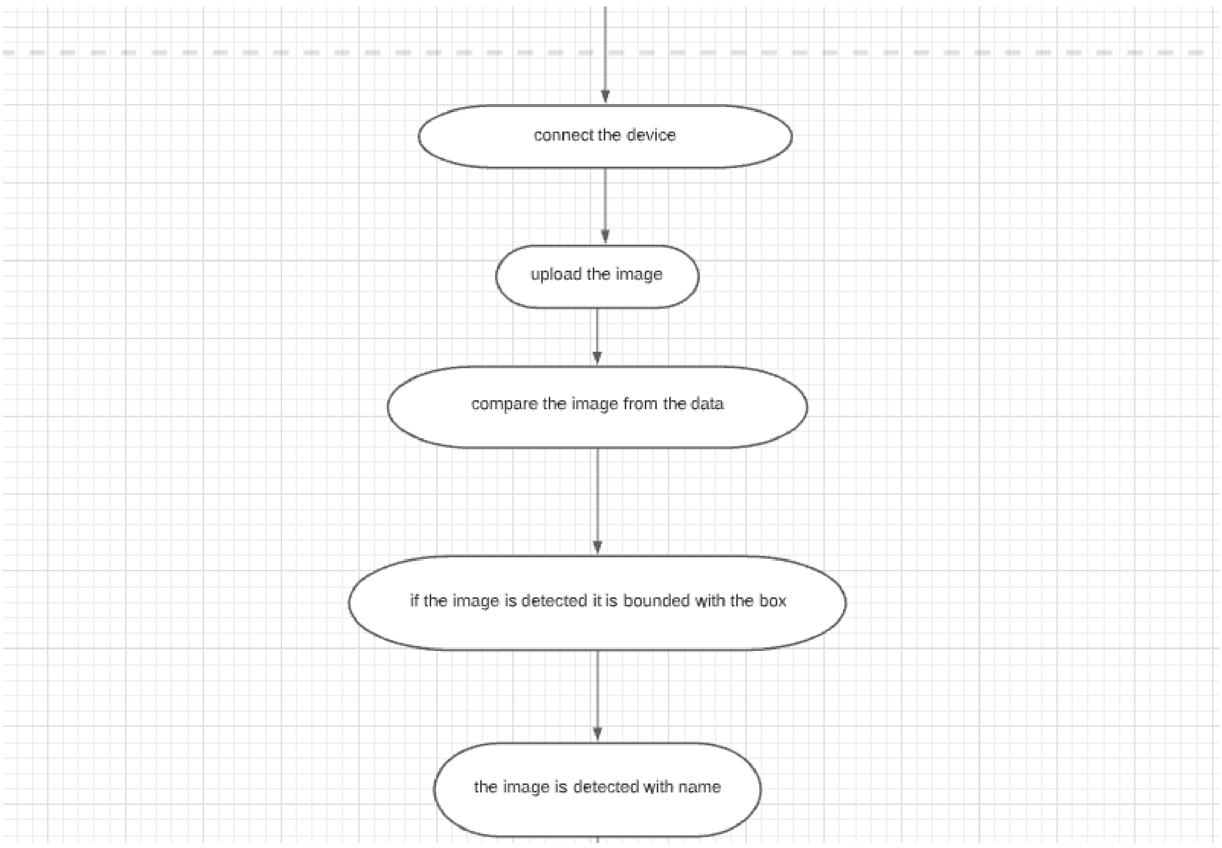

connect the device
upload the image
compare the image from the data
if the image is detected it is bounded with the box
the image is detected with name

FIG 8.2.2

E-R DIAGRAM:

UML stands for Unified Modelling Language. ER Diagram stands for Entity Relationship Diagram. It is a general modelling language which is used to visualize the design of a software system is shown in Fig. 9.3. An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

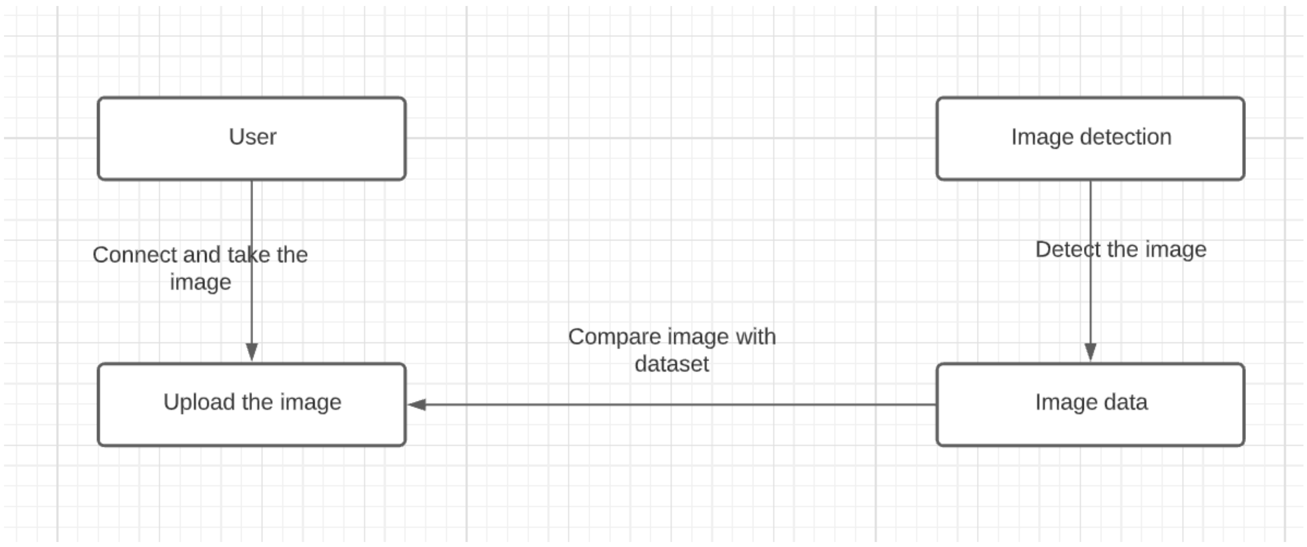


FIG 8.2.3

FLOWCHART DIAGRAM:

Flowcharts depict certain aspects of processes and are usually complemented by other types of diagrams. Similarly, in UML, a standard concept-modeling notation used in software development, the activity diagram, which is a type of flowchart, is just one of many different diagram types is shown in Fig. 9.4. UML stands for Unified Modeling Language. An activity diagram is a UML diagram. A flowchart, on the other hand, is a graphical diagram that represents an algorithm. Flow charts are simple diagrams that map out a process, so that you can easily communicate it to other people. You can also use them to define and analyze a process, build a step-by-step picture of it, and then standardize or improve it.

The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields. There are different types of flowcharts: each type has its own set of boxes and notations. The two most common types of boxes in a flowchart are.

- A processing step, usually called activity, and denoted as a rectangular box.

- A decision, usually denoted as a diamond.

●

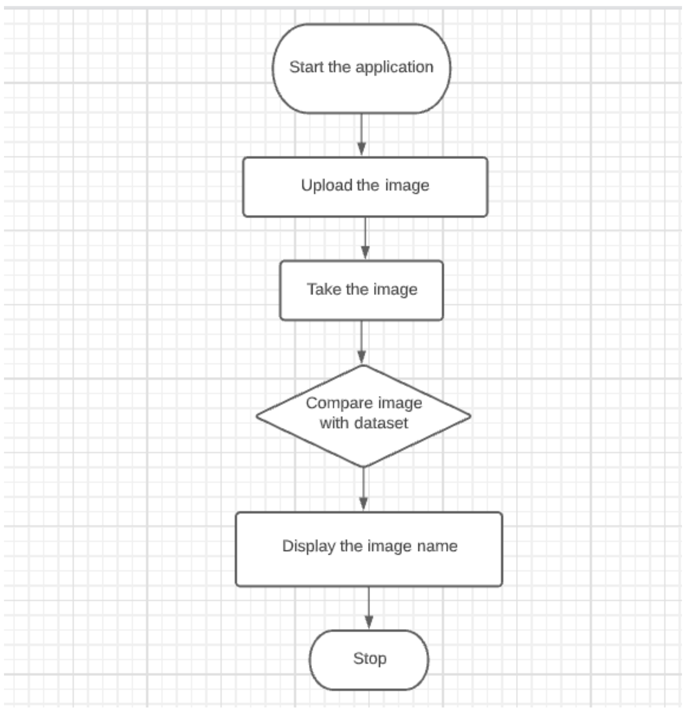


FIG 8.2.4

USE CASE DIAGRAM:

Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use case focus on the behavior of the system from an external point of view. The identification of actors and use cases results in the definition of the boundary of the system, which is, in differentiating the tasks accomplished by the system and the tasks accomplished by its environment. The actors are outside the boundary of the system, whereas the use cases are inside the boundary of the system is shown in Fig. 9.5. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

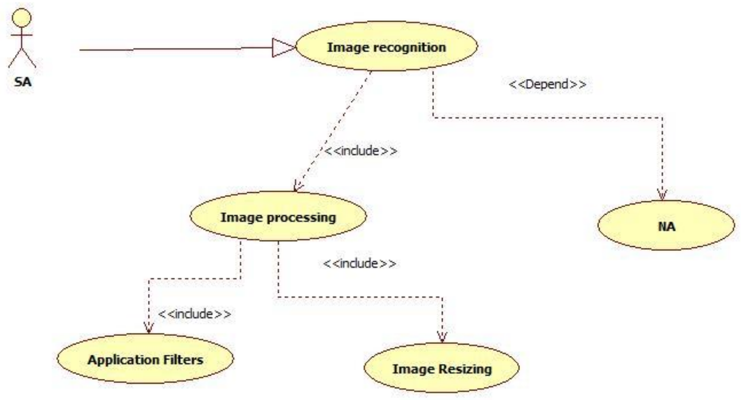


FIG 8.2.5

STATE CHART DIAGRAM:

A state diagram, sometimes known as a state machine diagram, is a type of behavioral diagram in the Unified Modeling Language (UML) that shows transitions between various objects. Using our collaborative UML diagram software, build your own state machine diagram with a free Lucid chart account today is shown in Fig. 9.7. State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of State chart diagram is to model lifetime of an object from creation to termination. Specifically, a state diagram describes the behavior of a single object in response to a series of events in a system. Sometimes it's also known as a Harel state chart or a state machine diagram. This UML diagram models the dynamic flow of control from state to state of a particular object within a system.

A class diagram shows classes in their relation and their properties and methods. A state diagram visualizes a class's states and how they can change over time. In both cases you are talking about diagrams which are only a window into the model. The class relations define how the single classes relate to each other

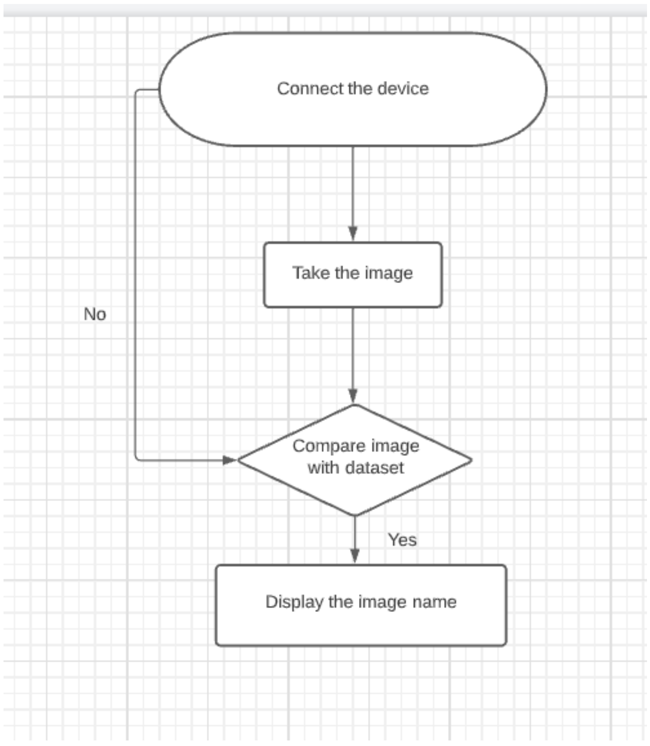


FIG 8.2.6

CLASS DIAGRAM:

Class Diagrams are used to describe the structure of the system. Classes are abstractions that specify the common structure and behavior of a set of objects. Objects are instances of classes that are created, modified and destroyed during the execution of a system. An object has state that includes the values of its attributes and links with other objects. The class diagram is used to refine the use cases diagrams and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the classes. Apart from this, each class may have certain "attributes" that uniquely identify the class. In the class diagram these classes are represented with boxes which contain three parts is shown in Fig. 8.2.7.

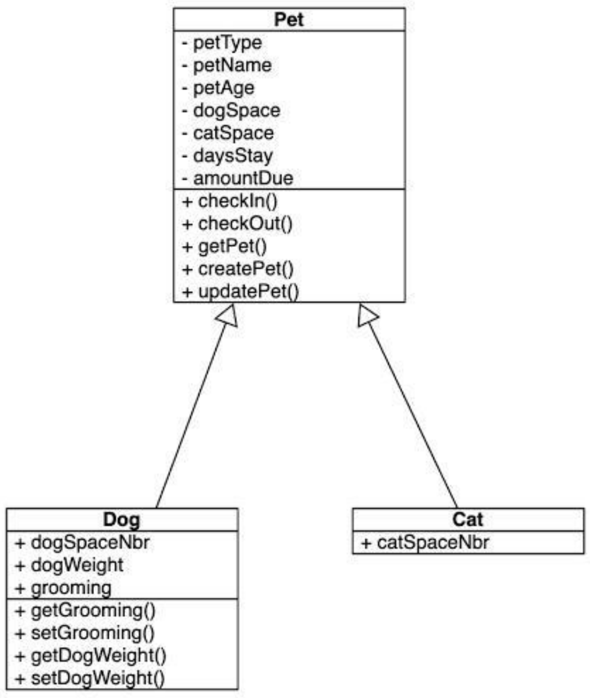


FIG 8.2.7

COMPONENT DIAGRAM:

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node is shown in Fig. 9.9.

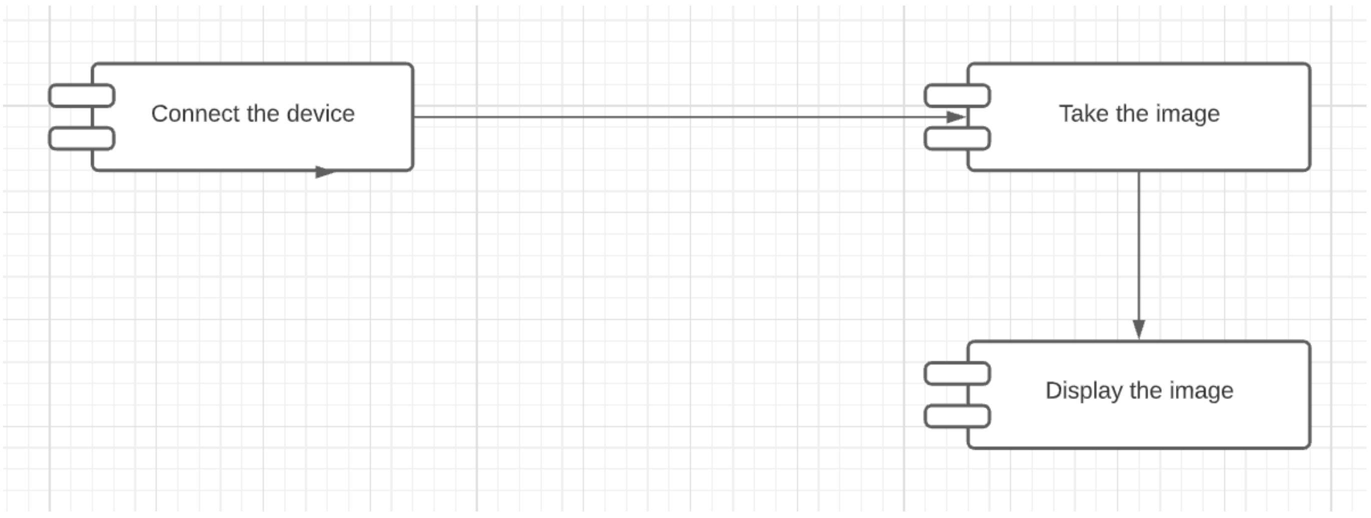


FIG 8.2.8

9. MODULES

9.1 Preparing Training Data

- We need to import numpy, pandas, keras, matplotlib.
- We need to load cats and dogs images from kaggle.
- A dataset is divided into training dataset and test dataset.
- In the training dataset we have 4,000 images of cats and 4,000 images of dogs
- In the test dataset we have 1,000 images of cats and 1,000 images of dogs

9.2 Build CNN model

A.CONVOLUTION: Convolutional layers are the layers where filters are applied to the original image, or to other feature maps in a deep CNN. This is where most of the user-specified parameters are in the network. The most important parameters are the number of kernels and the size of the kernels.

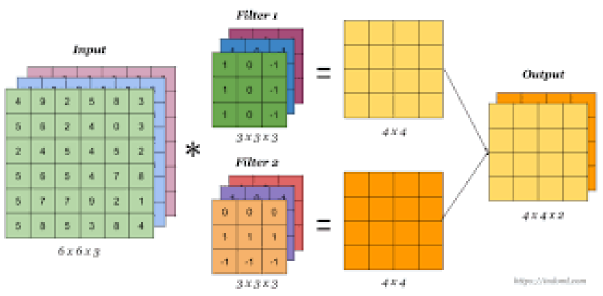


FIG 9.2.1

B.POOLING:Pooling layers are similar to convolutional layers, but they perform a specific function such as max pooling, which takes the maximum value in a certain filter region, or average pooling, which takes the average value in a filter region. These are typically used to reduce the dimensionality of the network.

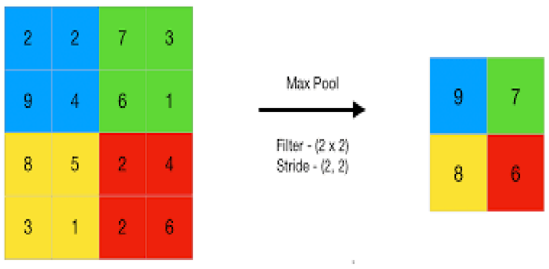


FIG 9.2.2

C.FULLY CONNECTED:Fully connected layers are placed before the classification output of a CNN and are used to flatten the results before classification. This is similar to the output layer of an MLP.

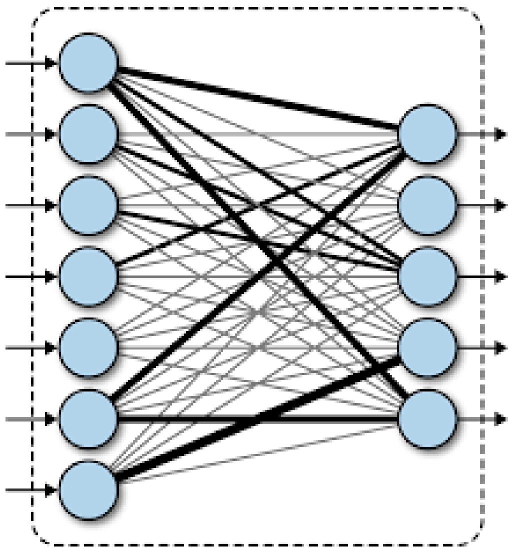


FIG 9.2.3

9.3 **Make predictions**

- We can use our saved model to make predictions on new images.
- First we load an image
- Then we resize it to a predefined size such as 224 x 224 pixels.
- Then scale the value of the pixels to the range [0,225].
- Then select a pre-trained model.
- Then run the pre-trained model.
- Display the result

10. SOURCE CODE

```
import tensorflow as tf
import scipy
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)
train_generator = train_datagen.flow_from_directory(
        r'C:\dogscats\dogscats\train',
        target_size=(64, 64),
        batch_size=32,
        class_mode='binary')
test_datagen = ImageDataGenerator(rescale=1./255)
validation_generator = test_datagen.flow_from_directory(
        r'C:\dogscats\dogscats\train',
        target_size=(64, 64),
        batch_size=32,
        class_mode='binary')
cnn = tf.keras.models.Sequential()
cnn.add(tf.keras.layers.Conv2D(filters = 32, kernel_size = 3, activation = 'relu',
input_shape = [64,64,3]))
cnn.add(tf.keras.layers.MaxPool2D(pool_size = 2,strides = 2))
cnn.add(tf.keras.layers.Flatten())
cnn.add(tf.keras.layers.Dense(units = 128, activation = 'relu'))

cnn.add(tf.keras.layers.Dense(units = 1, activation = 'sigmoid'))

cnn.compile(optimizer = 'adam',loss = 'binary_crossentropy',metrics = ['accuracy'])

history = cnn.fit(x =train_generator,validation_data = validation_generator, epochs = 2)

import numpy as np

from keras.preprocessing import image

test_image = image.load_img("Desktop/dataset/single_prediction/predict1.jpg",target_size
=(64,64))

test_image = image.img_to_array(test_image)

test_image = np.expand_dims(test_image, axis = 0)

result = cnn.predict(test_image)

train_generator.class_indices

if result[0][0] == 1:

    prediction = 'dog'

else:

    prediction = 'cat'

prediction

import matplotlib.pyplot as plt

print(history.history.keys())

plt.plot(history.history['accuracy'])
```

10.TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software manner. There are various types of test. Each test type addresses a specific testing requirement.

10.1 TESTING METHODOLOGIES:

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing: Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit.

Integration Testing: Integration testing addresses the issues associated with the dual problems of verification and program construction.

User Acceptance Testing: User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.

Output Testing: output testing is a type of software testing whereby the system is tested against the functional requirements/specifications. Functions (or features) are tested by feeding them input and examining the output. ... Determine the output based on the function's specifications. Execute the test case.

Validation Testing: Validation testing is the process of ensuring if the tested and developed software satisfies the client /user needs. The business requirement logic or scenarios have to be tested in detail. All the critical functionalities of an application must be tested here.

10.2 INTEGRATION TESTING:

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

Top Down Integration:

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner. In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

Bottom-up Integration:

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.

10.3 ACCEPTANCE TESTING:

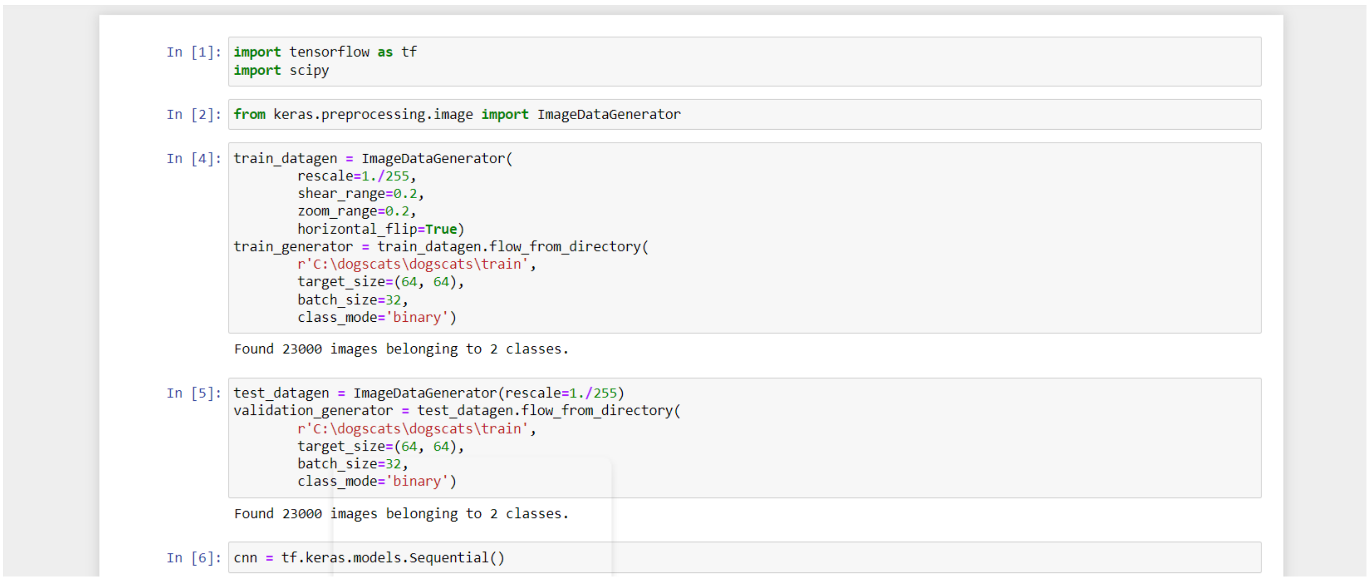
Acceptance testing, a testing technique performed to determine whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system's compliance with the business requirements and verify if it is has met the required criteria for delivery to end users.

There are various forms of acceptance testing:

- User acceptance Testing
- Business acceptance Testing
- Alpha Testing
- Beta Testing

11. SCREENSHOTS

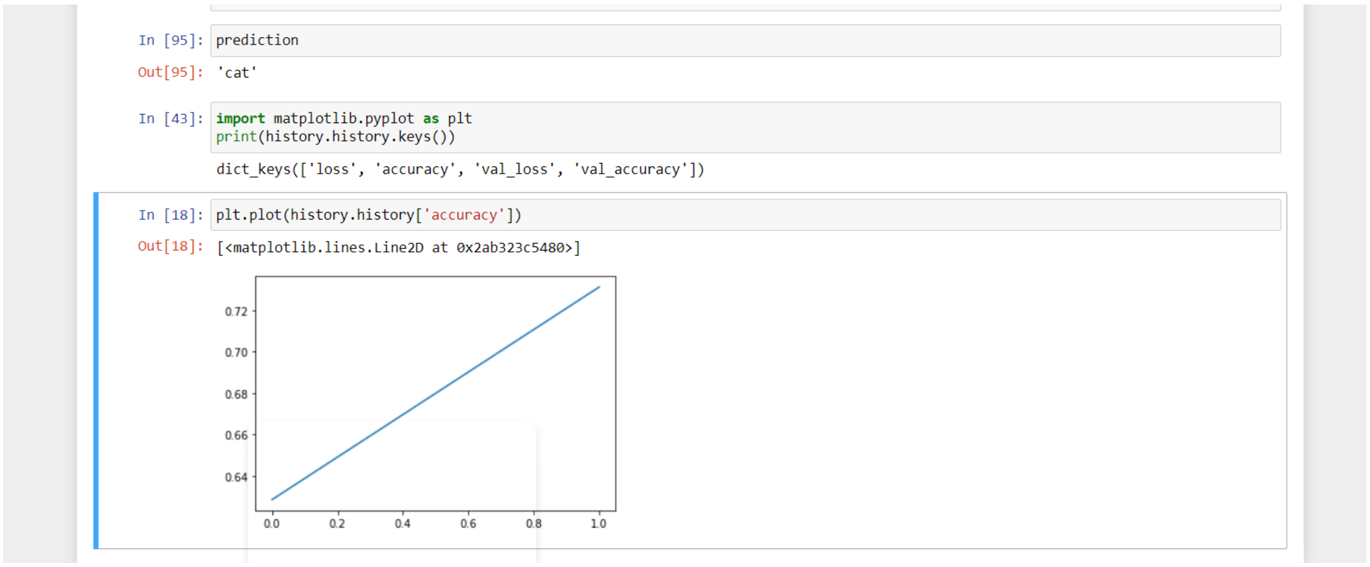
11.1 INPUT SCREENSHOTS



```
In [1]: import tensorflow as tf
        import scipy

In [2]: from keras.preprocessing.image import ImageDataGenerator

In [4]: train_datagen = ImageDataGenerator(
                rescale=1./255,
                shear_range=0.2,
                zoom_range=0.2,
                horizontal_flip=True)
        train_generator = train_datagen.flow_from_directory(
                r'C:\dogscats\dogscats\train',
                target_size=(64, 64),
                batch_size=32,
                class_mode='binary')

        Found 23000 images belonging to 2 classes.

In [5]: test_datagen = ImageDataGenerator(rescale=1./255)
        validation_generator = test_datagen.flow_from_directory(
                r'C:\dogscats\dogscats\train',
                target_size=(64, 64),
                batch_size=32,
                class_mode='binary')

        Found 23000 images belonging to 2 classes.

In [6]: cnn = tf.keras.models.Sequential()
```

FIG 11.1

11.2 OUTPUT SCREENSHOTS



```
In [95]: prediction
Out[95]: 'cat'

In [43]: import matplotlib.pyplot as plt
         print(history.history.keys())

         dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

In [18]: plt.plot(history.history['accuracy'])
Out[18]: [<matplotlib.lines.Line2D at 0x2ab323c5480>]
```

FIG 11.2

12.EXAMPLES

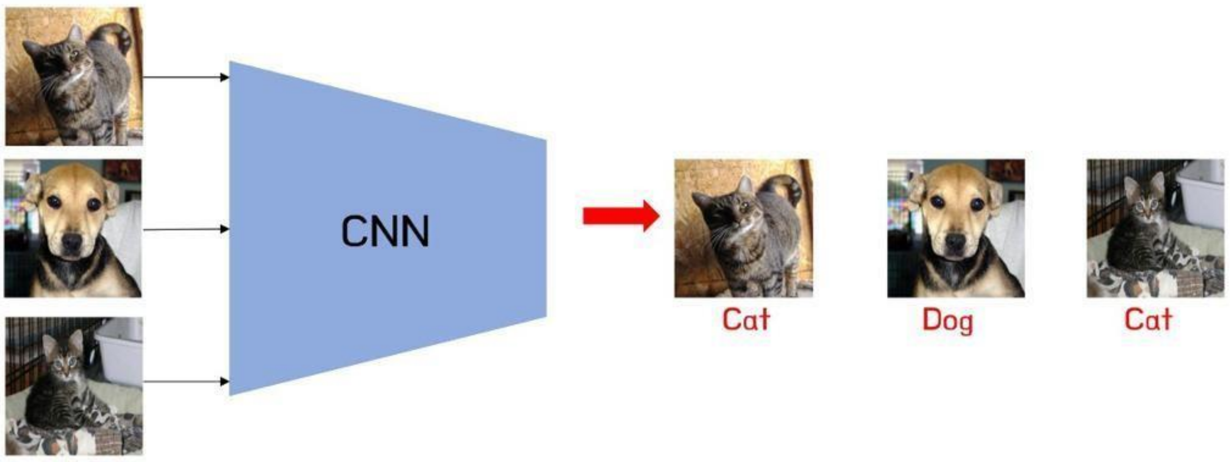
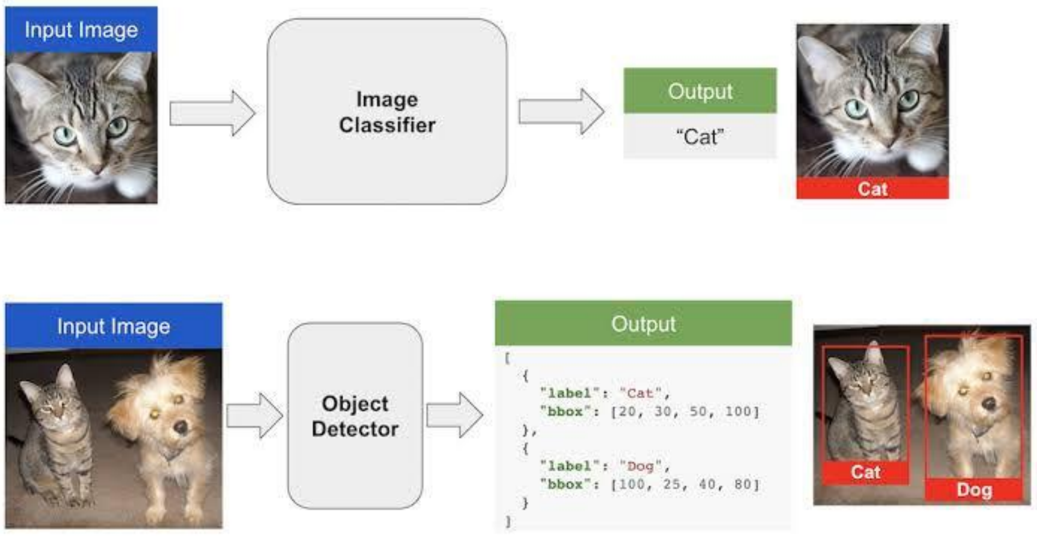


FIG 12.1



```
[
  {
    "label": "Cat",
    "bbox": [20, 30, 50, 100]
  },
  {
    "label": "Dog",
    "bbox": [100, 25, 40, 80]
  }
]
```

FIG 12.2

13.CONCLUSION AND FUTURE REMARKS

In this chapter the conclusion of the evaluation of the frameworks is provided and motivated, as well as suggestions for future work, and finally some concluding remarks by the authors.
The main goal of this project was to evaluate the two deep learning frameworks Google TensorFlow and Microsoft CNTK, primarily based on their performance in the training time of neural networks. In the aforementioned aspect CNTK performed better, than TensorFlow with Keras as frontend, see chapter 5.3 for a more detailed presentation of the benchmarking results. We chose to use the third-party API Keras instead of TensorFlow's own API when working with TensorFlow, we made this choice because we found TensorFlow's own API too cumbersome to work with, given the project's time span and our inexperience in the field of machine learning when the project began. When using Keras with TensorFlow as backend we found the development process in it to be easy and intuitive, see chapter 5.2 for our more detailed opinions on the frameworks.
In conclusion; even though CNTK performed better on the benchmarking tests, we found Keras with TensorFlow as backend to be much easier and more intuitive to work with, two aspects we think are more important when choosing a deep learning framework to work with. In addition; the fact that CNTKs underlying implementation of the machine learning algo- rithms and functions differ from that of the literature and of other frameworks, makes the development process tedious if you, like us, have just studied the literature and are new to machine learning. Therefore; based on the reasons just mentioned, if we had to choose a framework to continue working in, we would choose Keras with TensorFlow as backend, even though the performance is less compared to CNTK.

13.1 FUTURE WORKS

Regarding the possibility of future work in this area we found four interesting areas to explore, evaluate and to work on further development in: data gathering and preparation, integration with other applications, further exploration of CNTK and production environments.
The quantity and the quality of data available is one of the most critical factors, if not the most critical, influencing the performance of a trained model. The first step in getting the necessary quantity of data is to gather the raw data, a process that, all the practical difficulties aside, will take a considerable amount of time. The practical setup of the data gathering process would necessitate developing tools for automatic gathering of data, tools that would have to specialized depending on the kind of data required, e.g. production data from different processes in a paper mill. The raw data would then have to be cleansed, labeled and divided into training and test sets, a process that will entail additional costs in terms of time

and money. The legal and business aspects of the process would also be needed to be taken into account. Despite the work needed here we think that working with acquiring data could become an interesting project in it’s own right.

The question of how to integrate the finished, trained models into applications for practical use is a question we found interesting, but did not have time for in the project. One major obstacle here would be how make the predictions or the classifications of the model as fast as possible, a obstacle that must be overcome with both the proper hardware and the proper software. The application must in other words perform the chosen task, e.g. finding faces in pictures, without too much delay. The idea of developing such an application, with all the architectural choices, setup of hardware and other aspects that such a development process would entail, is an idea that we both found interesting. The topic of production environments is closely related to that of developing an application, but merits discussion in its own right. Having high-performing hardware is key to making the training and serving of models feasible with regards to time and latency, a fact we learned the hard way during the project. The hardware that is needed, particularly high-end GPU:s, is often too expensive to justify owning and administrating it yourself. There are platforms that could be used to lease or purchase the necessary computing power, platforms offering virtual machines for that purpose. Setting up an environment for production, development and research with the proper resources is both something we would like to do and could not do during the time span of the project.

The final area of potential future work, using CNTK in more depth, had to be cut from the project due to time constraints. Making CNTK work as well for us as we made Keras and Tensorflow do in implementing more advanced functionalities could be a good learning experience in several aspects. Learning to implement the building blocks behind deep learning and neural networks in several frameworks might be knowledge worth having since both the field at large and the development tools change rapidly. The frameworks that are being used for development today could be obsolete tomorrow - it is good to be prepared.

13.2 CONCLUDING REMARKS

Neural networks and deep learning is an area of research with a lot of unanswered questions, an area with such intensive research that it can be difficult to keep pace with all the new findings that are discovered constantly. A practitioner need to be well acquainted with the underlying theory to able to work with neural networks and deep learning and also need to be familiar with their building blocks to be able to use them effectively. A very important thing to emphasize here that in the process of developing machine learning applications the quality of the available data, not just the quality of the code, affects the end quality of the product, unlike other areas of software development. The performance of a machine learning model can almost always be improved with more and better data.

A final and perhaps uplifting remark is that not a lot of code is needed to develop a working prototype with the use of modern deep learning frameworks. The time saved due to the ease of scripting and designing models in these frameworks is valuable, especially when the time needed to tune and train the model is taken into account. Since a lot of the work that is involved with machine learning consists of testing, validating and discarding multiple hypotheses and ideas the turnaround time while developing the model should be as small as possible, and good frameworks help with that.

14.REFERENCES

[1] Golle, P. (2008, October). Machine learning attacks against the Asirra CAPTCHA. In Proceedings of the 15th ACM conference on Computer and communications security (pp. 535-542). ACM.

[2] J. Elson, J. Douceur, J. Howell and J. Saul. Asirra: a CAPTCHA that exploits interest-aligned manual image categorization. Proc. of ACM CCS 2007, pp. 366-374.

[3] Muthukrishnan Ramprasath, M.Vijay Anand and Shanmugasundaram Hariharan, Image Classification using Convolutional Neural Networks. International Journal of Pure and Applied Mathematics, Volume 119 No. 17 2018, 1307-1319

[4] Parkhi, O. M., Vedaldi, A., Zisserman, A., & Jawahar, C. V. (2012, June). Cats and dogs. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 3498-3505). IEEE.

[5] Zeiler, M. D., & Fergus, R. (2013). Visualizing and Understanding Convolutional Neural Networks. arXiv preprint arXiv:1311.2901.

[6] Bang Liu, Yan Liu, Kai Zhou “Image Classification for Dogs and Cats”.

16.RECOMMENDATIONS

Image classification is a complex process that may be affected by many factors. This paper examines current practices, problems, and prospects of image classification. The emphasis is placed on the summarization of major advanced classification approaches and the techniques used for improving classification accuracy. In addition, some important issues affecting classification performance are discussed. This literature review suggests that designing a suitable image-processing procedure is a prerequisite for a successful classification of remotely sensed data into a thematic map. Effective use of multiple features of remotely sensed data and the selection of a suitable classification method are especially significant for improving classification accuracy. Non-parametric classifiers such as neural network, decision tree classifier, and knowledge-based classification have increasingly become important approaches for multisource data classification. Integration of remote sensing, geographical information systems (GIS), and expert system emerges as a new research frontier. More research, however, is needed to identify and reduce uncertainties in the image-processing chain to improve classification accuracy

17.SELF-ASSESSMENT

Finally, I learn that there is no right or wrong in our application design, and there is no perfect way to achieve a design solution. The only way to reach a design goal and perfect our product is through iteration and incorporation of user feedback. Iteration makes my design get better and better, and each iteration keeps me focused on the problem I want to solve. Input from design critique makes me look at the problem from different directions and helps shape a better product.

Questioning: My question is clear, well-focused and requires high level thinking skills in order to research.

Planning: I made really good use of my time. I was able to remain focused on the tasks and make changes when I needed to. I was able to develop a clear method to organize my information. I was able to make revisions in my plan when needed.

Gathering: I used a variety of resources and carefully selected only the information that answered my question. I was able to continually revise my search based on information I found.

Sorting: I thoroughly selected and organized information that answered my question in a organized way. I selected information that was appropriate.

Synthesizing: My product answers the question in a way that reflects learning using some detail and accuracy.

18.APPENDIX

"C:\Users\pc\Downloads\SAMPADA REVIEW (1).pptx"