

# **CLASSIFICATION OF IMAGE USING CONVOLUTIONAL NEURAL NETWORK (CNN)**

**A PROJECT REPORT**

**Submitted by**

**LOKESH V - 510818104007**

**SURENDAR D - 510818104013**

**in partial fulfillment for the award of the degree**

**of**

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**GANADIPATHY TULSI'S JAIN ENGINEERING COLLEGE,  
VELLORE**



**ANNA UNIVERSITY: CHENNAI 600 025**

**JUNE 2021**

**ANNA UNIVERSITY: CHENNAI 600 025**

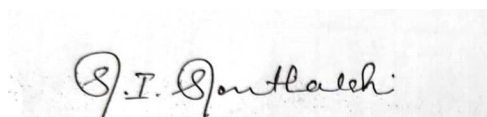
**BONAFIDE CERTIFICATE**

Certified that this project report “**CLASSIFICATION OF IMAGE USING CONVOLUTIONAL NEURAL NETWORK (CNN)**” is the bonafide work of

**LOKESH V - 510818104007**

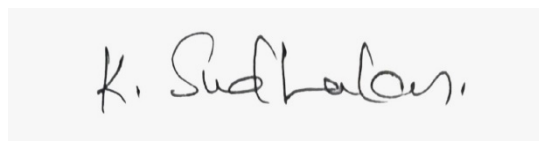
**SURENDAR D - 510818104013**

who carried out the project work under my supervision.



**SIGNATURE**

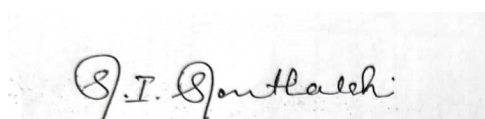
Mrs.S.I.Santhanalakshmi M.E.,  
ASSISTANT PROFESSOR,  
Department of Computer Science  
And Engineering  
GTEC  
Kaniyambadi



**SIGNATURE**

Mr.K SUDHAKAR M.TECH.,  
HEAD OF THE DEPARTMENT,  
Department of Computer Science and  
And Engineering  
GTEC  
Kaniyambadi

Submitted for the university examination held on 07/08/2021



**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

We first offer our deepest gratitude to our **GOD** the almighty who has given us strength and good health during the course of the project. We are conscious about our indebtedness to our **Managing Trustee Shri.N.Sugalchand Jain**, our Chairman **Shri.P.Pyarelal Jain**, our Secretary **Shri.T.Amarchand Jain**, our Principal **Dr.M.Barathi**, who inspired us to reach greater heights in the pursuits of knowledge.

We articulate our sincere gratitude to our Head of the Department **Mr.K.Sudhakar,M.Tech.**, who had taught us the way to do a successful project and without whose constant encouragement and whole idea, the project would not have been possible.

We wish to express our sincere thanks to our Internal Guide **Mrs.S.I.Santhana Lakshmi,M.E.**, who had helped us a lot and had given valuable suggestions, which made us finish this project in a very successful and neat way.

We wish to express our sincere thanks and honour to our Project Coordinator **Mrs.S.I.Santhanalakshmi,M.E.**, We extend our thanks to **Mr.K.Sudhakar,M.Tech., M.E., Mrs.S.Vennila, M.E., Mr.R.Venkatesan, M.E.** for their motivation. We also express our sincere thanks to the entire team of teaching and non-teaching staff of CSE Department.

We take this opportunity to thank our parents and friends who have been the source of inspiration and an encouraging factor throughout the project period.

## ABSTRACT

In recent year, with the speedy development in the digital contents identification, automatic classification of the images became most challenging task in the fields of computer vision. Automatic understanding and analysing of images by system is difficult as compared to human visions. In existing classification system the accurate classification of images is low. Our system present Convolutional Neural Network (CNN), a machine learning algorithm being used for automatic classification and to improve the speed and recognition accuracy of image. Java is used as a programming language because it comes together with TensorFlow framework. Convolutional neural network has been widely used in the field of image processing. Image classification model using a CNN with Tensor Flow. Inception-v3 is a pre-trained convolutional neural network model that is 48 layers deep. Inception layer is combination of convolutional layer and pooling layer.

Categorizes detected objects into predefined classes by using CNN that compares the image patterns with the target patterns. Our system uses the imagenet **ImageNet Large Scale Recognition Challenge (ILSVRC)** data set as a bench mark for classification of images. The images in the data set used for training which require more computational power for classification of images. By training the images using CNN network we obtain upto 98% accuracy result.

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	<b>ABSTRACT</b>	<b>IV</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>VIII</b>
	<b>LIST OF FIGURES</b>	<b>IX</b>
<b>1.</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Objective	1
	1.2 Overview	1
	1.3 Introduction	1
<b>2</b>	<b>LITERATURE SURVEY</b>	<b>3</b>
<b>3</b>	<b>SYSTEM ANALYSIS</b>	<b>9</b>
	3.1 Existing System	9
	3.1.1 Disadvantage	9
	3.2 Proposed System	10
	3.2.1 Advantages	10
<b>4</b>	<b>SYSTEM REQUIREMENTS</b>	<b>11</b>
	4.1 Hardware Requirements	11
	4.2 Software Requirements	11
	4.3 Technologies Used	12
	4.3.1 Java	12

	4.3.1.1 Applications Of Java Programming	14
	4.3.1.2 Java Platform	15
	4.3.1.3 Apache Netbeans	17
	4.3.1.4 Working Of A Java Program	18
	4.3.1.5 Java virtual machine	18
	4.3.4 Tensorflow Library	19
<b>5</b>	<b>SYSTEM DESIGN SPECIFICATIONS</b>	<b>21</b>
	5.1 Architecture Diagram	21
	5.2 Data Flow Diagram	22
	5.3 Use Case Diagram	23
	5.4 Activity Diagram	24
<b>6</b>	<b>SYSTEM DESIGN IMPLEMENTATION</b>	<b>25</b>
	6.1 Implementation	25
	6.2 Module Description	25
	6.2.1 Pre Processing	25
	6.2.2 Feature Extraction	26
	6.2.3 Object Detection	26
	6.2.4 Pre-Trained Inception V3 Model	27
<b>7</b>	<b>SYSTEM TESTING AND MAINTENANCE</b>	<b>29</b>
	7.1 Testing Objectives	29
	7.2 Types of Testing	30
	7.2.1 Unconventional Testing	30

	7.2.2 Conventional Testing	31
	7.3 Test Case Design	31
	7.3.1 Unit Testing	31
	7.3.2 Integration Testing	32
	7.3.3 Validation Testing	32
	7.4 Testing Strategies	32
	7.4.1 Integration Box Testing	33
	7.4.2 White Box Testing	33
	7.4.3 Black Box Testing	33
	7.4.4 Interface Testing	33
	7.4.5 Module Testing	34
	7.4.6 Smoke Testing	34
	7.5 Maintenance	34
<b>8</b>	<b>CONCLUSION AND FUTURE ENHANCEMENT</b>	<b>36</b>
	8.1 Conclusion	36
	8.2 Future Enhancement	36
	<b>APPENDIX 1</b>	<b>37</b>
	<b>APPENDIX 2</b>	<b>50</b>
	<b>REFERENCES</b>	<b>54</b>

## LIST OF ABBREVIATIONS

ABBREVIATIONS	DESCRIPTIONS
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
ILSRC	Imagenet Large Scale Recognition Challenge
CONV	Convolutional Layer
RELU	Rectified Linear Unit
POOL	Pooling Layer
FC	Fully Connected Layer



## **LIST OF FIGURES**

<b>FIGURE NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
4.3.1	JAVA TECHNOLOGY	13
4.3.2	JAVA VIRTUAL MACHINE	13
4.3.3	JAVA IDE	16
4.3.4	JAVA IMPLEMENTATION	19
5.1	ARCHITECTURE DIAGRAM	22
5.2	DATA FLOW DIAGRAM	24
5.3	UML DIAGRAM	25
5.3.1	ACTIVITY DIAGRAM	25
5.3.2	USE CASE DIAGRAM	26

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 OBJECTIVE**

The project's main goal is to learn and practically apply the concepts of Convolutional Neural Network. Extraction, analysis, and understanding of useful information from a single image.

### **1.2 OVERVIEW**

The overview of the project is Categorizes detected objects into predefined classes by using CNN that compares the image patterns with the target patterns. Our system uses the imagenet dataset for classify the images. By training the images using CNN network we obtain upto 98% accuracy result.

### **1.3 INTRODUCTION**

Image classification is a big problem in computer vision for the decades. In case of humans the image understanding, and classification is done very easy task, but in case of computers it is very expensive task. In general, each image is composed of set of pixels and each pixel is represented with different values.

Image classification model using a Convolutional neural network algorithm and TensorFlow library. Extraction of the features from Images by using Convolutional Neural Network (CNN) deep learning concept. its uses the different pooling layer in CNN for extraction of the feature (color and shape) from the image which are useful for perfect classification of images and target detection. One common way to execute image classification is through convolutional neural networks, a technique implementing deep learning, which is a subset of machine learning, which is in turn a subset of AI.

High resolution images in ImageNet data set having 15 million labelled images with 1000 different classes used for classification with help of deep convolutional Neural Network. The imagenet dataset contains thousands of images, various objects like: Cars, Birds, trucks, dogs, horses, cats and ships. The dataset is also split into three equal subsets: train, validation, and test. TensorFlow is an open-source software library for dataflow programming .It is a symbolic math library, and is also used for machine learning applications such as neural networks. CNN having four different layers such as input layer, Convolutional layer , Pooling layer and Fully connected layer. The output layer will be a fully connected layer usually to classify the image to which class it belongs to.

The convolution layer was consists of a convolution mask, bias terms and a function expression. Together, these generate output of the layer. CNN also make use of the concept of max-pooling, which is a form of non-linear down-sampling. In this method, the input image is partitioned into non-overlapping rectangles. The output for each sub-region is the maximum value.

## **CHAPTER 2**

### **LITERATURE SURVEY**

**[1]: TITLE: “Infrared Object Recognition Based On Monogenic Features And Multiple Kernel Learning”**

**AUTHOR :** CHEN NING , WENBO LIU, XIN WANG

**JOURNAL NAME AND YEAR:** IEEE TRANSACTIONS, 2018

**DESCRIPTION:** In this paper, a novel infrared object recognition method based on MONOGENIC FEATURES and multiple kernel learning is proposed. The proposed monogenic representation uniformly captures both spectral and spatial characteristics of the infrared objects. To provide efficient and compact feature description for subsequent recognition, a PRINCIPAL COMPONENTS ANALYSIS method is utilized to reduce the dimensionality of the features. By putting the different reduced features into the multiple kernel learning framework, the features can be fused effectively. The experimental results show that the proposed method can achieve good recognition performance for infrared objects

**LIMITATIONS:**

1. Monogenic feature extraction will decrease the computational speed
2. Data standardization is must before PCA.
3. PCA may miss some information as compared to the original dataset.

**[2]: TITLE: “3d Convolutional Object Recognition Using Volumetric Representations Of Depth Data”**

**AUTHORS:** ALI CAGLAYAN , AHMET BURAK CAN

**JOURNAL NAME AND YEAR:** IEEE TRANSACTIONS, 2017

**DESCRIPTION:** We have presented a 3D convolutional object recognition approach based on two volumetric representations using depth maps. Although depth maps do not give enough information to build a complete 3D model of objects, the constructed view-based incomplete 3D model is still useful to recognize objects. Our experiments show that the proposed model has achieved higher accuracy than many state-of-the-art approaches on the commonly used Washington RGB-D Object Dataset. To the best of our knowledge, the proposed model is the first volumetric approach on this dataset. We have demonstrated that 3D CNNs on volumetric representations make it possible to learn rich 3D structural information of objects. We believe this work opens up possibilities for learning rich 3D geometrical features.

**LIMITATIONS:**

1. CNN do not encode the position and orientation of object. The main component of a CNN is a convolutional layer.
2. Lack of ability to be spatially invariant to the input data. Artificial neurons output a single scalar.

**[3] : TITLE: “3d Object Recognition From Large-Scale Point Clouds With Global Descriptor And Sliding Window”**

**AUTHORS:** NAOYUKI GUNJI , HITOSHI NIIGAKI

**JOURNAL NAME AND YEAR:** IEEE TRANSACTIONS, 2016

**DESCRIPTION:** we proposed a novel approach for 3D object recognition from large-scale indoor point clouds. Our approach utilizes a sliding window with BoF representation of the model, which consists of aggregated local information and is robust to partial noises. We tackled two problems: one is the repetitive appearance of unhelpful primitive shapes (planar patches and curved surfaces), the other is the loss of detailed shape information due to noise such as clutter, occlusions, holes, and measurement error.

Although we used the same partitioning method in all experiment, optimal division of window may differ in many kinds of object. In future work, we will refine the division of sliding window

**LIMITATIONS:**

1. RANSAC is that there is no upper bound on the time it takes to compute these parameters (except exhaustion).
2. When the number of iterations computed is limited the solution obtained may not be optimal.
3. it may not even be one that fits the data in a good way.

**[4]: TITLE: “Object Detection In Images Using Artificial Neural Network And Improved Binary Gravitational Search Algorithm”**

**AUTHORS: FARZANEH AZADI POURGHAHESTANI**

**JOURNAL NAME AND YEAR: IEEE TRANSACTIONS, 2015**

**DESCRIPTION:** In this paper, object detection with ANN and IBGSA has been illustrated. KNN classifier is used to obtain fitness function in feature selection because of its high speed. After this stage, selected features are utilized for classification but due to low accuracy of KNN, ANN is used as the classifier. Target objects are hand and some hand tools. Recognition rate and elapsed time in this method proof that this approach is appropriate for automatically selecting salient features for object detection in specific jobs. This approach could be used in machine vision and robotic applications.

The purpose of using IBGSA is to decrease complexity by selecting salient features. At last, selected features are used in the ANN for detecting objects. Experimental results on detecting hand tools show that the proposed method could find salient features for object detection.

**LIMITATIONS:**

1. Accuracy depends on the quality of the data.
2. With large data, the prediction stage might be slow.
3. Sensitive to the scale of the data and irrelevant features.

**[5]: TITLE: “Detecting Object Affordances With Convolutional Neural Networks”**

**AUTHORS:** ANH NGUYEN, DIMITRIOS KANOULAS

**JOURNAL NAME AND YEAR:** IEEE TRANSACTIONS, 2016

**DESCRIPTION:** we present a novel method to detect object affordances using a deep convolutional neuron network. We have demonstrated that a large deep network can significantly improve the detection results compared to the state-of-the-art methods. Moreover, we have tested our method on grasping experiments with a full-size humanoid robot. Using our method, the inference procedure is real-time and the robot is able to perform grasping tasks using the detected affordances. We aim to develop a more general approach to overcome this limitation. Another interesting problem is to study the semantic relationship between object affordances that enables the completion of more types of tasks. Finally, we plan to release our new affordance dataset that has more challenging scenes and covers more types of objects.

**LIMITATIONS:**

1. It requires much computational power as well as resources as it builds numerous trees to combine their outputs.
2. It also requires much time for training as it combines a lot of decision trees to determine the class.



**[6]: TITLE: “3d Convolutional Object Recognition Using Volumetric Representations Of Depth Data”**

**AUTHORS:** DUMITRU ERHAN, CHRISTIAN SZEGEDY

**JOURNAL NAME AND YEAR:** IEEE TRANSACTIONS, 2014

**DESCRIPTION:** We aim at achieving a class-agnostic scalable object detection by predicting a set of bounding boxes, which represent potential objects. More precisely, we use a Deep Neural Network (DNN), which outputs a fixed number of bounding boxes. In addition, it outputs a score for each box expressing the network confidence of this box containing an object

In the future, we hope to be able to fold the localization and recognition paths into a single network, such that we would be able to extract both location and class label information in a single feed-forward pass through the network. Even in its current state, the two-pass procedure (localization network followed by categorization network) entails 5- 10 network evaluations. Importantly, this number does not scale linearly with the number of classes to be recognized.

**LIMITATIONS:**

1. It requires very large amount of data in order to perform better than other techniques.
2. It is extremely expensive to train due to complex data models.
3. deep learning requires expensive GPUs and hundreds of machines.

## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **3.1 EXISTING SYSTEM**

The existing system uses Recurrent neural network (RNN) and Double optimization algorithm is used. RNN is introduced into the CNN, and the deep features of the image are learned in parallel using the convolutional neural network and the recurrent neural network. A RNN is a class of artificial neural networks where connections between nodes form a directed graph . A RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.

##### **3.1.1 DISADVANTAGE**

1. The computation of this recurrent neural network is slow Due to its recurrent nature,.
2. Slow and Complex training procedures.
3. It faces issues like Exploding or Vanishing.
4. It cannot process very long sequences if using tanh or relu as an activation function

## **3.2 PROPOSED SYSTEM**

In our proposed system we used Convolutional neural network algorithm and TensorFlow library . A convolutional neural network is a class of deep neural networks most commonly applied to analyzing visual image. TensorFlow is a free and open-source software library for machine learning. Inception-v3 is a pre-trained convolutional neural network model is used . The proposed model is more efficient for image classification when comparing to the other methods .

### **3.2.1 ADVANTAGES**

1. TensorFlow is used to express your computation as a data flow graph.
2. To improve the speed and recognition accuracy we used convolutional neural network algorithm
3. CNN learns the filters automatically without mentioning it explicitly. These filters help in extracting the right and relevant features from the input data

## **CHAPTER 4**

### **SYSTEM REQUIREMENTS**

#### **4.1 HARDWARE CONFIGURATION**

System : Pentium IV

Hard Disk : 100 GB.

Keyboard : standard keyboard

Monitor : 15 VGA Colour.

Mouse : Logitech.

RAM : 512 Mb.

#### **4.2 SOFTWARE REQUIREMENTS**

Operating system : WINDOWS 7/8/8.1/10, LINUX, MAC

Coding Language : JAVA

Database : IMAGENET DATASET

Library : TENSORFLOW

## **4.3 TECHNOLOGIES USED**

### **4.3.1 JAVA**

Java is a high-level programming language originally developed by Sun Microsystems and released in 1995. Java runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. This tutorial gives a complete understanding of Java. This reference will take you through simple and practical approaches while learning Java Programming language.

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

## **I WILL LIST DOWN SOME OF THE KEY ADVANTAGES OF LEARNING JAVA PROGRAMMING:**

- **Object Oriented :**

In Java, everything is an Object. Java can be easily extended since it is based on the Object model.

- **Platform Independent :**

Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on.

- **Simple :**

Java is designed to be easy to learn. If you understand the basic concept of OOP Java, it would be easy to master.

- **Secure :**

With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.

- **Architecture-neutral :**

Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system.

#### 4.3.1.1 APPLICATIONS OF JAVA PROGRAMMING

- **Multithreaded :**

With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.

- **Interpreted :**

Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.

- **High Performance :**

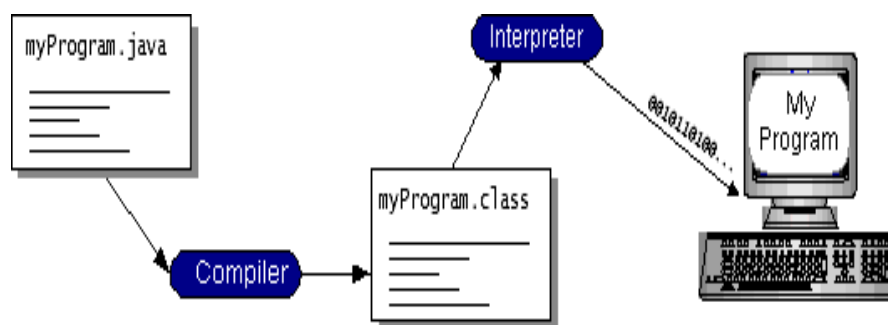
With the use of Just-In-Time compilers, Java enables high performance.

- **Distributed :**

Java is designed for the distributed environment of the internet.

- **Dynamic :**

Java is considered to be more dynamic than C or C++ since it is designed to adapt to an evolving environment. Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.



### 4.3.1.2 JAVA PLATFORM

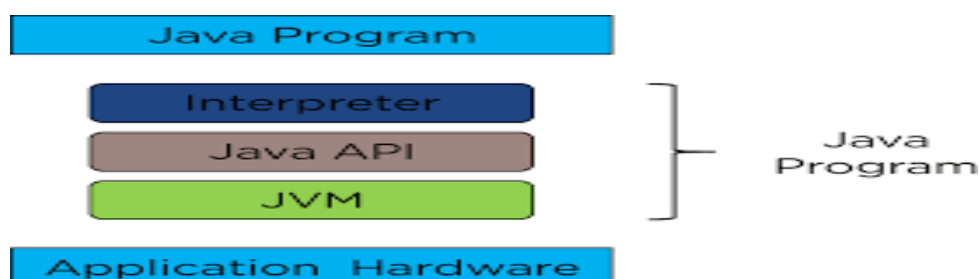
A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms. Most platforms can be described as a combination of the operating system and hardware. The **Java** platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine (Java VM)



- The Java Application Programming Interface (Java API)



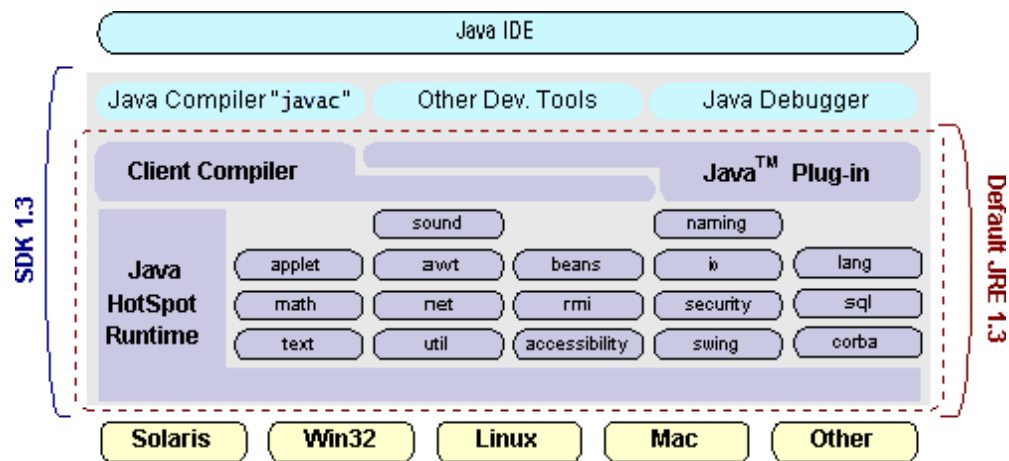


You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms. The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages.

The next section, *What Can Java Technology Do*, highlights what functionality some of the packages in the Java API provide. The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware. Native code is code that after you compile it, the compiled code runs on a specific hardware platform.

As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability. *What Can Java Technology Do?* The most common types of programs written in the Java programming language are applets and applications.

If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser. However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.



## JAVA JDE

### 4.3.1.3 APACHE NETBEANS

*NetBeans* is an integrated development environment (IDE) for Java.



#### Features:

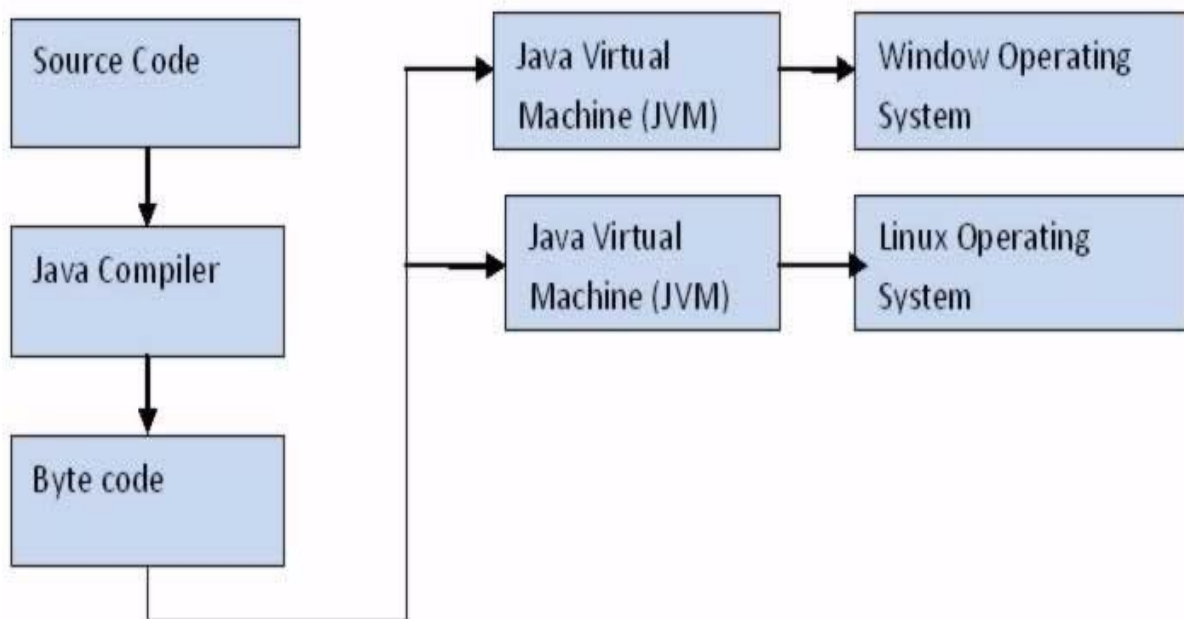
- You can easily see the structure of the Java class.
- It automatically completes the brackets.
- This app has service windows that show currently available external services.
- This Linux Java IDE offers readymade templates for writing a script.
- You can format a piece of code using a keyboard shortcut.
- The tool highlights Java variables and keywords.
- Netbeans provides parameter hints by typing \$ symbol.

**Platforms** : Windows, macOS, and Linux

#### 4.3.1.4 WORKING OF A JAVA PROGRAM

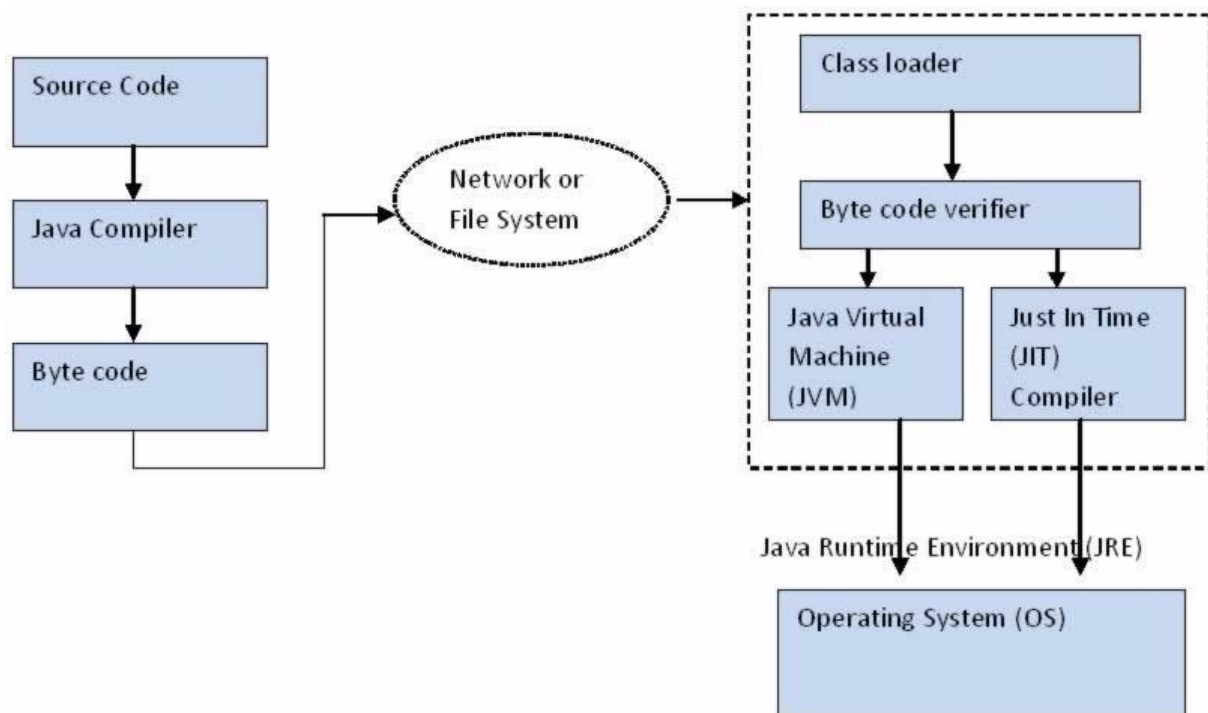
##### COMPILING SOURCE CODE

The source code must be compiled into machine code in order to become an executable program. This compilation process is referred to as compile time. Source code is written according to basic java notations and saved under extension .java and after compilation a generic byte code is produced under extension .class, this byte code is operating system independent. OS independent means once you produce a byte code then you can run it multiple times on multiple operating systems. Byte code is OS independent because JVM (Java Virtual Machine) is OS dependent.



#### 4.3.1.5 JAVA VIRTUAL MACHINE(JVM)

A Java virtual machine (JVM) is an abstract computing machine that enables a computer to run a Java program. JVM is different for every operating system as it produces different object code compatible to different OS. This object code is compatible as well as Executable.



## JVM classification

### 1. Class loader

Class loader is used to load the byte code into the virtual machine.

### 2. Byte code verification

Byte code is verified that it is correct and according to norms.

### 3. Execution engine

Just in time(JIT) compiler is used by JVM to run the queue of runnable commands.

## 4.3.4 TENSORFLOW LIBRARY

**TensorFlow** is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

Tensorflow is a symbolic math library based on dataflow and differentiable programming. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 in 2015. TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.

Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.



## CHAPTER 5

### SYSTEM DESIGN SPECIFICATIONS

#### 5.1 ARCHITECTURE DIAGRAM

Figure 5.1 explains that the input image send to pre processing section in that the image quality is increased and then send to feature extraction section , in this the shape and color of the object is extracted and then it send to object detection. Pre-trained inception will form the pattern for input image and then the detected pattern is compares with the target patterns. The compares will be done with the help of CNN. The target patterns are predefined in imagenet dataset. finally which get high accuracy will be printed in result.

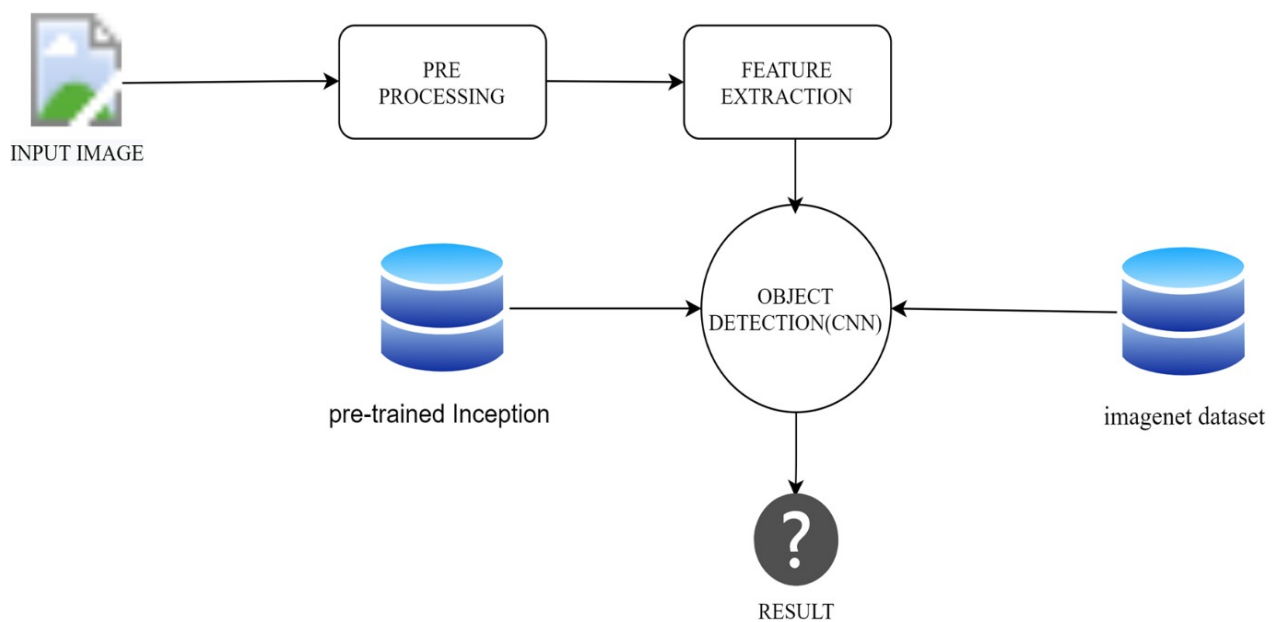


Fig 5.1 – Architecture Diagram

## 5.2 DATA FLOW DIAGRAM

The data flow diagram shows us the flow of data starting from input image then processed

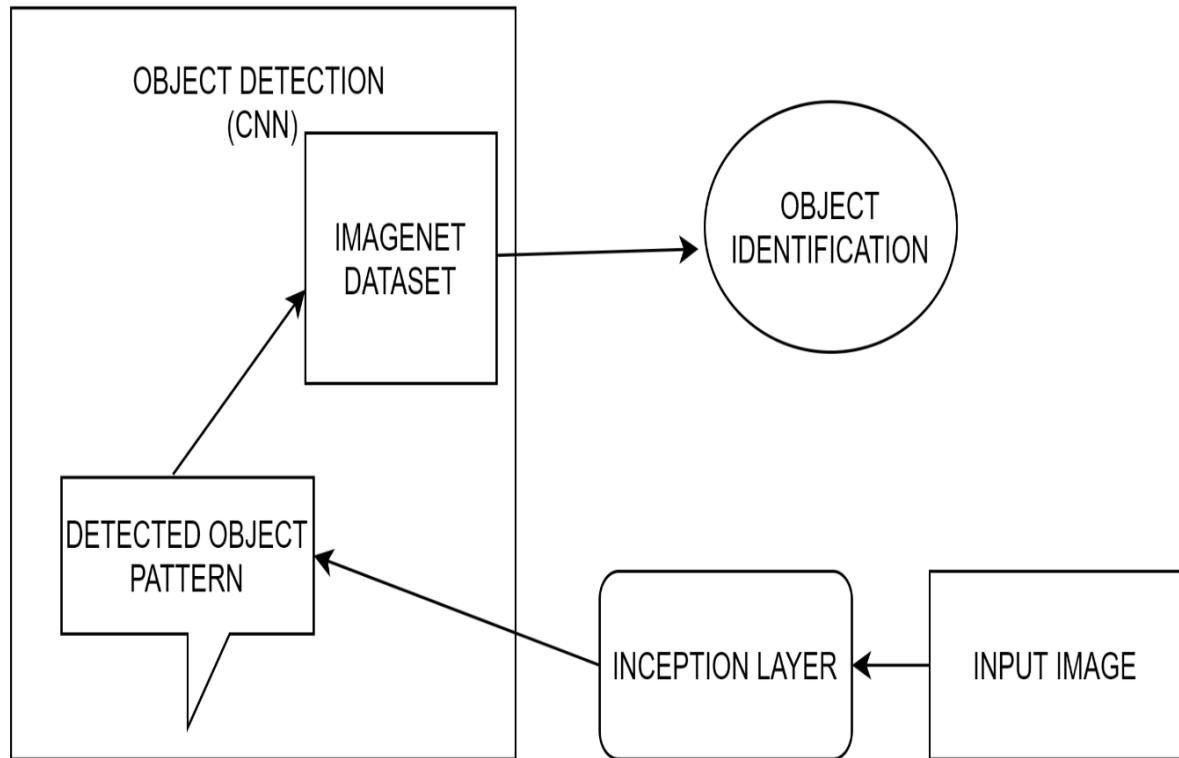


Fig 5.2 – Data Flow Diagram

By multiple processes then finally the image is classified with the help of CNN and done from the processed data as well as data from imagenet dataset.

### 5.3 USE CASE DIAGRAM

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

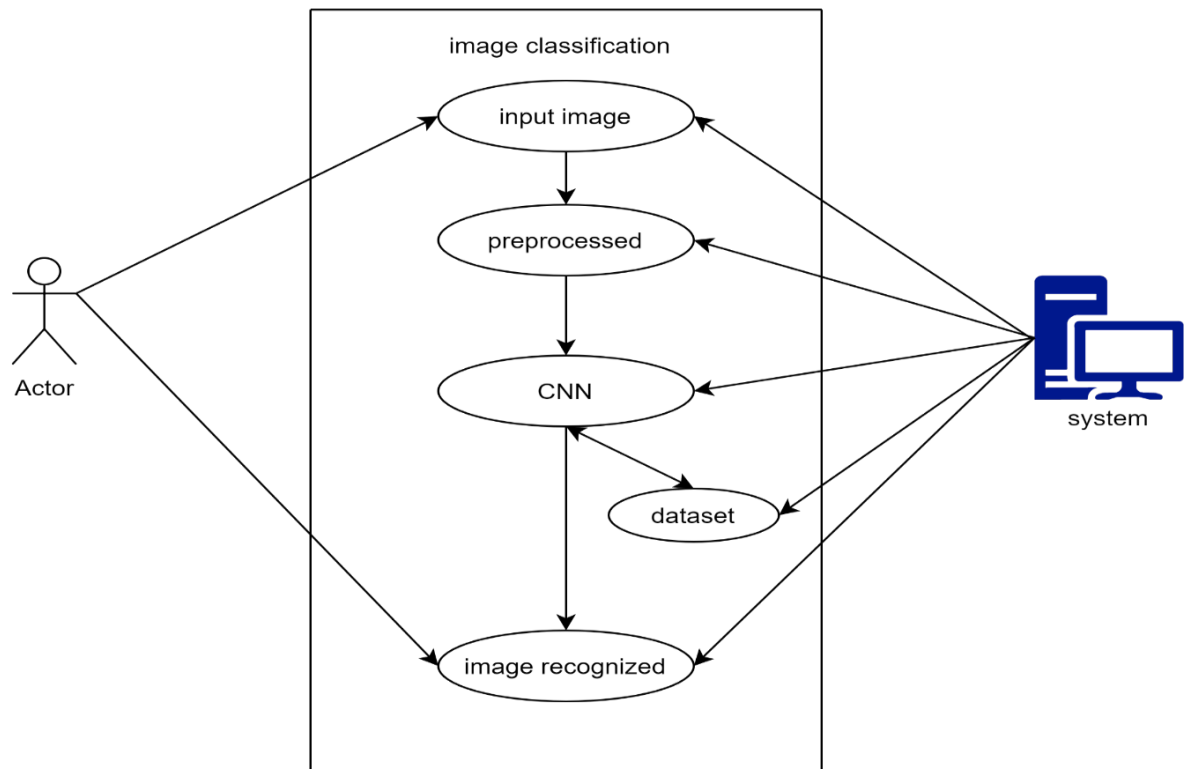


Fig 5.3 – Use case diagram

The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates the system's requirement, which includes both internal as well as external influences.



## 5.4 ACTIVITY DIAGRAM

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.

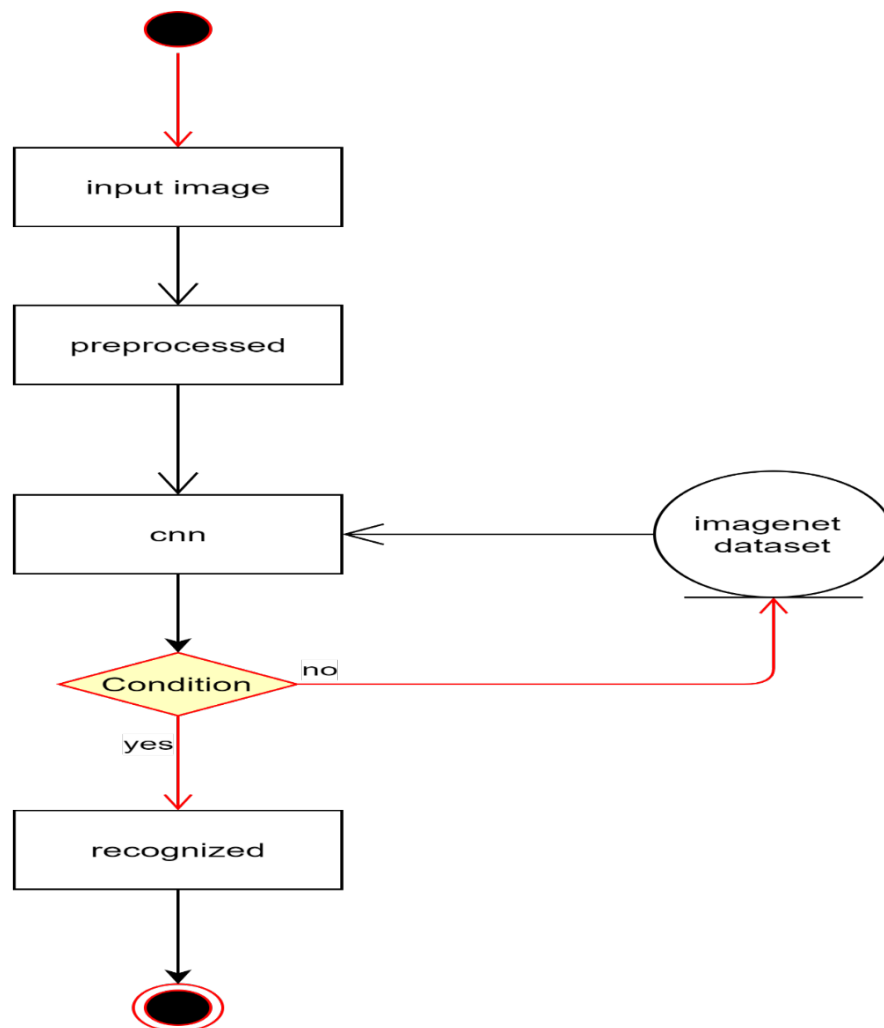


Fig 5.4 – Activity diagram

## **CHAPTER 6**

### **SYSTEM DESIGN IMPLEMENTATION**

#### **6.1 IMPLEMENTATION**

We begin the implementation in the input phase where the image is given as an input. Then image is pre-processing , CNN is categorized the detected objects into predefined classes by comparing the image patterns with the target patterns. And then finally the given input is classified to its matching target pattern.

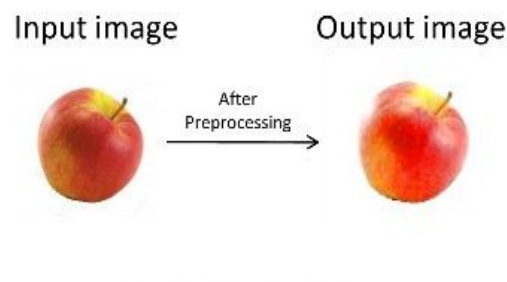
#### **6.2 MODULE DESCRIPTION**

The modules in this project are

- 1) PRE PROCESSING
- 2) FEATURE EXTRACTION
- 3) OBJECT DETECTION
- 4) PRE-TRAINED INCEPTION V3 MODEL

##### **6.2.1 PRE PROCESSING**

In this module, Raw data if applied to any classification methods does not produce good accuracy so Pre-processing will improve the quality of input image. Pre-processing is to improve the image data .The main objective of the data pre-processing step is to remove the noise (unwanted). It will remove unwanted distortion or noisy data. The Pixel and brightness will be increased.



### 6.2.2. FEATURE EXTRACTION

The process of measuring or calculation or detecting the features from the preprocessed image. In real life, all the data we collect are in large amounts. To understand this data, we need a process. Manually, it is not possible to process them. Here's when the concept of feature extraction comes in. The technique of extracting the features is useful when you have a large data set and need to reduce the number of resources without losing any important or relevant information. The two features extracted for the image are Geometric feature extraction and Color feature extraction. Feature extraction is done by implementing two techniques: LBP and Convolutional Network.

### 6.2.3 OBJECT DETECTION

In this module, detected objects are categorized into predefined classes that compare the image patterns with the target patterns by using CNN feature mapping. Supervised Classification and Unsupervised Classification are types of classification, and supervised classification is used.

**Supervised Classification** - The process of using samples of known informational classes (pre-training sets) to classify pixels of unknown identity.

The feature map is the output of one filter applied to the previous layer. A given filter is drawn across the entire previous layer, moved one pixel at a time. Each position results in an activation of the neuron and the output is collected in the feature map.

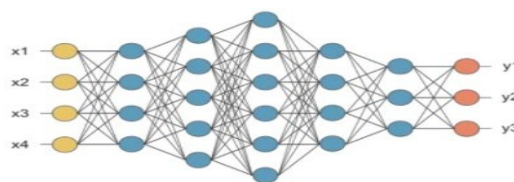
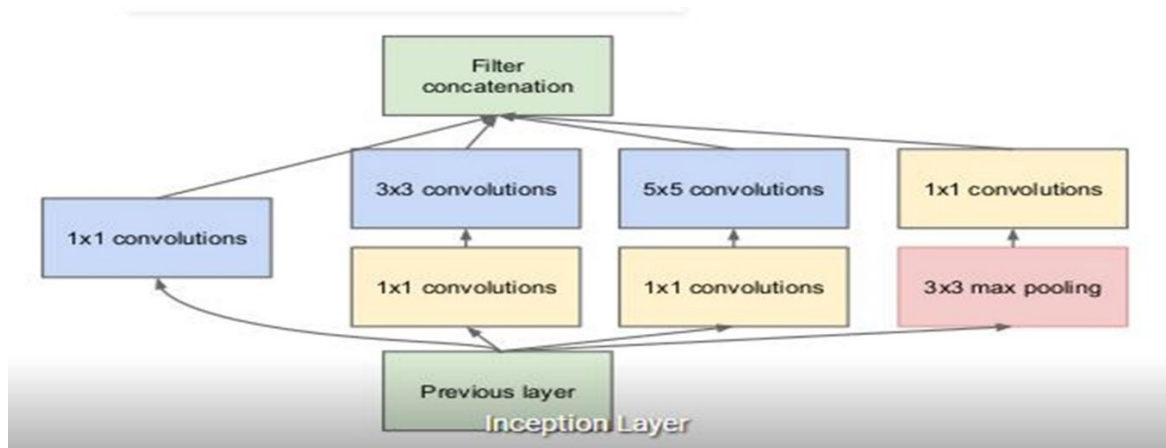


Fig 6.2.3 – feature mapping

## 6.2.4 PRE-TRAINED INCEPTION V3 MODEL

Inception-v3 is a pre-trained convolutional neural network model that is 48 layers deep. Inception layer is combination of convolutional layer and pooling layer with their output filter banks concatenated into a single output forming the input of the next stage.



## CONVOLUTIONAL LAYER

A CNN learns the values of these filters on its own during the training process. Although we still need to specify parameters such as number of filters, filter size, padding, and stride before the training process. An additional operation called Rectified Linear Unit (ReLU) has been used after every Convolution operation.  $1 \times 1$  Convolutional layer before applying another layer, which is mainly used for dimensionality reduction. A parallel Max Pooling layer, which provides another option to the inception layer

## POOLING LAYER

Pooling layer downsamples the volume spatially, independently in each depth slice of the input. The most common downsampling operation is max, giving rise to max pooling,

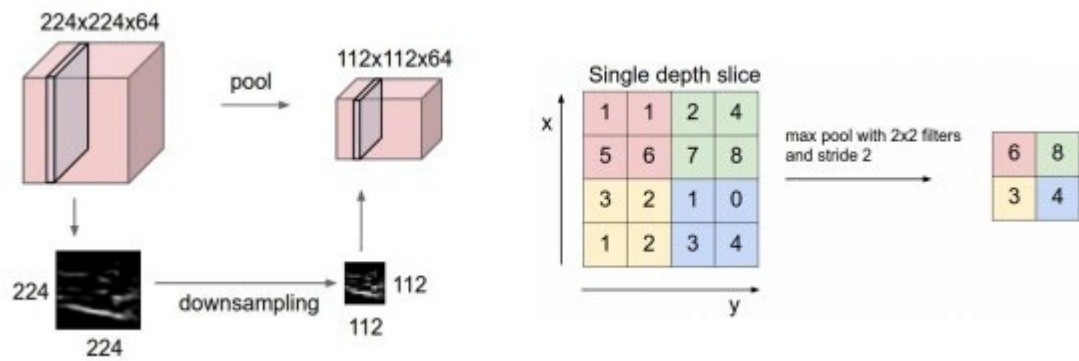


Fig 6.2.3 – pooling layer

## FULLY CONNECTED LAYER

Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular neural networks. The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like neural network.

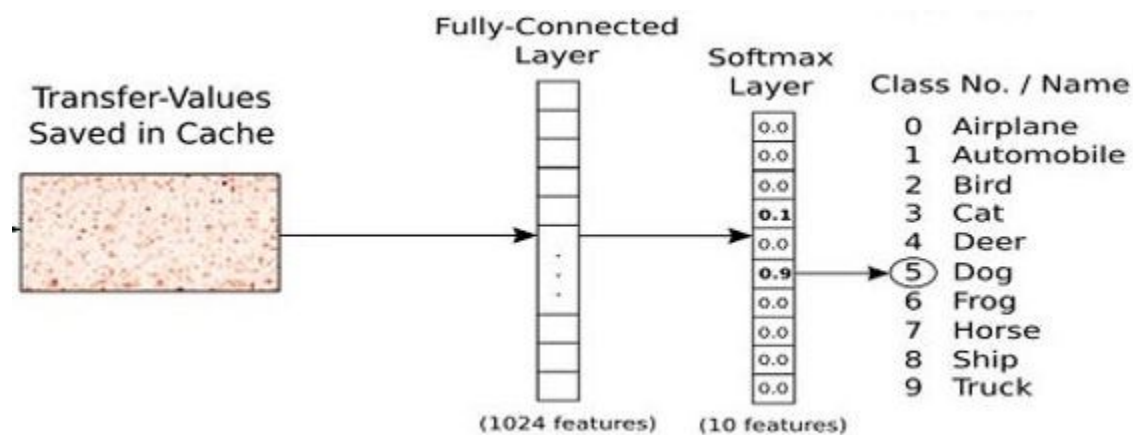


Fig 6.2.3 – fully connected layer

The output of the project which produces by fully connected layer that will be help to identify the image.

## **CHAPTER 7**

### **SYSTEM TESTING AND MAINTENANCE**

#### **7.1 TESTING OBJECTIVES**

Testing is a set of activities that can be planned in advance and conducted systematically. For this reason a template for software testing, a set of steps into which we can place specific test case design techniques and testing methods should be defined for the software process. Testing often accounts for more effort than any other software engineering activity. If it is conducted haphazardly, time is wasted, unnecessary effort is expended, and even worse, errors sneak through undetected. It would therefore seem reasonable to establish a systematic strategy for testing software.

In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently.

The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So we should be more conscious to give input data. It is important to give correct inputs to efficient testing. Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works.

Inadequate testing or non-testing leads to errors that may appear a few months later. This will create two problems, Time delay between the cause and

appearance of the problem. The effect of the system errors on files and records within the system. The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the system to its limits.

The testing process focuses on logical intervals of the software ensuring that all the statements have been tested and on the function intervals (i.e.,) conducting tests to uncover errors and ensure that defined inputs will produce actual results that agree with the required results. Testing has to be done using the two common steps Unit testing and Integration testing.

In the project system testing is made as follows:

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. This is the final step in the system life cycle. Here we implement the tested error-free system into a real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every months or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and reports.

## **7.2 TYPE OF TESTING**

There are two type of testing according their behaviours,

1. Unconventional Testing
2. Conventional Testing

### **7.2.1 UNCONVENTIONAL TESTING**

Unconventional testing is a process of verification which is done by the SQA (Software Quality Assurance) team. It is a prevention technique which is

performing from begging to ending of the project development. In this process SQA team verifies project development activities and ensures that developing projects are fulfilling the requirement of the client or not. In this testing the SQA team follows these methods:

1. Peer review
2. Code walk and throw
3. Inspection
4. Document Verification

## **7.2.2 CONVENTIONAL TESTING**

Conventional Testing is a process of finding the bugs and validating the project. Testing team is involved in this testing process and validating that the developed project is according to client requirement or not. This process is a correction technique where the testing team finds bugs and reports to the development team for correction on the developed project built.

## **7.3 TEST CASE DESIGN**

### **7.3.1 UNIT TESTING**

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use. In the company as well as seeker registration form, the zero length username and password are given and checked. Also the duplicate username is given and checked. In the job and question entry, the button will send data to the server only if the client side validations are made. The dates are entered in the wrong manner



and checked. Wrong email-id and website URL (Universal Resource Locator) is given and checked.

### **7.3.2 INTEGRATION TESTING**

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works.

### **7.3.3 VALIDATION TESTING**

The final step involves Validation testing, which determines whether the software function as the user expected. The end-user rather than the system developer conducts this test for most software developers as a process called “Alpha and Beta test” to uncover that only the end user seems able to find. The compilation of the entire project is based on the full satisfaction of the end users. In the project, validation testing is made in various forms. In question entry form, the correct answer only will be accepted in the answer box. The answers other than the four given choices will not be accepted.

## **7.4 TESTING STRATEGIES**

A number of software testing strategies have been proposed in the literature. All provide the software developer with a template for testing and all have the following generic characteristics:

Testing begins at the component level and works “outward” toward the integration of the entire computer-based system.

1. Different testing techniques are appropriate at different points in time.
2. The developer of the s/w conducts testing and for large projects, independent test groups.

#### **7.4.1 INTEGRATION BOX TESTING**

The strategies for integrating software components into a functioning product include the bottom-up strategy, the top-down strategy and to ensure that modules will be available for integration into the evolving software product when needed. The integration strategy dictates the order in which modules must be available and thus exerts a strong influence on the order in which modules are written, debugged and unit tested.

#### **7.4.2 WHITE BOX TESTING**

It is just the vice versa of the Black Box testing. There we do not watch the internal variables during testing. This gives a clear idea about what is going on during execution of the system. The point at which the bug occurs were all clear and were removed.

#### **7.4.3 BLACK BOX TESTING**

In this testing we give input to the system and test the output. Here we do not go for watching the internal file in the system and what are the changes made on them for the required output.

#### **7.4.4 INTERFACE TESTING**

The Interface Testing is performed to verify the interfaces between sub modules while performing integration of sub modules aiding master modules recursively.

### **7.4.5 MODULE TESTING**

Module Testing is a process of testing the system, module by module. It includes the various inputs given, outputs produced and their correctness. By testing in this method we would be very clear of all the bugs that have occurred.

### **7.4.6 SMOKE TESTING**

Smoke testing refers to physical tests made to closed systems of pipes to test for leaks. By metaphorical extension, the term is also used for the first test made after assembly or repairs to a system, to provide some assurance that the system under test will not catastrophically fail.

After a smoke test proves that the pipes will not leak, the keys seal properly, the circuit will not burn, or the software will not crash outright”, the system is ready for more stressful testing.

### **7.5 MAINTENANCE**

The objectives of this maintenance work are to make sure that the system gets into work all time without any bugs. Provision must be for environmental changes which may affect the computer or software system. This is called the maintenance of the system. Nowadays there is a change in the software world. Due to this rapid change, the system should be capable of adapting these changes. In our project the process can be added without affecting other parts of the system. Maintenance plays a vital role. The system will be able to accept any modification after its implementation. This system has been designed to favor all new changes. Doing this will not affect the system’s performance or its accuracy. This is the final step in the system life cycle.

As a rule, system testing takes, as its input, all of the “integrated” software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the “inter-assemblages” and also within the system as a whole.

## **CHAPTER 8**

### **CONCLUSION AND FUTURE ENHANCEMENT**

#### **8.1 CONCLUSION**

In order to improve the ability of the convolutional neural network to classify and recognize two-dimensional images and speed up the convergence of the algorithm, this proposes a new convolutional network algorithm. an image classifier using convolutional neural network, which is more efficient for image classification when comparing to the other methods. It is usefully for classifying larger number of image with in short time. In order to improve the ability of the convolutional neural network to classify and recognize two-dimensional images by using tensorflow . Inception v3 is a widely-used image recognition model that has been shown greater than 78.1% accuracy on the ImageNet dataset and upto 98.9% accuracy. . Experiments show that the proposed convolutional neural network algorithm can improve the feature extraction accuracy and image recognition ability of convolutional neural network.

#### **8.2 FUTURE ENHANCEMENT**

The future development of the concept might take it to another level by adding a feature like Image can be classified and keep in separate folders. Object recognition can be used for classifying the images automatically. N number of object can be identify at a time, it will save more time.

## **APPENDIX 1**

### **SOURCE CODE**

```
package com.emaraic.ObjectRecognition;

import com.esotericsoftware.tablelayout.swing.Table;

import java.awt.Image;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import java.io.IOException;

import java.nio.charset.Charset;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.util.Arrays;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.imageio.ImageIO;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFileChooser;

import javax.swing.JFrame;
```

```

import javax.swing.JLabel;

import javax.swing.JTextField;

import javax.swing.SwingUtilities;

import javax.swing.filechooser.FileNameExtensionFilter;

import org.tensorflow.DataType;

import org.tensorflow.Graph;

import org.tensorflow.Output;

import org.tensorflow.Session;

import org.tensorflow.Tensor;


public class Recognizer extends JFrame implements ActionListener {

    private Table table;

    private JButton predict;

    private JButton incep;

    private JButton img;

    private JFileChooser incepch;

    private JFileChooser imgch;

    private JLabel viewer;

    private JTextField result;

    private JTextField imgpth;

    private JTextField modelpth;

```

```

private FileNameExtensionFilter imgfilter = new

    FileNameExtensionFilter( "JPG & JPEG Images", "jpg", "jpeg");

private String modelpath;

private String imagepath;

private boolean modelselected = false;

private byte[] graphDef;

private List<String> labels;


public Recognizer() {

    setTitle("Object Recognition - surendar");

    setSize(500, 500);

    table = new Table();

    predict = new JButton("Predict");

    predict.setEnabled(false);

    incep = new JButton("Choose Inception");

    img = new JButton("Choose Image");

    incep.addActionListener(this);

    img.addActionListener(this);

    predict.addActionListener(this);


    incepch = new JFileChooser();

```



```
imgch = new JFileChooser();

imgch.setFileFilter(imgfilter);

imgch.setFileSelectionMode(JFileChooser.FILES_ONLY);

incepch.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);


result=new JTextField();

modelpth=new JTextField();

imgpth=new JTextField();

modelpth.setEditable(false);

imgpth.setEditable(false);

viewer = new JLabel();

getContentPane().add(table);

table.addCell(modelpth).width(250);

table.addCell(incep);

table.row();

table.addCell(imgpth).width(250);

table.addCell(img);


table.row();

table.addCell(viewer).size(200, 200).colspan(2);

table.row();

table.addCell(predict).colspan(2);
```

```

table.row();

table.addCell(result).width(300).colspan(2);


setLocationRelativeTo(null);


setResizable(false);

setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

@Override

public void actionPerformed(ActionEvent e) {

    if (e.getSource() == incep) {

        int returnVal = incepch.showOpenDialog(this);

        if (returnVal == JFileChooser.APPROVE_OPTION) {

            File file = incepch.getSelectedFile();

            modelpath = file.getAbsolutePath();

            modelpth.setText(modelpath);

            System.out.println("Opening: " + file.getAbsolutePath());

            modelselected = true;

```

```

        graphDef = readAllBytesOrExit(Paths.get(modelpath,
"tensorflow_inception_graph.pb"));

        labels = readAllLinesOrExit(Paths.get(modelpath,
"imagenet_comp_graph_label_strings.txt"));

    } else {

        System.out.println("Process was cancelled by user.");

    }

} else if (e.getSource() == img) {

    int returnVal = imgch.showOpenDialog(Recognizer.this);

    if (returnVal == JFileChooser.APPROVE_OPTION) {

        try {

            File file = imgch.getSelectedFile();

            imagepath = file.getAbsolutePath();

            imgpth.setText(imagepath);

            System.out.println("Image Path: " + imagepath);

            Image img = ImageIO.read(file);

            viewer.setIcon(new ImageIcon(img.getScaledInstance(200, 200,
200)));

            if (modelselected) {

                predict.setEnabled(true);

```

```

        }

    } catch (IOException ex) {

Logger.getLogger(Recognizer.class.getName()).log(Level.SEVERE, null, ex);

        }

    } else {

        System.out.println("Process was cancelled by user.");

    }

    } else if (e.getSource() == predict) {

        byte[] imageBytes = readAllBytesOrExit(Paths.get(imagepath));

        try (Tensor image = Tensor.create(imageBytes)) {

            float[] labelProbabilities = executeInceptionGraph(graphDef, image);

            int bestLabelIdx = maxIndex(labelProbabilities);

            result.setText("");

            result.setText(String.format(

                "BEST MATCH: %s (%.2f%% likely)",

                labels.get(bestLabelIdx), labelProbabilities[bestLabelIdx] *

100f));

            System.out.println(

                String.format(

                    "BEST MATCH: %s (%.2f%% likely)",

```

```

        labels.get(bestLabelIdx), labelProbabilities[bestLabelIdx] *
100f));
    }

}

}

```

```

private static float[] executeInceptionGraph(byte[] graphDef, Tensor image)
{
    try (Graph g = new Graph()) {
        g.importGraphDef(graphDef);
        try (Session s = new Session(g);
            Tensor result = s.runner().feed("DecodeJpeg/contents",
image).fetch("softmax").run().get(0)) {
            final long[] rshape = result.shape();
            if (result.numDimensions() != 2 || rshape[0] != 1) {
                throw new RuntimeException(
                    String.format(
                        "Expected model to produce a [1 N] shaped tensor where
N is the number of labels, instead it produced one with shape %s",
                        Arrays.toString(rshape)));
            }
        }
    }
}

```

```

        int nlabels = (int) rshape[1];

        return result.copyTo(new float[1][nlabels])[0];

    }

}

```

```

private static int maxIndex(float[] probabilities) {

    int best = 0;

    for (int i = 1; i < probabilities.length; ++i) {

        if (probabilities[i] > probabilities[best]) {

            best = i;

        }

    }

    return best;

}

```

```

private static byte[] readAllBytesOrExit(Path path) {

    try {

        return Files.readAllBytes(path);

    } catch (IOException e) {

        System.err.println("Failed to read [" + path + "]: " + e.getMessage());

        System.exit(1);

    }

}

```

```

    }

    return null;

}

private static List<String> readAllLinesOrExit(Path path) {

    try {

        return Files.readAllLines(path, Charset.forName("UTF-8"));

    } catch (IOException e) {

        System.err.println("Failed to read [" + path + "]: " + e.getMessage());

        System.exit(0);

    }

    return null;

}

static class GraphBuilder {

    GraphBuilder(Graph g)

```

```
this.g = g;
```

```
}
```

```
Output div(Output x, Output y) {
```

```
    return binaryOp("Div", x, y);
```

```
}
```

```
Output sub(Output x, Output y) {
```

```
    return binaryOp("Sub", x, y);
```

```
}
```

```
Output resizeBilinear(Output images, Output size) {
```

```
    return binaryOp("ResizeBilinear", images, size);
```

```
}
```

```
Output expandDims(Output input, Output dim) {
```

```
    return binaryOp("ExpandDims", input, dim);
```

```
}
```

```
Output cast(Output value, DataType dtype) {
```

```
    return g.opBuilder("Cast", "Cast").addInput(value).setAttr("DstT",  
dtype).build().output(0);
```



```
}
```

```
Output decodeJpeg(Output contents, long channels) {
```

```
    return g.opBuilder("DecodeJpeg", "DecodeJpeg")
```

```
        .addInput(contents)
```

```
        .setAttr("channels", channels)
```

```
        .build()
```

```
        .output(0);
```

```
}
```

```
Output constant(String name, Object value) {
```

```
    try (Tensor t = Tensor.create(value)) {
```

```
        return g.opBuilder("Const", name)
```

```
            .setAttr("dtype", t.dataType())
```

```
            .setAttr("value", t)
```

```
            .build()
```

```
            .output(0);
```

```
    }
```

```
}
```

```
private Output binaryOp(String type, Output in1, Output in2) {
```

```
        return g.opBuilder(type,  
type).addInput(in1).addInput(in2).build().output(0);  
    }
```

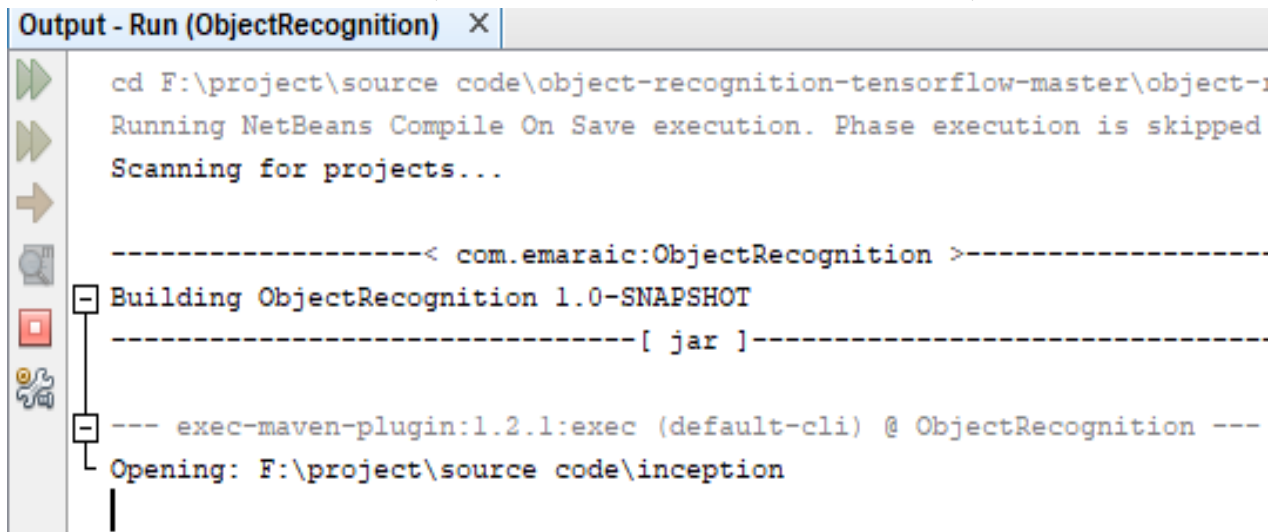
```
    private Graph g;  
}
```

```
public static void main(String[] args) {  
    SwingUtilities.invokeLater(new Runnable() {  
        public void run() {  
            new Recognizer().setVisible(true);  
  
        }  
    });  
}  
  
}
```

## APPENDIX 2

### OUTPUT SCREENSHOTS

#### OUTPUT-(SET PATH TO INCEPTION)

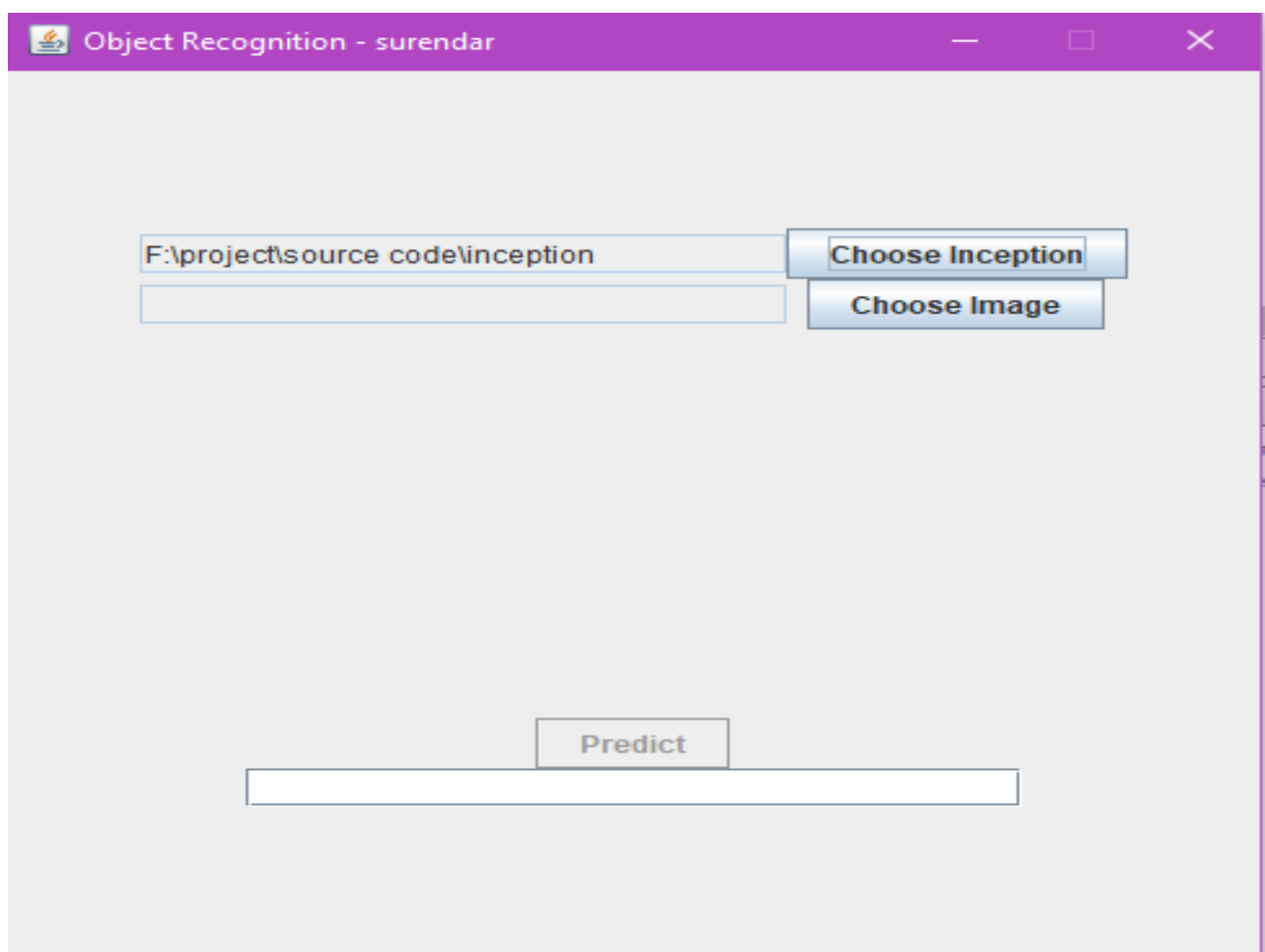


Output - Run (ObjectRecognition) X

```
cd F:\project\source code\object-recognition-tensorflow-master\object-1
Running NetBeans Compile On Save execution. Phase execution is skipped
Scanning for projects...

-----< com.emaraic:ObjectRecognition >-----
Building ObjectRecognition 1.0-SNAPSHOT
-----[ jar ]-----

--- exec-maven-plugin:1.2.1:exec (default-cli) @ ObjectRecognition ---
Opening: F:\project\source code\inception
|
```

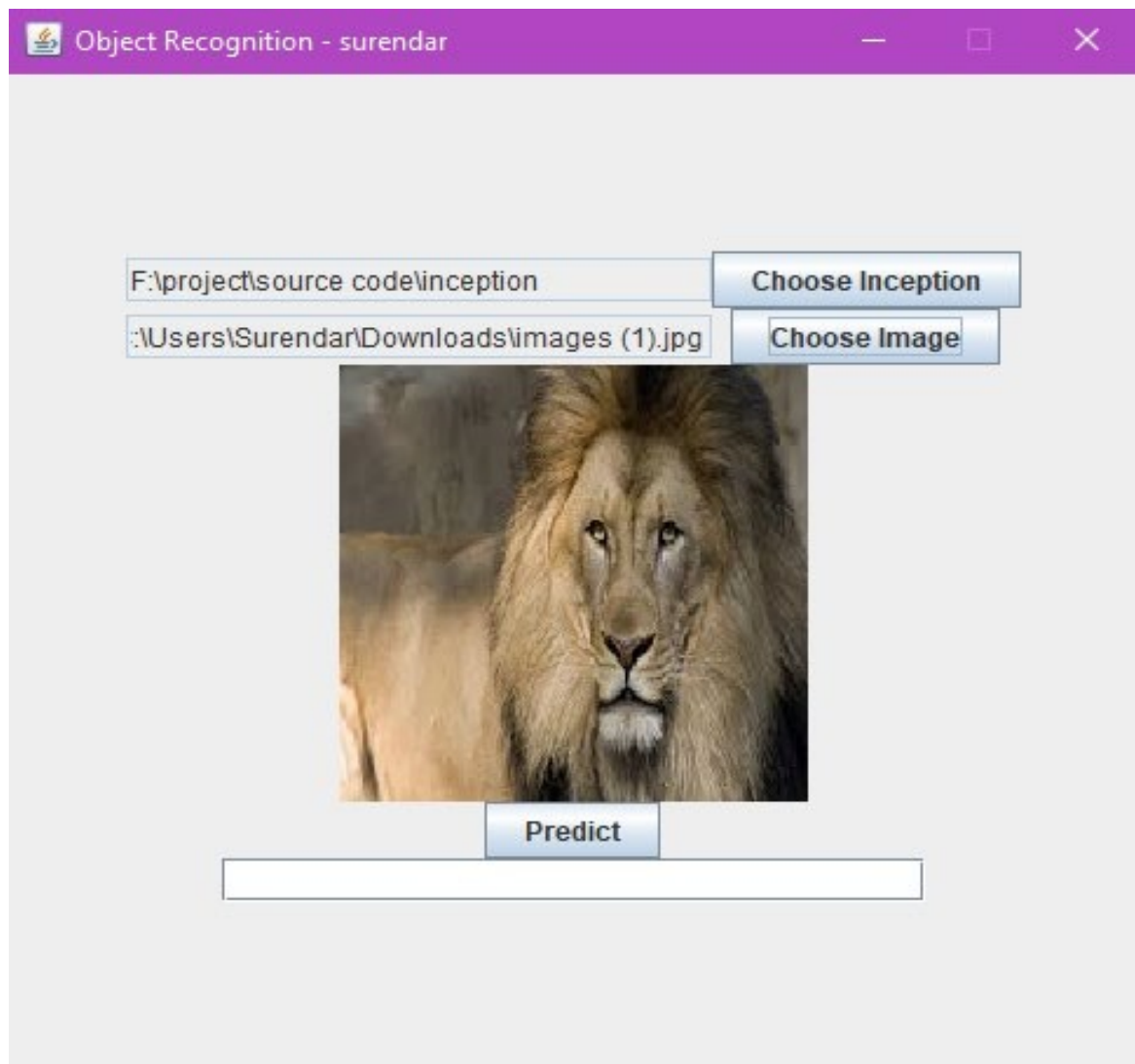


## OUTPUT-(INSERT THE IMAGE)

```
Output - Run (ObjectRecognition) X
cd F:\project\source code\object-recognition-tensorflow-master\object-recognition-master
Running NetBeans Compile On Save execution. Phase execution is skipped and
Scanning for projects...

-----< com.emaraic:ObjectRecognition >-----
Building ObjectRecognition 1.0-SNAPSHOT
-----[ jar ]-----

--- exec-maven-plugin:1.2.1:exec (default-cli) @ ObjectRecognition ---
Opening: F:\project\source code\inception
Image Path: C:\Users\Surendar\Downloads\images (1).jpg
```

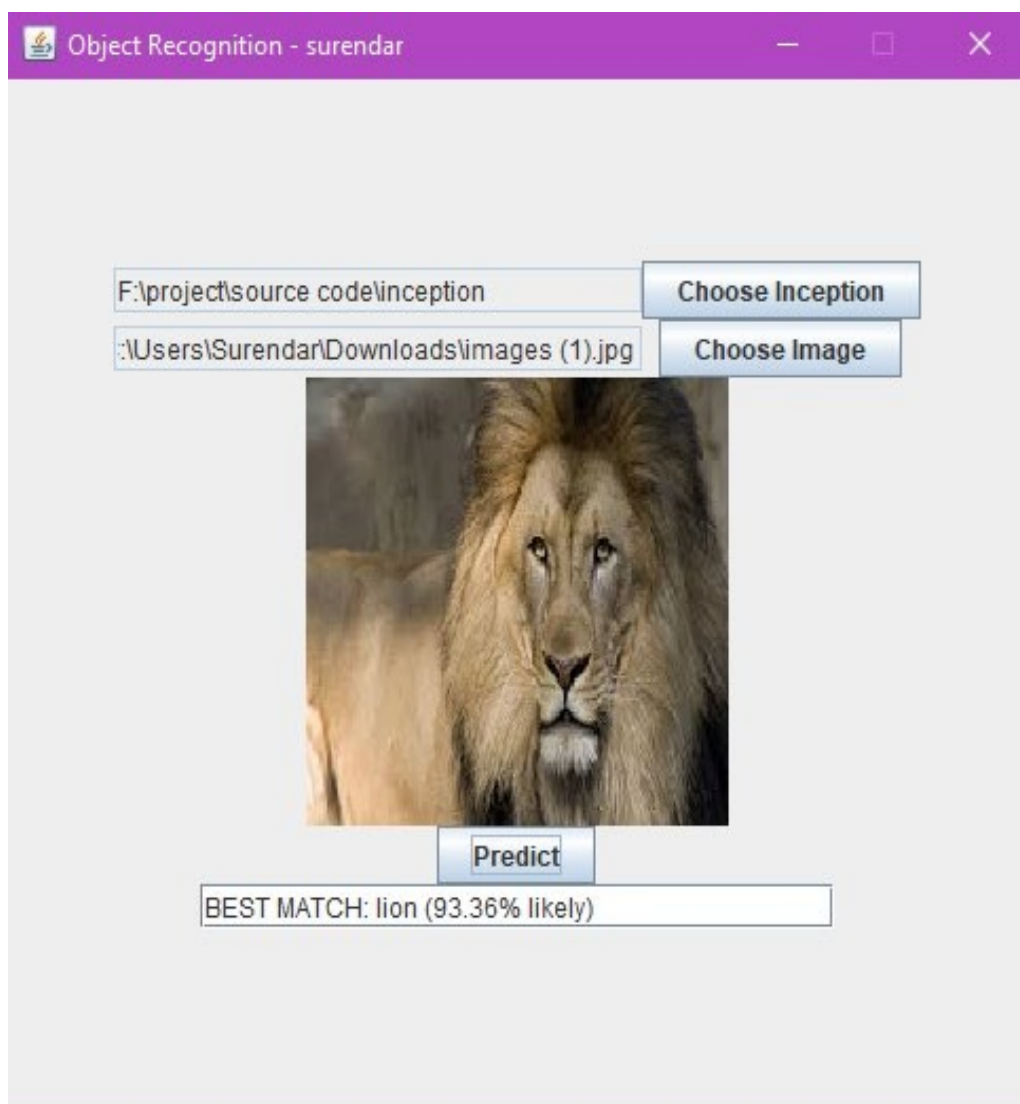


## OUTPUT-(FINAL PREDICTION 1)

```
Output - Run (ObjectRecognition) X
cd F:\project\source code\object-recognition-tensorflow-master\obje
Running NetBeans Compile On Save execution. Phase execution is skip
Scanning for projects...

-----< com.emaraic:ObjectRecognition >-----
Building ObjectRecognition 1.0-SNAPSHOT
-----[ jar ]-----

--- exec-maven-plugin:1.2.1:exec (default-cli) @ ObjectRecognition -
Opening: F:\project\source code\inception
Image Path: C:\Users\Surendar\Downloads\images (1).jpg
BEST MATCH: lion (93.36% likely)
```



## OUTPUT-(FINAL PREDICTION 2)



## REFERENCES

- [1] Youhui Tian, (July 21,2020). “Artificial Intelligence Image Recognition Method Based on Convolutional Neural Network Algorithm”,.
- [2] S. Liansheng, Z. Xiao, H. Chongtian, T. Ailing, and A. Krishna Asundi, (Feb 2019), “Silhouette-free interference-based multiple-image encryption using cascaded fractional Fourier transforms,” *Opt. Lasers Eng.*, vol. 113, pp.29–37,.
- [3] X. Zhu, Z. Li, X.-Y. Zhang, P. Li, Z. Xue, and L. Wang, (Oct 2019), “Deep convolutional representations and kernel extreme learning machines for image classification,” *Multimedia Tools Appl.*, vol. 78, no. 20, pp. 29271–29290,.
- [4] F. Wang, D. Jiang, H. Wen, and H. Song, (Nov 2019), “AdaBoost-based security level classification of mobile intelligent terminals,” *Supercomput.*, vol. 75, no. 11, pp. 7460–7478,.
- [5] L. Wen, K. Zhou, and S. Yang, (Mar 2019) ,“A shape-based clustering method for pattern recognition of residential electricity consumption,” *Cleaner Prod.*, vol. 212, pp. 475–488,.
- [6] T. Zan, Z. Liu, H. Wang, M. Wang, and X. Gao, (Mar 2020), “Control chart pattern recognition using the convolutional neural network,” *J. Intell. Manuf.*, vol. 31, no. 3, pp. 703–716,.
- [7] J. Yu, X. Zheng, and S. Wang, (July 2019), “A deep autoencoder feature learning method for process pattern recognition,” *J. Process Control*, vol. 79, pp. 1–15,.
- [8] D. Freire-Obregón, F. Narducci, S. Barra, and M. Castrillón-Santana, (Sep 2019), “Deep learning for source camera identification on mobile devices,” *Pattern Recognit. Lett.*, vol. 126, pp. 86–91,.