1

औद्योगिक प्रशिक्षण के लिए राष्ट्रीय संस्थान

National Institute for Industrial Training

One Premier Organization with Non Profit Status | Registered Under Govt. of WB
Empanelled Under Planning Commission Govt. of India
Inspired By: National Task Force on IT & SD Government of India

National Institute for Industrial Training- One Premier Organization with Non Profit Status Registered Under Govt.of West Bengal,Empanelled Under Planning Commission Govt.of India , Empanelled Under Central Social Welfare Board Govt. of India , Registered with National Career Services , Registered with National Employment Services.

PYTHON WITH ARTIFICIAL INTELLIGENCE

Subject: Driver Drowsiness detection system
Submitted By: Arkadeep Basu,Ambarish Ganguly
Email: basuarkadeep2001@gmail.com ,
Submitted To: Soumotanu Mazumdar

2

STUDENT PROFILE

Name: Arkadeep Basu
College: Kalinga Institue of Industrial technology,Bhubaneswar
Department: Aerospace engineering
Year: 3rd year (5th Semester)
Email: basuarkadeep2001@gmail.com
LinkedIn: https://www.linkedin.com/in/arkadeep-basu-431604187
Project Topic: driver drowsiness detection system

Acknowledgment

In the accomplishment of this project successfully, many people have bestowed upon us their blessings and heart pledged support, this time we are utilising this oppurtunity to thank all the people who have been concerned with this project. I would express my gratitude to my supervisor Mr Soumotanu Mazumdar of National Institute for Industrial Training who gave us the golden opportunity to do this project and guided me and gave constant supervision as well as provided necessary information regarding the project. It helped us a lot in doing a lot research related to this topic and we came to know about various new things. we would also like to thank our parents for their encouragement and kind co-operation which helped us a lot in completion of the project.

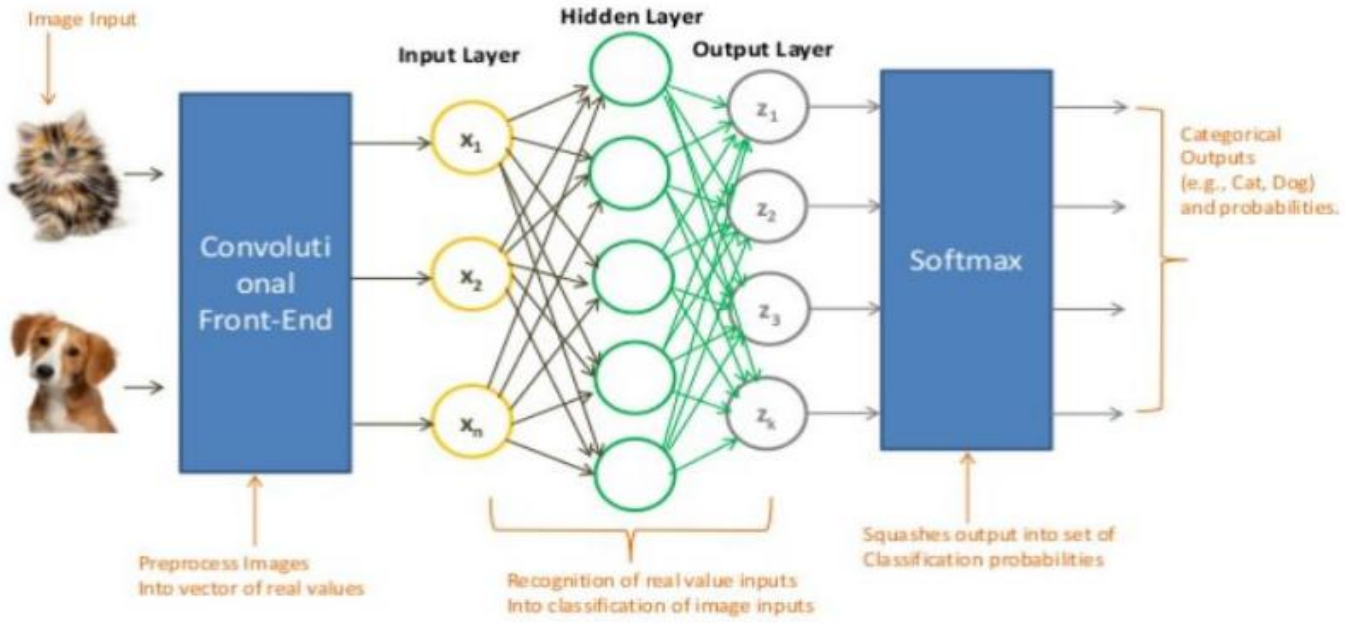
Content

Abstract :

This project uses computer vision and machine learning techniques to predict the correct dog breed from their images. This deals with a fine grained image-recognition problem that have multi-class classification, which determines whether the eyes are closed or open in an given image. In this exertion, we are using convolutional neural networks.

INTRODUCTION:

In machine learning, convolution Neural Networks (CNN) are complicated feed forward neural networks. CNNs are used for image classification and recognition due to its high accuracy. CNN follows a hierarchal model that specializes in processing data that has a grid like topology, such as an image. A digital image is a binary representation of visual data. It contains a series of pixels arranged in a grid like fashion that contains pixel values to denote the colour of each pixel. We present experiments on source and accuracy trade-offs and exhibits live achievements in contrast to different appreciated models on ImageNet classification.



Convolution Neural Networks. Convolution is a front end for a Neural Network for image classification.

5

MOTIVATION

This project hopes to identify the eyes and classify them as close or open. This is a fine grained classification problem.we know that road accidents are pretty common in india and more so drowsiness related accidents are very common.About 167 people in india alone die each yer due to sleep related car accidents.so this project aims to detect whether a person is feeling sleepy and if the eyes are closed for a stipulated amount of time and the score becomes more than 15 then an alarm rings thereby alerting the driver and preventing the accident.The alarm does not stop ringing until the score becomes less than 15.This system will help prevent accidents .Also, I hope to expand my knowledge in this type of fine-tuned classification problems.

OBJECTIVE

In this paper our aim is to implement Image Classification with Deep learning and Convolution Neural Network using Tensorflow.
Different techniques has been applied for this purpose and a comparative study has been made between different accuracies shown by different models.
The steps involved are:

- DATA MINING: It is a process of extracting and discovering pattern in large dataset involving intelligent methods to transform the information to a comprehensive structure for further use. Computers can not only acknowledge pictures, but they additionally describe the assorted parts in photos and create transient sentences. This is done by convolution neural network that learns pattern that occur in pictures.
- EXPLORATORY DATA ANALYSIS: This is used by data scientist to analyse and investigate dataset and summarize their main characteristics, often using data visualization methods and graphical methods.
- PRE PROCESSING: It is a data mining technique that involves transforming raw data into understandable format. Real world data is often incomplete, lacking certain behaviour, and is likely to contain many errors. Data pre-processing is proven method to solve these problems.
- DIVING THE DATSET INTO TRAINING AND TESTING SET: We partition the data into two portion usually for cross validation purpose. One portion of the data is used to develop the predictive model and the other part is used to evaluate the model.
- APPLYING VARIOUS MODELS: We also use neural networks from scratch and transfer learning which is using some pre-trained models on a new problem, to train our model. This exploits the knowledge gained from a previous task to improve

generalization about another.

DATASET

The Model used by me (cnncat.h5) is a deep learning model which used over 10000 pictures of closed and open eyes to train.I do not have a pc of good confguration to train data to a deep learning model so I have used this model.

A convolutional neural network is a special type of deep neural network which performs extremely well for image classification purposes. A CNN basically consists of an input layer, an output layer and a hidden layer which can have multiple layers. A convolution operation is performed on these layers using a filter that performs 2D matrix multiplication on the layer and filter.

The CNN model architecture consists of the following layers:

Convolutional layer; 32 nodes, kernel size 3
Convolutional layer; 32 nodes, kernel size 3
Convolutional layer; 64 nodes, kernel size 3
Fully connected layer; 128 nodes
The final layer is also a fully connected layer with 2 nodes. A Relu activation function is used in all the layers except the output layer in which we used Softmax.

IMAGE CLASSIFICATION

We have used the modern machine learning frameworks like Tensorflow and keras to build models to classify the dog breeds which will be discussed in the later part of this
paper in details. Convolution neural networks (CNNs) are the most popular neural networks
used for image classification purpose.

7

LITERATURE SURVEY

First, we shed light on the fact as to how these concepts work. A substantial increase in images available online in the recent years has managed to obtain a lot of attention towards image classification in industry and academics. With digital images playing a typical role with respect to multimedia content, the process of automation of the image classification has become an open research problem. Image classification is a fundamental task that attempts to comprehend an entire image as a whole. The goal is to classify the image by assigning it to a specific label. The computer just sees the grid of pixels of an image. The computer sees an image in the form of matrix of numbers between 0-255.

Human vision

Computer Vision

```
[[  9   1  29  70 114  76   0   8   4   5   5   0 111 162   9   8  62  62]
 [  3   0  33  61 102 106  34   0   0   0   0  49 182 150   1  12  65  62]
 [  1   0  40  54 123  90  72  77  52  51  49 121 205  98   0  15  67  59]
 [  3   1  41  57  74  54  96 181 220 170  90 149 208  56   0  16  69  59]
 [  6   1  32  36  47  81  85  90 176 206 140 171 186  22   3  15  72  63]
 [  4   1  31  39  66  71  71  97 147 214 203 190 198  22   6  17  73  65]
 [  2   3  15  30  52  57  68 123 161 197 207 200 179   8   8  18  73  66]
 [  2   2  17  37  34  40  78 103 148 187 205 225 165   1   8  19  76  68]
 [  2   3  20  44  37  34  35  26  78 156 214 145 200  38   2  21  78  69]
 [  2   2  20  34  21  43  70  21  43 139 205  93 211  70   0  23  78  72]
 [  3   4  16  24  14  21 102 175 120 130 226 212 236  75   0  25  78  72]
 [  6   5  13  21  28  28  97 216 184  90 196 255 255  84   4  24  79  74]
 [  6   5  15  25  30  39  63 105 140  66 113 252 251  74   4  28  79  75]
 [  5   5  16  32  38  57  69  85  93 120 128 251 255 154  19  26  80  76]
 [  6   5  20  42  55  62  66  76  86 104 148 242 254 241  83  26  80  77]
 [  2   3  20  38  55  64  69  80  78 109 195 247 252 255 172  40  78  77]
 [ 10   8  23  34  44  64  88 104 119 173 234 247 253 254 227  66  74  74]
 [ 32   6  24  37  45  63  85 114 154 196 226 245 251 252 250 112  66  71]]
```

Figure 2.1: *The cat image above is what a human can see vs. the image what the computer sees in the form of matrix.*

In this project, for collecting images from webcam we will be using OpenCV and feed these images to our Deep learning model which will classify that the person's eyes is 'Open' or 'Closed'. So we will follow these steps:

- We will take image input from the camera
- Detect face and eyes in the image.
- Create a Region of Interest(ROI), for both detected face and eyes.
- Feed this to our classifier(model), which will categorize whether eyes are open or closed.
- At last, we will calculate the time to check if the person is drowsy or not.

8

RESEARCH

After doing various surveys on various existing literature on fine-tuned image classification , open cv , tensorflow 2 and keras and other machine learning models we first use the prexisting model cat2cnn.h5. Then we highlight the advanced methods and their comparative study in this paper. We will learn about the structure of the model used which will help us to better understand the underlying principal under each model.

GENERAL INTRODUCTION TO RELATED TOPICS

Artificial intelligence (AI) refers to simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. The term may also be applied to any machine that exhibits traits associated with a human mind such as learning and problem solving. AI is an interdisciplinary science with multiple approaches, but advancement in machine learning and deep learning are creating a paradigm shift in virtually every sector of the technical industry. To meet the market current opportunity, we should know the basic difference between artificial intelligence, machine learning, deep learning and data science.

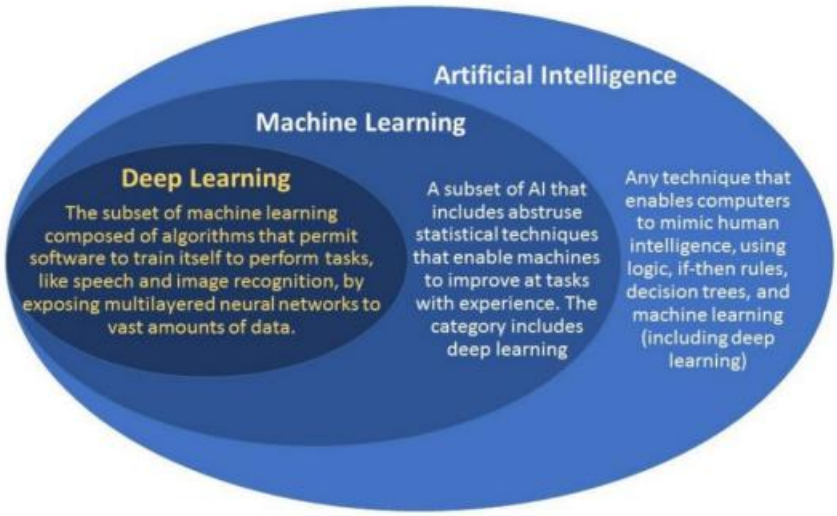


Figure 5.1: *Interrelationship between Artificial intelligence, machine learning, and deep learning.*

Since the development of digital computer in the 1940s, it has been demonstrated that computers can be programmed to carry out very complex task, for example discovering proofs of mathematical theorems and playing chess with great proficiency. But there are certain programs that have attained the performance level of human experts and professionals in performing certain specific tasks, so that AI in this limited sense is found in applications as diverse as medical diagnosis, computer search engines, and voice or handwriting recognition.

Machine Learning: This is the study of computer algorithms that improve automatically through experience and by the use of data. It is seen as a part of AI. Machine learning builds a model based on sample data, known as 'training data'. Machine learning algorithms are widely used in applications such as email filtering, computer vision and such. In the past decade, machine learning has given us self-driving cars, practical speech recognition, effect web search and a vastly improved understanding of human genome.

There are various steps involved in machine learning:

➢
Collecting the data from the web.

➢
Filtering or cleaning the data,

Analysing the data.

➢

Splitting the data into training and testing part.

➢

Training the data on various algorithms.

➢

Testing the data.

➢

Using the algorithm for the future predictions.

There are two types of algorithms involved in machine learning:

• SUPERVISED LEARNING

It is process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y). One example of this can be text classification problem where our goal is to predict the sentiment of the text like a tweet or a product review. There are only two steps involved in supervised learning:

1. Training
2. Testing

Some of the popular supervised learning algorithm are given as follows:

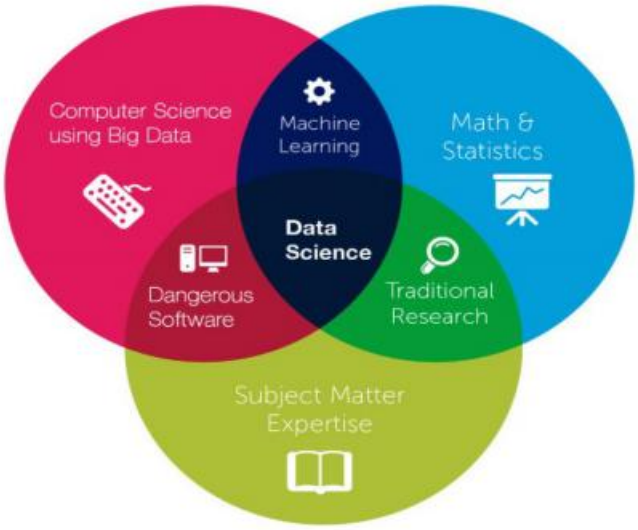
1. Linear Regression.
2. Logistic Regression.
3. Decision Tree.
4. Support Vector Machine (SVM).
5. Naïve Bayes.
6. K-nearest neighbors.

Generally, most of the neural networks used are supervised machine learning models are used to either predict (regression) or to classify. CNN are widely used in image classification and object detection.

• UNSUPERVISED LEARNING:

It is a machine learning technique in which the user does not need to supervise the model. It allows the model to work on its own discovered patterns and information that was previously detected. It mainly used unlabelled data. They are further classified into clustering and association problems.

DATA SCIENCE: This is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from structured and unstructured data, and apply knowledge and actionable insights from data across a broad range of application domains. Data science helps organization identify and refine target audience by combining existing data with other data points for developing useful insights. The definition of data science thus emphasizes statistical inference, data visualization, experiment design, domain knowledge and communication. Data scientist may use some tools to visualize data or plot graphs, whatever they use their goal is to gain a better understanding of their data



13

PROGRAMMING LANGUAGE-PYTHON

Python is an interpreted high-level general purpose programming language designed by Guido van Rossum, first appeared in the year 1991. Ever since then, the language has undergone several versions and updates and is now used in many big enterprises and IT companies. The latest version of python is 3.x and is under active development. Its language constructs as well as its object orientation approach aims to help programmers write clear, logical code for small and large scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms including structured, objected oriented and functional programming. Python is often described as 'batteries included' language due to its comprehensive library. Python is meant to be an easily readable language. It often uses English keywords unlike other programming languages which uses punctuations and semicolon. It has fewer syntactic exceptions and special cases than C or Pascal. Python uses whitespace indentation rather than curly braces to delimit blocks as in other programming language like C, C++, or Java. Python has achieved a great next-gen reputation because of its vast industrial application and its international glory. It has been adopted into many top level computer science programs by major educators and institutes as an undergrad course requirement.

There are various advantages of using python, some of them are listed below:

1. Presence of third part modules.
2. Extensive support of libraries.
3. Open source and community development.
4. Easy to learn.
5. User friendly data structure.
6. High-level language.
7. Dynamically typed language.
8. Object-Oriented Language.
9. Portable and interactive.
10. Portable across operating system.
11. Python is safe

- **Presence of third part modules:** Python has a huge collection of third-party modules. Some of the common list of external libraries include **matplotlib, Numpy, Scipy, Django, IPython** etc.
- **Extensive support of libraries:** There are various libraries in python that are very useful for machine learning as well as web development. Some of them are **Pandas, Seaborn, Numpy, Tensorflow, Scikit-learn, Pytorch** and such. In addition to the standard libraries, there are extensive collections of freely available add-on modules, libraries, frameworks and tool kits. Almost all database adapters conform to python DBAPI and thus can mostly be accessed using the same code. It is usually easy to modify python code to support any database engine.
- **Open source and community development:** Python is developed under an OSI-approved open source license, making it freely usable and distributable, even for commercial use.
- **Easy to learn:** Python is easy to learn, so both programmers as well as non-programmers can code along.
- **User friendly data structure:** Python has built-in list and dictionary data structure that can be used to construct runtime data structures. Besides, Python also offers the option of dynamic high-level data typing that reduces the length of support code required.

- **High-level language:** High level language means it is easier to read. Python is very similar to English language and hence it is very simple to read. There are no punctuations and semicolons. Other languages turn into Assembly when compiled, and run directly in the processor. Hence, being an interpreted language, which is not subject to processor, makes Python a high-level language. Python offers convenience of code readability -- which makes the syntax of the program much easier and shorter, resulting in less coding steps for developers than imposed by Java or C++.
- **Dynamically typed language:** This means that we do not to give the data type of the variable while initializing it, it stores the variable and gets its data type. This means that python interpreter checks the type of the variable while executing the code.
Other languages like C, C++, or Java strictly define the type of the data before the variable name before assigning any value to it.
- **Object-Oriented Language:** Objected Oriented programming (OOP) is a programming paradigm based on the concept of objects which can contain attributes and methods. Objects provide a better and clear structure of program.
For example, a bike can be an object. If bike is considered to be an object then all its features like colour, model type, price etc. will be the properties of the object.
Programming languages like Java, Python, JavaScript, PHP, Scala and C++ are the main languages that provide object oriented approach.
- **Portable and interactive:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code. Python can run on wide variety of hardware platform and has same interface on all.16
- **Portable across operating system:** Python can operate on all operating systems such as Microsoft windows, Linux, MacOS, and such.

- **Python is safe:** Like C and C++ python does not use the concept of pointers, which makes it more reliable. It is safe from memory leaks. Python runtime checks that type system rules are satisfied, so python is safe.

14
APPLICATIONS OF PYTHON

Applications of Python language for building software for various industries is greatly diverse, with each application drawing a unique potential from the language. Software development companies can therefore leverage the diversity element of Python to gain advantage in domain like:

- Web applications.
- Testing frameworks
- Embedded apps
- Gaming apps
- Enterprise solutions
- Prototype creation
- Language development
- Graphic design applications

Such broad usage of the language across assorted industries means a strong upper hand of Python over other contemporary programming languages used by programmers.

II.
FUTURE SCOPES OF PYTHON

Python is a high level multi paradigm programming language that has the feature of all conventional programming languages like C, C++, Java and such.

It is one of the fastest growing language that has undergone a successful span of more than 25 years as far as its adoption is concerned. In fact, it has been continuously serving as the best programming language application development, web development, game development, system administration, scientific and numeric computing, GIS and mapping etc.

The reason behind the immense popularity of python programming language across the globe is the feature it provides. We have listed some of the features below:

1.**Python Supports Multiple Programming Paradigms:** Python is a multi-paradigm programming language including features such as object-oriented, imperative, procedural, functional, reflective etc.

2.**Python Has Large Set of Library and Tools:** Python has very extensive standard libraries and tools that enhance the overall functionality of python language and also helps python programmers to easily write codes. Some of the important python libraries and tools are:

- Built-in functions, constants, types, and exceptions.
- File formats, file and directory access, multimedia services.
- GUI development tools such as Tkinter
- Custom Python Interpreters, Internet protocols and support, data compression and archiving, modules etc.
- Scrappy, wxPython, SciPy, matplotlib, Pygame, PyQT, PyGTK etc.

3.**Python Has a Vast Community Support:** This is what makes python a favourable choice for development purposes. If you are having problems writing python a program, you can post directly to python community and you will get the response with the solution of your problem.

4.**Python is designed for Better Code Readability:** Python provides a much better code readability as compared to another programming language. For example, it uses whitespace indentation in place of curly brackets for delimiting the block of codes.

5.**Python Contains Fewer Lines Of Codes:** Codes written in python programming language complete in fewer lines thus reducing the efforts of programmers.

15

FUTURE TECHNOLOGIES COUNTING ON PYTHON

We know that python programming language is extensively used for web development, application development, system administration, developing games etc. But, there are certain future technologies that rely on python. As a matter of fact, Python has become the core language as far as the success of these technologies is concerned. Let's dive into the technologies which use python as a core element for research, production and further developments.

1.**Artificial Intelligence (AI)**

Python programming language is undoubtedly dominating the other languages when future technologies like Artificial Intelligence (AI) comes into the play.

There are plenty of python frameworks, libraries, and tools that are specifically developed to direct Artificial Intelligence to reduce human efforts with increased accuracy and efficiency for various development purposes.

It is only the Artificial Intelligence that has made it possible to develop speech recognition system, autonomous cars, interpreting data like images, videos etc.

We have shown below some of the python libraries and tools used in various Artificial Intelligence branches.

- **Machine Learning-** PyML, PyBrain, scikit-learn, MDP Toolkit, GraphLab Create, and MIPy etc.
- **General AI-** pyDatalog, AIMA, EasyAI, SimpleAI etc.
- **Neural Networks-** PyAnn, pyrenn, ffnet, neurolab etc.
- **Natural Language & Text Processing-** Quepy, NLTK, gensim

Big Data

The future scope of python programming language can also be predicted by the way it has helped big data technology to grow. Python has been successfully contributing in analysing a large number of data sets across computer clusters through its high performance toolkits and libraries.

Following are some of the libraries and toolkit used for data analysis and handling other big data issue:

- Pandas
- Scikit-Learn
- NumPy
- SciPy
- GraphLab Create
- IPython
- Bokeh
- Agate
- PySpark
- Dask

.

Networking: Networking is another field in which python has a brighter scope in the future. Python programming language is used to read, write and configure routers and switches and perform other networking automation tasks in a cost-effective and secure manner.
For these purposes, there are many libraries and tools that are built on the top of the python language. Here we have listed some of these python libraries and tools especially used by network engineers for network automation.

- Ansible
- Netmiko
- NAPALM
- Pyeapi
- Junos PyEZ
- PySNMP
- Paramiko SSH

REAL LIFE PYTHON SUCCESS STORIES

Python has seemingly contributed as a core language for increasing productivity regarding various development purposes at many of the IT organizations. We have shown below some of the real-life python success stories.

- Australia's RMA Department D-Link has successfully implemented python for creating DSL Firmware Recovery System.19
- Python has helped Gusto.com, an online travel site, in reducing development costs and time.
- ForecastWatch.com also uses python in rating the accuracy of weather forecast reports provided by companies such as Accuweather, MyForecast.com and The Weather Channel.
- Python has also benefitted many product development companies such as Acqutek, AstraZeneca, GravityZoo, Carmanah Technologies Inc. etc. in creating autonomous devices and software.
- Test&Go uses python scripts for Data Validation.
- Industrial Light & Magic (ILM) also uses python for batch processing that includes modelling, rendering and compositing thousands of picture frames per day.

There is a huge list of success stories of many organizations across the globe which are using python for various purposes such as software development, data mining, unit testing, product development, web development, data validation, data visualization etc

TOP COMPETITORS OF PYTHON

The future scope of python programming language also depends on its competitors in the IT market. But, due to the fact that it has become a core language for future technologies such as artificial intelligence, big data, etc., it is surely going to rise further and will be able to beat its competitors.
The top competitors of Python are listed below:

- **ASP.NET**
- **Java**
- **C#**

WEBSITE DEVELOPED USING PYTHON

We already know that python programming is used for web development, so some of the world's most famous websites are created using python.
Some of them are listed below:

- YouTube
- Quora
- Instagram
- Pinterest
- Spotify
- Flipkart
- Slack
- Uber
- Cloudera
- Zenefits

18

PROJECT REQUIREMENTS :

. IMPORTS

The libraries that have been imported for this project is listed below:

1.**Numpy**: NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements. It is an extension module of Python which is mostly written in C. It provides various functions which are capable of performing the numeric computations with a high speed.

2.**Pandas:** Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. It is fast, powerful, flexible and easy to use open source data analysis and manipulation tool, build on the top of Python programming language.

The key features of pandas are:

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of datasets.
- Label-based slicing, indexing and subsetting of large datasets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

3.**Matplotlib:** Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers also.

4.**Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. Seaborn can be installed using pip on Windows. It is used to provide graphical representation of random distribution. Seaborn besides being a statistical plotting library also provides some default datasets, example ‘tips’. The ‘tips’ dataset contains information about people who probably had food at a restaurant and whether or not they left a tip for the waiters, their gender, whether they smoke and so on.

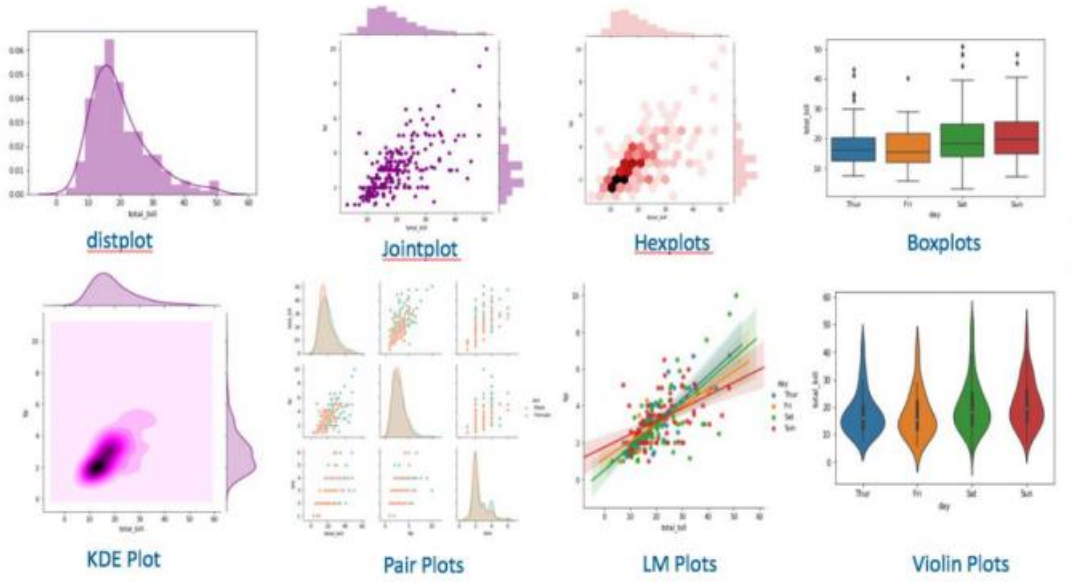


Figure 6.1: *Different types of plots in seaborn*

5.**Pygame:**This library is used mainly to create games but here we are using it to ring an alarm.
6.**Tensorflow:** TensorFlow is an open source library for fast numerical computing. It was created and is maintained by Google and released under the Apache 2.0 open source license. The API is nominally for the Python programming language, although there is access to the underlying C++ API. To develop and research on fascinating ideas on artificial intelligence, Google team created TensorFlow.

TensorFlow is designed in Python programming language, hence it is considered an easy to understand framework. Most models are made of layers. Layers are functions with a known mathematical structure that can be reused and have trainable variables. In TensorFlow, most high-level implementations of layers and models, such as Keras are built on the same foundational class. It has three layers, they are described as Follows:

• I**nput Layer**: In Keras, the input layer itself is not a layer, but a tensor. It's the starting tensor you send to the first hidden layer. This tensor must have the

same shape as your training data. They consist of artificial neurons.

•**Hidden Layer:** A hidden layer is an artificial neural network that is in between input layers and output layers, where the artificial neurons take in a set of weighted inputs and produce an output through an activation function.

•**Output Layer:** The output layer in an artificial neural network is the last layer of neurons that produces given outputs for the program.

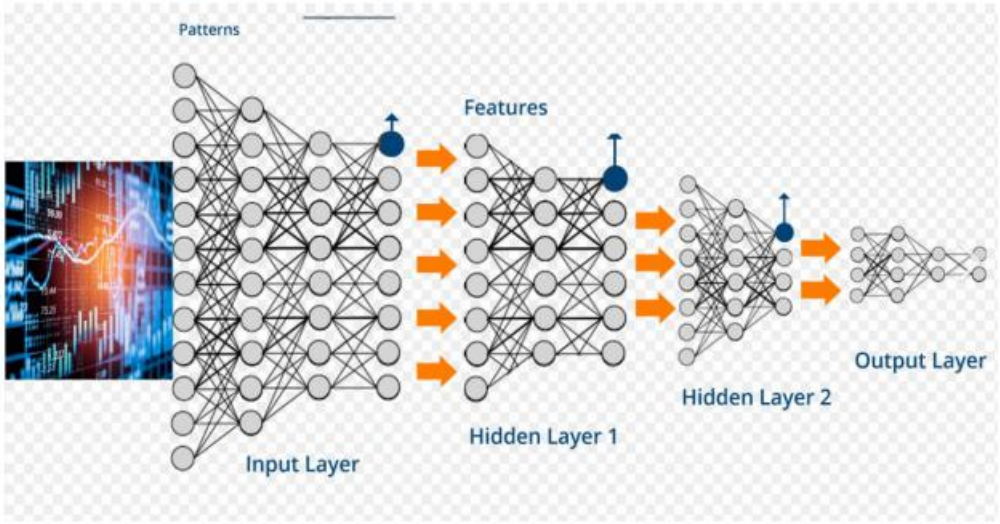


Figure 6.2: *The layers in convolutional neural network.*

7.**Keras**: Keras is an open source deep learning framework for python. It has been developed by an artificial intelligence researcher at Google named **Francois Chollet**

8.OS:provides functions for interacting with the environment

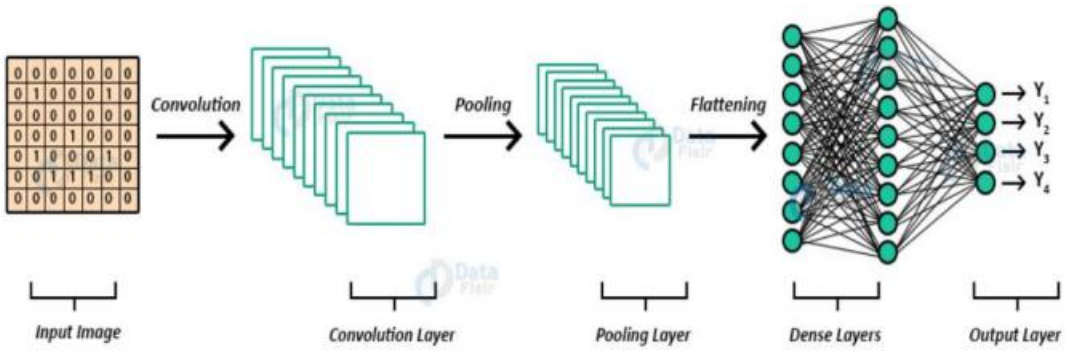


Figure 6.3: *Dense Layers in Keras*

Sklearn: Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality 23 reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

HARDWARE AND SOFTWARE REQUIREMENTS

Software Requirement:
- Operating System – Windows/Linux.
- Programming Language – Python 3.6 or above.
- Software – Jupyter Notebook/Pycharm/Google colab (Open source)/VSCode

Hardware Requirement:
- Speed – 233 MHz or above.
- Hard disk – 10 GB
- RAM – 512 GB

DESCRIPTION OF TRAINING MODELS USED

Convolutional Neural Network:

Artificial Intelligence has been observing an enormous growth in fulfilling the gap between the capabilities of humans and machines. Researchers work on numerous aspects of the field to make amazing things happen and one of such areas is the domain of **Computer Vision**. The agenda for this field is to enable machines to view the world as humans do, perceive it in a similar manner and even use the knowledge for a multitude of tasks such as Image and Video recognition, Image Analysis and Classification, Media Recreation, Recommendation Systems, Natural Language Processing, etc. The advancements in Computer Vision with Deep Learning has been constructed and perfected with time, primarily over one particular algorithm — a **Convolutional Neural Network**. Neural networks are a subset of machine learning, and they are at the heart of deep learning algorithms. They are comprised of node layers, containing an input layer, one or more hidden layers, and an output layer. Each node connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

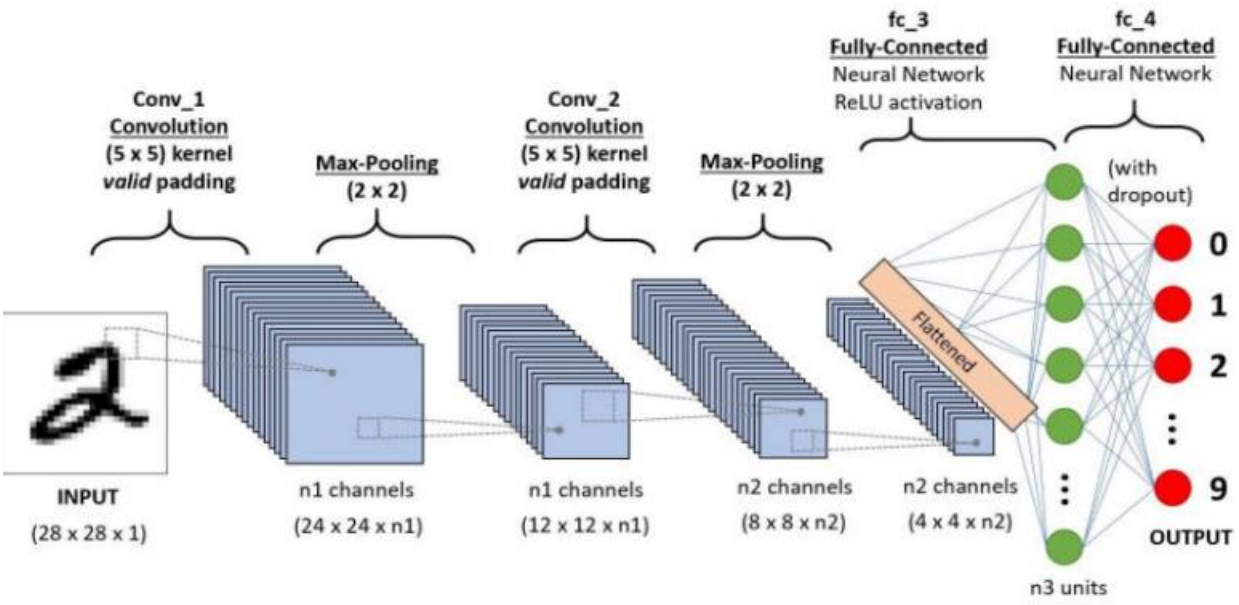


Figure 6.4: *Different layers in Convolutional neural networks (CNN)*

Before diving into the Convolution Neural Network, let us first revisit some concepts of Neural Network. In a regular Neural Network there are three types of layers:

1.**Input Layers:** It's the layer in which we give input to our model. The number of neurons in this layer is equal to total number of features in our data (number of pixels in case of an image).

2.**Hidden Layer:** The input from Input layer is then fed into the hidden layer. There can be many hidden layers depending upon our model and data size. Each hidden layers can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of output of the previous layer with learnable weights of that layer and then by addition of learnable biases followed by activation function which makes the network nonlinear.

3.**Output Layer:** The output from the hidden layer is then fed into a logistic function like sigmoid or softmax which converts the output of each class into probability score of each class.

Input Image: In the figure, we have an RGB image which has been separated by its three colour planes — Red, Green, and Blue. There are a number of such colour spaces in which images exist in Grayscale, RGB, HSV, CMYK, etc. You can imagine how computationally intensive things would get once the images reach dimensions, say 8K (7680×4320). The role of the convent is to reduce the images into a form which is easier to process, without losing features which are important for getting a good prediction.

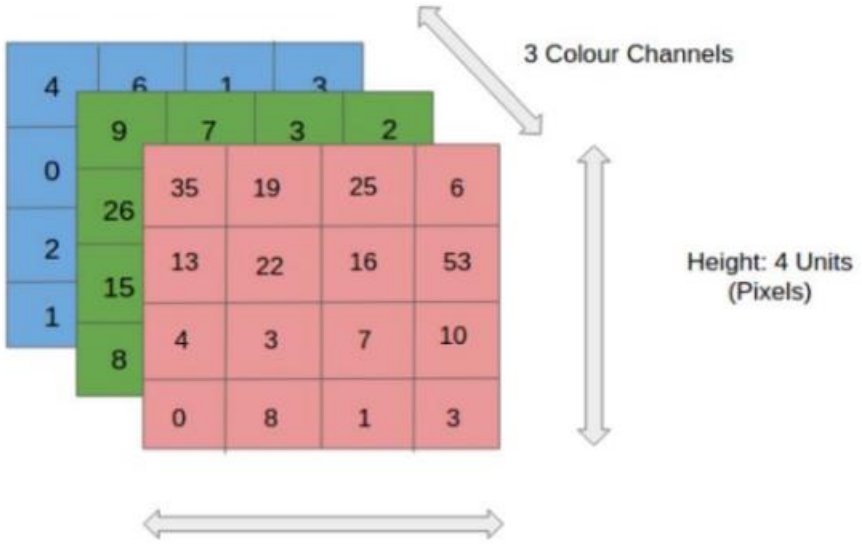


Figure 6.5: *Input Image of size 4x4x3 RGB image.*

Convolution Layer – the kernel

Image Dimensions = 5 (Height) x 5 (Breadth) x 1 (Number of channels, e.g. RGB)
In the image below, the bigger matrix section resembles our 5x5x1 input image, I. The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the Kernel/Filter, K, represented in the colour yellow. We have selected K as a 3x3x1 matrix.

Kernel/Filter,
K =
1 0 1
0 1 0
0 0 1
The Kernel shifts 9 times because of Stride Length = 1 (Non-Strided), every time performing a matrix multiplication operation between K and the portion P of the image over which the kernel is hovering.

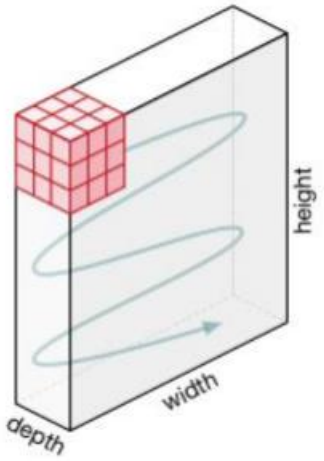


Figure 6.7: *Movement of the kernel*

The filter moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.

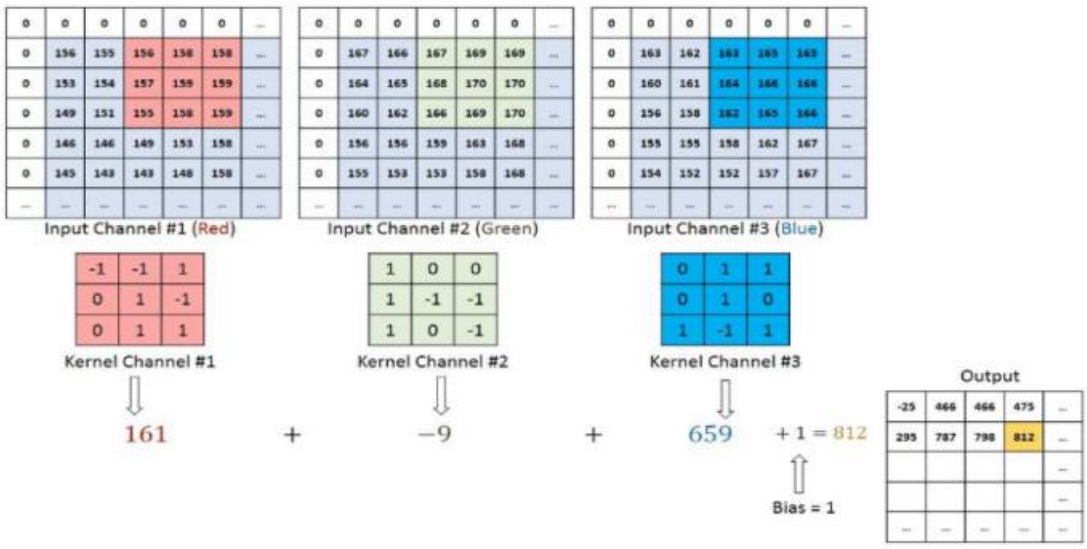


Figure 6.8: *Convolution operation on MxNx3 image matrix with a 3x3x3 kernel.*

In the case of images with multiple channels (e.g. RGB), the Kernel has the same depth as that of the input image. Matrix Multiplication is performed between Kn and In stack ([K1, I1]; [K2, I2]; [K3, I3]) and all the results are summed with the bias to give us a squashed one-depth channel Convoluted Feature Output.
The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, colour, gradient orientation, etc. With added layers, the architecture adapts to the High Level features as well, giving us a network which has the wholesome understanding of images in the dataset, similar to how we would.

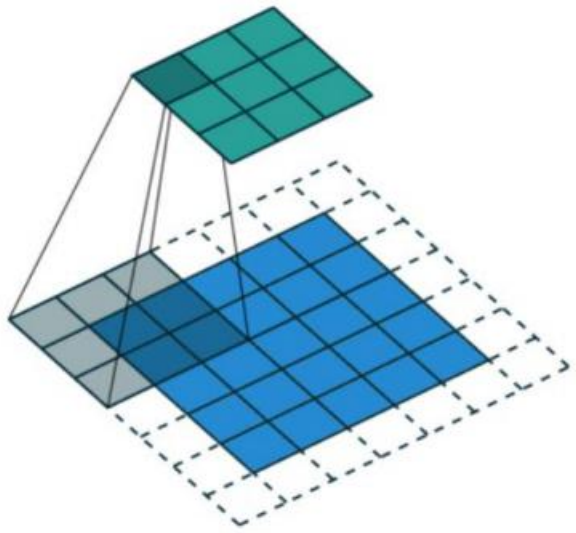


Figure 6.9: *Convolution operation with stride length 2.*

There are two types of results to the operation — one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying **Valid Padding** in case of the former, or **Same Padding** in the case of the latter.
When we augment the 5x5x1 image into a 6x6x1 image and then apply the 3x3x1 kernel over it, we find that the convolved matrix turns out to be of dimensions 5x5x1. Hence the name **Same Padding**.
If we perform the same operation without padding, we are presented with a matrix which has dimensions of the Kernel (3x3x1) itself. Hence the name **Valid Padding**.

Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.
There are **two** types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel and, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average

Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.

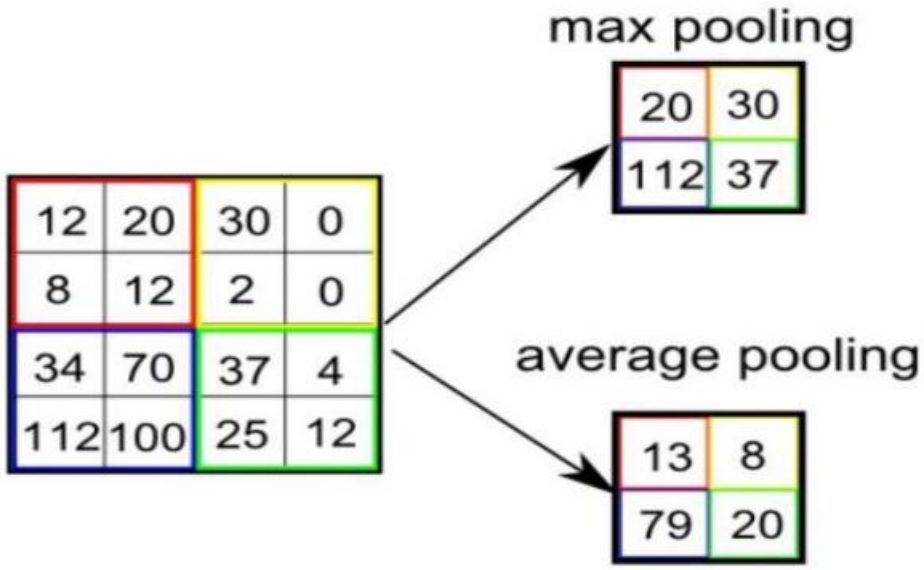


Figure 6.10: *Types of Pooling.*

The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, but at the cost of more computational power.

Classification – Fully Connected Layer

Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully Connected layer is learning a possibly non-linear function in that space.

Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and back propagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the **Softmax Classification** technique.

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future. Some of them have been listed below:

1.LeNet
2.AlexNet
3.VGGNet
4.GooLeNet
5.ResNet
6.ZFNet

Optimizers:

Optimizer updates the model in response to the output of the loss function. Optimizer assist in minimizing the loss function. It changes the attributes of the neural network such as weights and learning rate to reduce the loss.
Gradient Descent :Gradient descent is the most popular and widely used optimizer of all. It is simple and effective to find the minimum value of the neural network. The objective of all optimizers is to reach the global minima where the cost function attains the least possible value. When there are one or more inputs, we use a process of optimising the values of the coefficient by iteratively minimising the error of the model on the training data. The sum of the squared error are calculated for each pair of input and output. The coefficients are updated in the direction towards minimising the error. The process until a minimum sum squared error is achieved.

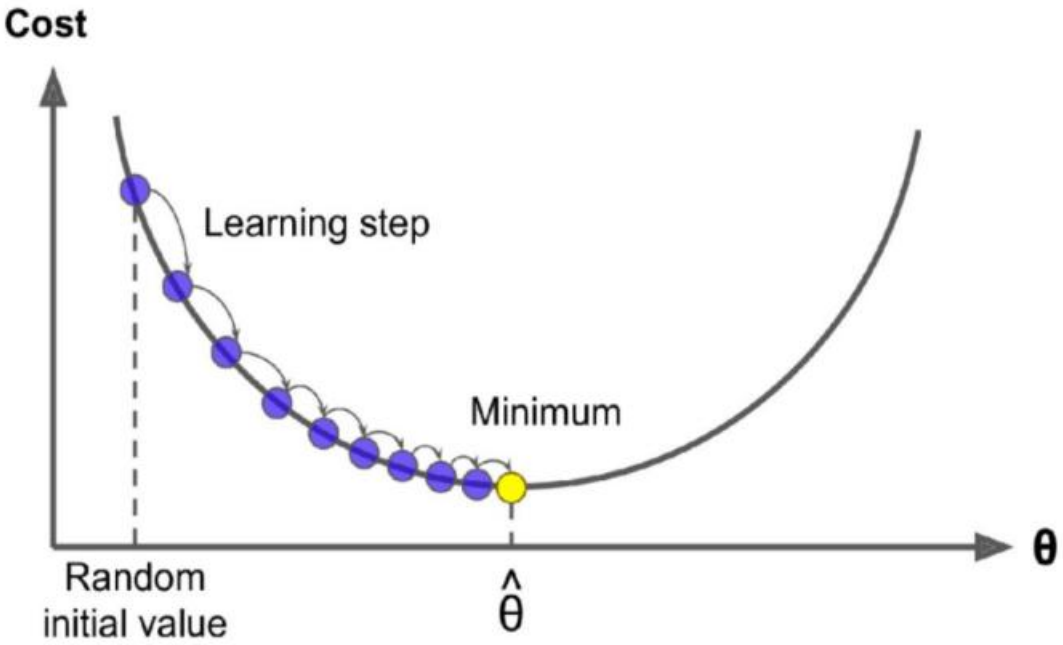


Figure 6.15: *Graphical representation of Gradient descent optimization.*

Stochastic Gradient Optimizer
The SGD or Stochastic Gradient Optimizer is an optimizer in which the weights are updated for each training sample or a small subset of data. SGD performs frequent updates with a high variance that cause the objective function to fluctuate. In this, the
`
model parameters are altered after computation of loss on each training example. So, if the dataset contains 1000 rows SGD will update the model parameters 1000 times in one cycle of dataset instead of one time as in Gradient Descent.

Momentum based Gradient Descent:

If we consider, Simple Gradient Descent completely relies only on calculation i.e. if there are 10000 steps, then our model would try to implement Simple Gradient Descent for 10000 times that would be obviously too much time consuming and computationally expensive. In laymens language, suppose a man is walking towards his home but he don't know the way so he ask for direction from by passer, now we expect him to walk some distance and then ask for direction but man is asking for direction at every step he takes, that is obviously more time consuming, now compare man with Simple Gradient Descent and his goal with minima. In order to avoid drawbacks of vanilla Gradient Descent, we introduced momentum based Gradient Descent where the goal is to lower the computation time and that can be achieved when we introduce the concept of experience i.e. the confidence using previous steps. The momentum based Gradient descend is taking large steps due to momentum it is burdening along. But due to larger steps it overshoots its goal by longer distance as it oscillate around minima due to steep slope, but despite such hurdles it is faster than vanilla Gradient Descent.

Adagrad:

One of the disadvantages of all the optimizers explained is that the learning rate is constant for all parameters and for each cycle. This optimizer changes the learning rate. It changes the learning rate 'η' for each parameter and at every time step 't'. It's a type second order optimization algorithm. It works on the derivative of an error function. The advantage of using this optimizer is that it can work on sparse data and learning rate need not to be tuned manually. But it is computationally Expensive.AdaDelta:

It is an extension of AdaGrad which tends to remove the decaying learning rate problem of it. Instead of accumulating all previously squared gradients, this limits the window of accumulated past gradients to some fixed size weight. In this exponentially moving average is used rather than the sum of all the gradients.

Nesterov Accelerated Gradient Descent (NAG):

To overcome the problems of momentum based Gradient Descent we use NAG, in this we move first and then compute gradient so that if our oscillations overshoots then it must be insignificant as compared to that of Momentum Based Gradient Descent. Nesterov accelerated Gradient (NAG) is a way to provide history to our momentum. We can now adequately look forward by computing the angle not with respect to to our present parameters θ.

With momentum:

$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_t$$
$$w_{t+1} = w_t - update_t$$

With NAG:

$$w_{look\_ahead} = w_t - \gamma \cdot update_{t-1}$$
$$update_t = \gamma \cdot update_{t-1} + \eta \nabla w_{look\_ahead}$$
$$w_{t+1} = w_t - update_t$$

Adam optimizer

Adaptive Moment Estimation is an algorithm for optimization technique for gradient descent. The method is really efficient when working with large problem involving a lot of data or parameters. It requires less memory and is efficient. Intuitively, it is a combination of the 'gradient descent with momentum' algorithm and the 'RMSP' algorithm.

Momentum:

This algorithm is used to accelerate the gradient descent algorithm by taking into consideration the 'exponentially weighted average' of the gradients. Using averages makes the algorithm converge towards the minima in a faster pace.

$$w_{t+1} = w_t - \alpha m_t$$

where,

$$m_t = \beta m_{t-1} + (1 - \beta) \left[\frac{\delta L}{\delta w_t} \right]$$

m_t = aggregate of gradients at time t [current] (initially, $m_t = 0$)
m_{t-1} = aggregate of gradients at time t-1 [previous]
W_t = weights at time t
W_{t+1} = weights at time t+1
α_t = learning rate at time t
∂L = derivative of Loss Function
∂W_t = derivative of weights at time t
β = Moving average parameter (const, 0.9)

Root Mean Square Propagation (RMSP):

Root mean square prop or RMSprop is an adaptive learning algorithm that tries to improve AdaGrad. Instead of taking the cumulative sum of squared gradients like in AdaGrad, it takes the 'exponential moving average'.

$$w_{t+1} = w_t - \frac{\alpha_t}{(v_t+\varepsilon)^{1/2}} * \left[\frac{\delta L}{\delta w_t}\right]$$

where,

$$v_t = \beta v_{t-1} + (1-\beta) * \left[\frac{\delta L}{\delta w_t}\right]^2$$

W_t = weights at time t
W_{t+1} = weights at time t+1
α_t = learning rate at time t
∂L = derivative of Loss Function
∂W_t = derivative of weights at time t
V_t = sum of square of past gradients. [i.e sum($\partial L/\partial Wt-1$)] (initially, $V_t = 0$)
β = Moving average parameter (const, 0.9)
ϵ = A small positive constant (10^{-8})

Mathematics involved in Adam optimization:
Taking the formulas used in above two equations:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\left[\frac{\delta L}{\delta w_t}\right] \; v_t = \beta_2 v_{t-1} + (1-\beta_2)\left[\frac{\delta L}{\delta w_t}\right]^2$$

Parameters Used :
1. ϵ = a small +ve constant to avoid 'division by 0' error when (v_t -> 0). (10^{-8})
2. β_1 & β_2 = decay rates of average of gradients in the above two methods. (β_1 = 0.9 &
3. α – Step size parameter / learning rate (0.001)

Performance:
Adam optimizer gives much higher performance than the previously used optimisers and outperforms them by a big margin into giving an optimized gradient descent. The plot is shown below clearly depicts how Adam Optimizer outperforms the rest of the optimizer by a considerable margin in terms of low training cost and high performance.

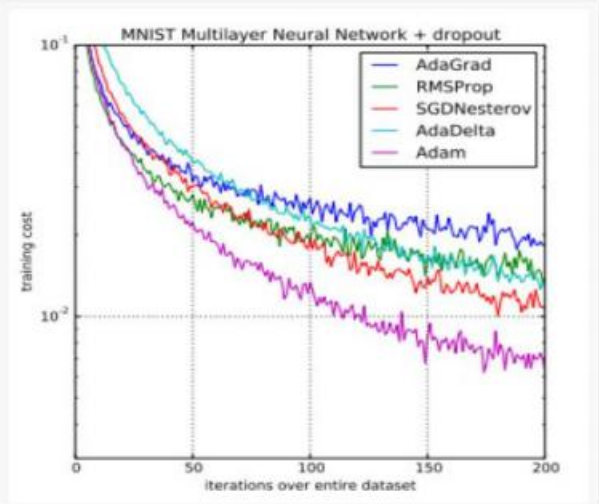


Figure 6.17: *Comparison of Adam optimizer with other optimizers.*

Loss Functions:

In Machine learning, the loss function is determined as the difference between the actual output and the predicted output from the model for the single training example while the average of the loss function for all the training example is termed as the cost function. This computed difference from the loss functions (such as Regression Loss, Binary Classification and Multiclass Classification loss function) is termed as the error value; this error value is directly proportional to the difference in the actual and the predicted value. Loss function will define how good our prediction is, so choosing a right loss function is important.

Loss function is of two types:

1.Regression Loss Function.

2.Classification Loss Function.

Regression loss functions establishes a linear relationship between a dependent variable (Y) and an independent variable (X); hence we try to fit the best line in space n these variables.

In **classification**, one can predict the output from a set of finite categorical values, i.e. categorizing a broad data set of handwritten digits into one of 0–9 digits.

Binary cross Entropy loss function:

It's a default loss function for binary classification problems. Cross-entropy loss calculates the performance of a classification model, which gives an output of a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability value deviate from the actual label.

Categorical cross Entropy loss function:

The categorical cross-entropy loss is exclusively used in multi-class classification tasks, where each sample belongs exactly to one of the classes. It is basically binary cross entropy extended to multiple classes. The output label is assigned one-hot category encoding value in form of 0s and 1s. The output label, if present in integer form, is converted into categorical encoding using keras.utils to_categorical method.

Activation Functions:

To put in simple terms, an artificial neuron calculates the 'weighted sum' of its inputs and adds a bias, as shown in the figure below by the net input.

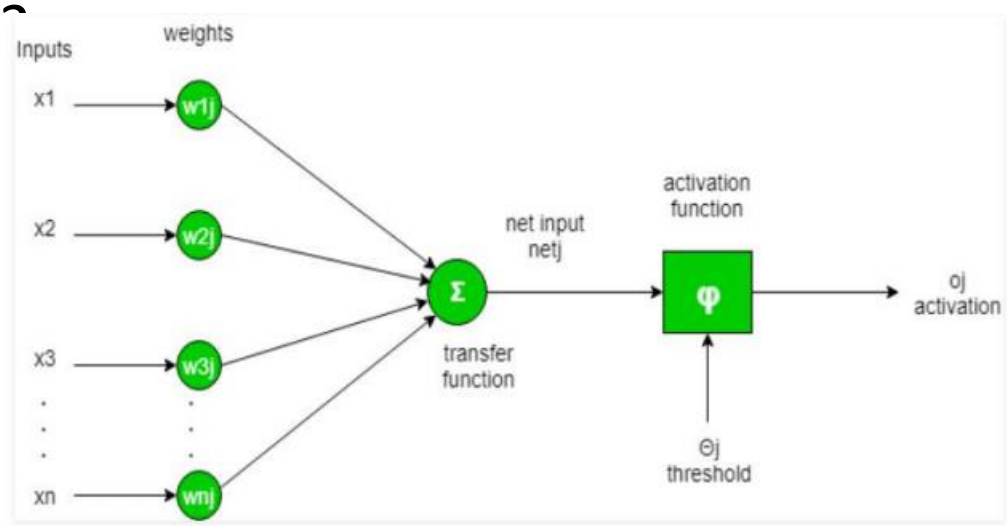


Figure 6.18: *How an activation function works.*

Now the value of net input can be any anything from -inf to +inf. The neuron doesn't really know how to bind to value and thus is not able to decide the firing pattern. Thus the activation function is an important part of an artificial neural network. They basically decide whether a neuron should be activated or not. Thus it bounds the value of the net input. The activation function is a non-linear transformation that we do over the input before sending it to the next layer of neurons or finalizing it as output.

Some of the important activation functions are as follows:
1.Sigmoid or Logistic activation function. (Non-Linear activation function)
2.Rectified Linear Unit or ReLU. (Non-Linear activation function).
3.Hyperbolic tangent (tanh) activation function. (Non-Linear activation function).

Sigmoid or Logistic activation function:

Sigmoid functions are bounded, differentiable, real functions that are defined for all real input values, and have a non-negative derivative at each point. The sigmoid function have output in the range of 0-1.
The equation of the sigmoid function is given below:

$$\phi(s_k) = \frac{1}{1+e^{-s_k}} = \frac{e^{s_k}}{1+e^{s_k}}$$

The output of the activation function is always going to be in range (0, 1) compared to (-inf, inf) of linear function. It is non-linear, continuously differentiable, monotonic, and has a fixed output range. But it is not zero centred.

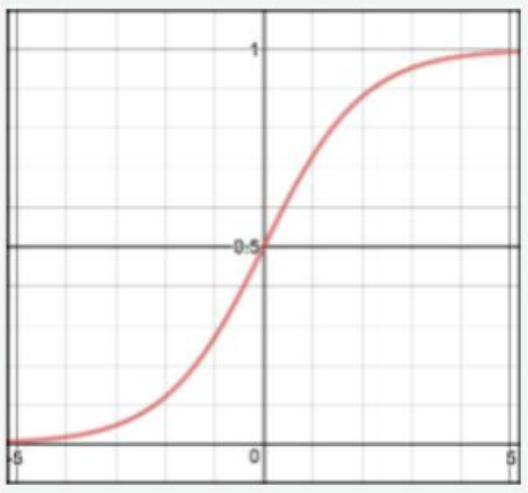


Figure 6.19: *Graphical representation of sigmoid function.*

Rectified Linear Unit or ReLU:
The ReLU function is the Rectified linear unit. It is the most widely used activation function. It is defined as:

$$f(x) = \max(0, x)$$

The main advantage of using the ReLU function over other activation functions is that it does not activate all the neurons at the same time

7. SOURCE CODE SNIPPETS.

```
import cv2
import os
from keras.models import load_model
import numpy as np
from pygame import mixer
import time

mixer.init()
sound = mixer.Sound('alarm.wav')
# These xml files are used to detect the face, left eye, and righ
t eye of a person.
face = cv2.CascadeClassifier(
'haar cascade files\haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier(
'haar cascade files\haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier(
'haar cascade files\haarcascade_righteye_2splits.xml')

lbl=['Close','Open']
# load the model, that we have created
model = load_model('models/cnncat2.h5')
path = os.getcwd()
# to capture each frame
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
#check if the webcam is opened correctly
if cap.isOpened():
    cap = cv2.VideoCapture(0)
if not cap.isOpened():
    raise IOError("Cannot open webcam")
count=0
score=0
thicc=2
rpred=[99]
lpred=[99]
```

```
while(True):
    ret, frame = cap.read()
    height,width = frame.shape[:2]
    #convert the captured image to grey color:
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)


#perform detection(this will return x,y coordinates , height , wi
dth of the boundary boxes object)
    faces = face.detectMultiScale(gray,minNeighbors=5,scaleFactor
=1.1,minSize=(25,25))
    left_eye = leye.detectMultiScale(gray)
    right_eye = reye.detectMultiScale(gray)

    cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0
) , thickness=cv2.FILLED )
    classes_lpred=None
    classes_rpred=None

#iterating over faces and drawing boundary boxes for each face:
    for (x,y,w,h) in faces:
     cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )
    print(right_eye)
     #iterating over right eye:
    for (x,y,w,h) in right_eye:
        r_eye=frame[y:y+h,x:x+w]
        count=count+1
        r_eye = cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
        r_eye = cv2.resize(r_eye,(24,24))
        r_eye= r_eye/255
        r_eye= r_eye.reshape(24,24,-1)
        r_eye = np.expand_dims(r_eye,axis=0)
        rpred = model.predict(r_eye)
        classes_rpred=np.argmax(rpred,axis=1)
        if(classes_rpred[0]==1):
            lbl='Open'
        if(classes_rpred[0]==0):
            lbl='Closed'
        break
    print(left_eye)
     #iterating over left eye:
    for (x,y,w,h) in left_eye:

        l_eye=frame[y:y+h,x:x+w]
        count=count+1
        l_eye = cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
        l_eye = cv2.resize(l_eye,(24,24))
        l_eye= l_eye/255
        l_eye=l_eye.reshape(24,24,-1)
        l_eye = np.expand_dims(l_eye,axis=0)
        lpred = model.predict(l_eye)
        classes_lpred=np.argmax(lpred,axis=1)
        if(classes_lpred[0]==1):
            lbl='Open'
        if(classes_lpred[0]==0):
            lbl='Closed'
        break
```

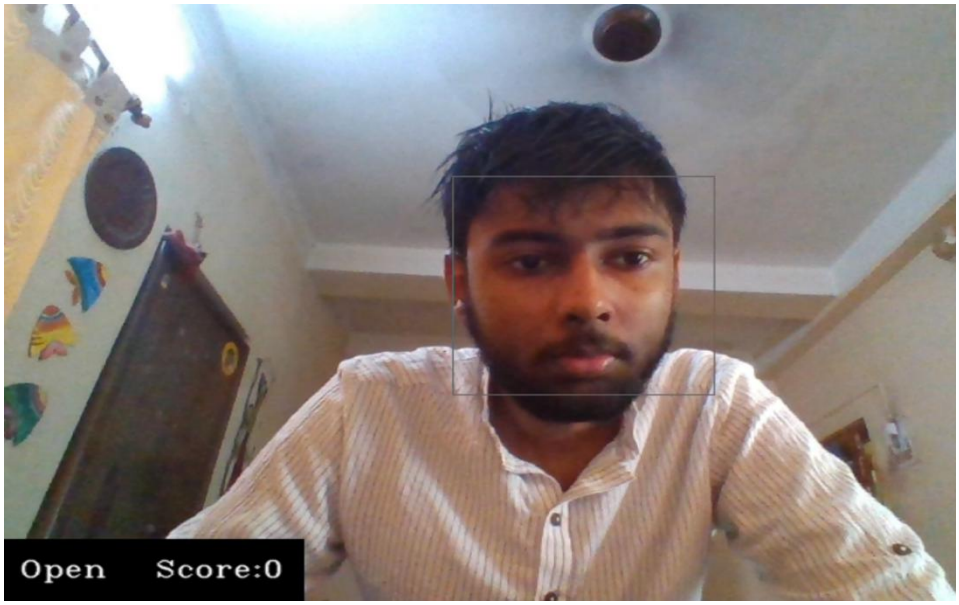
```
#print(classes_lpred,classes_rpred)
    if not classes_rpred and not classes_lpred:
        score=score+1
        cv2.putText(frame,"Closed",(10,height-20), font, 1,(255,
255,255),1,cv2.LINE_AA)
    elif  classes_rpred[0]==0 and classes_lpred[0]==0:
        score=score+1
        cv2.putText(frame,"Closed",(10,height-20), font, 1,(255,
255,255),1,cv2.LINE_AA)
    # if(rpred[0]==1 or lpred[0]==1):
    else:
        score=score-1
        cv2.putText(frame,"Open",(10,height-20), font, 1,(255,255
,255),1,cv2.LINE_AA)

    if(score<0):
        score=0
    cv2.putText(frame,'Score:'+str(score),(100,height-20), font,
1,(255,255,255),1,cv2.LINE_AA)
    if(score>15):
        #person is feeling sleepy so we beep the alarm
        cv2.imwrite(os.path.join(path,'image.jpg'),frame)
        try:
            sound.play()

        except: # isplaying = False
            pass
        if(thicc<16):
            thicc= thicc+2
        else:
            thicc=thicc-2
            if(thicc<2):
                thicc=2
        cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

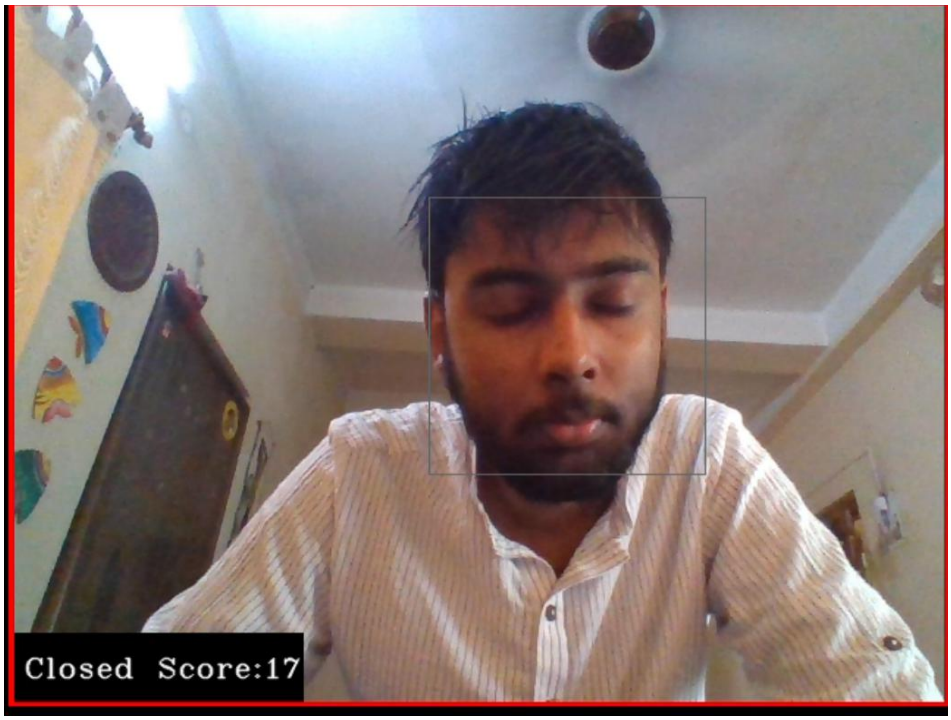
Model used(cnnCat2.h5):

```
import os
from keras.preprocessing import image
import matplotlib.pyplot as plt
import numpy as np
from keras.utils.np_utils import to_categorical
import random,shutil
from keras.models import Sequential
from keras.layers import Dropout,Conv2D,Flatten,Dense, MaxPooling2D, BatchNormalization
from keras.models import load_model


def generator(dir, gen=image.ImageDataGenerator(rescale=1./255), shuffle=True,batch_size=1,target_size=(24,24),class_mode='categorical' ):

    return gen.flow_from_directory(dir,batch_size=batch_size,shuffle=shuffle,color_mode='grayscale',class_mode=class_mode,target_size=target_size)

BS= 32
TS=(24,24)
train_batch= generator('data/train',shuffle=True, batch_size=BS,target_size=TS)
valid_batch= generator('data/valid',shuffle=True, batch_size=BS,target_size=TS)
SPE= len(train_batch.classes)//BS
VS = len(valid_batch.classes)//BS
print(SPE,VS)


# img,labels= next(train_batch)
# print(img.shape)

model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(24,24,1)),
    MaxPooling2D(pool_size=(1,1)),
    Conv2D(32,(3,3),activation='relu'),
    MaxPooling2D(pool_size=(1,1)),
#32 convolution filters used each of size 3x3
#again
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(1,1)),

#64 convolution filters used each of size 3x3
#choose the best features via pooling

#randomly turn neurons on and off to improve convergence
    Dropout(0.25),
#flatten since too many dimensions, we only want a classification output
    Flatten(),
#fully connected to get all relevant data
    Dense(128, activation='relu'),
#one more dropout for convergence' sake :)
    Dropout(0.5),
#output a softmax to squash the matrix into output probabilities
    Dense(2, activation='softmax')
])

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

model.fit_generator(train_batch, validation_data=valid_batch,epochs=15,steps_per_epoch=SPE ,validation_steps=VS)

model.save('models/cnnCat2.h5', overwrite=True)
```

8. RESULTS

We can see that from the following screenshots that the model works perfectly only in the absence of light its accuracy decreases drastically.





CONCLUSION

The model works with great accuracy until the the amount of light decreases,

REFERENCES

The information of this project have been collected from the following sources:

- www.google.com
- www.geeksforgeeks.com
- www.youtube.com
-

Self-performed

Also
Various other sites like tensorflow official documentation , towards data science have been used,