***A Project Report on***

# OBJECT DETECTION WEBAPP USING TENSORFLOW, OPENCV AND FLASK

***Submitted to***

## C:\Users\student\Desktop\jntua.pngJAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, ANANTHAPURAMU

*In partial fulfillment of requirement for the award of the degree of*

**Bachelor of Technology**

In

**Computer Science & Engineering**

*By*

## P.Sravani (17F11A0577)

**N.Sri Lavanya (17F11A0574)**

## P.Tejaswini (17F11A0588)

**V.G.Sowmya Sree (17F11A05B2)**

*Under the esteemed Guidance of*

# Dr. P. Venkateswara Rao Professor



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

NARAYANA ENGINEERING COLLEGE :: GUDUR

**(An ISO 9001:2008 Certified Institution, Approved by AICTE New Delhi &Permanently Affiliated to JNTUA, Ananthapuramu )**

**Accredited with “A” Grade by NAAC**

**Dhurjati Nagar, Gudur – 524101, SPSR Nellore Dt., A.P., India**

## [www.necg.ac.in](http://www.necg.ac.in/)

**2017-2021**

NARAYANA ENGINEERING COLLEGE :: GUDUR

**(An ISO 9001:2008 Certified Institution, Approved by AICTE New Delhi &Permanently Affiliated to JNTUA, Ananthapuramu )**

**Accredited with “A” Grade by NAAC**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**BONAFIDE CERTIFICATE**

This is to certify that the project entitled **OBJECT DETECTION WEBAPP USING TENSORFLOW, OPENCV AND FLASK** that is being submitted by **P. SRAVANI,REG NO.17F11A0577, N. SRI LAVANYA, REG NO.17F11A0574, P. TEJASWINI, REG**

**NO.17F11A0588, V.G. SOWMYA SREE, REG NO.17F11A05B2** in partial fulfillment of the requirements for the award of degree of **Bachelor of Technology** in **COMPUTER SCIENCE AND ENGINEERING** to JNTUA Ananthapuramu is recorded to be the bonafied work carriedout by her under my guidance and supervision.

## PROJECT GUIDE HEAD OF THE DEPARTMENT

Dr. P. Venkateswara Rao, Ph D Dr. P. Venkateswara Rao, Ph D

Professor. Professor.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

# ACKNOWLEDGEMENT

We extremely thankful to **Dr.P.Narayana**, the **Founder Chairman** of **Narayana Group** for his good initiation starting technical institution in Gudur like rural area for helping economically poor students. We also thankful to **Mr.K.Puneeth**, the **Chairman** of **NarayanaGroup** for providing the infrastructural facilities to work in, without this the work would not have been possible.

We would like to express our deep sense of gratitude to **Dr.V. Ravi prasad, Principal, Narayana Engineering College, Gudur** for his continuous effort in creating a competitive environment in our college and encouraging throughout this course.

We would like to convey our heartfelt thanks to **Dr. P.Venkateswara Rao, Professor& HOD** of Computer Science and Engineering for giving the opportunity to embark up on thistopic and for her continues encouragement throughout the preparation of the project.

We would like to thank our guide **Dr. P.Venkateswara Rao, Professor & HOD**, Department of CSE for his/her invaluable guidance, constant assistance, support, endurance and constructive suggestions for the betterment of the project.

We also wish to thank all the **staff members** of the Department of Computer Science & Engineering for helping us directly or indirectly in completing this project successfully.

Finally we are thankful to our **parents and friends** for their continued moral and material support throughout the course and in helping us to finalize the report.

**P.SRAVANI (17F11A0577) N.SRI LAVANYA(17F11A0574) P.TEJASWINI(17F11A0588) V.G.SOWMYA SREE (17F11A05B2)**

# DECLARATION

We hereby declare that the project entitled **OBJECT DETECTION WEBAPP USING TENSORFLOW, OPENCV AND FLASK** has been done by us under the guidance of **Dr. P.VENKATESWARA RAO**, **Professor ,** Department of Computer Science & Engineering. This project work has been submitted to **NARAYANA ENGINEERING COLLEGE, GUDUR** as a part of partial fulfillment of the requirements for the award of degreeof **Bachelor of Technology**.

We also declare that this project report has not been submitted at any time to another institute or University for the award of any degree.

## P. SRAVANI (17F11A0577) N.SRI LAVANYA (17F11A0574)

**P. TEJASWINI (17F11A0588)**

## V.G. SOWMYA SREE (17F11A05B2)

**Place: Gudur Date:**

# INDEX

|  |  |  |
| --- | --- | --- |
| **SL NO** | **CONTENT** | **PAGE NO** |
| **I** | **ABSTRACT** | **i** |
| **II** | **LIST OF FIGURES** | **ii** |
| **III** | **LIST OF TABLES** | **iii** |
| **IV** | **LIST OF ABBREVIATIONS** | **iv** |
| **1** | **INTRODUCTION** | **1** |
| **2** | **FEASIBILITY STUDY**   * 1. Technical Feasibility   2. Economical Feasibility   3. Social Feasibility | **3** |
| **3** | **SYSTEM ANALYSIS**   * 1. System Study and Environment   2. Deep Learning   3. Convolutional Neural Networks   4. Transfer Learning   5. Existing System      1. Disadvantages   6. Proposed System      1. Applications      2. Advantages & Limitations   7. System Requirements      1. Hardware Requirements      2. Software Requirements | **6** |

|  |  |  |
| --- | --- | --- |
| **4** | **SYSTEM DESIGN**   * 1. System Architecture   2. UML Diagrams      1. Use Case Diagram      2. Class Diagram      3. Component Diagram      4. Deployment Diagram      5. Sequence Diagram      6. Activity diagram      7. Dataflow diagram | **26** |
| **5** | **IMPLEMENTATION**   * 1. Modules      1. Transfer learning      2. Object Detection      3. Building the web interface | **35** |
| **6** | 1. **SYSTEM TESTING**    1. Unit Testing    2. Functional Testing    3. System Testing    4. Performance Testing    5. Integration Testing    6. Acceptance Testing    7. Test Cases | **50** |
| **7** | **RESULTS** | **55** |
| **8** | **CONCLUSION & FUTURE SCOPE** | **67** |
| **9** | **BIBLIOGRAPHY** | **71** |

**ABSTRACT**

Detecting and Naming Objects automatically is one of the heart of the human visual system. There are various advantages if there is an application which automatically detects the objects present in the image and name it automatically. In this paper, we presenta model based on CNN based neural networks which automatically detects the objects in the images and generates naming conventions for the images. It uses various pre-trained models to perform the task of detecting objects and uses CNN to generate the names. It uses Transfer Learning based pre-trained models for the task of object Detection.

The model is to detect visible objects in the image using Convolutional Neural Networks and to name them using pre-trained a model trained with Flickr dataset. The interface of the model is developed using flask rest API, which is a web development framework of python. The main use case of this project is to strengthen the medical diagnosis, improve the Human Computer Interaction, effective tracking of miscellaneous objects in surveillance, traffic checking, improving the vision of Robots etc,.

Object detection is one of the interesting and focused areas of Artificial Intelligence which has many challenges to pass on. Object detection involves various complex scenarios starting from picking the dataset, training the model, validating the model, creating pre-trained models to test the images ,detecting the images and finally naming the objects.

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIG.NO** | **NAMES** | **PAGE NO** |
| **1** | A Fully-connected multi-layer neural network | **8** |
| **2** | Convolutional Neural Network | **13** |
| **3** | System Architecture | **26** |
| **4** | Use Case diagram | **28** |
| **5** | Class diagram | **29** |
| **6** | Component diagram | **30** |
| **7** | Deployment diagram | **31** |
| **8** | Sequence diagram | **32** |
| **9** | Activity diagram | **33** |
| **10** | Data Flow diagram | **34** |
| **11** | Classifiers-Transfer Learning | **36** |
| **12** | Tribal Knowledge Growth Before Language | **37** |
| **13** | Tribal Knowledge Growth After Language | **37** |
| **14** | Architecture of Inception-v3 model | **39** |
| **15** | Architecture of VGG-16 model | **40** |
| **16** | Examples of MSCOCO dataset | **43** |
| **17** | Comparisons to other Detectors | **44** |
| **18** | Sample Images | **55** |
| **19** | Visual Studio Code Interface | **55** |
| **20** | Web Application Interface | **56** |
| **21** | User Image Upload | **56** |
| **22** | Detected Object | **57** |
| **23** | Before Detection Example 1 | **57** |
| **24** | After Detection Example 1 | **58** |
| **25** | Object Detection Example 2 | **58** |
| **26** | Object Detection Example 3 | **59** |
| **27** | Object Detection Example 4 | **59** |
| **28** | Before Detection Example 5 | **60** |
| **29** | After Detection Example 5 | **60** |
| **30** | Object Detection Example 6 | **61** |
| **31** | Object Detection Example 7 | **61** |
| **32** | Object Detection Example 8 | **62** |
| **33** | Console result for above images | **62** |
| **34** | Graph for Training and Validation Accuracy | **63** |
| **35** | Graphs for Accuracy and Loss | **64** |
| **36** | Graph for model Accuracy | **64** |
| **37** | Training Curve | **65** |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **NAME** | **PAGE NO.** |
| 1 | Unit Test Case for user image upload | 52 |
| 2 | Unit Test Case for naming the detected objects | 53 |
| 3 | Acceptance Test Case for detecting objects | 54 |
| 4 | Overall Success rates of detected images | 63 |

# LIST OF ABBREVATIONS

1. CNN - Convolutional Neural Network
2. OPENCV - Open Computer Vision.
3. NLTK - Natural Language Tool Kit
4. NLP - Natural Language Processing.
5. PYPI - Python Package Installer
6. TF - Tensorflow

# INTRODUCTION

Object detection has emerged as a challenging and important research area following advances in statistical language modelling and image recognition. The detecting the objects from images has various practical benefits, ranging from aiding the visually impaired, to enabling the automatic and cost-saving labelling of the millions of images uploaded to the Internet every day. The field also brings together state-of-the-art models in Natural Language Processing and Computer Vision, two of the major fields in Artificial Intelligence.

There are two main approaches to Object Detection: bottom-up and top-down. Bottom- up approaches, such as those by [1] [2] [3], generate items observed in an image. Top-down approaches, such as those by [4] [5] [6], attempt to generate a semantic representation of an image that is then decoded into a names using various architectures, such as Natural Language processing. The latter approach follows in the footsteps of recent advances in statistical machine translation, and the state- of-the-art models mostly adopt the top-down approach.

Our approach draws on the success of the top-down object detection models listed above.We use a deep convolutional neural network to generate a vectorized representation of an image, which then naming the objects.

One of the main challenges in the field of Object detection is overfitting the training data. This is because the largest datasets, such as the Microsoft Common Objects in Context (MSCOCO) dataset, only have 160000 labelled examples, from which any top- down architecture must learn

(a) a robust image representation, (b) a robust hidden-state based representation to capture image semantics and (c) language modelling for syntactically-sound oriented design for the unique purpose of naming the objects.

Our CNN architecture, modelled after the NIC architecture describedin [6]. We use a deep convolutional neural network to create a semantic representation of an image, which we then decode using a NLP. (Right) The vectorized imagerepresentation is fed into the network, followed by a special start of sentence token. The hidden state produced is then used by the detected the objects for the given image. Figures taken from [6] manifests itself in the memorization of inputs and the use of similar images which differ in their specific details.

To cope with this, recent advances in the field of Object detection have innovated atthe architecture-level, with the most successful model to date on the Microsoft Common Objects in Context competition using the basic architecture augmented with an attention mechanism [7]. This allows it to deal with the main challenge of top-down approaches, i.e.the inability to focus the objects and specific details in the image.

In this paper, we approach the problem via thorough hyper-parameter experimentation on the basic architecture. For most computer vision researchers the classification task has always been dominant in the field. Either it was a scene understanding in the pioneer 1960sor a traffic sign detection in the modern days, the task has been rooted in the soil of computer vision. It is not surprising that one of the most significant competition in the field comprises the image classification task among others.

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) awards annually the algorithm which is most successful at predicting the class of an image in its five estimates (known as top-5 error). For the record, the lowest top-5 classification error reached 28.2% at the ILSVRC2010 and 25.8% a year later, respectively [1]. Nonetheless, an unexpected breakthrough came in the year 2012 when Krizhevsky et al. [2] presented decades old algorithms [3, 4] enhanced by novel training techniques achieving so-far-not-seen results. In particular, the top-5 classification error was pushed to 16.4%.

At the latest contest in 2015, the lowest top-5 error was brought to 3.5%, drawing on the work of Krizhevsky et al. After this success, neural networks has revolutionized the field and brought in new challenges that had not been merely considerable before. One of those newly feasible techniques

* object detection- is discussed in this thesis.

In fact, as an arising discipline with promising potential, object detection still is an active area of research nowadays, striving to answer unsolved questions. Consecutively, sincethe field has not been entirely established yet, one must rely mainly on recently published papers and on-line lectures only. Considering recent work, we define object detecting as a task in which an algorithm describes a particular image with a statement.

# FEASIBILITY STUDY

Preliminary investigation examine project practicability, the chance the system arehelpful to the organization. The most objective of the practicability study is to check the Technical, Operational and Economical practicability for adding new modules and debugging previous running system. All system is possible if they're unlimited resourcesand infinite time. There are unit aspects within the practicability study portion of the preliminary investigation

* + Technical Feasibility
  + Economical Feasibility
  + Social Feasibility

## Technical Feasibility

The technical issue typically raised throughout the practicableness stage of the investigation includes the following:

* + - Does the mandatory technology exist to try to what's suggested?
    - Do the planned equipments have the technical capability to carry the info needed to usethe new system?
    - Will the planned system offer adequate response to inquiries, despite the amount or location of users?
    - Can the system be upgraded if developed?
    - Are there technical guarantees of accuracy, responsibleness, simple access and information security?

Earlier no system existed to cater to the requirements of ‘Secure Infrastructure Implementation System’. this system developed is technically possible. it's an internet primarily based interface for audit work flow at NIC-CSD. therefore it provides a simple access to the users.

The database’s purpose is to make, establish and maintain a work flow among numerous entities so as to facilitate all involved users in their numerous capacities or roles. Permission to the users would be granted supported the roles nominative. Therefore, itprovides the technical guarantee of accuracy, responsibleness and security. The package and laborious needs for the event of this project aren't several and area unit already out there in- house at NIC or area unit out there as free as open supply.

The work for the project is finished with this instrumentality and existing package technology. Necessary information measure exists for providing a quick feedback to the users nomatter the amount of user’s victimization the system.

## Economical Feasibility

A system is developed technically which are used if put in should still be an honest investment for the organization. within the economical practicableness, the event price in making the system is evaluated against the last word profit derived from the new systems. money advantages should equal or exceed the prices.

The system is economically possible. It doesn't need any addition hardware or code. Since the interface for this technique is developed mistreatment the prevailing resources and technologies out there at NIC, there's nominal expenditure and economical practicableness sure.

## Social Feasibility

Proposed comes square measure useful given that they will be clad into data system. That may meet the organizations in operation needs. Operational feasibleness aspects of the project square measure to be taken as a vital a part of the project implementation. a number of the vital problems raised square measure to check the operational feasibleness of a project includes the following: -

* + - Is there spare support for the management from the users?
    - Will the system be used and work properly if it's being developed and implemented?
    - Will there be any resistance from the user that may undermine the potential application benefits?

This system is targeted to be in accordance with the above-named problems. Beforehand,the management problems and user needs are taken into thought. Therefore there's absolute confidence of resistance from the users that may undermine the potential application edges. The well-planned style would make sure the optimum utilization of the pc resources and would facilitate within the improvement of performance standing.

# SYSTEM ANALYSIS

## System Study and Environment

In order to tackle the object detection task, recent work shows it is in one's interest to utilize neural networks [7]. This frequently used term dates back to 1950s when notions such as the Perceptron Learning Algorithm were introduced [8]. Modern neural networks draw on notions discovered in the era of a Perceptron. In this section, we first define a neuron asa fundamental part of modern neural networks. Then we elaborate on Convolutional Networks,

## Perceptron

For the purposes of this work, a perceptron is defined generally as it became a funda- mental part of modern neural networks and the notation is utilized further on. Thus, a perceptron is compounded of one neuron. The neuron's output, known as the activation a, is mapped by : RN ! R as follows:

a = (x) = (wT x + b) - - - - - - -  (1)

where x 2 RN is a feature vector, w 2 RN and b 2 R are weights and ( ) is a non- linear function. In case of the Perceptron, ( ) stands for

1 if z > 0

(z) =0 otherwise - - - - - - - - - - -  (2)

In other words, a perceptron is a non-linear function separating data into two classes each associated with either 1 or 0. A perceptron is parametrized by weights w and b. By setting proper weights, one e ects the output and the perceptron's behaviour for a given feature vector. Therefore, such weights trimming is essential, yet non-trivial task. In order to find the weights, a learning algorithm was introduced, named the Perceptron Learning Algorithm [8]. This algorithm has a limiting property such that successful learning is achievedif and only if the data are linearly separable which is a major drawback pointed out in by Minsky and Papert in 1969 [9]. For example, there is no vector w and bias b that would make a perceptron mimic the XOR function.

## Multi-Layer Neural Network

Taking the perceptron as inspiration, the XOR problem can be overcome by aligning neurons into layers and interconnecting those layers. This function is called a Feedforward Neural Network, an Artificial Neural Network or simply a Neural Network.

In a neural network, each layer comprises N neurons processing inputs coming from the previous layer and producing activations used later in the following layer. For the sakeof simplicity, it is now assumed that the number of neurons N is same for all layers. However,this varies very often, for example, usually the output layer consists of fewer neurons corresponding to the nature of the problem being solved. To conclude, the activations of the k-th layer (a(1k); a(2k); : : : ; a(Nk)) = a(k) 2 RN are each a function of the activations of the previouslayer

k=1, noted as a(k 1) 2 RN : a1 (k) = 1k (a(k 1)) a2(k) = 2k (a(k 1))

A(k)=(k) a(k 1)N (3)

where (ik)( ) is a function defining the properties of the i-th neuron in the layer k, often having the form similar to Eq. (1). However, there are exceptions that proved to be essential to modern neural networks designs [2, 7, 10 ,12]. We discuss these in the later sections.

In practice, to lower the complexity of a network, in the given layer k all the functions (ik)( ) are always of the same form, only distinct in weights. Therefore, it is convenient to use vector notation. This is done by simplifying Eq. (3) into the following form:

a(k) = (k)(a(k 1)) (4)

As an example, we show a neural network with one hidden layer. The network takes in a vector that is propagated forward into the hidden layer. The processes in the hidden layer are noted here as (1). Further, the activations of the hidden layer are again propagated, analogically, into the second layer (2) whose activations are the output of the network. The network's design is shown in Fig. 3. Formally, the network is fed with a feature vector x 2 RN producing a vector y 2 RM :

y = (x) = (2)( (1)(x)) (5)

where 1 : RN ! RH is called the hidden layer and 2 : RH ! RM is called the output layer.

Note that the number of neurons in the hidden layer H is a hyper-parameter.

Although the structure of the network in the example has been defined, there are still other hyper-parameters to be determined. For example, the form of the layer mappings (1) and

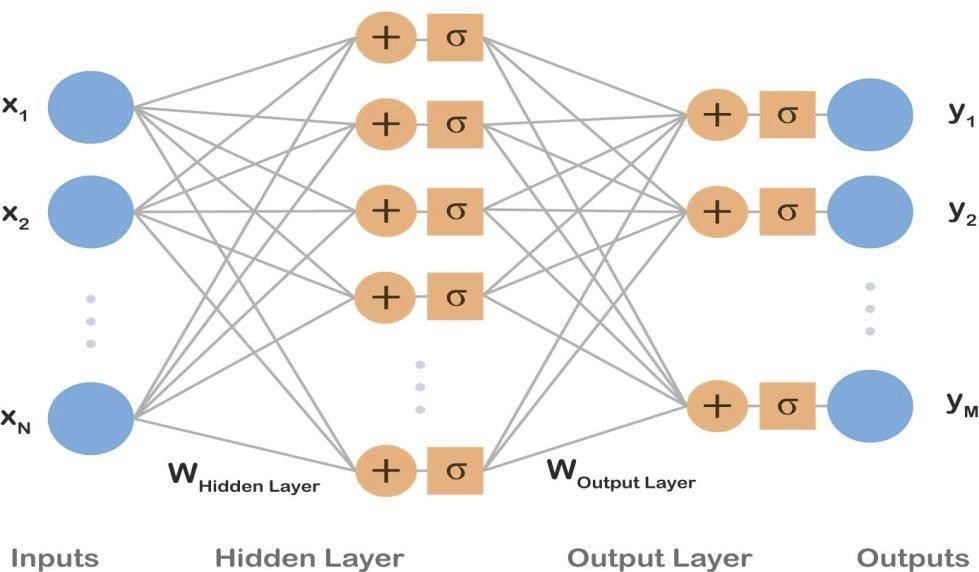
(2) needs to be specified. A layer with the most simple form of its mapping is called a Fully Connected Layer and is discussed in the following subsection.

## Fully Connected Layer

In the most basic neural network |a feed forward neural network comprising fully- connected layers only { each neuron processes activations of all neurons in the previous layer and is activated using ( ) defined in Eq. (1). Thus, based on vector notation in Eq. (4), the activations of the k-th fully-connected layer are de ned as

a(k) = (W(k 1;k)a(k 1) + b(k 1;k)) (6)

where W(k 1;k) 2 RN N is a weights matrix with weights vectors aligned in rows, b(k 1;k) 2 RN isa vector of biases and : RN ! RN is a non-linear function. Note that, since in practise the elements i( ) are identical single-variable functions, we refer to them as simply ( ).



## Figure 1: A fully-connected multi-layer neural network.

Inputs x1; : : : ; xN are processed by a hidden layer and, consecutively, by an output layer producing outputs y1; : : : ; yM . Biases b were omitted for the sake of simplicity.

In contrast to a perceptron, ( ) is generally required to be differentiable due to the nature of learning algorithms used in the field. For example, ( ) used to be set to a sigmoid curve as similar to the perceptron's activation function shown in Eq. (2). Most commonly, tanh() or thelogistic function (Eq. (7)) were used.

Z= 1+e z (7)

Nevertheless, when used in deep learning sigmoids suffer from problems such as vanishingor exploding gradients, therefore those were replaced with a Rectified Linear Unit (ReLU) [13]:

(z) = max(0; z) (8)

In modern networks, it is recommended to use ReLUs as they proved to provide better results and are thus the most common activation function used nowadays [7].

Drawing on the example presented above, we now assume that both layers are fully connected, meaning that layer mappings have a form of Eq. (6). Then Eq. (5) can be rewritten as follows:

y = (W(1;2)(W(0;1)x + b(0;1)) + b(1;2))  (9)

## Number of Parameters

Let us now assume x 2 RN , the activations of the hidden layer a 2 RH and y 2 RM . Then we can calculate the number of parameters as N +N H+H+H M. Considering a small gray- scale image, 28 28, of a hand-written number taken from the MNIST dataset [14], that is classi ed as 0-9 digit, N = 784 and M = 10. Then the number of parameters, needed to be learned, is 795 H + 784 where H, the number of hidden layers, is a hyper-parameter. For a hidden layer having the same width as the input vector, i.e. H = 28 in this example, the number of parameters reaches 23; 044.

Truly, this is a large number for such a shallow network suggesting that fully connected layers extensively increase the number of parameters.

## Hornik's Theorem

The network mentioned above is a common design that in past was believed to yield promising results. It was shown by Hornik [15], a multilayer feedforward neural network is able to approximate any continuous function that is bounded. Yet, a possibly great number of hidden neurons might be needed in order to do so because the theorem does not quantify this important hyper-parameter.

## Name Origin

As shown in Fig. 1, the structure is called a network since it can be drawn as a directed acyclic graph. Wondering about the name's background, one may notice the word neural and mistakenly assume a relation to biological neurons. However, as stated for example in [7], it is a common misconception as the name, neural networks, was derived in1950s from former biological models serving as motivation.

Those models are now, however, considered outdated, and conversely, modern neural networks are not designed to be realistic models, rather going beyond neuroscience perspective. Since that, endeavours to infer an algorithm from the brain's functioning have not faded.

For example, Je Hawkins et al. developed Hierarchical Temporal Memory (HTM) [16] as a strict mathematization of human neocortex based on current neuroscience. Internally, HTM as a biologically inspired algorithm is distinct from deep learning and, admittedly, its results have not been as fruitful as those obtained by deep nets [17], which are discussed int the following section.

Those models are now, however, considered outdated, and conversely, modern neural networks are not designed to be realistic models, rather going beyond neuroscience perspective. Since that, endeavours to infer an algorithm from the brain's functioning have not faded. These models are now, however, considered outdated, and conversely, modern neural networks are not designed to be realistic models, rather going beyond neuroscience perspective.

Since that, endeavours to infer an algorithm from the brain's functioning have not faded. For example, Je Hawkins et al. developed Hierarchical Temporal Memory as a strict mathematization of human neocortex based on current neuroscience. Internally, HTM as a biologically inspired algorithm is distinct from deep learning and, admittedly, its results have not been as fruitful as those obtained by deep nets , which are discussed in the following section.

Those models are now, however, considered outdated, and conversely, modern neural networks are not designed to be realistic models, rather going beyond neuroscience perspective. Since that, endeavours to infer an algorithm from the brain's functioning have not faded. For example, Je Hawkins et al. developed Hierarchical Temporal Memory (HTM) [16] as a strict mathematization of human neocortex based on current neuroscience.

Internally, HTM as a biologically inspired algorithm is distinct from deep learning and, admittedly, its results have not been as fruitful as those obtained by deep nets [17], which are discussed int the following section.

## Deep Learning

In spite of former beliefs, it was found that [7] it is more e client to insert severalhidden layers one by one and propagate information sequentially creating a deep structure, instead of utilizing a shallow network given in the example. This concept is called deep learning and, surprisingly, has its roots already in the pioneer 1960s as it was assumed that anintelligent algorithm solving complex problems shall work with hierarchy of concepts [7] that was rather deep. This is why we get the name deep learning.

The notion was later found in the idea of modern neural networks which, as stated above,consist of numerous nested layers each extracting more abstract and complex features as information is propagated forward the network. Therefore, the modern neural networks and techniques used for learning them are usually nowadays referred to as deep learning.

Deep neural networks were introduced already in 1998 [18] and the optimization algorithm (back-propagation) was known by then and used frequently [3]. Yet, the deep nets were found too complex to be trained. In their books, Ian Goodfellow etal. list those reason that enabled the boom of neural networks in 2012: rstly, more data were available as well, therefore, the deep nets have started to outperform other models.

Secondly, deeper models require decent architectures both in software and hardware and those had become available. Then on, promising results enabled advent of neural networks, especially in their deep form. Models such as convolutional neural networks or recurrent nets are considered state-of-the-art building blocks nowadays. Their detailed design is discussed in the following subsections.

## Convolutional Neural Networks

In image analysis, many of recent advances in deep learning are built on the work of LeCun et al. [18] who introduced a Convolutional Neural Network (CNN) which had a large impact on the field. A CNN is a type of a neural network that is designed to process an image and represent it with a vector code. The architecture of CNN draws on fully-connected neuralnetworks. Similarly, a convolutional neural network is a compounded structure of several layers processing signals and propagating them forward.

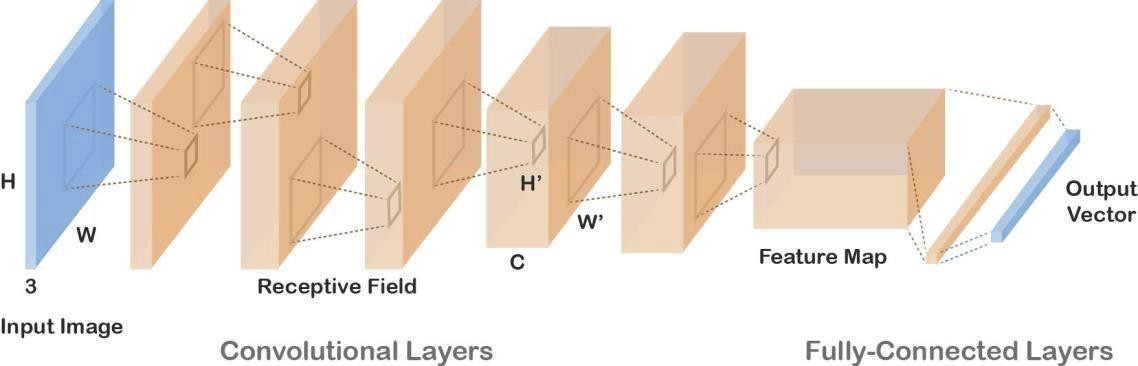
However, in contrast to a vector activation in a fully-connected layer, activations in CNNs have a shape of three-dimensional tensors. Commonly, this output tensor is called a feature map. For instance, an input image of shape 3 W H is transformed by the first convolutional layer into a feature map of shape C W 0 H0, where C is the number of features.

In other words, a convolutional layer transforms a volume into a volume.A typical CNN consists of several convolutional layers and, at the top, fully connected layers that convolutional volumes into a vector output. In the field's terminology, this vector code of an image is often called fc7 features as it used to be extracted from the seventh fully connected layer of AlexNet [2].

Convolutional layers are designed in such a way the spatial dimensions are preserved andthe depth is increased along the network flow. However, it is practical to reduce spatialdimensions, especially in higher layers. Dimensions reduction can be obtained by using stride when convolving, leading to dilution of receptive fields overlap.

Even though AlexNet has already been outperformed by many and the state-of-the-art designs are different from AlexNet, the term maintained its popularity. Additionally, depending on a problem the network is supposed to solve, an additional layer, such as soft-max, can be added on top of fc7 features. A common design of aCNN is depicted in Fig. 2.

## Receptive Field



**Fig 2:Convolutional Neural Network**

As mentioned above, a convolutional layer takes a tensor on input and produces a ten- sor, too. Note that these tensors have two spatial dimensions W and H, and one feature dimension C as they copy the form images are stored in. The context conveyed by the spatial dimensions is utilized in the CNN design which takes into account correlations in small areas of the input tensor called receptive fields. Concretely, in contrast to a neuron in a fully connectedlayer that processes all activations of the previous layer, a neuron in a convolutional layer "sees" only activations in its receptive field.

Instead of transforming layer's activations it restraints to a specific small rectangular shaped subset of the activations. When mentioninga receptive field, it is often expected only spatial dimensions of the input volume are referred to, i.e. a receptive field defines an area in the W H grid. The shape of the receptive field is a hyper-parameter and varies across the models.

A convolutional neural network takes an image on input (in blue) and transformsit into a vector code (in blue). Convolutional Neural Networks are characteristic for processingvolumes. An output of each layer is illustrated as an orange volume. Each neuron process onlyactivations in the previous layer that belong to its receptive field. The same set of weights is usedfor neurons across the whole grid. On top of convolutional layers, fully-connected layers are commonly connected.

## 3.3.2.Convolution in CNNs

A neuron's receptive field is processed similarly to fully connected layer neurons. The values below the receptive field along the input tensor's full depth are transformed by a non- linear function, typically ReLU (Eq. (8)).

However, in contrast to fully connected layer neurons, the same set of weights (referred to as a kernel) is used for all receptive fields in the input volume resulting into a transformationthat has a form of convolution across the input. A kernel is convolved across W and H spatial dimensions. Then, a different kernel is again convolved across the input volume producing another 2D tensor. Aligning up the output tensors into a C W 0 H0 volume assembles the layer's output feature map.

This is an important property of convolutional neural networks because each kernel detectsa specific feature in the input. For example, in the first layer, the first kernel would detect presenceof horizontal lines in the receptive fields, the second kernel would look for vertical lines, and similarly further on. In fact, learning such types of detectors in the bottom layers is typical for CNNs.

For example, in the first layer, the first kernel would detect presenceof horizontal lines in the receptive fields, the second kernel would look for vertical lines, and similarly further on. In fact, learning such types of detectors in the bottom layers is typical for CNNs.Fully Connected Layer is simply, [feed forward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network)*.* Fully Connected Layers form the last few layers in the network. The input to the fully connected layer is the output from the *final* Pooling or Convolutional Layer, which is flattened and then fed into the fully connected layer.

The design of CNNs has an immensely practical implication since a kernel is convolvedacross the input utilizing the same set of weights and it covers only the receptive eld, the numberof parameters is significantly reduced. Therefore, convolutional layers are less costly in terms of memory usage and the training time is shorter.

However, in contrast to fully connected layer neurons, the same set of weights (referred to as a kernel) is used for all receptive fields in the input volume resulting into a transformationthat has a form of convolution across the input. A kernel is convolved across W and H spatial dimensions. Then, a different kernel is again convolved across the input volume producing another 2D tensor. Aligning up the output tensors into a C W 0 H0 volume assembles the layer's output feature map.

This is an important property of convolutional neural networks because each kernel detectsa specific feature in the input. For example, in the first layer, the first kernel would detect presenceof horizontal lines in the receptive fields, the second kernel would look for vertical lines, and similarly further on. In fact, learning such types of detectors in the bottom layers is typical for CNNs.

The design of CNNs has an immensely practical implication since a kernel is convolvedacross the input utilizing the same set of weights and it covers only the receptive eld, the numberof parameters is significantly reduced. Therefore, convolutional layers are less costly in terms of memory usage and the training time is shorter.

## 3.3.3 Pooling Layer

Convolutional layers are designed in such a way the spatial dimensions are preserved andthe depth is increased along the network flow. However, it is practical to reduce spatialdimensions, especially in higher layers. Dimensions reduction can be obtained by using stride when convolving, leading to dilution of receptive fields overlap. Nevertheless, a more straightforward technique was developed called a pooling layer. An input is partitioned into non-overlapping rectangles and the layer simply outputs a grid of maximum values of each rectangle. In practice, pooling layers are inserted often in between convolutional layers to reduce dimensionality.

## Transfer Learning

Transfer Learning is computer vision’s popular method because it allows us to build accurate Models in a time saving manner. With transfer learning ,instead of starting the learning process from scratch ,you start from patterns that have been learned when solving a different problem.

In this phase, Transfer learning methodology is used to extract the previously used knowledge. We have used pre-trained model n to detect the objectsfrom the image , which contains the functionality of convolutional neural network *.*

## Existing System

In the last 5 years, a large number of articles havebeen published on detecting objects with deep machine learning being popularly used. Deep learning algorithms can handle complexities and challenges of detecting objects quite well. So far, only three survey papers [8, 13, 75] have been published on this research topic. Although the papers have presented a good literature survey of object detecting, they could only cover a few papers on deep learning because the bulk of them was published after the survey papers.

These survey papers mainly discussed template based, retrieval based,and a very few deep learning-based novel object detection models. However, a large number of works have been done on deep learning-based object detection. Moreover, the availability of large and new datasets has made the learning-based object detection an interesting research area. To provide an abridged version of the literature, we present a surveymainly focusing on the deep learning-based papers on image captioning.

## A.R-CNN

To circumvent the problem of selecting a huge number of regions, Ross Girshick et al. proposed a method where we use the selective search for extract just 2000 regions from the image and he called them region proposals. Therefore, instead of trying to classify the huge number of regions, you can just work with 2000 regions. These 2000 region proposals are generated by using the selective search algorithm which written below.

Selective Search:

1. Generate the initialsub-segmentation, we generate many candidate regions .
2. Use the greedy algorithm to recursively combine similar regions into larger ones .
3. Use generated regions to produce the final candidate region proposal.

## B.Fast R-CNN

The same author of the previous paper(R-CNN) solved some of the drawbacks of R-CNN to build a faster object detection algorithm and it was called Fast R-CNN. The approach is similar to the R-CNN algorithm. But, instead of feeding the region proposals to the CNN, we feed the input image to the CNN to generate a convolutional feature map.

From the convolutional feature map, we can identify the region of the proposals and warp them into the squares and by using an RoI pooling layer we reshape them into the fixed size so that it can be fed into a fully connected layer. From the RoI feature vector, we can use a softmax layer to predict the class of the proposed region and also the offset values for the bounding box.

## C.YOLO –You Only Look Once

All the previous object detection algorithms have used regions to localize the object within the image. The network does not look at the complete image. Instead, parts of the image which has high probabilities of containing the object. YOLO or You Only Look Once is an object detection algorithm much is different from the region based algorithms which seen above. In YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes.

YOLO works by taking an image and split it into an SxS grid, within each of the grid we take m bounding boxes. For each of the bounding box, the network gives an output a class probability and offset values for the bounding box. The bounding boxes have the class probability above a threshold value is selected and used to locate the object within the image. YOLO is orders of magnitude faster(45 frames per second) than any other object detection algorithms.

The limitation of YOLO algorithm is that it struggles with the small objects within the image, for example, it might have difficulties in identifying a flock of birds. This is due to the spatial constraints of the algorithm.

## 3.5.1.Disadvantages

The problem of object detection is a complex and widely interested research topic since the evolution of deep learning. There are many proposed solutions for this problem which are replacing the previous solutions every single day. In [1] Karpathy proposed a system which uses multimodel neural networks to detect the objects in an image. In [2], Deng proposed a model which uses a database called ImageNet which is build using the core called WordNet.

This model uses ImageNet to generate names from the image.Kelvin at el [3] proposed an attention based model, which generate names of the images based on the region of interest. It generates the names based on the region the image is surrounded. In [4] , Yang proposed a multimodal neural network based model, which detecting the objects and naming the objects in an image, . which is almost similar to human visual system .

In [5], Aneja proposed a convolutional neural network based modal to detect the objects from the imageafter the rigorous training given to the model. In [6], Pan proposed a multiple neural network model, which is experimented with large sets of datasets to generate the accurateobjects from the image. In [7], Vinyals proposed a model that uses Natural Language Processing and Computer Vision to detect the objects in the image and name objects based on word processing.

It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image. It cannot be implemented real time as it takes around 47 seconds for each test image. The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.

You can infer that Fast R-CNN is significantly faster in training and testing sessions over R- CNN. When you look at the performance of Fast R-CNN during testing time, including region proposals slows down the algorithm significantly when compared to not using region proposals. Therefore, the region which is proposals become bottlenecks in Fast R-CNN algorithm affecting its performance.

Both of the above algorithms(R-CNN & Fast R-CNN) uses selective search to find out the region proposals. Selective search is the slow and time-consuming process which affect the performance of the network. Similar to Fast R-CNN, the image is provided as an input to a convolutional network which provides a convolutional feature map.

Instead of using the selective search algorithm for the feature map to identify the region proposals, a separate network is used to predict the proposals. The predicted the region which is proposals are then reshaped using an RoI pooling layer which is used to classify the image within the proposed region and predict the offset values for the bounding boxes.

In [6], Pan proposed a multiple neural network model, which is experimented with large sets of datasets to generate the accurateobjects from the image. In [7], Vinyals proposed a model that uses Natural Language Processing and Computer Vision to detect the objects in the image and name objects based on word processing.

## 3.6.Proposed System

Our model uses the neural networks to detect the objects. The neural network is Convolutional Neural Network(CNN), which is used to train the images as well as to detect theobjects in the image with the help of various pre-trained models like VGG, Inception or YOLO. The natural language processing used to understand the human language.

As, there is lot of data involved to train and validate the model, generalized machine learning algorithms will not work. Deep Learning has been evolved from the recent times to solve the data constraints on Machine Learning algorithms. GPU based computing is required to perform the Deep Learning tasks more effectively.

Object popularity is to explain a set of associated laptop imaginative and prescient duties that contain sports like ﬁguring out items in virtual photographs. Image class includes sports including predicting the magniﬁcence of 1 item in an photograph. Object localization is refers to ﬁguring out the area of 1 or extra items in an photograph and drawing an abounding ﬁeld round their extent.

Object detection does the paintings of combines those duties and localizes and classiﬁes one or extra items in an photograph. When a consumer or practitioner refers back to the term “item popularity“, they frequently mean “item detection“. It can be hard for novices to diﬀerentiate among distinctive associated laptop imaginative and prescient duties.So, we are able to distinguish among those 3 laptop imaginative and prescient duties with this example:

Image Classiﬁcation: This is executed with the aid of using Predict the kind or magniﬁcence of an item in an photograph.

Input: An photograph which includes a unmarried item, including a photograph.

Output: A magniﬁcence label (e.g. one or extra integers that are mapped to magniﬁcence labels). Object Localization: This is executed through, Locate the presence of items in an photograph and suggest their area with a bounding ﬁeld.

Input: An photograph which includes one or extra items, including a photograph. Output: One or extra bounding boxes (e.g. described with the aid of using a point, width, and height).

Object Detection: This is executed through, Locate the presence of items with a bounding ﬁeld and brands or lessons of the positioned items in an photograph.

Output: One or extra bounding boxes (e.g. described with the aid of using a point, width, and height), and a category label for every bounding ﬁeld.

One of the in addition extension to this breakdown of laptop imaginative and prescient duties is item segmentation, additionally called “item example segmentation” or “semantic segmentation,” wherein times of identiﬁed items are indicated with the aid of using highlighting the unique pixels of the item rather than a rough bounding ﬁeld. From this breakdown, we are able to apprehend that item popularity refers to a collection of hard laptop imaginative and prescient duties.

For example, photograph class is virtually immediately forward, however the variations among item localization and item detection may be confusing, mainly whilst all 3 duties can be simply as similarly called item popularity.

Object popularity refers to a set of associated duties for ﬁguring out items in virtual photographs. Region-primarily based totally Convolutional Neural Networks, or R-CNNs, is a own circle of relatives of strategies for addressing item localization and popularity duties, designed for version performance. You Only Look Once, or YOLO is called the second one own circle of relatives of strategies for item popularity designed for pace and real-time use.

Object detection is an important task, yet challenging vision task. It is a critical part of many applications such as image search, image auto-annotation and scene understanding, object tracking. Moving object detection of video image sequences was one of the most important subjects in computer vision. It had already been applied in many computer vision fields, such as smart video surveillance (Arun Hampapur 2005), artificial intelligence, military guidance, safety detection and robot navigation, medical and biological application.

In recent years, a number of successful single-object detection system appeared, but in the presence of several objects, object detection becomes difficult and when objects are fully or partially occluded, they are obtruded from the human vision which further increases the problem of detection. Decreasing illumination and acquisition angle.

Object detection is a computer technology that is related to image processing and computer vision. The technology deals with detecting the instances of the semantic objects of different classes like building, human beings, cars, and others in videos and digital images. Some of the domains of object detection that have gone through proper research are pedestrian detection and face detection. There are numerous applications of object detection in areas like image retrieval, computer vision, and video surveillance.

Some of the major applications of object detection are related to computer vision and include face recognition, video object co-segmentation, etc. It is used in instances like tracking objects, tracking a person in a video, tracking the movement of a cricket bat, and many more.

People often confuse image classification with object detection. When the main aim is to classify the image into a certain category, image classification is used. On the other hand, to identify the location of the objects in an image or count the number of instances of an object, object detection is to be used. Labelled data is needed in order to train a custom model. The labelled data in the context of object detection are images that have corresponding labels and bounding box coordinates.

In a typical object detection algorithm, an image is sent to the network, which is then sent through lots of convolutions and pooling layers. The output would be an object of the class. For each input image, there is a corresponding class as output. After taking the image as an input, the image is divided into various regions.

Each of these regions is considered a separate image. The regions are then passed to the Convolution Neural Networks (CNN) to classify them into various classes. Once each of the regions has been divided into corresponding classes, all the regions are combined to get the original image with the detected objects.However, there are some problems with such trivial algorithms as the images might have different aspect ratios and spatial locations.

These factors could lead to a large number of regions and the computational time would increase. In this section, the proposed method for detecting the items in real-time from photographs through the use of convolutional neural network(CNN) deep gaining knowledge of procedure for that we've got used OpenCV libraries. Our version makes use of Convolutional Neural Network(CNN), that is used to teach the photographs in addition to to locate the items with inside the photograph with the assist of numerous pre-skilled fashions like VGG, YOLO.

As, there's lot of records worried to teach and validate the version, generalized system gaining knowledge of algorithms will now no longer work. Deep Learning has been developed from the latest instances to clear up the records constraints on Machine Learning algorithms. GPU primarily based totally computing is needed to carry out the Deep Learning responsibilities greater eﬀectively. The proposed method includes 3 modules i.e, growing pre-skilled version(Transfer Learning),Object detection and deployment to internet server.Transfer gaining knowledge of is pc vision’s famous method,as it permits us to construct correct fashions in a timesaving manner.

With switch gaining knowledge of, as opposed to beginning the gaining knowledge of procedure from scratch, you begin from styles which have been found out while ﬁxing a one of a kind problems.In this module, Convolutional Neural Network plays the challenge of Object Detection from the photographs. In this phase, Transfer gaining knowledge of method is used to extract the formerlyused knowledge.

We have used pre-skilled version named ssd\_mobilenet\_coco to locate the items from the photograph, which incorporates the capability of convolutional neural community.In this module, Used FLASK to installation our undertaking as a REST-API withinside the shape of an internet software. FLASK is an internet software framework of PYTHON used by and large to installation system gaining knowledge of fashions

## Applications of Object Detection

Object Detection has a lot of real-life applications and can be used in different scenarios. New algorithms and models keep on outperforming the previous ones and object detection is one of the areas of computer vision which is maturing very rapidly. Below here are its applications.

## Face Recognition

For instance, a group of researchers at Facebook had developed the DeepFace, which is a facial recognition system based on deep learning. Google also has its own facial recognition system which can automatically segregate the photos based on the person in the images.

Object Detection is one of the computer technologies that is connected to image processing and computer vision. It detects the instances of an object like building, human faces, cars, trees, and others. The primary job of face detection is to ensure whether there is any face in the image. face detection is the first and most essential step and it detects the faces in images. It is used in areas like security, law enforcement, biometrics, personal safety, and entertainment.

Faces can be detected in real-time and it helps to track persons or objects. The face detection methods can be appearance-based, feature-based, knowledge-based, or template matching.

## People Counting

Another important use of object detection is people counting. It can be used for analyzing store performance or recording crowd statistics during festivals or other activities. However, it can be difficult at times as people move out of the frames very quickly.

Off-the-shelf people counters are not very expensive but the data generated by them is tied to proprietary systems that limit the options for data extraction and KPI optimization. An embedded DIP using your own camera and SBC would save time and money and offer the freedom to tailor the application to the KPIs you need. Insights can be extracted from the cloud that would not be possible in other cases.

The overall functionality for your DIP IoT application can be enhanced using the cloud. The [visualization](https://mindmajix.com/data-visualization-for-business), alerting, reporting offer increased capabilities and so do the cross-referencing outside data sources.

## Industrial Quality Check

Object Detection is often used in industrial processes to identify products. Using visual inspection to find a specific object is a basic task and it is involved in various industrial processes. This includes inventory management, sorting, quality management, machining, and packaging. Inventory management is sometimes quite tricky as it could be hard to track items in real-time. Localization and automatic object counting allow improving inventory accuracy.

Several challenges need to be taken into account while object detection is being performed. The objects come in different sizes, shapes, colors, and orientation. There is additional noise which occurs through variation in illumination, viewpoint, shadows, and occlusions. Ensuring the desired accuracy is important without arranging too many training examples.

## Self-Driving Cars

Self-driving cars are something evident in the future. However, the working is very tricky as a lot of different techniques are required to perceive the surroundings like laser light, GPS, radar, computer vision, and odometry. Sensory information is interpreted to identify appropriate navigation paths and obstructions with the help of advanced control systems. When a sign of a living being is found in the path, the car automatically stops. The process is very fast and is a huge step towards Self-Driving cars.

Self-driving cars are being designed with the intention to save lives. A lot of people are involved in road accidents every year. Autonomous vehicles allow accurate and safer transportation and needless death tools are lowered. Object detection is performed in two steps - image classification and image localization. Image classification determines what the objects look like and image localization provides the specific location of the objects.

## Security

A very important role is played by Object Detection in terms of Security. It is used by police personnel to access security feed and match with the existing database. It helps to detect criminals or their vehicles. It can even be used to locate stolen products. There could be limitless applications. The abilities of a machine to look out for objects have surpassed the capabilities of human beings.

Using technology to perform surveillance is a lot more efficient. As surveillance is a repetitive and mundane task, performance dips can result in human beings. Letting technology do the task can help human beings to focus on the actions to be taken if something goes wrong. A lot of personnel might be needed to survey a large strip of land. Mobile surveillance bots, along with stationary cameras can mitigate the problems.

## Advantages

There are various advantages of Object detection in multiple disciplines.

* + - * It can be used for visually impaired people to understand the environment.
      * It can be used in areas where text is more used and it can be used to infer text fromimages.
      * Image captioning can also be used in self driving cars.
      * It can be used by social networks to describe the image being uploaded by the user.
      * It can be used in various NLP applications, where insights and summary is neededfrom the images.

## System Requirements

The following are the software and hardware requirements:

## Software Requirements

Language : Python 3.x

Database : Flickr8k

Operating System : OS Independent

IDE : Visual Studio Code

Front End : BOOTSTRAP, HTML and CSS

## Hardware Requirements

Processor : Intel I3

Speed : 1.6 Ghz

RAM : 4GB (min)

Hard Disk : 500 GB

## Deployment Tools

Flask Rest API Flask-Python

# SYSTEM DESIGN

## 4.1.System Architecture

Implemen ted using keras with Tensor Flow backend with python



**image**

**Input**

**Deployment**

**Output**

NLTK(Data processing and vocabulary training

CNN(VGG16

or inception4 or pretrained model

User interface build using Flask rest API of python

Detected objects

Text

Dataset

Image Dataset

## Flickr or mscoco Dataset

**Fig 3: System Architecture**

## UML Diagrams

**Unified Modeling Language**:

The Unified Modeling Language permits the technologist to specific AN analysis model mistreatment the modeling notation that's ruled by a group of grammar linguistics and pragmatic rules.

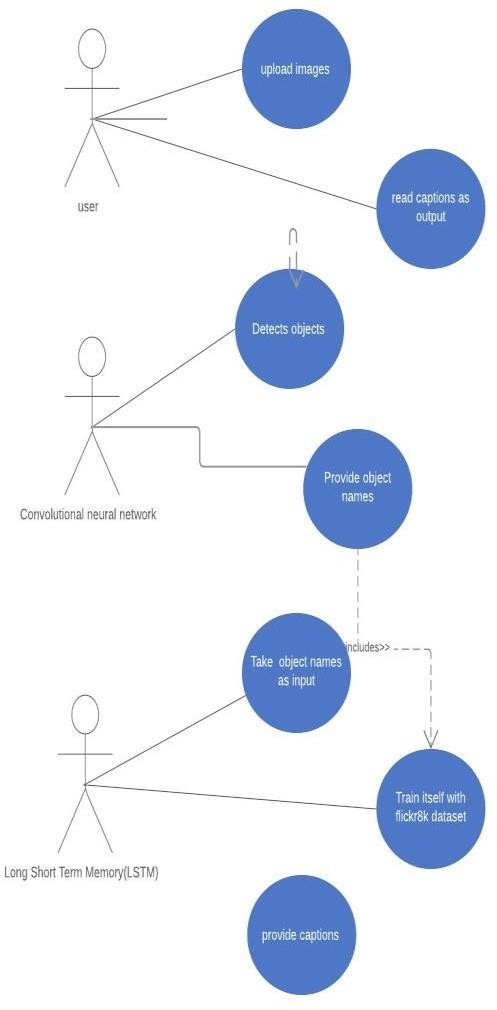
A UML system is diagrammatical mistreatment 5 completely different views that describethe system from clearly different perspective. Every read is outlined by a group of diagram, that is as follows.

It represents the dynamic of behavioral as elements of the system, portrayal the interactions of assortment between varied structural components delineated within the user model and structural model read.

Use case Diagrams represent the practicality of the system from a user’s purpose of read. Use cases are used throughout needs induction and analysis to represent the practicality of thesystem. Use cases specialize in the behavior of the system from external purpose of read.

Actors are external entities that move with the system. Samples of actors embody users like administrator, bank client …etc., or another system like central info.

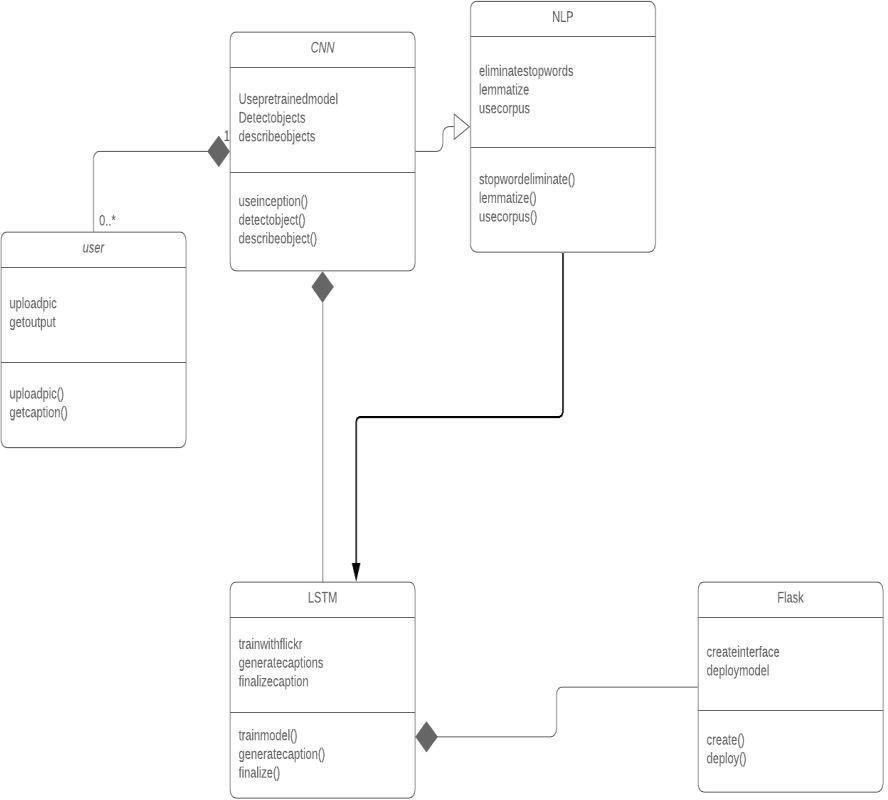
## Use Case Diagrams:

Use case diagrams model the practicality of system treatment actors and use cases.Use cases are services or functions provided by the system to its users.

## Fig 4: Use Case diagram

* + 1. **Class Diagram:**

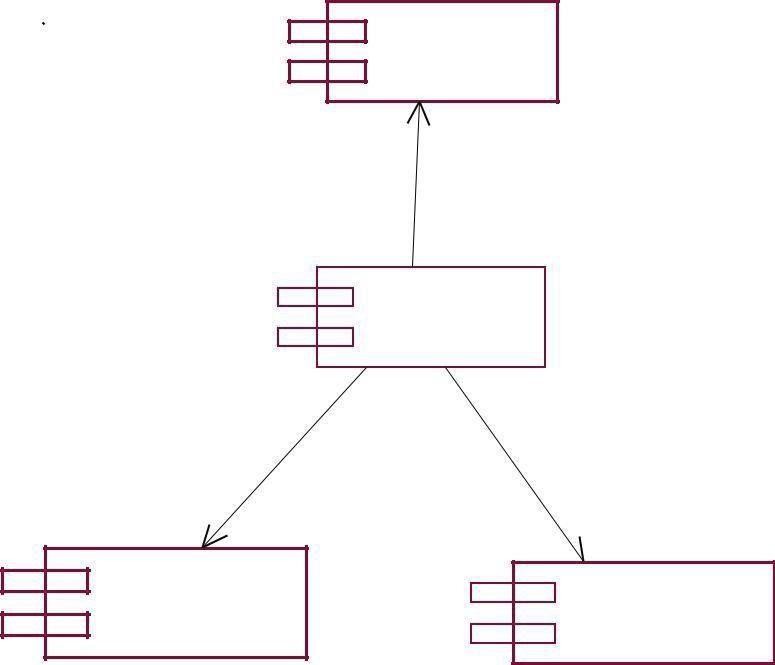
Class diagrams are the backbone of virtually each object-oriented methodology aswell as UML. They describe the static structure of a system. Categories represent associate degree abstraction of entities with common characteristics. Associations represent the relationships between categories.



## Fig 5: Class Diagram

* + 1. **Component Diagram:**

An element diagram describes the organization of the physical elements in a very system. An element could be a physical building block of the system. it's pictured as a parallelogram with tabs.



FLICK R8K DATA SET

SERVER

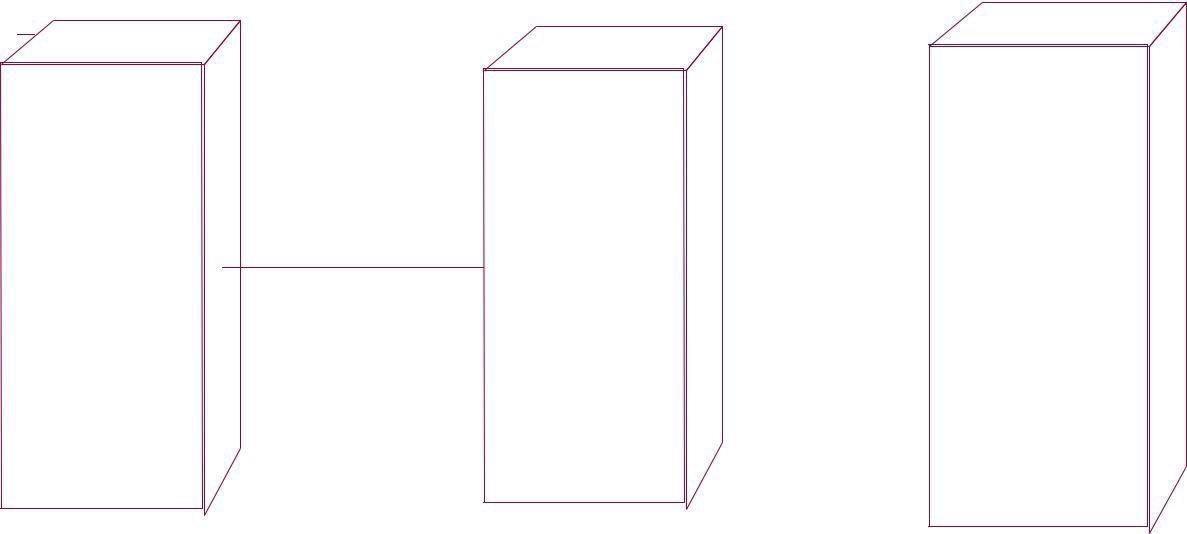
HTML/ BOOTSTRAP

FLASK

## Fig 6: Component Diagram

* + 1. **Deployment Diagram:**

Deployment diagrams depict the physical resources in an exceedingly system as well asnodes, components, and connections. A node may be a physical resource that executes code parts.



HTML/ CSS

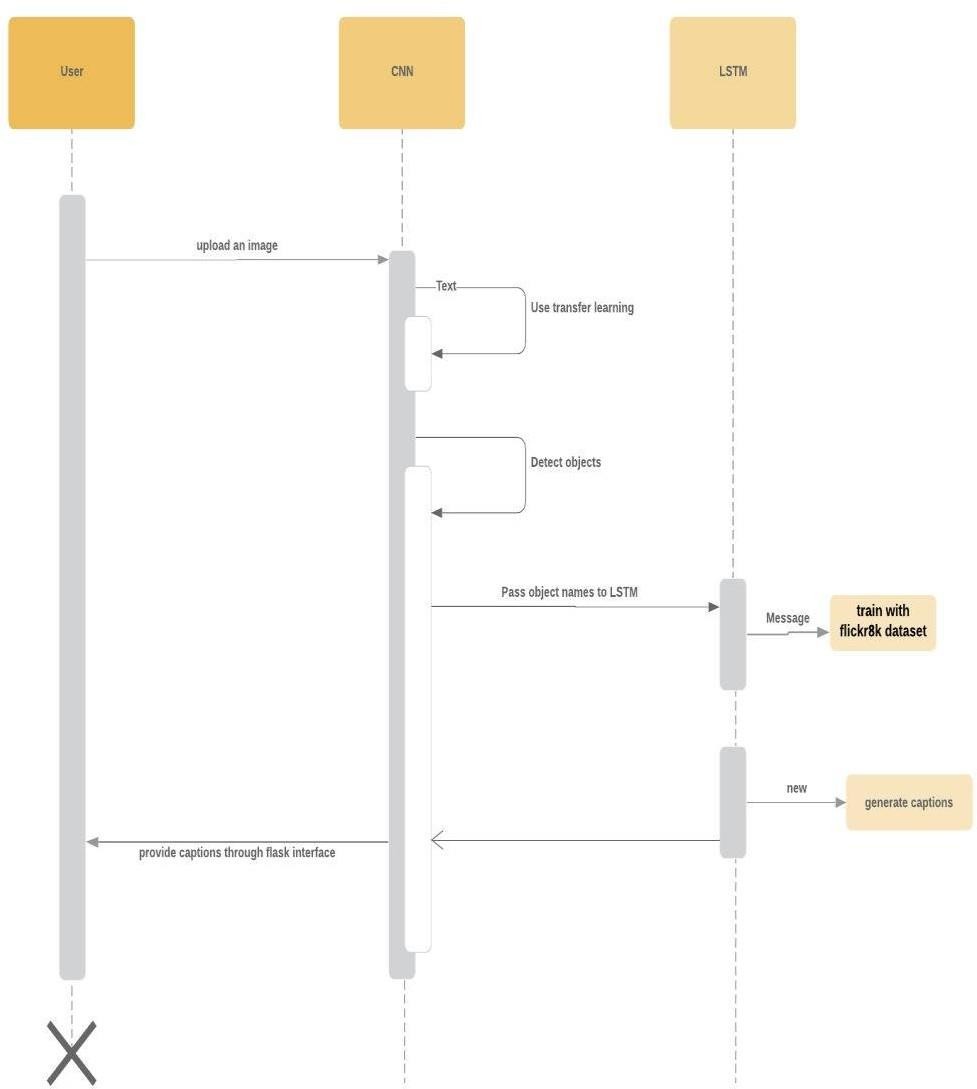
FLICKR8K

FLASK WEB SERVER

DATASET

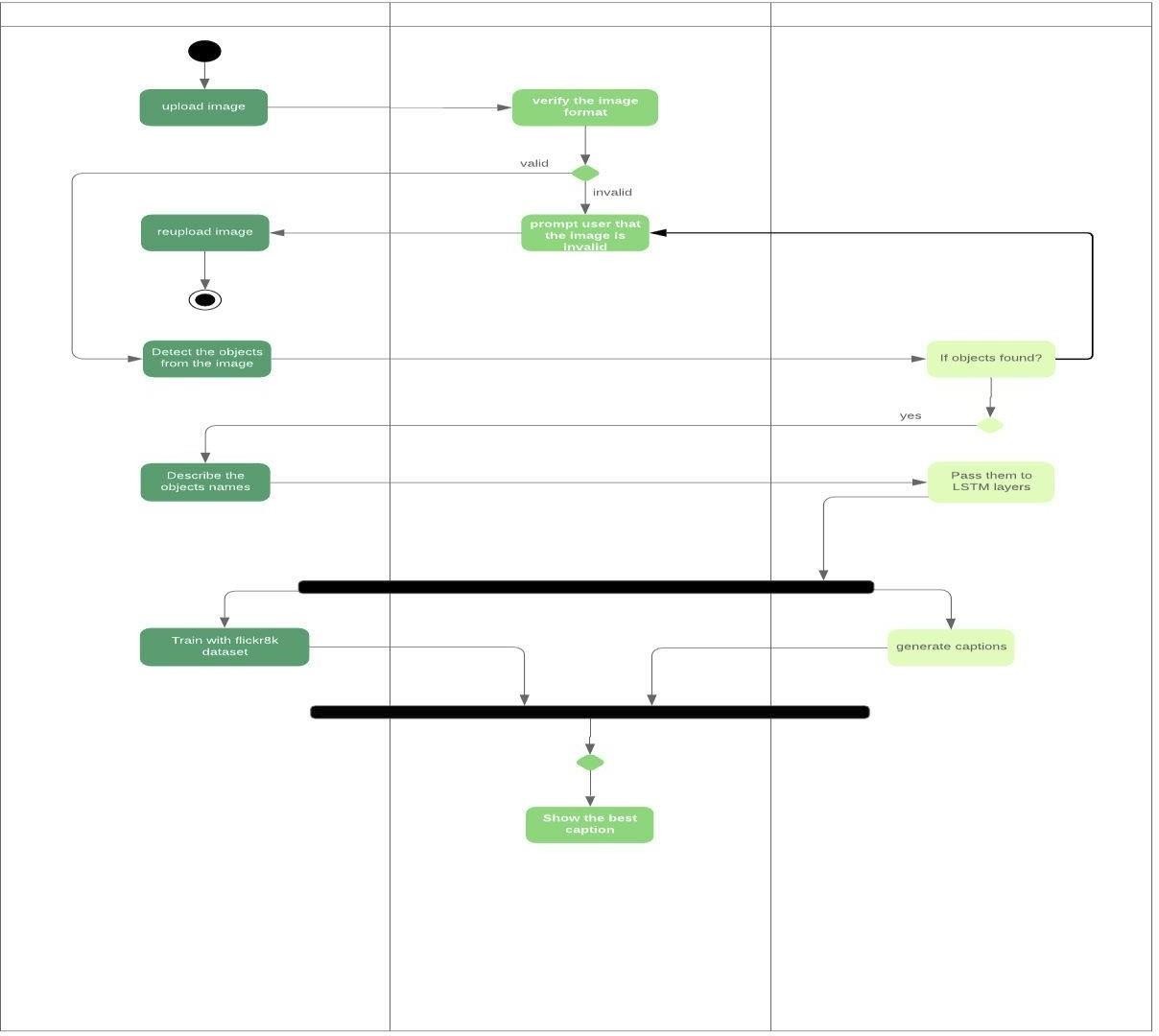
## Fig 7: Deployment Diagram

* + 1. **Sequence Diagram:**

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged betweenthe objects needed to carry out the functionality of the scenario

**Fig 8: Sequence Diagram**

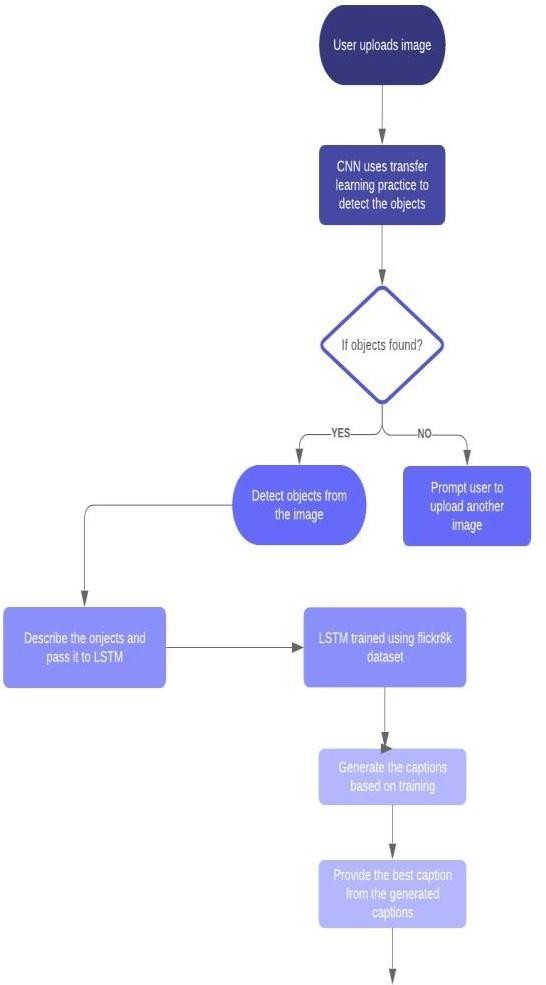
## Activity Diagram:

An activity diagram illustrates the dynamic nature of a system by modeling the flow of management from activity to activity. An activity represents AN operation on some category within the system that leads to an amendment within the state of the system. Typically, activity diagrams are accustomed model progress or business processes and internal operation. As a result of AN activity diagram may be a specialquite state chart diagram, it uses a number of constant modeling conventions.

## Fig 9: Activity Diagram

* + 1. **Dataflow Diagram:**

A data-flow diagram is a way of representing a flow of a data of a process or a system The DFD also provides information about the outputs and inputs of each entity and the process itself. A data- flow diagram has no control flow, there are no decision rules and no loops.



**Fig 10: Data Flow Diagram**

# IMPLEMENTATION

## Modules

* + 1. **Transfer learning (Using the Pre-trained model)**

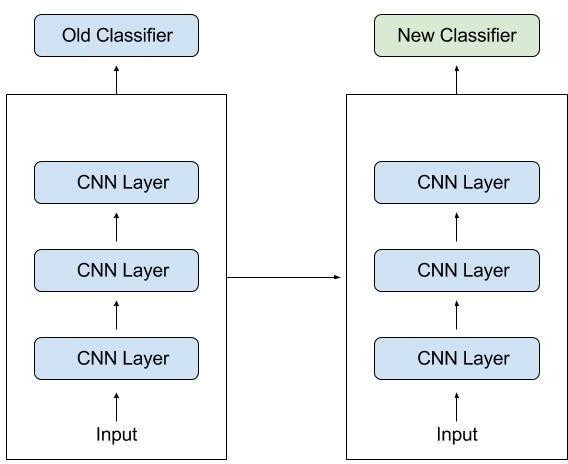
In this phase, Transfer learning methodology is used to extract the previouslyused knowledge. We have used pre-trained model named ssd\_mobilenet\_coco to detect the objects from the image , which contains the functionality of convolutional neural network. Transfer learning is a machine learning method where a model developed for a task is reused at the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

In computer vision, for example, neural networks usually try to detect edges in the earlier layers, shapes in the middle layer and some task-specific features in the later layers. In transfer learning, the early and middle layers are used and we only retrain the latter layers. It helps leverage the labeled data of the task it was initially trained on.

Let’s go back to the example of a model trained for recognizing a backpack on an image, which will be used to identify sunglasses. In the earlier layers, the model has learned to recognize objects, because of that we will only retrain the latter layers so it will learn what separates sunglasses from other objects.

In transfer learning, we try to transfer as much knowledge as possible from the previous task the model was trained on to the new task at hand. This knowledge can be in various forms depending on the problem and the data. For example, it could be how models are composed, which allows us to more easily identify novel objects.



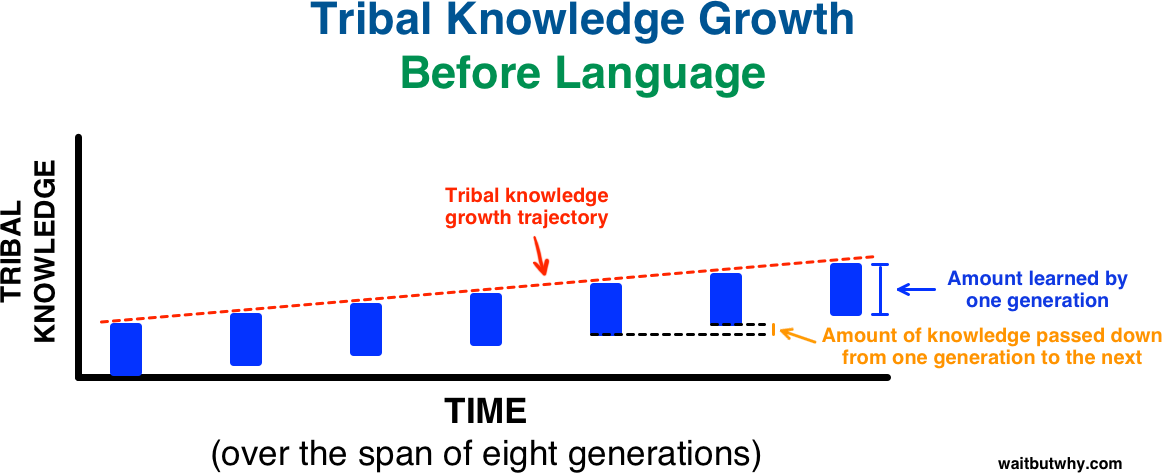
## Fig 11 :Classifiers-Transfer Learning

Transfer learning has several benefits, but the main advantages are saving training time, better performance of neural networks (in most cases), and not needing a lot of data. Usually, a lot of data is needed to train a neural network from scratch but access to that data isn't always available — this is where transfer learning comes in handy. With transfer learning a solid machine learning model can be built with comparatively little training data because the model is already pre-trained.

This is especially valuable in natural language processing because mostly expert knowledge is required to create large labeled datasets. Additionally, training time is reduced because it can sometimes take days or even weeks to train a deep neural network from scratch on a complex task. There are a lot of these models out there, so make sure to do a little research. How many layers to reuse and how many to retrain depends on the problem.

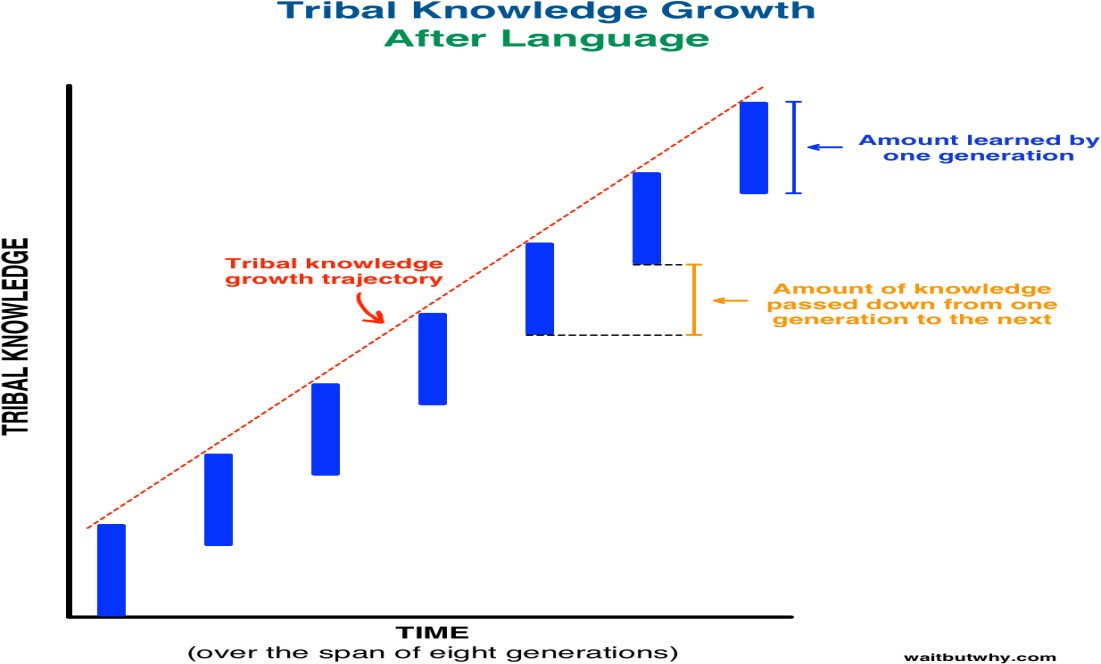
Keras, for example, provides nine pre-trained models that can be used for transfer learning, prediction, feature extraction and fine-tuning. You can find these models, and also some brief tutorials on how to use them, [here.](https://keras.io/applications/) There are also many research institutions that release trained models. This type of transfer learning is most commonly used throughout deep learning.

Tim explains that before language was invented, every generation of humans had to re-invent the knowledge for themselves and this is how knowledge growth was happening from one generation to other:



## Fig 12:Tribal Knowledge Growth Before Language

Then, we invented language ! A way to transfer learning from one generation to another this is what happened over same time frame:



## Fig 13:Tribal Knowledge Growth After Language

By using pre-trained models which have been previously trained on large datasets, we can directly use the weights and architecture obtained and apply the learning on our problem statement. This is known as transfer learning. We “transfer the learning” model to our specific problem statement.

You should be very careful while choosing what pre-trained model you should use in your case. If the problem statement we have at hand is very different from the one on which the pre- trained model was trained – the prediction we would get would be very inaccurate. For example, a model previously trained for speech recognition would work horribly if we try to use it to identify objects using it.

The pre-trained architectures are directly available for us in the Keras library. **Imagenet** data set has been widely used to build various architectures since it is large enough (1.2M images) to create a generalized model. The problem statement is to train a model that can correctly classify the images into 1,000 separate object categories. These 1,000 image categories represent object classes that we come across in our day-to-day lives, such as species of dogs, cats, various household objects, vehicle types etc.

These pre-trained networks demonstrate a strong ability to generalize to images outside the ImageNet dataset via transfer learning. We make modifications in the pre-existing model by fine- tuning the model. Since we assume that the pre-trained network has been trained quite well, we would not want to modify the weights too soon and too much. While modifying we generally use a learning rate smaller than the one used for initially training the model.

We can use a pre-trained model as a feature extraction mechanism. What we can do is that we can remove the output layer( the one which gives the probabilities for being in each of the 1000 classes) and then use the entire network as a fixed feature extractor for the new data set. What we can do is that we use architecture of the model while we initialize all the weights randomly and train the model according to our dataset again.

If the original model was trained using TensorFlow, you can simply restore it and retrain some layers for your task. Keep in mind, however, that transfer learning only works if the features learned from the first task are general, meaning they can be useful for another related task as well. In machine learning, features are usually manually hand-crafted by researchers and domain experts. Fortunately, deep learning can extract features automatically.

Of course, this doesn't mean feature engineering and domain knowledge isn’t important anymore — you still have to decide which features you put into your network. That said, neural networks have the ability to learn which features are really important and which ones aren’t. A

representation learning algorithm can discover a good combination of features within a very short timeframe, even for complex tasks which would otherwise require a lot of human effort.

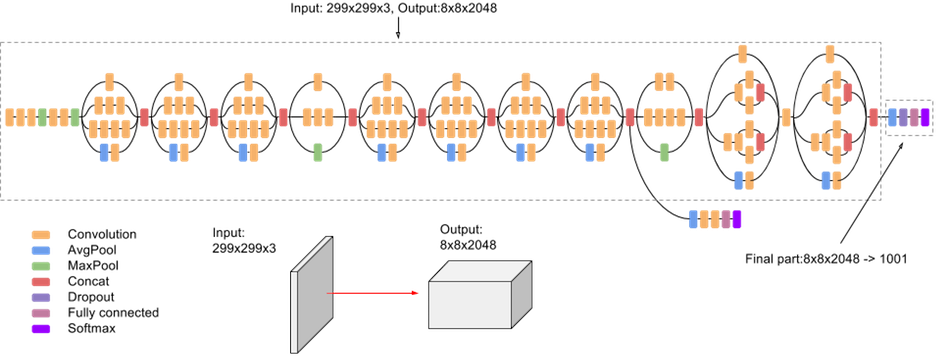
The learned representation can then be used for other problems as well. Simply use the first layers to spot the right representation of features, but don’t use the output of the network because it is too task-specific. Instead, feed data into your network and use one of the intermediate layers as the output layer. This layer can then be interpreted as a representation of the raw data.This approach is mostly used in computer vision because it can reduce the size of your dataset, which decreases computation time and makes it more suitable for traditional algorithms, as well.

## A. Popular Pre-Trained Models

There are a some pre-trained machine learning models out there that are quite popular. One of them is the Inception-v3 model, which was trained for the [ImageNet](http://image-net.org/) “Large Visual Recognition Challenge."

# Inception-v3 model

**Inception-v3** is a convolutional neural network architecture from the Inception family that makes several improvements including using Label Smoothing, Factorized 7 x 7 convolutions, and the use of an auxiliary classifier to propagate label information lower down the network (along with the use of batch normalization for layers in the side head).

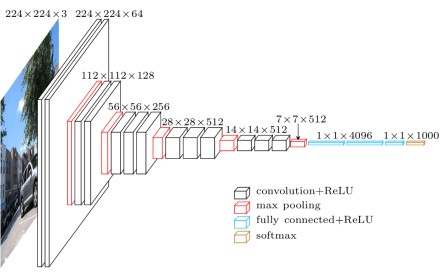


**Fig 14: Architecture of Inception-v3 model**

# VGG16 model

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

Here I first importing all the libraries which i will need to implement VGG16. I will be using Sequential method as I am creating a sequential model. Sequential model means that all the layers of the model will be arranged in sequence. Here I have imported ImageDataGenerator from keras.preprocessing. The objective of ImageDataGenerator is to import data with labels easily into the model. It is a very useful class as it has many function to rescale, rotate, zoom, flip etc. The most useful thing about this class is that it doesn’t affect the data stored on the disk. This class alters the data on the go while passing it to the model.



## Fig 15 : Architecture of VGG16 model

It is a complete implementation of VGG16 in keras using ImageDataGenerator. Here I have loaded the image using image method in keras and converted it to numpy array and added an extra dimension to the image to image for matching NHWC (Number, Height, Width, Channel) format of keras.This is a complete implementation of VGG16 in keras using ImageDataGenerator. We can

make this model work for any number of classes by changing the the unit of last softmax dense layer to whatever number we want based on the classes which we need to classify.

## Object Detection

In it is module, Object detection is one of the classical problems of computer vision and is often described as a difficult task. In many respects, it is similar to other computer vision tasks, because it involves creating a solution that is invariant to deformation and changes in lighting and viewpoint. What makes object detection a distinct problem is that it involves both locating and classifying regions of an image.

The locating part is not needed in, for example, whole image classification. To detect an object, we need to have some idea where the object might be and how the image is segmented. This creates a type of chicken-and-egg problem, where, to recognize the shape (and class) of an object, we need to know its location, and to recognize the location of an object, we need to know its shape.

Some visually dissimilar features, such as the clothes and face of a human being, may be parts of the same object, but it is difficult to know this without recognizing the object first. On the other hand, some objects stand out only slightly from the background, requiring separation before recognition. Low-level visual features of an image, such as a saliency map, may be used as a guide for locating candidate objects.

The location and size is typically defined using a bounding box, which is stored in the form of corner coordinates. Using a rectangle is simpler than using an arbitrarily shaped polygon, and many operations, such as convolution, are performed on rectangles in any case. The sub-image contained in the bounding box is 28 then classified by an algorithm that has been trained using machine learning.

The boundaries of the object can be further refined iteratively, after making an initial guess. During the 2000s, popular solutions for object detection utilized feature descriptors, such as scale- invariant feature transform (SIFT) developed by David Lowe in 1999 and histogram of oriented gradients (HOG) popularized in 2005. In the 2010s, there has been a shift towards utilizing convolutional neural networks.

Before the widescale adoption of CNNs, there were two competing solutions for generating bounding boxes. In the first solution, a dense set of region proposals is generated and then most of these are rejected. This typically involves a sliding window detector. In the second solution, a sparse set of bounding boxes is generated using a region proposal method, such as Selective Search.

Combining sparse region proposals with convolutional neural networks has provided good results and is currently popular.

In a typical object detection algorithm, an image is sent to the network, which is then sent through lots of convolutions and pooling layers. The output would be an object of the class. For each input image, there is a corresponding class as output. After taking the image as an input, the image is divided into various regions.

Each of these regions is considered a separate image. The regions are then passed to the Convolution Neural Networks (CNN) to classify them into various classes. Once each of the regions has been divided into corresponding classes, all the regions are combined to get the original image with the detected objects. However, there are some problems with such trivial algorithms as the images might have different aspect ratios and spatial locations. These factors could lead to a large number of regions and the computational time would increases.

There could be multiple objects in the image and this is something that would be very common in self-driving cars. The algorithm would not only need to detect other cars but motorcycles, pedestrians, trees, and other objects. When it comes to the context of deep learning, the basic algorithmic difference would be choosing the relevant inputs and outputs.

An input image is convoluted by n-filters. The output of the convolution is then treated with non-linear transformations, like RELU and MaxPool. The above operations of Convolution, MaxPool and RELU are performed multiple times. The output of the final layer is sent to the Softmax layer, where the numbers between 0 and 1 are converted and a probability is considered, declaring them a member of a particular class. The losses are minimized so that the predictions from the last layer can be as close as possible to the actual values.

The output labels are changed to make the bounding boxes around an object. This helps the programming model to learn the class of the object and the position of the object in the image. Four parameters are added in the output layer which includes the centroid, the proportion of height and width of the bounding box. A bunch of output units is added to get the cartesian coordinates of the different positions to be recognized. The different positions or landmark would be consistent for particular objects.

Multiple object detection and localization, If we are trying to detect multiple objects in the image, we can use the same technique that was being used in object localization. The difference is that we would want the algorithm to be able to classify and localize all the different objects in the

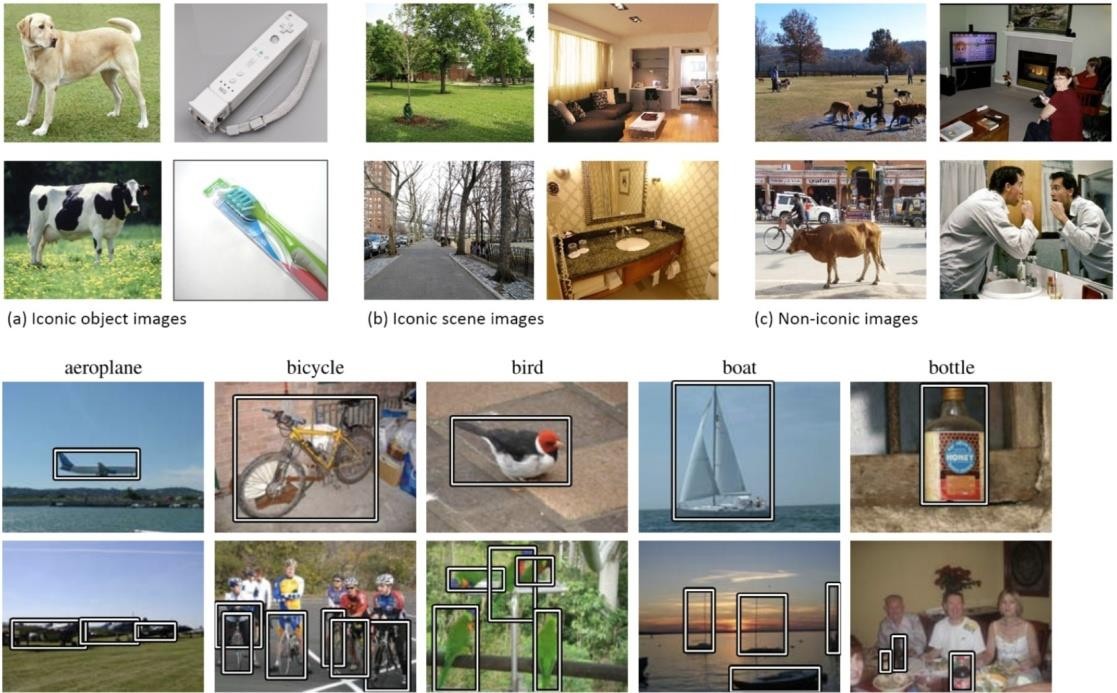
image and not just one. The simple idea is to crop the image into multiple images and run the same algorithm for all these cropped images.

In the algorithm, a window of much smaller size than the actual image size is made. It is cropped and passed to the CNN for it to make the predictions.The window is to be kept on sliding and these cropped images are to be passed into CNN.

After all the portions of the image with the window size have been cropped, the steps are to repeated all over again for bigger window sizes. These images are then to be passed to CNN for predictions.There would be a set of cropped images at the end where there would be an object, along with a class and the bounding box of the object.

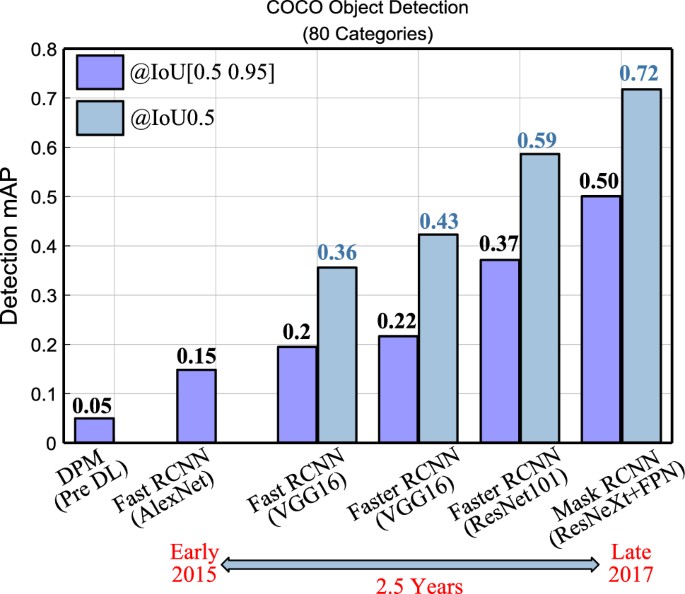
## MSCOCO dataset

The MS **COCO** (**Microsoft Common Objects in Context**) dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset. The dataset consists of 328K images.

**Splits:** The first version of MS COCO dataset was released in 2014. It contains 164K images split into training (83K), validation (41K) and test (41K) sets. In 2015 additional test set of 81K images was released, including all the previous test images and 40K new images.Based on community feedback, in 2017 the training/validation split was changed from 83K/41K to 118K/5K. The new split uses the same images and annotations. The 2017 test set is a subset of 41K images of the 2015 test set. Additionally, the 2017 release contains a new unannotated dataset of 123K images.

## Fig 16:The first two lines are examples from the MS COCO dataset [5].

The images show three different types of images sampled in the dataset, including iconic objects, iconic scenes and non-iconic objects.



**Fig 17:comparisons to Other Detectors Annotations:** The dataset has annotations for

* + object detection: bounding boxes and per-instance segmentation masks with 80 object categories.
  + captioning: natural language descriptions of the images (see MS COCO Captions).
  + keypoints detection: containing more than 200,000 images and 250,000 person instances labeled with keypoints (17 possible keypoints, such as left eye, nose, right hip, right ankle).
  + stuff image segmentation – per-pixel segmentation masks with 91 stuff categories, such as grass, wall, sky (see MS COCO Stuff).
  + panoptic: full scene segmentation, with 80 thing categories (such as person, bicycle, elephant) and a subset of 91 stuff categories (grass, sky, road),dense pose: more than 39,000 images and 56,000 person instances labeled with Dense.
  + Pose annotations – each labeled person is annotated with an instance id and a mapping between image pixels that belong to that person body and a template 3D model. The annotations are publicly available only for training and validation images.

## Flickr8k dataset

Generating a caption for a given image is a challenging problem in the deep learning domain. In this article, we will use different techniques of computer vision and NLP to recognize the context of an image and describe them in a natural language like English. we will build a working model of the image caption generator by using CNN (Convolutional Neural Networks) and LSTM (Long short term memory) units.

For training our model I’m using [Flickr8K](https://www.kaggle.com/shadabhussain/flickr8k) dataset. It consists of 8000 unique images and each image will be mapped to five different sentences which will describe the image.

## C.NLTK Library

Simply and in short, natural language processing (NLP) is about developing applications and services that can understand human languages.We are talking here about practical examples of natural language processing (NLP) like speech recognition, speech translation, understanding complete sentences, understanding synonyms of matching words, and writing complete grammatically correct sentences and paragraphs.

This is not everything; you can think about the industrial implementations about these ideas and their benefits. In this module, we will talk about natural language processing (NLP) using Python. This NLP will use Python NLTK library. NLTK is a popular Python library which is used for NLP.

As all of you know, there are millions of gigabytes every day are generated by blogs, social websites, and web pages.Many companies are gathering all of these data for understanding users and their passions and give reports to the companies to adjust their plans.These data could show that the people of Brazil are happy with product A which could be a movie or anything while the people of the US are happy with product B.

And this could be instant (real-time result). Like what search engines do, they give the appropriate results to the right people at the right time.You know what, search engines are not the only implementation of natural language processing (NLP), and there are a lot of awesome implementations out there.

Natural language toolkit (NLTK) is the most popular library for natural language processing (NLP) which is written in Python and has a big community behind it.NLTK also is very easy to learn; it’s the easiest natural language processing (NLP) library that you’ll use.

# Building the web interface:

## Flask

Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by **Armin Ronacher** who leads an international group of python enthusiasts (POCCO). It is based on WSGI toolkit and jinja2 template engine. Flask is considered as a micro framework.

## Keras

Keras is a minimalist Python library for deep learning that can run on top of Theano or TensorFlow.It was developed to make implementing deep learning models as fast and easy as possible for research and development.It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks.

It is released under the permissive MIT license.Keras was developed and maintained by [François Chollet](https://www.linkedin.com/in/fchollet), a Google engineer using four guiding principles:

* + **Modularity**: A model can be understood as a sequence or a graph alone. All the concerns of a deep learning model are discrete components that can be combined in arbitrary ways.
  + **Minimalism**: The library provides just enough to achieve an outcome, no frills and maximizing readability.
  + **Extensibility**: New components are intentionally easy to add and use within the framework, intended for researchers to trial and explore new ideas.
  + **Python**: No separate model files with custom file formats. Everything is native Python.

## Tensorflow

The most famous deep learning library today is TensorFlow. It is owned by Google. Machine learning is used in all of the Google products to improve translation, search engine, image captioning, and recommendations. Google users get to have a faster and refined search with Artificial Intelligence. Google uses machine learning to take advantage of the massive datasets to help users get the best experience. The researchers, programmers, and data scientists all use machine learning. TensorFlow was built as a framework to help developers and researchers work together on an AI model. Lots of people can use it once it has been developed and scaled.

Creating an object detection algorithm is the best way to understand how everything works. The necessary algorithms are provided with TensorFlow. You can create an entire object detection algorithm as follows. However, you need to take care of two things before you start:

* + Getting prerequisites
  + Setting up the environment

## Getting prerequisites

A few prerequisites would be required to get the job done. A few things need to be installed on the system.Python Tensorflow Tensorboard Ptorobuf v3.4 and above **Setting up the environment**

Tensorflow can be downloaded using the pip or conda commands: # For CPU

pip install tensorflow # For GPU

pip install tensorflow-gpu

The other libraries are also to be installed using the pip or conda commands. The following code would work.

pip install --user Cython pip install --user contextlib2 pip install --user pillow

pip install --user lxml pip install --user jupyter

pip install --user matplotlib

Protocol Buffers are the language-neutral, platform-neutral, extensible mechanism, which is like XML, but smaller and much simpler. Version 3.4 or above of the same needs to be downloaded. TensorFlow's model needs to be cloned or downloaded from GitHub. Both the models and protobuf should be placed in the same folder. After that, it is time to run protofbuf from the research folder.

"path\_of\_protobuf's bin"./bin/protoc object\_detection/proto

To perform real-time object detection through TensorFlow, the same code can be used but a few tweakings would be required. OpenCV would be used here and the camera module would use the live feed from the webcam.

## OpenCv

OpenCV was started at Intel in 1999 by **Gary Bradsky**, and the first release came out in 2000. **Vadim Pisarevsky** joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project.

OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day. OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS.

Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development. OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language. OpenCV-Python is a library of Python bindings designed to solve computer vision problems.

Python is a general purpose programming language started by **Guido van Rossum** that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++.

OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation. OpenCV- Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

OpenCV introduces a new set of tutorials which will guide you through various functions available in OpenCV-Python. **This guide is mainly focused on OpenCV 3.x version** .Prior knowledge of Python and Numpy is recommended as they won't be covered in this guide. **Proficiency with Numpy is a must in order to write optimized code using OpenCV- Python.**

In this undertaking, in a item detection algorithm, an photograph is dispatched to the community, that is then dispatched via plenty of convolutions and pooling layers. The output might be an item of the elegance. For every enter photograph, there's a corresponding elegance as output. After taking the photograph as an enter, the photograph is split into numerous areas.Each of those areas is taken into consideration a separate photograph.

The areas are then handed to the Convolution Neural Networks (CNN) to categorise them into numerous classes. Once every of the areas has been divided into corresponding classes, all of the areas are mixed to get the unique photograph with the detected items. . Decreasing illumination and acquisition angle.

The proposed MLP primarily based totally item monitoring gadget is made strong with the aid of using an most excellent choice of particular functions and additionally with the aid of using enforcing the Adaboost sturdy classmethod.

Send the image as input to Convolutional Neural Network which is trained with the pre-trained models such as Inception-v3 model,VGG16 model to detect the objects from the image .Here NLTK library performs data pre-processing and vocabulary training takes place using NLP to the CNN .

Flickr or MSCOCO datasets consists of 8000 to 10000 images. Flickr or MSCOCO dataset consists of image dataset and text dataset.The models are trained with these datasets. The CNN extracts the features and classify them into corresponding classes. The detected objects are bounded with labels with weights.

It is implemented using Keras with Tensorflow backend written in Python. Then we deploy this into Web interface with the help of Flask Rest API of python to build a web application.Finally

,our Object detection WebApp is ready to detect the objects from the images.

# SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying todiscover 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 isthe process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used.

The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

## Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application

.it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique pathof a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

## Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems/Procedures : interfacing systems or procedures must be invoked.

## System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing isbased on process descriptions and flows, emphasizing pre-driven process links and integration points.

## Performance Testing

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being sendto the system for to retrieve the results.

## Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused byinterface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or - one step up - software applications at the company level - interact without error.

## Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

## Acceptance testing for Data Synchronization:

* + - The auditors audit operation is done only when there is a request from user
    - The Status of data information on the cloud is viewed only by the cloud server

## Test Cases

|  |  |
| --- | --- |
| Test case Number | 1 |
| Test case Name | Unit Test Case for uploading images by the  user. |
| Feature to be tested | Image upload |
| Description | When a new image is uploaded by the user, the model verifies the image format. |
| Sample Input | Example.jpg, Example.jpeg, Example.png |
| Expected output | The image should be uploaded successfully  without any errors and the uploaded image  should satisfy the mentioned format described |
| Actual output | If the image uploaded is invalid, the prompt shows "Please upload valid image". |
| Remarks | Success |

**Table 1: Unit Test Case for Image Upload**

|  |  |
| --- | --- |
| Test case Number | 2 |
| Test case Name | Unit Test Case for naming the detected objects |
| Feature to be tested | Naming the detected objects |
| Description | When an image is uploaded by the user using  the web interface, the name /labelling should be generated after clicking the upload button. |
| Sample Input | Naming the objects |
| Expected output | The name of the image, which descibes the  image based on the objects detected from the  image. |
| Actual output | If there is no image selected, it prompts  "Please upload the image to generate caption" |
| Remarks | Successful |

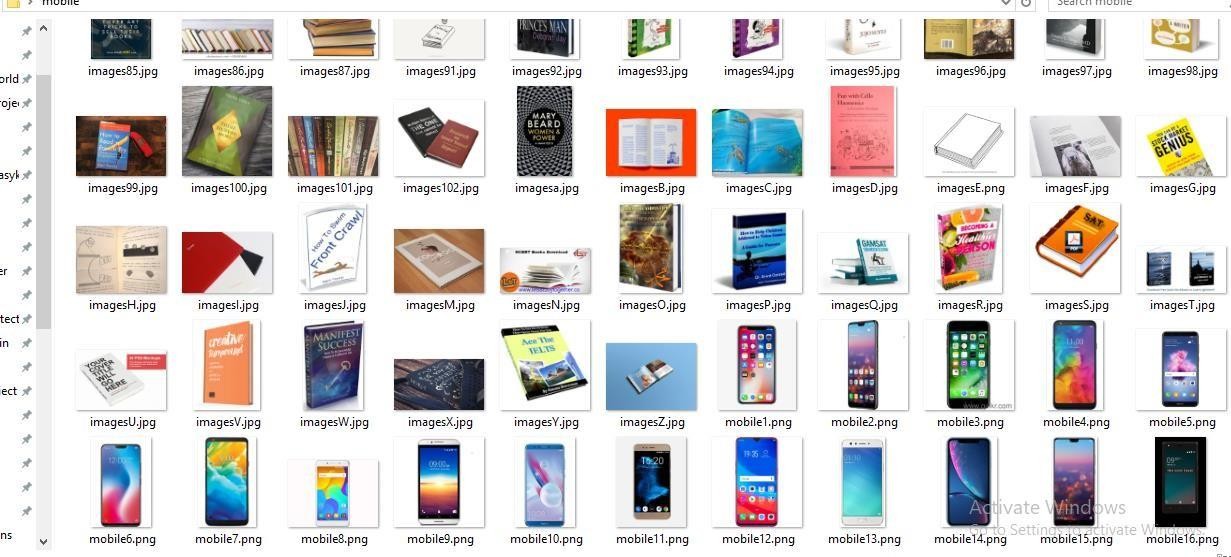
## Table 2: Unit Test Case for Naming the detected objects

|  |  |  |  |
| --- | --- | --- | --- |
| Sl-no | Test-case scenario | Expected result | Actual result |
| 1 | Invalid image | Prompt for “Please upload  Valid image”. | Focus on image Format |
| 2 | No image is selected | Prompt for “Please upload an  image". | Focus on labeling. |

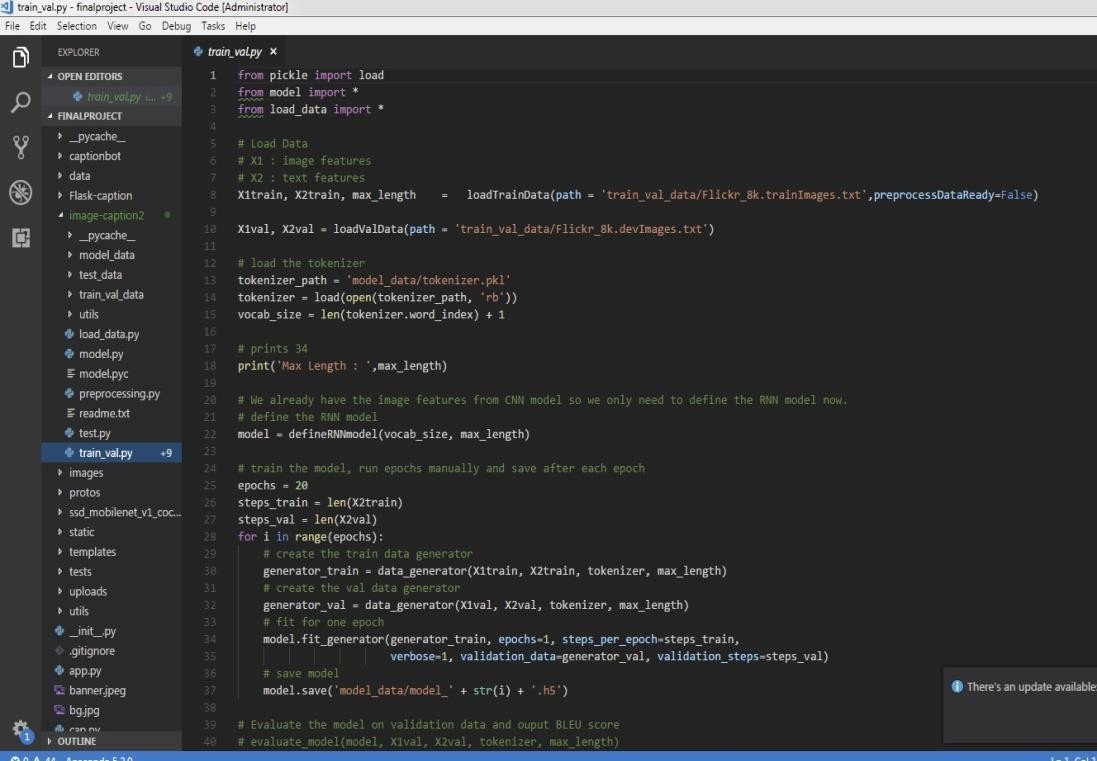
**Table 3: Acceptance Test Case for detecting objects**

# RESULTS

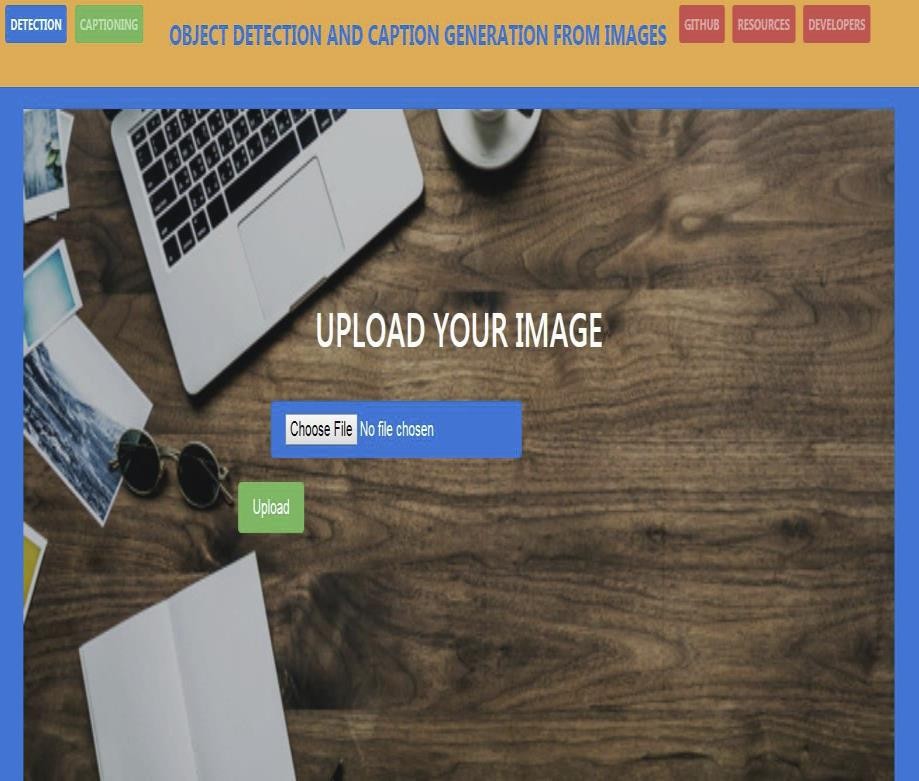
The first step is collecting images for our project. Download them from goggle .I ensured that images were taken from multiple angles, brightness, scale etc.so that the detector can work under different conditions of lightning and angles. Overall 100–150pics will suffice. Some sample images below:



## Fig 18: Sample Images

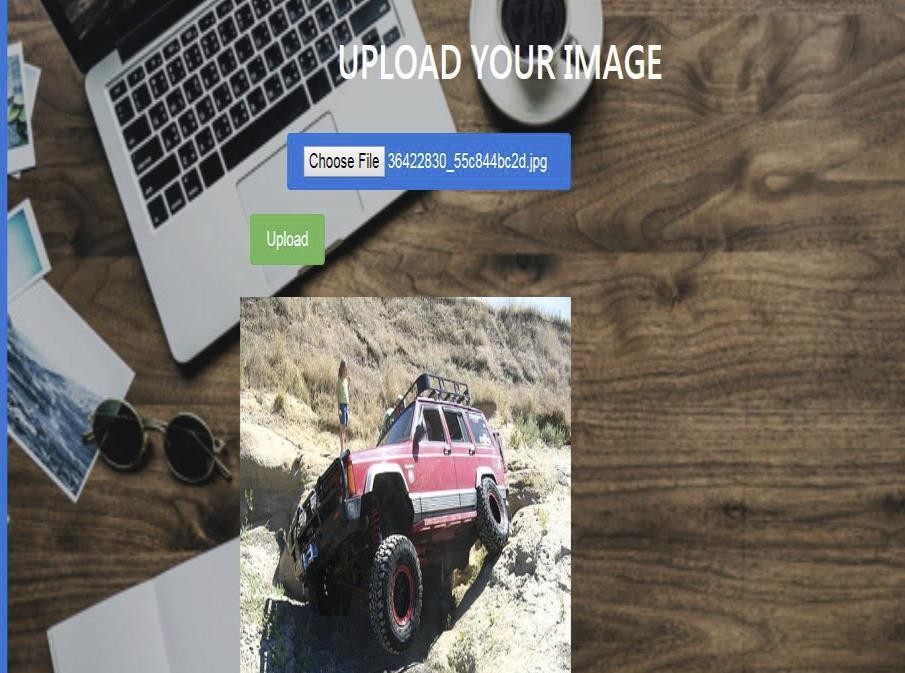


**Fig 19: Visual Studio Code Interface**



## Fig 20:Web Application Interface

After running the program a new window will open, which can be used to detect objects .



## Fig 21:User Image Upload



**Fig 22: Detected Object**

It identifies a Truck with 96% confidence . It show the accuracy of detecting the object.



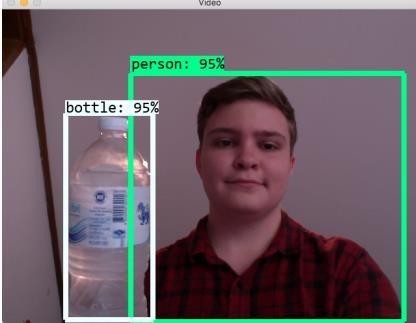
## Fig 23: Before Detection Example1

Here, we are expecting to detect the person in an image uploaded with high confidence.



**Fig 24:After Detection Example1**

It identifies a Truck with 96% confidence . It show the accuracy of detecting the object.



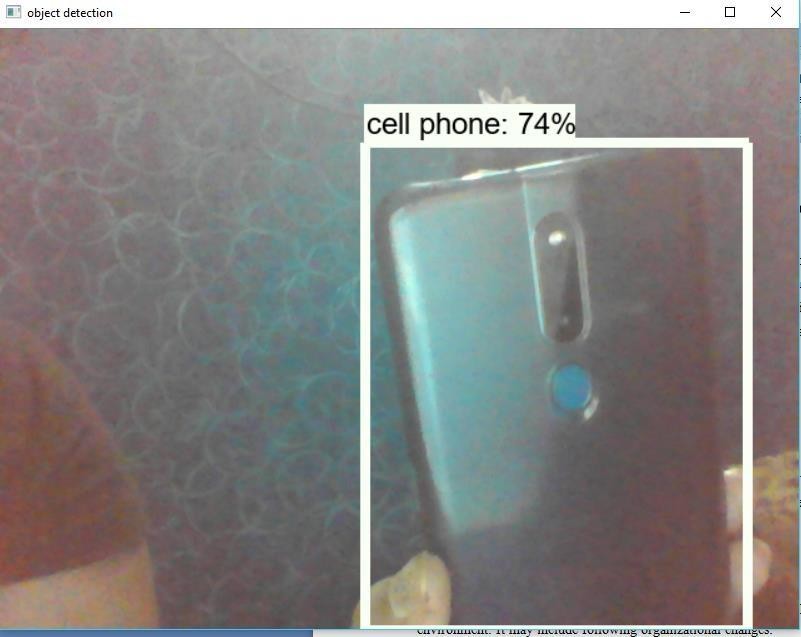
## Fig 25:Object Detection Example2

It identifies me as a person with 95% confidence and water bottle also with 95% confidence. It show the accuracy of detecting the object.



**Fig 26:Object Detection Example 3**

It identifies the persons with 55% and 76% confidence respectively .It show the accuracy of detecting the object.



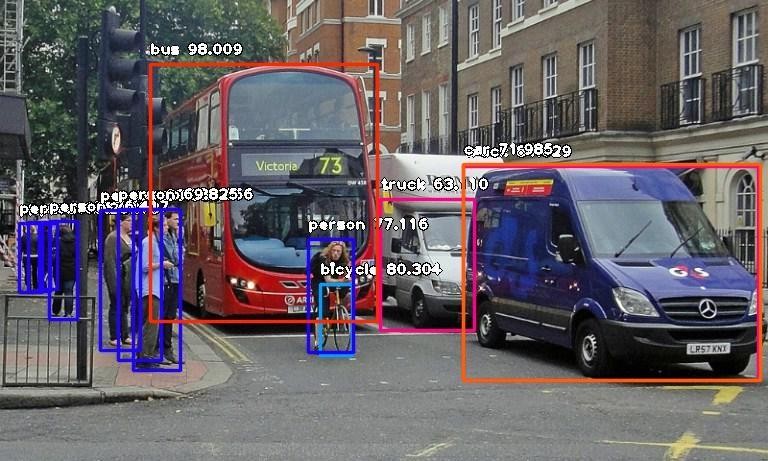
## Fig 27:Object Detection Example 4

It identifies a Cell phone with 74% confidence .It show the accuracy of detecting the object.



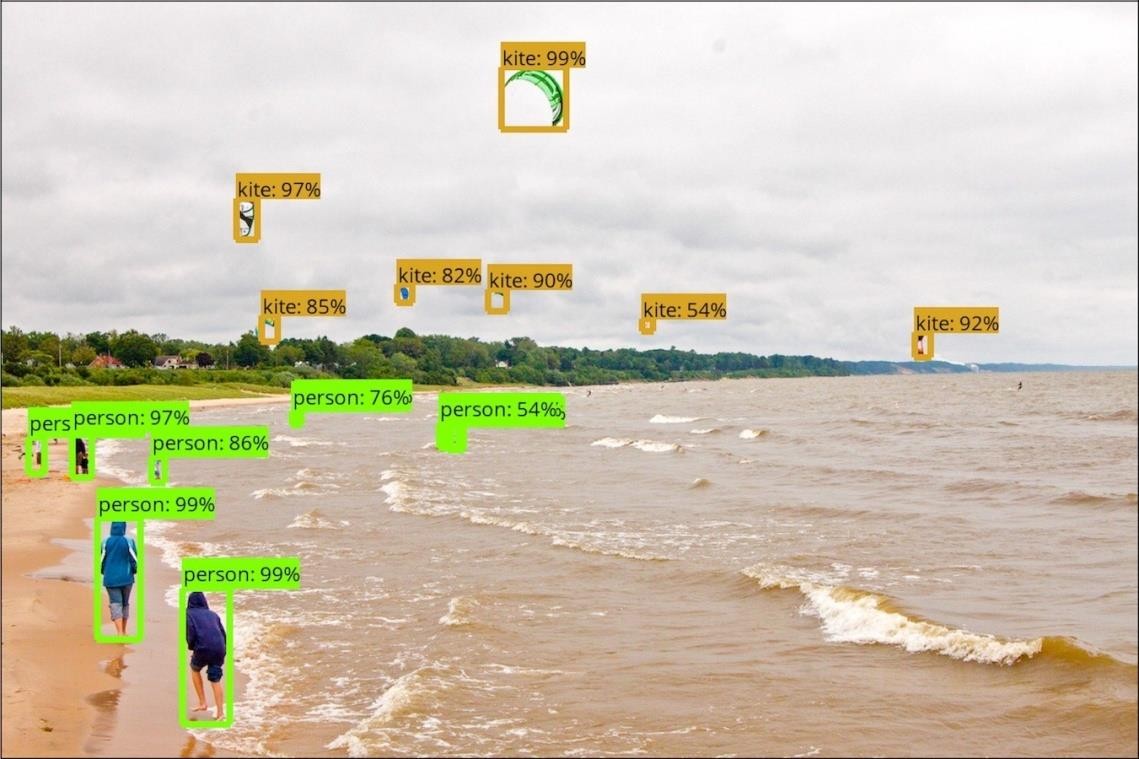
## Fig 28:Before Detection Example 5

This is a sample image we feed to the algorithm and expect our algorithm to detect and identify objects in the image and label them according to the class assigned to it.



## Fig 29:After Detection Example5

As expected our algorithm identifies the objects by its classes ans assigns each object by its tag and has dimensions on detected image.



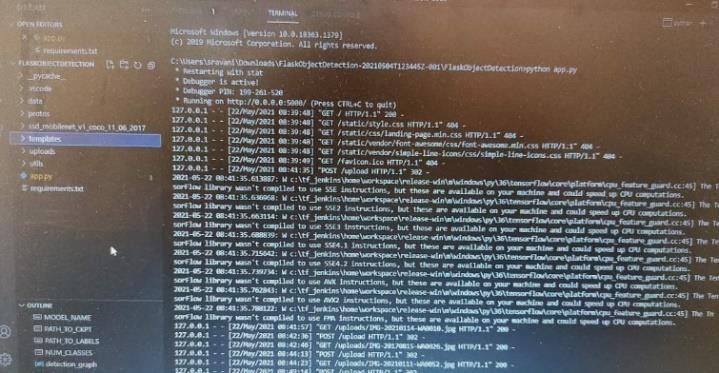
## Fig 30:Object Detection Example 6



**Fig 31:Object Detection Example 7**

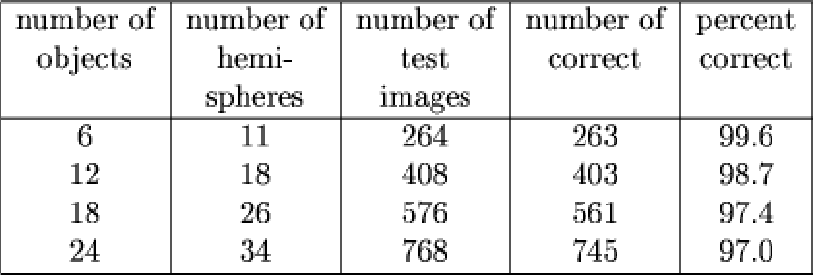


## Fig 32:Object Detection Example 8

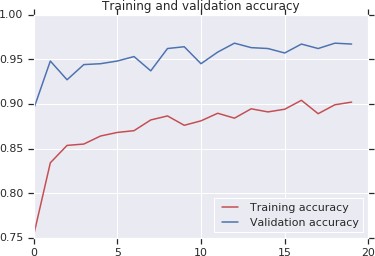


**Fig 33:Console result for above images**

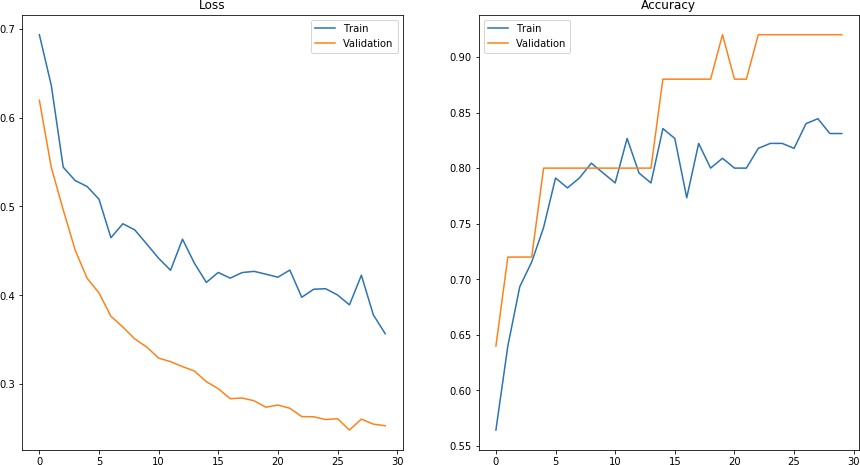
We ran tests with databases built for 6,12,18,24 objects and obtained overall success rates(correct classification on forced choice) of 99.6%, 98%, 97.4% and 97% respectively.The worst cases were the book and the pen in 24 object test,with 19/24 and 20/24 correct respectively



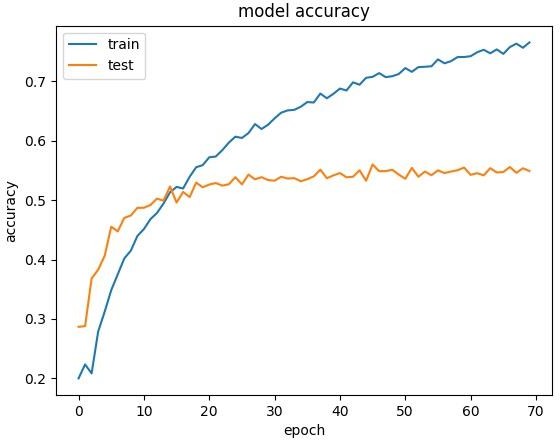
## Table 4:Overall Success rates of detected images



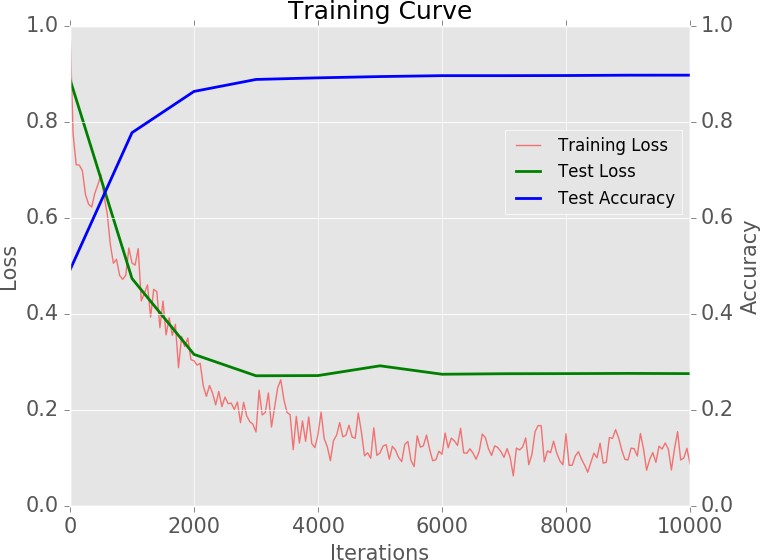
**Fig 34:Graph for Training and Validtaion Accuracy**



## Fig 35:Graphs for Accuracy and Loss



**Fig 36:Graph for model Accuracy**



## Fig 37: Graph forTraining Curve

The time to identify an object depends more or less linearly on the number of key featuresfed to the system, and the size of the database. At the moment, overall recognition time ona single processor are about 20 seconds for the 6 object database, and about 2 min for the 24 object database. This could also be improved substantially by pushing on the indexing methods.

The program updates the image frame with a new frame every between 0.25 sec and 0.5 seconds, which means an average of 2 - 4 FPS. In this project we detect live object with help of camera.

ImageAI provides many more features useful for customization and production capable deployments for object detection tasks. Some of the features supported are:

-Adjusting Minimum Probability: By default, objects detected with a probability percentage of less than 50 will not be shown or reported. You can increase this value for high certainty cases or reduce the value for cases where all possible objects are needed to be detected.

-Custom Objects Detection: Using a provided CustomObject class, you can tell the detection class to report detections on one or a few number of unique objects.

-Detection Speeds: You can reduce the time it takes to detect an image by setting the speed of detection speed to “fast”, “faster” and “fastest”.

-Input Types: You can specify and parse in file path to an image, Numpy array or file stream of an image as the input image

-Output Types: You can specify that the detectObjectsFromImage function should return the image inthe form of a file or Numpy array.

# CONCLUSION & FUTURE SCOPE

Object detection has many advantages in almost every complex area of Artificial Intelligence. The main use case of our model is to help visually impaired to understand the environment and made them easy to act according to the environment. As, this is a complex task to do, with the help of pre trained models and powerful deep learning frameworks like Tensorflow and Keras, we made it possible. This is completely a Deep Learning project, which makes use of multiple Neural Networks like Convolutional Neural Network to detect objects and naming the images. To deploy our model as a web application, we have used Flask, which is a powerful Python's web framework.

We are going to extend our work in the next higher level by enhancing our model is to detect the objects even for the live video frame. Our present model detects objects only forthe image, which itself a complex task and detecting live video frames is much complex to create. This is completely GPU based and detecting live video frames cannot be possiblewith the general CPUs. Objects detection in Video is a popular research area in which it is going to change the lifestyle of the people with the use cases being widely usable in almost every domain. It automates the major tasks like video surveillance and other security tasks.

By the use of this thesis and primarily based totally on experimental outcomes we're capable of discover object extra precisely and pick out the gadgets in my opinion with genuine area of an object with inside the photo in x,y axis. This paper additionally oﬀer experimental outcomes on exceptional strategies for item detection and identity and compares each approach for their eﬃciencies. The item popularity gadget may be implemented withinside the place of surveillance gadget, face popularity, fault detection, character popularity etc.

The goal of this thesis is to expand an item popularity gadget to apprehend the 2D and 3-D gadgets withinside the picture. The overall performance of the item popularity gadget relies upon at the capabilities used and the classiﬁer hired for popularity. This studies paintings tries to propose a unique function extraction approach for extracting worldwide capabilities and acquiring nearby capabilities from the area of interest. Detecting an object accurately in a surveillance video is one of the major research areas in computer vision due to its wide range of applications.

It is very challenging one, to process the image obtained from a surveillance video due to the following reasons low resolution, illumination variation, dynamic objects in the background, small changes in the background like waving of leaves. We have presented an overview of recent

developments in object detection methods. The detection process occurs in background modeling, object detection, object classification.

In this project, all available object detection techniques are categorized into background subtraction, optical flow and spatiotemporal filter methods and discussed the advantages and disadvantages of the methods applied in various types of dataset. The object classification techniques are categorized into shape-based, motion-based and texture based methods. The state-of-the-art of existing methods in each key issue is discussed and made to point the future work needed to improve the object detection process in surveillance videos.

Also the studies paintings tries to hybrid the conventional classiﬁers to apprehend the item. The item popularity gadget evolved on this studies changed into examined with the benchmark datasets like COIL100, Caltech 101, ETH80 and MNIST. The item popularity gadget is applied in MATLAB

7.5. It is important to mention the difficulties observed during the experimentation of the object recognition system due to several features present in the image. The research work suggests that the image is to be preprocessed and reduced to a size of 128 x 128.

The proposed feature extraction method helps to select the important feature. To improve the efficiency of the classifier, the number of features should be less in number. Specifically, the contributions towards this research work are as An object recognition system is developed, that recognizes the two-dimensional and three follows, The feature extracted is sufficient for recognizing the object and marking the location of the dimensional objects.

The proposed global feature extraction requires less time, compared to the traditional feature object. x The proposed classifier is able to recognize the object in less computational cost. The performance of the SVM-kNN is greater and promising when compared with the BPN and extraction method. The performance of the One-against-One classifier is efficient SVM. Along with the local features, the width and height of the object computed through projection. Local feature PCA-SIFT is computed from the blobs detected by the Hessian-Laplace detector.

Global feature extracted from the local parts of the image method is used. The methods presented for feature extraction and recognition are common and can be applied to any application that is relevant to object recognition. The proposed object recognition method combines the state-of-art classifier SVM and k-NN to recognize the objects in the image. The multiclass SVM is used to hybridize with the k-NN for the recognition. The feature extraction method proposed in this research work is efficient and provides unique information for the classifier.

The image is segmented into 16 parts, from each part the Hu’s Moment invariant is computed and it is converted into Eigen component. The local feature of the image is obtained by using the Hessian-Laplace detector. This helps to obtain the objects feature easily and mark the object Features either the local or global used for recognition can be increased, to increase the efficiency location without much difficulty. As a scope for future enhancement,Geometric properties of the image can be included in the feature vector of the object recognition system

The proposed object recognition system uses grey-scale image and discards the color information. The colour information in the image can be used for recognition of the object. Colour based object. Using unsupervised classifier instead of a supervised classifier for recognition of the object recognition plays vital role in Robotics Although the visual tracking algorithm proposed here is robust in many of the conditions, it can be In the Single Visual tracking, the size of the template remains fixed for tracking.

If the size of the made more robust by eliminating some of the limitations as listed below: object reduces with the time, the background becomes more dominant than the object being Foreground object extraction depends on the binary segmentation which is carried out by applying Fully occluded object cannot be tracked and considered as a new object in the next frame tracked. In this case the object may not be tracked. Splitting and merging cannot be handled very well in all conditions using the single camera due to threshold techniques.

So blob extraction and tracking depends on the threshold value. the loss of information of a 3D object projection in 2D images. For Night time visual tracking, night vision mode should be available as an inbuilt feature in the CCTV camera. To make the system fully automatic and also to overcome the above limitations, in future, multi- view tracking can be implemented using multiple cameras. Multi view tracking has the obvious advantage over single view tracking because of wide coverage range with different viewing angles for the objects to be tracked.

In this thesis, an effort has been made to develop an algorithm to provide the base for future In this research work, the object Identification and Visual Tracking has been done through the use applications such as listed below of ordinary camera. The concept is well extendable in applications like Intelligent Robots, Automatic Guided Vehicles, Enhancement of Security Systems to detect the suspicious behaviour along with detection of weapons, identify the suspicious movements of enemies on boarders with In the proposed method, background subtraction technique has been used that is simple and fast.

This technique is applicable where there is no movement of camera. For robotic application or the help of night vision cameras and many such applications. automated vehicle assistance system,

due to the movement of camera, backgrounds are continuously changing leading to implementation of some different segmentation techniques

Object identification task with motion estimation needs to be fast enough to be implemented for like single Gaussian mixture or multiple Gaussian mixture model in the real time system. Still there is a scope for developing faster algorithms for object identification .Such algorithms can be implemented using FPGA or CPLD for fast execution.

# BIBLIOGRAPHY

References for this project development were taken from the following books and websites:

1. Abhaya Agarwal and Alon Lavie. 2008. Meteor, m-bleu and m-ter: Evaluation metricsfor high-correlation with human rankings of machine translation output. In Proceedings of theThird Workshop on Statistical Machine Translation. Association for Computational Linguistics, 115-118.
2. Ahmet Aker and Robert Gaizauskas. 2010. Generating image descriptions using dependency relational patterns. In Proceedings of the 48th annual meeting of theassociation for computational linguistics. Association for Computational Linguistics, 1250- 1258.
3. Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In European Conference on ComputerVision. Springer, 382-398.
4. Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2017. Bottom-up and top-down attention for image captioning and vqa.arXiv preprint arXiv:1707.07998 (2017).
5. Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing. 2018. Convolutional image captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 5561-5570.
6. Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney,Kate Saenko, Trevor Darrell, Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, et al. 2016. Deep compositional captioning: Describing novel object categories without paired training data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
7. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machinetranslation by jointly learning to align and translate. In International Conference on Learning Representations (ICLR).
8. Shuang Bai and Shan An. 2018. A Survey on Automatic Image Caption Generation. Neurocomputing.
9. Shuang Bai and Shan An. 2018. A Survey on Automatic Image Caption Generation. Neurocomputing.
10. Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, Vol. 29. 65-72.
11. Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In Advances in Neural Information Processing Systems. 1171-1179.
12. Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. Journal of machine learning research 3, Feb, 1137-1155.
13. Adam L Berger, Vincent J Della Pietra, and Stephen A Della Pietra. 1996. A maximum entropy approach to natural language processing. Computational linguistics 22, 1 (1996), 39- 71.
14. Raffaella Bernardi, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, Barbara Plank, et al. 2016. Automatic Description Generation from Images: A Survey of Models, Datasets, and Evaluation Measures. Journal of Artificial Intelligence Research (JAIR) 55, 409-442.
15. David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. Journal of machine Learning research 3, Jan (2003), 993-1022.
16. Cristian Bodnar. 2018. Text to Image Synthesis Using Generative Adversarial Networks. arXiv preprint arXiv:1805.00676.
17. Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. InProceedings of the 2008 ACM SIGMOD international conference on Management of data. AcM, 1247-1250.
18. Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. InProceedings of the 2008 ACM SIGMOD international conference on Management of data. AcM, 1247-1250.
19. Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. 1992. A training

algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory. ACM, 144-152.

1. Andreas Bulling, Jamie A Ward, Hans Gellersen, and Gerhard Troster. 2011. Eye

movement analysis for activity recognition using electrooculography. IEEE transactions on pattern analysis and machine intelligence 33, 4 (2011), 741-753.

1. Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. 2011. Learning photographic global tonal adjustment with a database of input/output image pairs. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. IEEE, 97-104.
2. Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluation the Roleof Bleu in Machine Translation Research.. In EACL, Vol. 6. 249-256.
3. Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, and Tat-Seng Chua. 2017. SCA-CNN: Spatial and Channel-wise Attention in Convolutional Networks for Image Captioning. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). 6298- 6306.
4. Tseng-Hung Chen, Yuan-Hong Liao, Ching-Yao Chuang, Wan-Ting Hsu, Jianlong Fu,and Min Sun. 2017. Show, Adapt and Tell: Adversarial Training of Cross-domain Image Captioner. In The IEEE International Conference on Computer Vision (ICCV), Vol. 2.
5. Xinlei Chen and C Lawrence Zitnick. 2015. Mind’s eye: A recurrent visual representation for image caption generation. In Proceedings of the IEEE conference oncomputer vision and pattern recognition. 2422-2431.
6. Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In Association for Computational Linguistics. 103-111.

25] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

1. Bo Dai, Dahua Lin, Raquel Urtasun, and Sanja Fidler. 2017. Towards Diverse and Natural Image Descriptions via a Conditional GAN. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). 2989-2998.
2. Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 1. IEEE, 886-893.
3. Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. 2006. Generating typed dependency parses from phrase structure parses. In Proceedings of LREC, Vol. 6. Genoa Italy, 449-454.
4. Etienne Denoual and Yves Lepage. 2005. BLEU in characters: towards automatic MT evaluation in languages without word delimiters. In Companion Volume to the Proceedingsof the Second International Joint Conference on Natural Language Processing. 81-86.
5. Jacob Devlin, Hao Cheng, Hao Fang, Saurabh Gupta, Li Deng, Xiaodong He, Geoffrey Zweig, and Margaret Mitchell. 2015. Language models for image captioning: The quirks and what works. arXiv preprint arXiv:1505.01809 (2015).
6. Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2625-2634.
7. Desmond Elliott and Frank Keller. 2013. Image description using visual dependency representations. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 1292-1302.ACM Computing Surveys, Vol. 0, No. 0, Article 0.

Acceptance Date: October 2018.A Comprehensive Survey of Deep Learning for ImageCaptioning 0:31

1. Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, PiotrDollÃąr, Jianfeng Gao, Xiaodong He, Margaret Mitchell, and John C Platt. 2015. Fromcaptions to visual concepts and back. In Proceedings of the IEEE conference on computervision and pattern recognition.1473-1482.
2. Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: Generating sentences from images. In European conference on computer vision. Springer, 15-29.
3. Alireza Fathi, Yin Li, and James M Rehg. 2012. Learning to recognize daily actions using gaze. In European Conference on Computer Vision. Springer, 314-327.
4. William Fedus, Ian Goodfellow, and Andrew M Dai. 2018. Maskgan: Better text generation via filling in the \_. arXiv preprint arXiv:1801.07736.
5. Nicholas FitzGerald, Yoav Artzi, and Luke Zettlemoyer. 2013. Learning distributions over logical forms for referring expression generation. In Proceedings of the 2013 conference on empirical methods in natural language processing. 1914-1925.