REALTIME VISUAL RECOGNITION IN DEEP CONVOLUTIONAL NEURAL NETWORKS

A PROJECT REPORT

Submitted by

SREE SATHYA SHREYA.T

[REGISTER NO:211417104267]

VAISHNAVI KARISHMA NAIDU.M

[REGISTER NO:211417104287]

VARSHAA PARAMESSH

[REGISTER NO:211417104293]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2021

BONAFIDE CERTIFICATE

Certified that this project report **“..........REALTIME VISUAL RECOGNITION IN DEEP CONVOLUTIONAL NEURAL NETWORK..........”**is the bonafide work of “**..........SREE SATHYA SHREYA.T(211417104267),VAISHNAVI KARISHMA NAIDU.M(211417104287),VARSHAA PARAMESSH(211417104293)..........**” who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.S.MURUGAVALLI,M.E.,Ph.D.,	**Mrs.SANGEETHA KRISHNAN**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	ASSISTANT PROFESSOR
DEPARTMENT OF CSE,	DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,	PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI,	NAZARATHPETTAI,
POONAMALLEE,	POONAMALLEE,
CHENNAI-600 123.	CHENNAI-600 123.

Certified that the above candidate(s) was/were examined in the Anna University Project Viva-Voce Examination held on 06.08.2021.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like express our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHIKUMAR, M.E., Ph.D.,** and

Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A., for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank our **Project Guide Mrs.SANGEETHA KRISHNAN** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

NAME OF THE STUDENTS

SREE SATHYA SHREYA .T

VAISHNAVI KARISHMA NAIDU.M

VARSHAA PARAMESSH

ABSTRACT

Detecting the objects from images and videos has always been the point of active research area for the applications of computer vision and artificial intelligence namely robotics, self-driving cars, automated video surveillance, crowd management, home automation and manufacturing industries, activity recognition systems, medical imaging, and biometrics. The recent years witnessed the boom of deep learning technology for its effective performance on image classification. Convolutional neural networks have provided promising results for object detection by alleviating the need for human expertise for manually handcrafting the features for extraction. It allows the model to learn automatically by letting the neural network to be trained on large-scale image data using powerful and robust GPUs in a parallel way, thus, reducing training time. This paper aims to highlight the state-of-the-art approaches based on the convolutional neural networks especially designed for object detection from images.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

YOLO-	You Only Live Once
CNN-	Convolutional Neural Networks
SSD-	Single Shot Detector
R-CNN-	Region Based Convolutional Neural Networks

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Well-researched domains of object detection include face detection and pedestrian detection. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance. Object detection is the task of detecting instances of objects of a certain class within an image. The state-of-the-art methods can be categorized into two main types: one-stage methods and two stage-methods. One-stage methods prioritize inference speed, and example models include YOLO, SSD and RetinaNet. Two-stage methods prioritize detection accuracy, and example models include Faster R-CNN, Mask R-CNN and Cascade R-CNN. Human beings have the ability of identifying the region or object in an image. Visual Recognition aims to detecting and identifying the object or region in an image automatically. Earlier methods for salient detection are mainly inspired by cognitive studies of visual attention. It is widely used in computer vision tasks such as image annotation, activity recognition, face detection, face recognition, video object co-segmentation. It is also used in tracking objects, for example tracking a ball during a football match, tracking movement of a cricket bat, or tracking a person in a video. In our project, we train nearly 60-65 images for object detection. It can recognize images in blur stage or less brightness. After detection it specifies the name of the object detected.

1.2 PROBLEM DEFINITION

In recent days the detection of salient objects is done with convolutional neural networks(CNN). Later the development of fully convolutional neural networks (FCN), moreover there are chances for improvisation over generic FCN models which deals with scale space problems. Also on the other side holistically-nested edge detector provides a skip layer structure with deep supervision for edge and boundary detection. Our framework takes full advantage of extracting from FCN providing more advanced representation at each layer, this property is used for segment detection. It has advantages in terms of efficiency i.e. ,0.08 second per image, effectiveness and simplicity over existing algorithms. It can be analyzed on the role of training data on performance, the experimental results provide a more reasonable and powerful training set for future research.

CHAPTER 2

Literature Survey

1.P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627 1645, Sep. 2010.

Described an object detection system based on mixtures of multi scale deformable part models. Their system was able to represent highly variable object classes and achieves state-of-the-art results in the PASCAL object detection challenges. They combined a margin-sensitive approach for data-mining hard negative examples with a formalism we call latent SVM. This led to an iterative training algorithm that alternates between fixing latent values for positive examples and optimizing the latent SVM objective function. Their system relied heavily on new methods for discriminative training of classifiers that make use of latent information. It also relied heavily on efficient methods for matching deformable models to images. The described framework allows for exploration of additional latent structure. For example, one can consider deeper part hierarchies (parts with parts) or mixture models with many components.

2. B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," Int. J. Comput. Vis., vol. 77, nos. 1-3, pp. 259-289, May2008.

Presented a novel method for detecting and localizing objects of a visual category in cluttered real-world scenes. Their approach considered object categorization and figure-ground segmentation as two interleaved processes that closely collaborate towards a common goal. The tight coupling between those two processes allows them to benefit from each other and improve the combined performance. The core part of their approach was a highly flexible learned representation for object shape that could combine the information observed on different training examples in a probabilistic extension of the Generalized Hough Transform. As they showed, the resulting approach can detect categorical objects in novel images and automatically infer a probabilistic segmentation from the recognition result. This segmentation was then in turn used to again improve recognition by allowing the system to focus its efforts on object pixels and to discard misleading influences from the background. Their extensive evaluation on several large data sets showed that the proposed system was applicable to a range of different object categories, including both rigid and articulated objects. In addition, its flexible representation allowed it to achieve competitive object detection performance already from training sets that were between one and two orders of magnitude smaller than those used in comparable systems.

3. J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classi cation of texture and object categories: A comprehensive study," in Proc. Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW), Jun. 2006, p.13.

A large-scale evaluation of an approach that represented images as distributions (signatures or histograms) of features extracted from a sparse set of key-point locations and learnt a Support Vector Machine classifier with kernels based on two effective measures for comparing distributions. They first evaluated the performance of the proposed approach with different key-point detectors and descriptors, as well as different kernels and classifiers. Then, they conducted a comparative evaluation with several modern recognition methods on 4 texture and 5 object databases.

4. P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), vol. 1. Dec. 2001, pp.511-518.

In a conference on pattern recognition described a machine learning approach for visual object detection which was capable of processing images extremely rapidly and achieving high detection rates. Their work was distinguished by three key contributions. The first was the introduction of a new image representation called the "integral image" which allowed the features used by their detector to be computed very quickly. The second was a learning algorithm, based on AdaBoost, which used to select a small number of critical visual features from a larger set and yield extremely efficient classifiers. The third contribution was a method for combining increasingly more complex classifiers in a "cascade" which allowed background regions of the image to be quickly discarded while spending more computation on promising object-like regions. The cascade could be viewed as an object specific focus-of-attention mechanism which unlike some of the previous approaches provided statistical guarantees that discarded regions were unlikely to contain the object of interest. They had done some testing over face detection where the system yielded detection rates comparable to the best of previous systems. Used in real-time applications, the detector runs at 15 frames per second without resorting to image differencing or skin color detection.

5. M. Weber, M. Welling, and P. Perona, "Towards automatic discovery of object categories," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., vol. 2. Jun. 2000, pp.101-108.

A method to learn heterogeneous models of object classes for visual recognition. The training images, that they used, contained a preponderance of clutter and the learning was unsupervised. Their models represented objects as probabilistic constellations of rigid parts (features). The variability within a class was represented by a join probability density function on the shape of the constellation and the appearance of the parts. Their method automatically identified distinctive features in the training set. The set of model parameters was then learned using expectation maximization. When trained on different, unlabeled and non-segmented views of a class of objects, each component of the mixture model could adapt to represent a subset of the views. Similarly, different component models could also specialize on sub-classes of an object class. Experiments on images of human heads, leaves from different species of trees, and motor-cars demonstrated that the method works well over a wide variety of objects.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

This paper describes a machine learning approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates. This work is distinguished by three key contributions. The first is the introduction of a new image representation called the "integral image" which allows the features used by our detector to be computed very quickly. The second is a learning algorithm, based on AdaBoost, which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers. The third contribution is a method for combining increasingly more complex classifiers in a "cascade" which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions. The cascade can be viewed as an object specific focus-of-attention mechanism which unlike previous approaches provides statistical guarantees that discarded regions are unlikely to contain the object of interest. In the domain of face detection the system yields detection rates comparable to the best previous systems.

3.2 PROPOSED SYSTEM

Deep CNNs have been extensively used for object detection.CNN is a type of feed-forward neural network and works on principle of weight sharing. Convolution is an integration showing how one function overlaps with other function and is a blend of two functions being multiplied. Image is convolved with activation function to get feature maps. To reduce spatial complexity of the network, feature maps are treated with pooling layers to get abstracted feature maps. This process is repeated for the desired number of filters and accordingly feature maps are created. Eventually, these feature maps are processed with fully connected layers to get output of image recognition showing confidence score for the predicted class labels.

3.3 HARWARE REQUIREMENTS

Processor	: Intel Pentium Dual Core 2.00GHz
Hard disk	: 500 GB
RAM	: 8 GB (minimum)

3.4 SOFTWARE REQUIREMENTS

- Python 3.6.4Version

3.5 SOFTWARE SPECIFICATION

3.5.1 Machine learning

Machine learning (**ML**) is the study of computer algorithms that improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

The types of machine learning algorithms are mainly divided into four categories:

- **Supervised learning,**
- **Un-supervised learning,**

- **Semi-supervised learning,**
- **Reinforcement learning**.

❖ **Supervised learning**

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

CLASSIFICATION

As the name suggests, Classification is the task of "classifying things" into sub- categories. But, by a machine. If that doesn't sound like much, imagine your computer being able to differentiate between you and a stranger. Between a potato and a tomato. Between an A grade and a F. In Machine Learning and Statistics, Classification is the problem of identifying to which of a set of categories (sub populations), a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.

TYPES OF CLASSIFICATION

Classification is of two types:

- Binary Classification
- Multiclass Classification

Binary Classification

When we have to categorize given data into 2 distinct classes. Example – On the basis of given health conditions of a person, we have to determine whether the person has a certain disease or not.

Multiclass Classification

The number of classes is more than 2. For Example

– On the basis of data about different species of flowers, we have to determine which specie does our observation belong to

Fig 2 : Binary and Multiclass Classification. Here x1 and x2 are our variables upon which the class is predicted. Suppose we have to predict whether a given patient has a certain disease or not, on the basis of 3 variables, called features. Which means there are two possible outcomes:

1. The patient has the said disease. Basically a result labeled “Yes” or “True”.
2. The patient is disease free. A result labeled “No” or “False”.

This is a binary classification problem. We have a set of observations called training data set, which comprises of sample data with actual classification results. We train a model, called Classifier on this data set, and use that model to predict whether a certain patient will have the

1. X : pre-classified data, in the form of a N*M matrix. N is the no. of observations and M is the number of features.

2. y : An N-d vector corresponding to predicted classes for each of the N observations.

3. Feature Extraction : Extracting valuable information from input X using a series of transforms.

4. ML Model : The “Classifier” we’ll train.

5. y’ : Labels predicted by the Classifier.

6. Quality Metric : Metric used for measuring the performance of the model.

7. ML Algorithm : The algorithm that is used to update weights w’, which update the model and “learns” iteratively.

Types of Classifiers (Algorithms)

There are various types of classifiers. Some of them are :

- Linear Classifiers : Logistic Regression
- Tree Based Classifiers : Decision Tree Classifier
- Support Vector Machines
- Artificial Neural Networks
- Bayesian Regression
- Gaussian Naive Bayes Classifiers
- Stochastic Gradient Descent (SGD)Classifier
- Ensemble Methods : Random Forests, AdaBoost, Bagging Classifier, Voting Classifier, Extra Trees Classifier

Practical Applications of Classification

- Google’s self driving car uses deep learning enabled classification techniques which enables it to detect and classify obstacles.
- Spam E-mail filtering is one of the most widespread and well recognized uses of Classification techniques.
- Detecting Health Problems, Facial Recognition, Speech Recognition, Object Detection, Sentiment Analysis all use Classification at their core.

REGRESSION

A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points.

❖ **Un-supervised learning**

Un-supervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

CLUSTERING

It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them. For example, the data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, These data points are clustered by using the basic concept that the data point lies within the given constraint from the cluster center. Various distance methods and techniques are used for calculation of the outliers. Clustering is very much important as it determines the intrinsic grouping among the unlabeled data present. There are no criteria for a good clustering. It depends on the user, what is the criteria they may use which satisfy their need. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding "natural clusters" and describe their unknown properties ("natural" data types), in finding useful and suitable groupings ("useful" data classes) or in finding unusual data objects (outlier detection). This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

3.5.1.1 Clustering Methods :

1. **Density-Based Methods :** These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters. Example DBSCAN

(Density-Based Spatial Clustering of Applications with Noise) , OPTICS (Ordering Points to Identify Clustering Structure) etc.

2. **Hierarchical Based Methods :** The clusters formed in this method forms a tree type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two category

- Agglomerative (bottom up approach)
- Divisive (top down approach).

3. **Partitioning Methods :** These methods partition the objects into k clusters and each partition for more cluster. This method is used to

optimize an objective criterion similarity function such as when the distance is a major parameter example K-means, CLARANS (Clustering Large Applications based upon randomized Search) etc.

4. **Grid-based Methods :** In this method the data space are formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (ClusteringIn Quest)etc.

Clustering Algorithms:

- K-Means Clustering.
- Mean-Shift Clustering for a single sliding window.
- The entire process of Mean-Shift Clustering.
- DBSCAN Smiley Face Clustering.
- EM Clustering using GMMs.
- Agglomerative Hierarchical Clustering.

❖ Semi-supervised learning

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

3.6 ANACONDA

Anaconda is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. Anaconda Distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and MacOS.

Python is a high-level programming language devised by Guido van Rossum & first released in 1991. It's the most popular coding language used by software developers to build, control, manage and for testing. It is also an interpreter which executes Python programs. The python interpreter is called python.exe on Windows.

Python Packages

Packages or additional libraries help in scientific computing and computational modeling. In Python, the packages are not the part of the Python standard library. Few major packages are –

numpy (NUMeric Python): matrices and linear algebra

scipy (SCIentific Python): many numerical routines matplotlib: (PLOTting LIBrary) creating plots of data

sympy (SYMbolic Python): symbolic computation

pytest (Python TESTing): a code testing framework

Together with a list of Python packages, tools like editors, Python distributions include the Python interpreter. Anaconda is one of several Python distributions. Anaconda is a new distribution of the Python and R data science package. It was formerly known as Continuum Analytics. Anaconda has more than 100 new packages.

This work environment, Anaconda is used for scientific computing, data science, statistical analysis, and machine learning. The latest version of Anaconda 5.0.1 is released in October 2017.The released version 5.0.1 addresses some minor bugs and adds useful features, such as updated R language support. All of these features weren't available in the original 5.0.0 release.

This package manager is also an environment manager, a Python distribution, and a collection of open source packages and contains more than 1000 R and Python Data Science Packages.

IPYTHON NOTEBOOKS

IPython is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history. IPython provides the following features:

- Interactive shells (terminal and Qt-based).
- A browser-based notebook interface with support for code, text, mathematical expressions, inline plots and other media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.
- Tools for parallel computing.

IPython is based on an architecture that provides parallel and distributed computing. IPython enables parallel applications to be developed, executed, debugged and monitored interactively. Hence, the I (Interactive) in IPython.[3]This architecture abstracts out parallelism, which enables IPython to support many different styles of parallelism[4]including:

With the release of IPython 4.0, the parallel computing capabilities have been made optional and released under the ipyparallel python package. IPython frequently draw from SciPy stack[5] libraries like NumPy and SciPy, often installed alongside from one of many Scientific Python distributions. IPython provide integration some library of the SciPy stack like matplotlib, like inline graph when in used with the Jupyter notebook. Python libraries can implement IPython specific hooks to customize object Rich object display.SymPyfor example implement rendering of Mathematical Expression as rendered LaTeX when used within IPython context.

Other features:

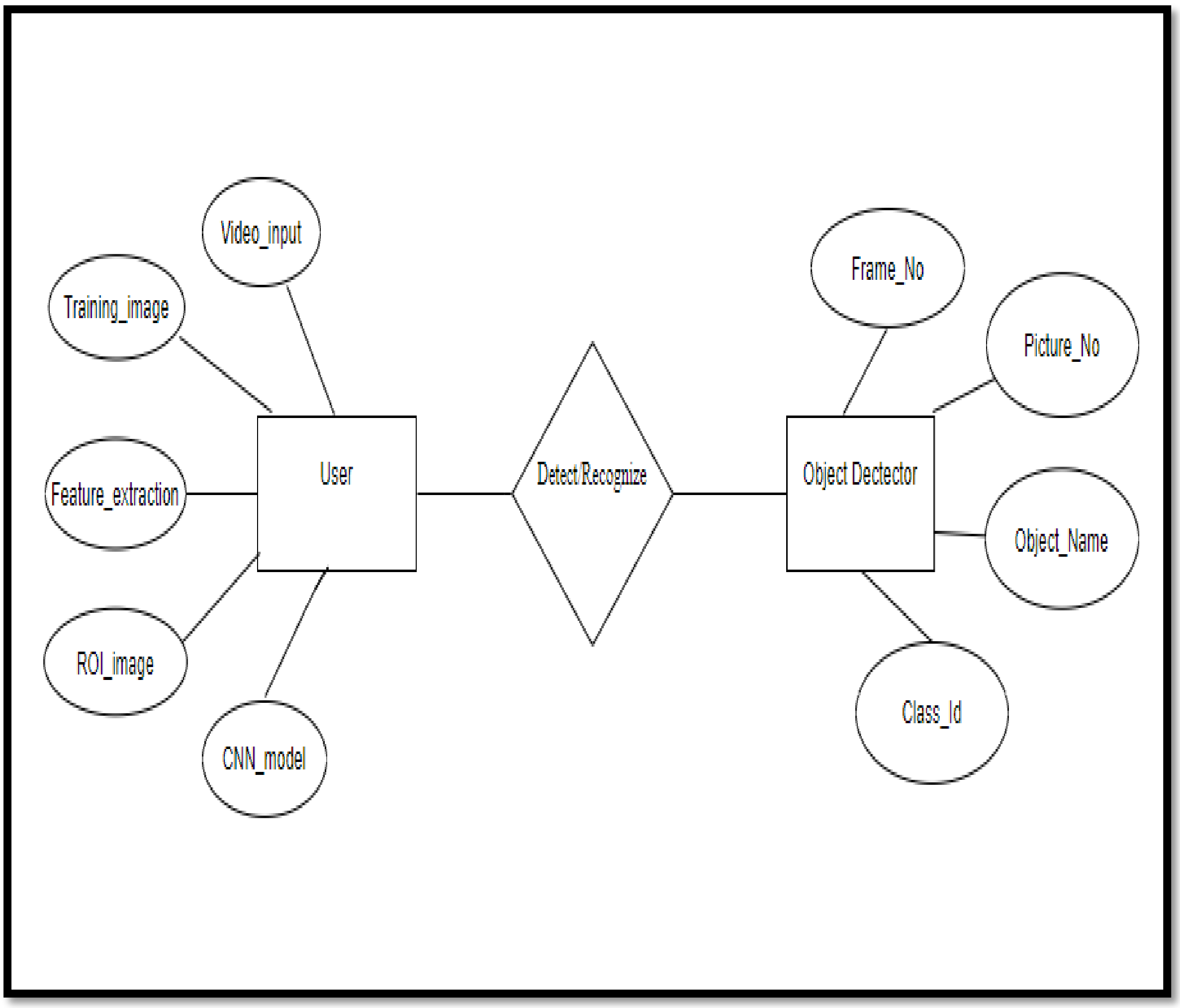
IPython also allows non-blocking interaction with Tkinter, PyGTK, PyQt/PySide and wxPython (the standard Python shell only allows interaction with Tkinter). IPython can interactively manage parallel computing clusters using asynchronous status call-backs and/or MPI. IPython can also be used as a system shell replacement. Its default behavioris largely similar to Unix shells, but it allows customization and the flexibility of executing code in a live Python environment. Using IPython as a shell replacement is less common and it is now recommended to use Xonsh which provide most of the IPython feature with better shell integrations.

CHAPTER 4

SYSTEM DESIGN

4.1 ER Diagram

Entity Relationship Diagram (ERDs) illustrate the logical structure of databases. An entity-relationship (ER) diagram is a specialized graphic that illustrates the interrelationships between entities in a database.

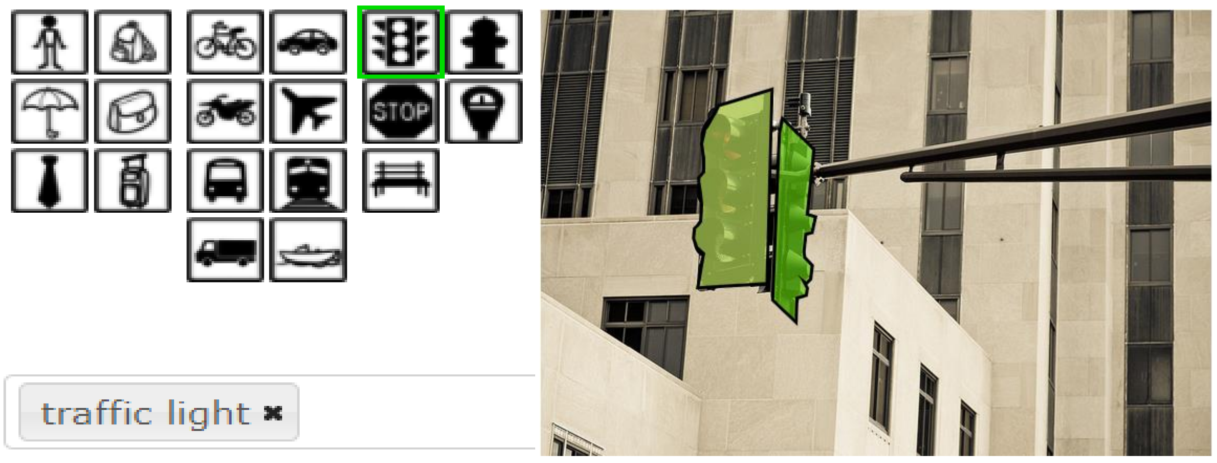


4.2 Data Dictionary

A data set is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where every column of a table represents a particular variable, and each row corresponds to a given record of the data set in question.

Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. Products: Competitions, Kaggle Kernels, Kaggle. Industry: Data science.

COCO is a large-scale object detection, segmentation, and captioning dataset. The COCO Dataset. Common Objects in Context (COCO) literally implies that the images in the dataset are everyday objects captured from everyday scenes. This adds some "context" to the objects captured in the scenes.



Common Objects in Context (COCO) is a database that aims to enable future research for object detection, instance segmentation, image captioning, and person key points localization.

4.3 UML DIAGRAMS

4.3.1 Use case Diagram

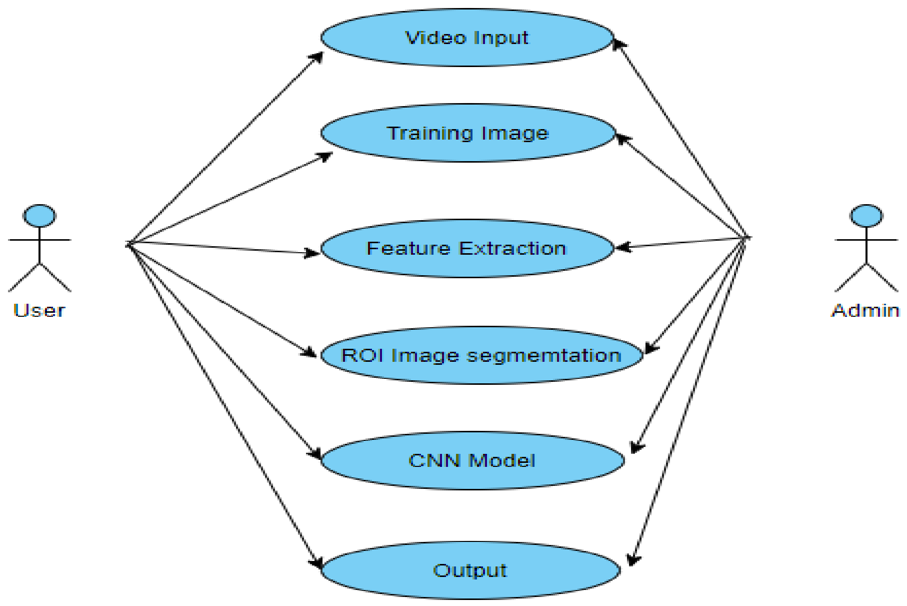
Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

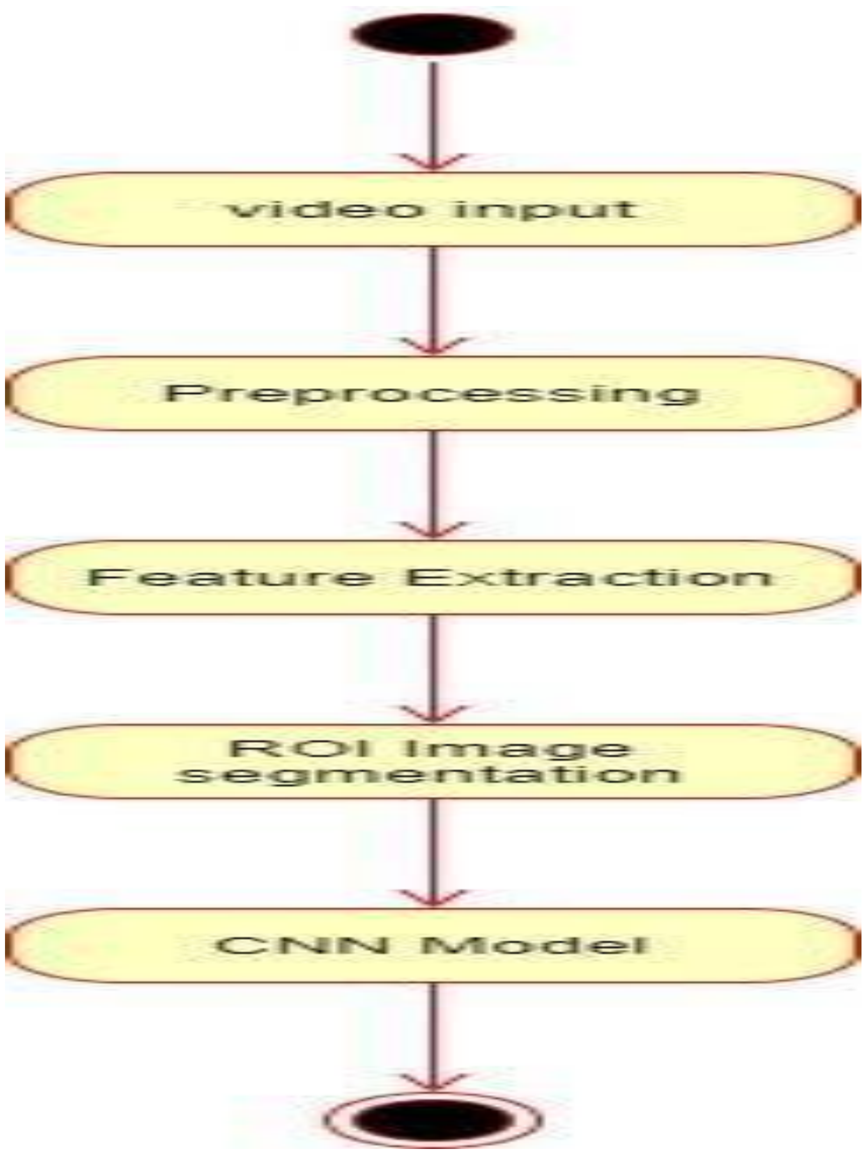


4.3.2 Activity Diagram

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

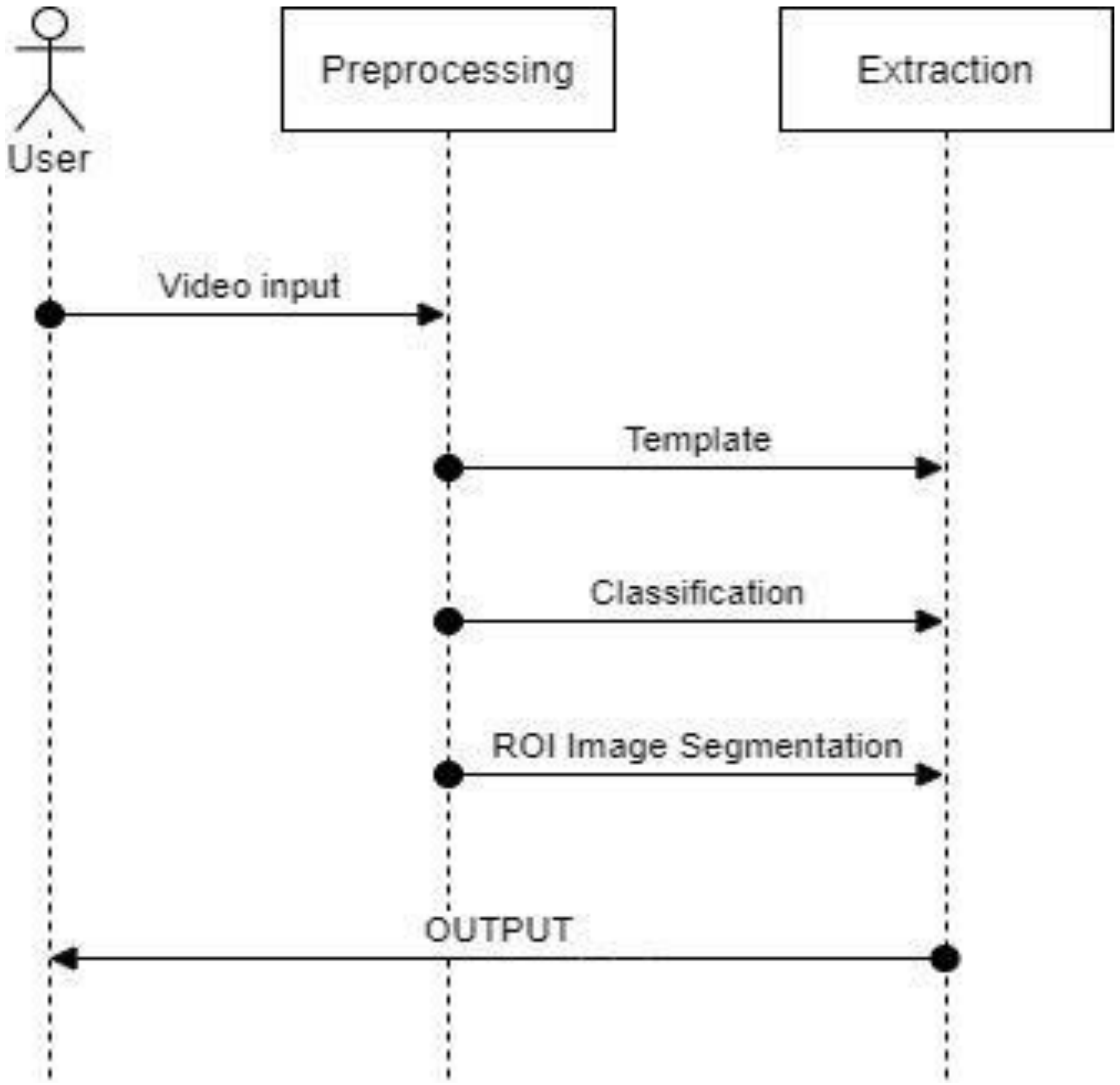
The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.



4.3.3 Sequence Diagram

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

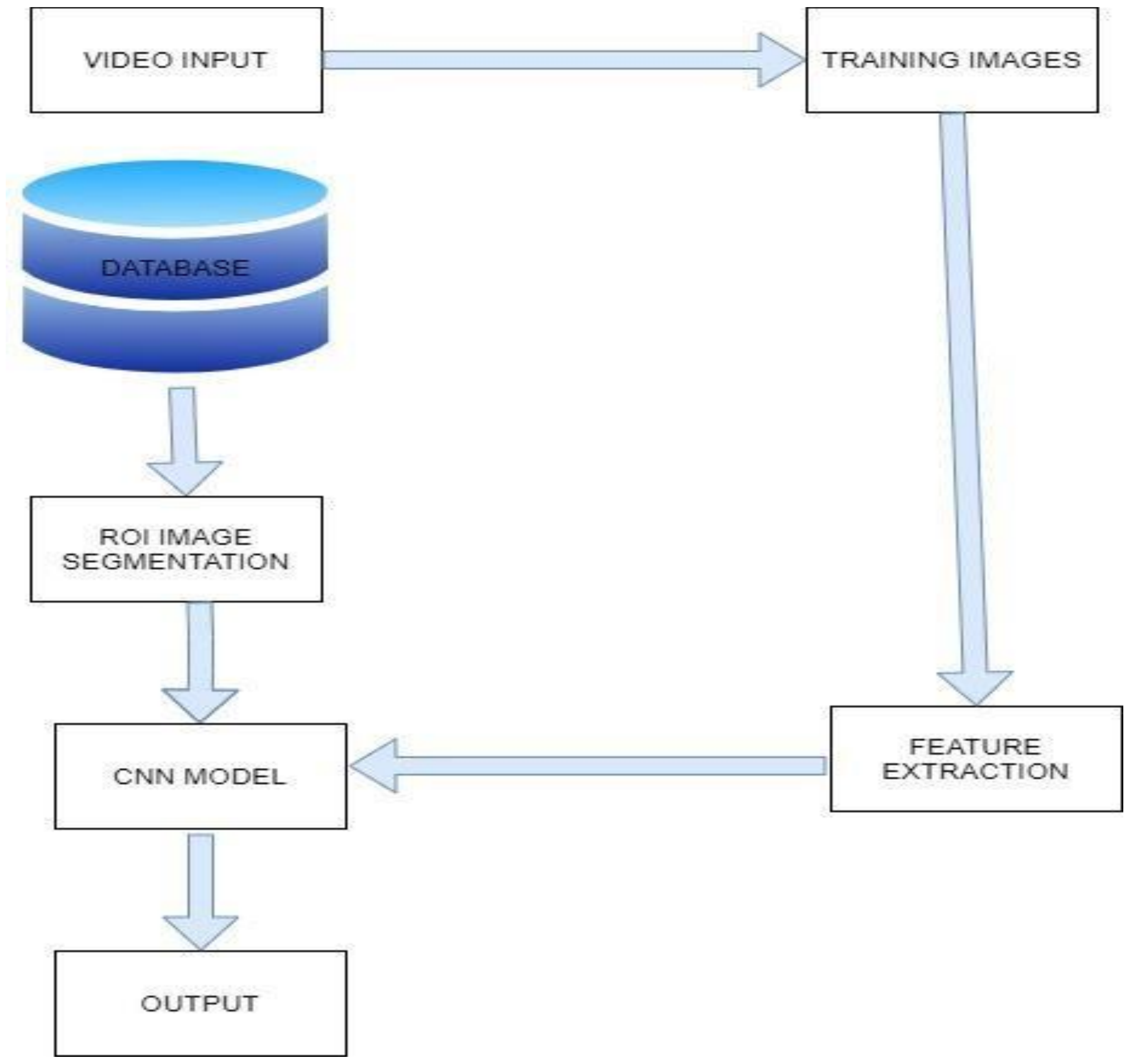


CHAPTER 5

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE OVERVIEW

System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.



5.2 MODULE DESIGN SPECIFICATION

- Object Data collection.
- Data Preprocessing.
- Object Detection.

5.2.1 MODULE DESCRIPTION

Object Data collection

Real time data collected from kaggle .Collection of data is one of the major and most important tasks of any machine learning projects. Because the input we feed to the algorithms is data. So, the algorithms efficiency and accuracy depends upon the correctness and quality of data collected. So as the data same will be the output.

Data Preprocessing

- Collecting the data is one task and making that data useful is an-other vital task.
- Data collected from various means will be in an unorganized format and there may be lot of null values, in-valid data values and unwanted data.
- Cleaning all these data and replacing them with appropriate or approximate data and removing null and missing data and replacing them with some fixed alternate values are the basic steps in pre processing of data.

- Even data collected may contain completely garbage values. It may not be in exact format or way that is meant to be. All such cases must be verified and replaced with alternate values to make data meaning meaningful and useful for further processing. Data must be kept in a organized format.

Object Detection

- Yolo is an algorithm that uses convolutional neural networks for object detection.
- In comparison to recognition algorithms, a detection algorithm does not only predict class labels, but detects locations of objects as well.
- The algorithm divides the image into grids and runs the image classification and localization algorithm (discussed under object localization) on each of the grid cells. For example, we have an input image of size 256 × 256. We place a 3 × 3 grid on the image.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 AUDIO VISUAL

```
import numpy as np
import cv2

confidenceThreshold = 0.5
NMSThreshold = 0.3

modelConfiguration = 'c.cfg'
modelWeights = 'obj.weights'

labelsPath = 'coco.names'
labels = open(labelsPath).read().strip().split('\n')

np.random.seed(10)
COLORS = np.random.randint(0, 255, size=(len(labels), 3), dtype="uint8")

net = cv2.dnn.readNetFromDarknet(modelConfiguration, modelWeights)

outputLayer = net.getLayerNames()
outputLayer = [outputLayer[i[0] - 1] for i in net.getUnconnectedOutLayers()]

video = cv2.VideoCapture('traffic.mp4')
writer = None
(W, H) = (None, None)

try:
    prop = cv2.CAP_PROP_FRAME_COUNT
    total = int(video.get(prop))
    print("[INFO] {} frame conversion".format(total))
except:
    printf("no frames in video")

count = 0
while True:
    (ret, frame) = video.read()
    if not ret:
        break
    if W is None or H is None:
        (H,W) = frame.shape[:2]
    blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416), swapRB = True,
crop = False)
    net.setInput(blob)
    layersOutputs = net.forward(outputLayer)

    boxes = []
    confidences = []
    classIDs = []

    for output in layersOutputs:
        for detection in output:
            scores = detection[5:]
            classID = np.argmax(scores)
            confidence = scores[classID]
            if confidence > confidenceThreshold:
                box = detection[0:4] * np.array([W, H, W, H])
                (centerX, centerY,  width, height) = box.astype('int')
                x = int(centerX - (width/2))
                y = int(centerY - (height/2))

                boxes.append([x, y, int(width), int(height)])
                confidences.append(float(confidence))
                classIDs.append(classID)

    #Apply Non Maxima Suppression
    detectionNMS = cv2.dnn.NMSBoxes(boxes, confidences,
confidenceThreshold, NMSThreshold)
    if(len(detectionNMS) > 0):
        for i in detectionNMS.flatten():
            (x, y) = (boxes[i][0], boxes[i][1])
            (w, h) = (boxes[i][2], boxes[i][3])

            color = [int(c) for c in COLORS[classIDs[i]]]
            cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)
            text = '{}: {:.4f}'.format(labels[classIDs[i]], confidences[i])
            cv2.putText(frame, text, (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX,
0.5, color, 2)

            if writer is None:
                fourcc = cv2.VideoWriter_fourcc(*'MJPG')
                writer = cv2.VideoWriter('output.avi', fourcc, 30, (frame.shape[1],
frame.shape[0]), True)
```

```
    if writer is not None:
        writer.write(frame)
        print("Writing frame" , count+1)
        count = count + 1

writer.release()
video.release()
```

6.2 WEBCAM

```
import numpy as np
import cv2

confidenceThreshold = 0.5
NMSThreshold = 0.3

modelConfiguration = 'c.cfg'
modelWeights = 'obj.weights'

labelsPath = 'coco.names'
labels = open(labelsPath).read().strip().split('\n')

np.random.seed(10)
COLORS = np.random.randint(0, 255, size=(len(labels), 3), dtype="uint8")

net = cv2.dnn.readNetFromDarknet(modelConfiguration, modelWeights)

outputLayer = net.getLayerNames()
outputLayer = [outputLayer[i[0] - 1] for i in net.getUnconnectedOutLayers()]

video_capture = cv2.VideoCapture(0)

(W, H) = (None, None)

while True:
    ret, frame = video_capture.read()
    frame = cv2.flip(frame, 1)
    if W is None or H is None:
        (H,W) = frame.shape[:2]

    blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416), swapRB = True,
crop = False)
    net.setInput(blob)
    layersOutputs = net.forward(outputLayer)
    boxes = []
    confidences = []
    classIDs = []

    for output in layersOutputs:
        for detection in output:
            scores = detection[5:]
            classID = np.argmax(scores)
            confidence = scores[classID]
            if confidence > confidenceThreshold:
                box = detection[0:4] * np.array([W, H, W, H])
                (centerX, centerY,  width, height) = box.astype('int')
                x = int(centerX - (width/2))
                y = int(centerY - (height/2))

                boxes.append([x, y, int(width), int(height)])
                confidences.append(float(confidence))
                classIDs.append(classID)

    #Apply Non Maxima Suppression
    detectionNMS = cv2.dnn.NMSBoxes(boxes, confidences,
confidenceThreshold, NMSThreshold)
    if(len(detectionNMS) > 0):
        for i in detectionNMS.flatten():
            (x, y) = (boxes[i][0], boxes[i][1])
            (w, h) = (boxes[i][2], boxes[i][3])

            color = [int(c) for c in COLORS[classIDs[i]]]
            cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)
            text = '{}: {:.4f}'.format(labels[classIDs[i]], confidences[i])
            cv2.putText(frame, text, (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX,
0.5, color, 2)

    cv2.imshow('Output', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

#Finally when video capture is over, release the video capture and
destroyAllWindows
video_capture.release()
cv2.destroyAllWindows()
```

CHAPTER 7

SYSTEM TESTING

7.1 TESTING TECHNIQUES

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet – undiscovered error. A successful test is one that uncovers an as-yet-undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- ➢ Static analysis is used to investigate the structural properties of the Source code.
- ➢ Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding an yet undiscovered error. A successful test is one that uncovers an yet undiscovered error.

7.2 TEST DATA AND OUTPUT

UNITS TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

FUNCTIONAL TESTING

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files. Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

PERFORMANCE TESTING

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

STRESS TESTING

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

STRUCTURED TESTING

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.

- Checking attributes for their correctness.
- Handling end of file condition, I/O errors, buffer problems and textual errors in output information

INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

7.3 TESTING STRATEGIES

WHITE BOX TESTING

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been design to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

- Flow graph notation
- Kilo metric complexity
- Deriving test cases
- Graph matrices Control

BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

- Graph based testing methods
- Equivalence partitioning
- Boundary value analysis
- Comparison testing

SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity 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 methods should be strategy should have the following characteristics:

- Testing begins at the module level and works “outward” toward the integration of the entire computer based System.
- Different testing techniques are appropriate at different points in time.
- The developer of the software and an independent test group conducts testing.
- Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

7.4 TEST CASES

TEST CASE 1

S.No	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT	TEST COMMENTS
1.	Object Detection	traffic.avi	Object-bus to be detected	Object bus is detection	Pass	Status will indicate as object-bus is detected.

TEST CASE 2

S.No	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT	TEST COMMENTS
2.	Object Detection	traffic.avi	Object-bus to be detected	Object-bus is not detected	Fail	Status will indicate as object not detected.

TEST CASE 3

S.No	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT	TEST COMMENTS
3.	Object Detection	webcam.py	Object shown to be detected	Object shown is detected	Pass	Status will indicate that the object is detected.

TEST CASE 4

S.No	ACTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT	TEST COMMENTS
4.	Object Detection	webcam.py	Object shown to be detected	Object shown is not detected	Fail	Status will indicate that the object is not detected.

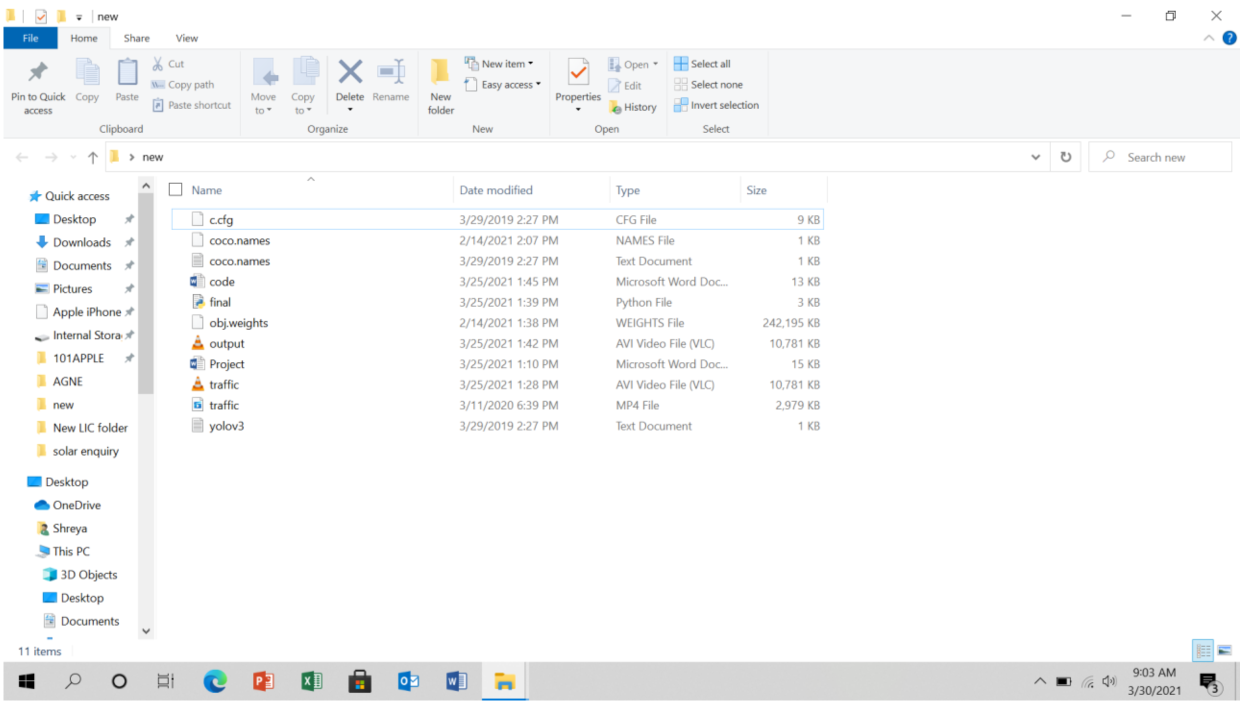
CHAPTER 8

CONCLUSION

Our framework detects all the objects based on training set provided to it. Main view of this project is to increase the accuracy rate of the detection i.e., it recognize the object even in blur stage or under less brightness .This framework can be constructed using deep learning, which is a subset of machine learning in AI that has network capability of learning unsupervised from data that is unstructured or unable.

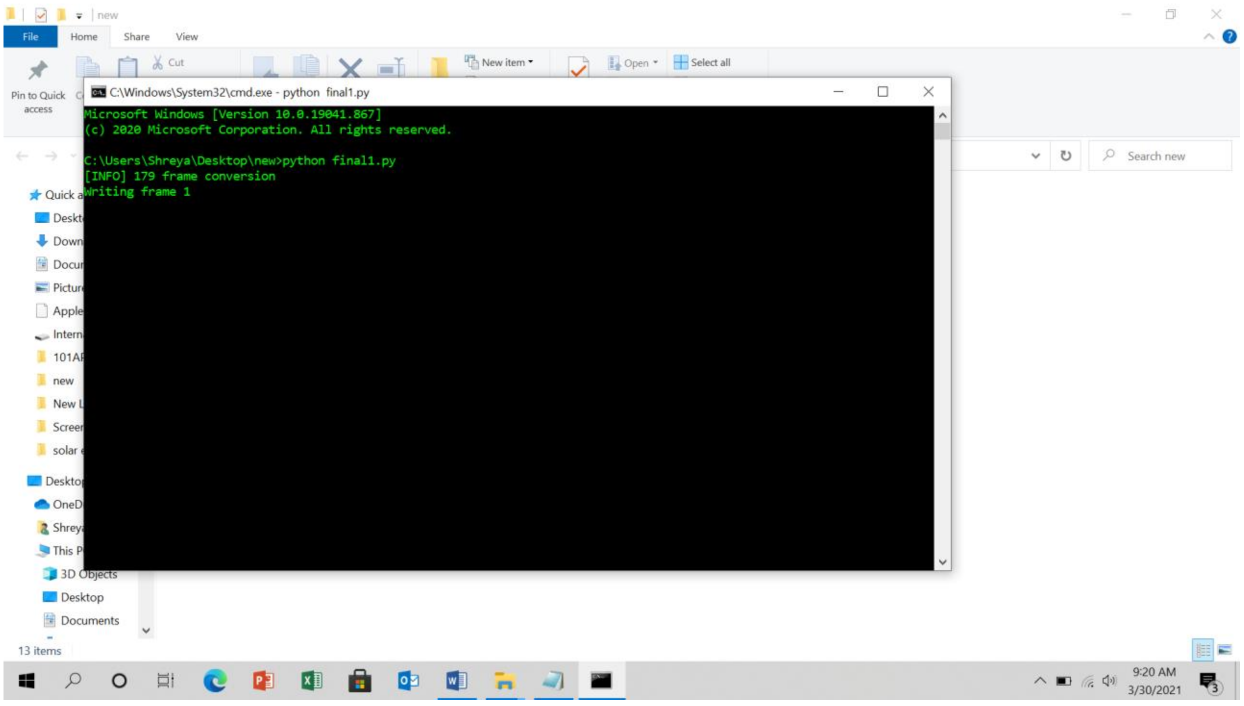
APPENDICES

A.1 SCREENSHOT



Name	Date modified	Type	Size
c.cfg	3/29/2019 2:27 PM	CFG File	9 KB
coco.names	2/14/2021 2:07 PM	NAMES File	1 KB
coco.names	3/29/2019 2:27 PM	Text Document	1 KB
code	3/25/2021 1:45 PM	Microsoft Word Doc...	13 KB
final	3/25/2021 1:39 PM	Python File	3 KB
obj.weights	2/14/2021 1:38 PM	WEIGHTS File	242,195 KB
output	3/25/2021 1:42 PM	AVI Video File (VLC)	10,781 KB
Project	3/25/2021 1:10 PM	Microsoft Word Doc...	15 KB
traffic	3/25/2021 1:28 PM	AVI Video File (VLC)	10,781 KB
traffic	3/11/2020 6:39 PM	MP4 File	2,979 KB
yolov3	3/29/2019 2:27 PM	Text Document	1 KB

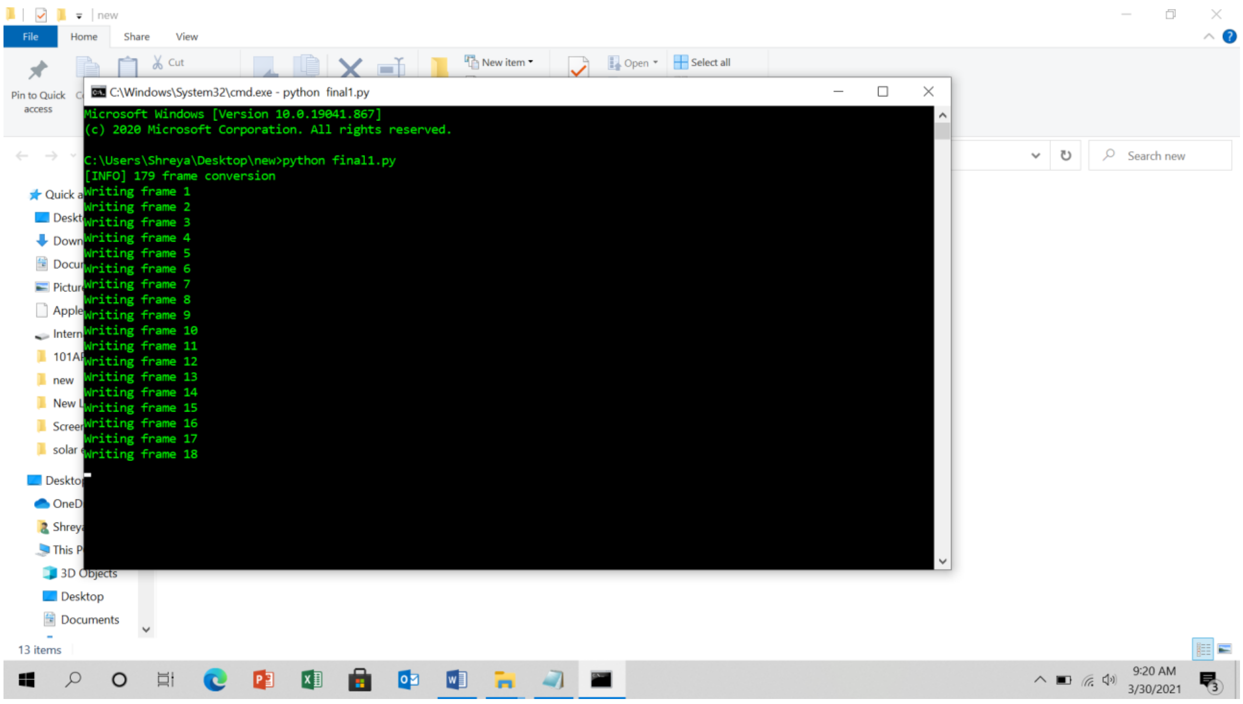
11 items



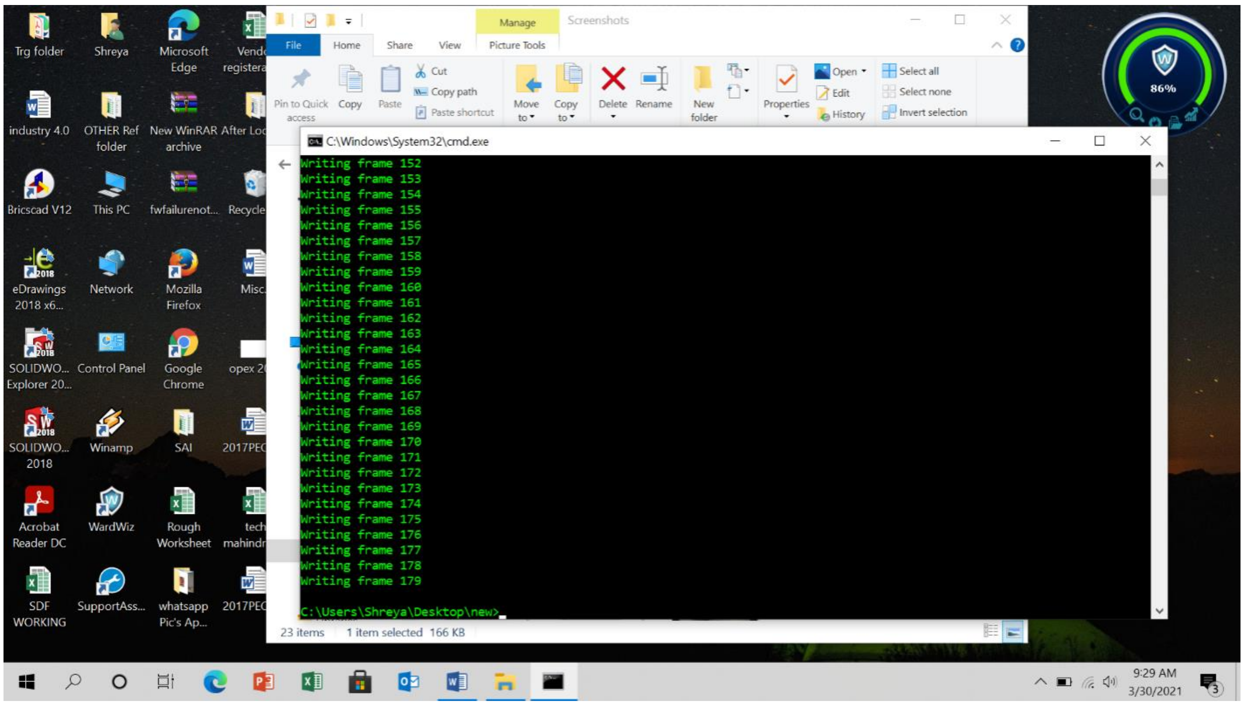
C:\Windows\System32\cmd.exe - python final1.py

```
Microsoft Windows [Version 10.0.19041.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Shreya\Desktop\new>python final1.py
[INFO] 179 frame conversion
Writing frame 1
```

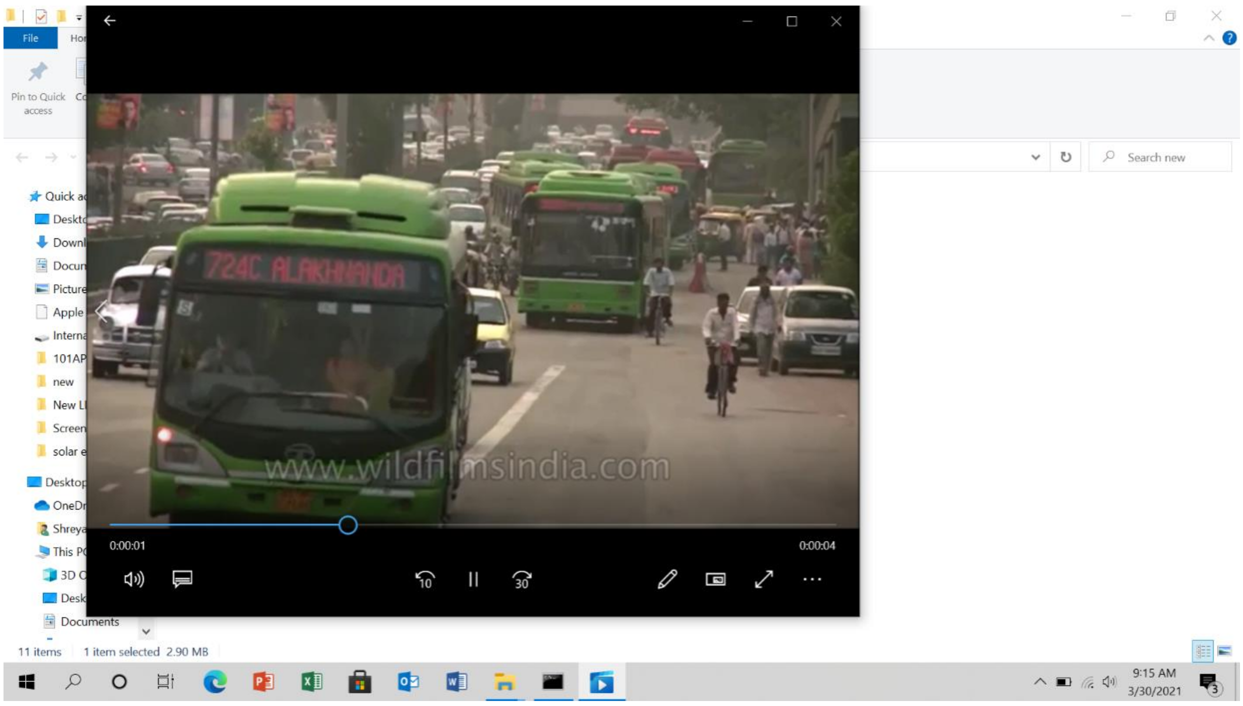
13 items



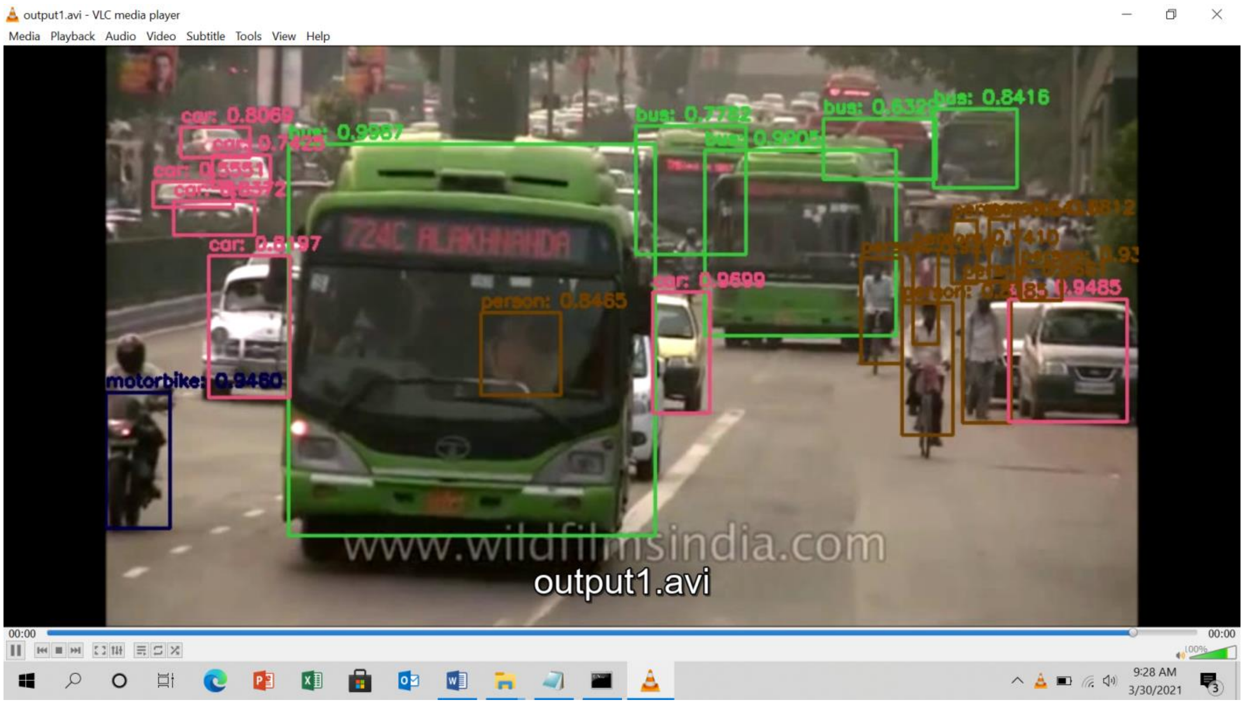
```
C:\Windows\System32\cmd.exe - python  final1.py
Microsoft Windows [Version 10.0.19041.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Shreya\Desktop\new>python final1.py
[INFO] 179 frame conversion
Writing frame 1
Writing frame 2
Writing frame 3
Writing frame 4
Writing frame 5
Writing frame 6
Writing frame 7
Writing frame 8
Writing frame 9
Writing frame 10
Writing frame 11
Writing frame 12
Writing frame 13
Writing frame 14
Writing frame 15
Writing frame 16
Writing frame 17
Writing frame 18
```



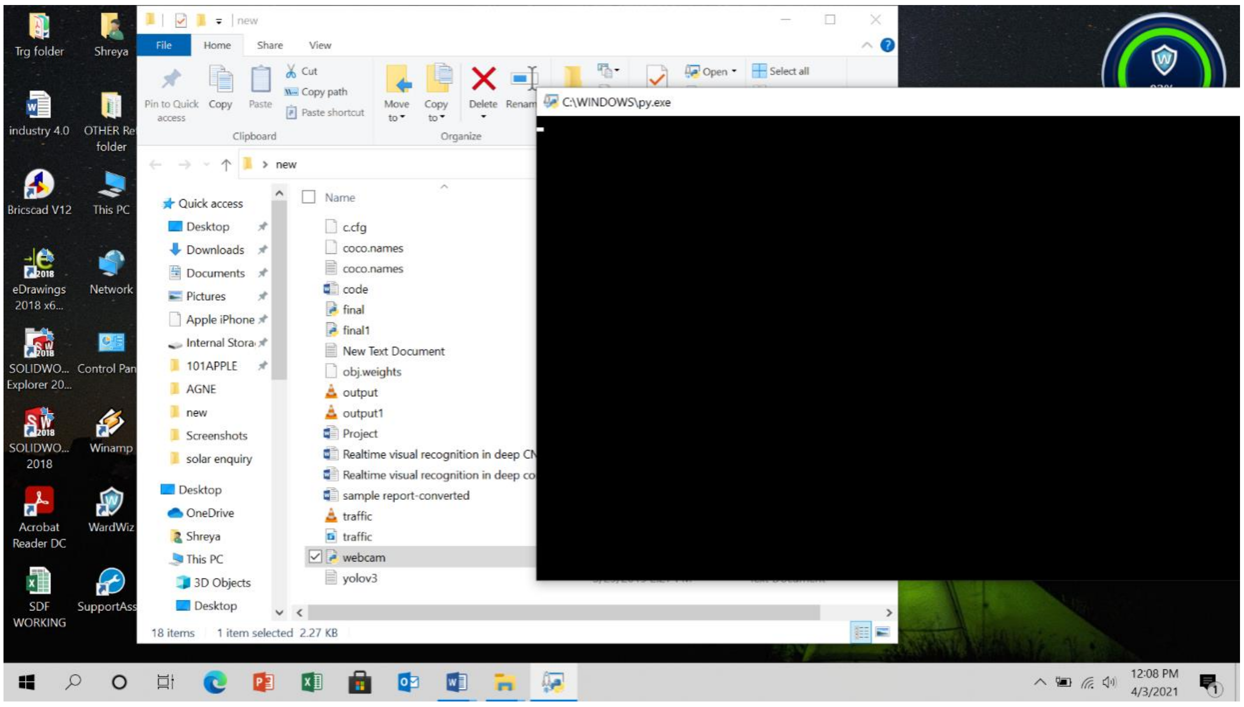
```
C:\Windows\System32\cmd.exe
Writing frame 152
Writing frame 153
Writing frame 154
Writing frame 155
Writing frame 156
Writing frame 157
Writing frame 158
Writing frame 159
Writing frame 160
Writing frame 161
Writing frame 162
Writing frame 163
Writing frame 164
Writing frame 165
Writing frame 166
Writing frame 167
Writing frame 168
Writing frame 169
Writing frame 170
Writing frame 171
Writing frame 172
Writing frame 173
Writing frame 174
Writing frame 175
Writing frame 176
Writing frame 177
Writing frame 178
Writing frame 179

C:\Users\Shreya\Desktop\new>
```

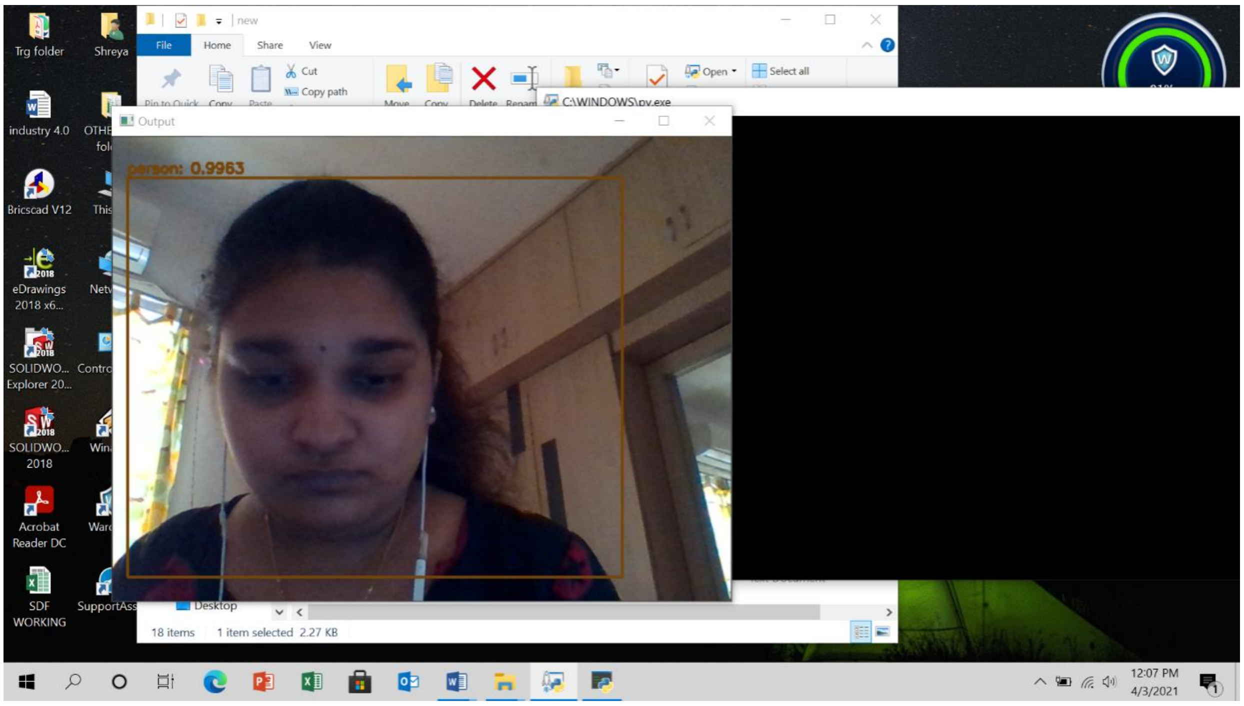


0:00:01
0:00:04
www.wildfilmsindia.com
11 items 1 item selected 2.90 MB
9:15 AM
3/30/2021

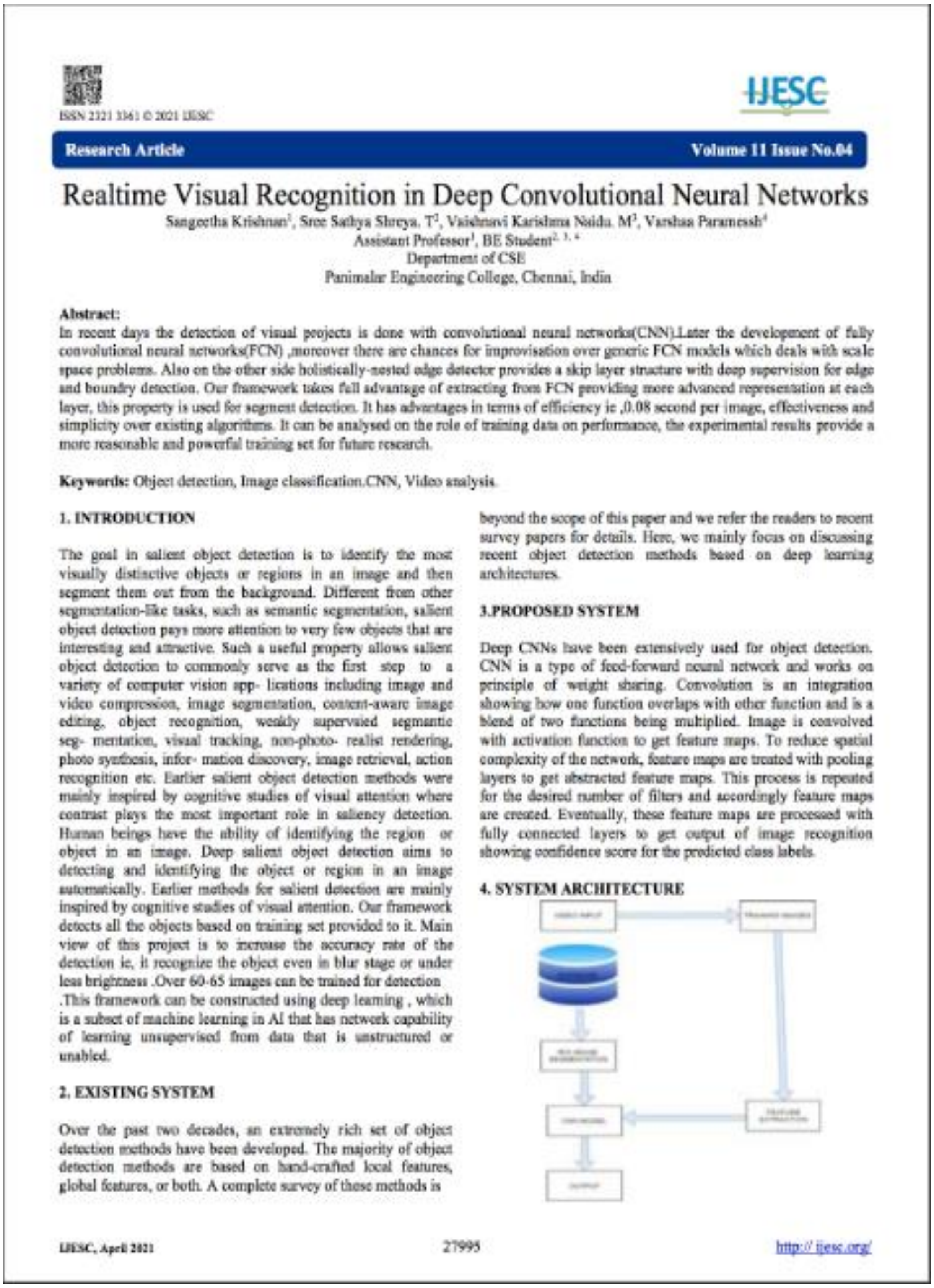


output1.avi - VLC media player
Media Playback Audio Video Subtitle Tools View Help
car: 0.8069
bus: 0.9987
bus: 0.7782
bus: 0.8416
bus: 0.9905
car: 0.9699
person: 0.8465
motorbike: 0.9460
output1.avi
9:28 AM
3/30/2021


new
C:\WINDOWS\py.exe
c.cfg
coco.names
coco.names
code
final
final1
New Text Document
obj.weights
output
output1
Project
sample report-converted
traffic
traffic
webcam
yolov3
18 items 1 item selected 2.27 KB
12:08 PM
4/3/2021


Output
C:\WINDOWS\py.exe
person: 0.9963
12:07 PM
4/3/2021

A.2 PUBLICATIONS



ISSN 2321 3361 © 2021 IJESC

IJESC

Research Article Volume 11 Issue No.04

Realtime Visual Recognition in Deep Convolutional Neural Networks

Sangeetha Krishnan[1], Sree Sathya Shreya. T[2], Vaishnavi Karishma Naidu. M[3], Varshaa Paramessh[4]
Assistant Professor[1], BE Student[2, 3, 4]
Department of CSE
Panimalar Engineering College, Chennai, India

Abstract:
In recent days the detection of visual projects is done with convolutional neural networks(CNN).Later the development of fully convolutional neural networks(FCN) ,moreover there are chances for improvisation over generic FCN models which deals with scale space problems. Also on the other side holistically-nested edge detector provides a skip layer structure with deep supervision for edge and boundry detection. Our framework takes full advantage of extracting from FCN providing more advanced representation at each layer, this property is used for segment detection. It has advantages in terms of efficiency ie ,0.08 second per image, effectiveness and simplicity over existing algorithms. It can be analysed on the role of training data on performance, the experimental results provide a more reasonable and powerful training set for future research.

Keywords: Object detection, Image classification.CNN, Video analysis.

1. INTRODUCTION

The goal in salient object detection is to identify the most visually distinctive objects or regions in an image and then segment them out from the background. Different from other segmentation-like tasks, such as semantic segmentation, salient object detection pays more attention to very few objects that are interesting and attractive. Such a useful property allows salient object detection to commonly serve as the first step to a variety of computer vision app- lications including image and video compression, image segmentation, content-aware image editing, object recognition, weakly supervsied segmantic seg- mentation, visual tracking, non-photo- realist rendering, photo synthesis, infor- mation discovery, image retrieval, action recognition etc. Earlier salient object detection methods were mainly inspired by cognitive studies of visual attention where contrast plays the most important role in saliency detection. Human beings have the ability of identifying the region or object in an image. Deep salient object detection aims to detecting and identifying the object or region in an image automatically. Earlier methods for salient detection are mainly inspired by cognitive studies of visual attention. Our framework detects all the objects based on training set provided to it. Main view of this project is to increase the accuracy rate of the detection ie, it recognize the object even in blur stage or under less brightness .Over 60-65 images can be trained for detection .This framework can be constructed using deep learning , which is a subset of machine learning in AI that has network capability of learning unsupervised from data that is unstructured or unabled.

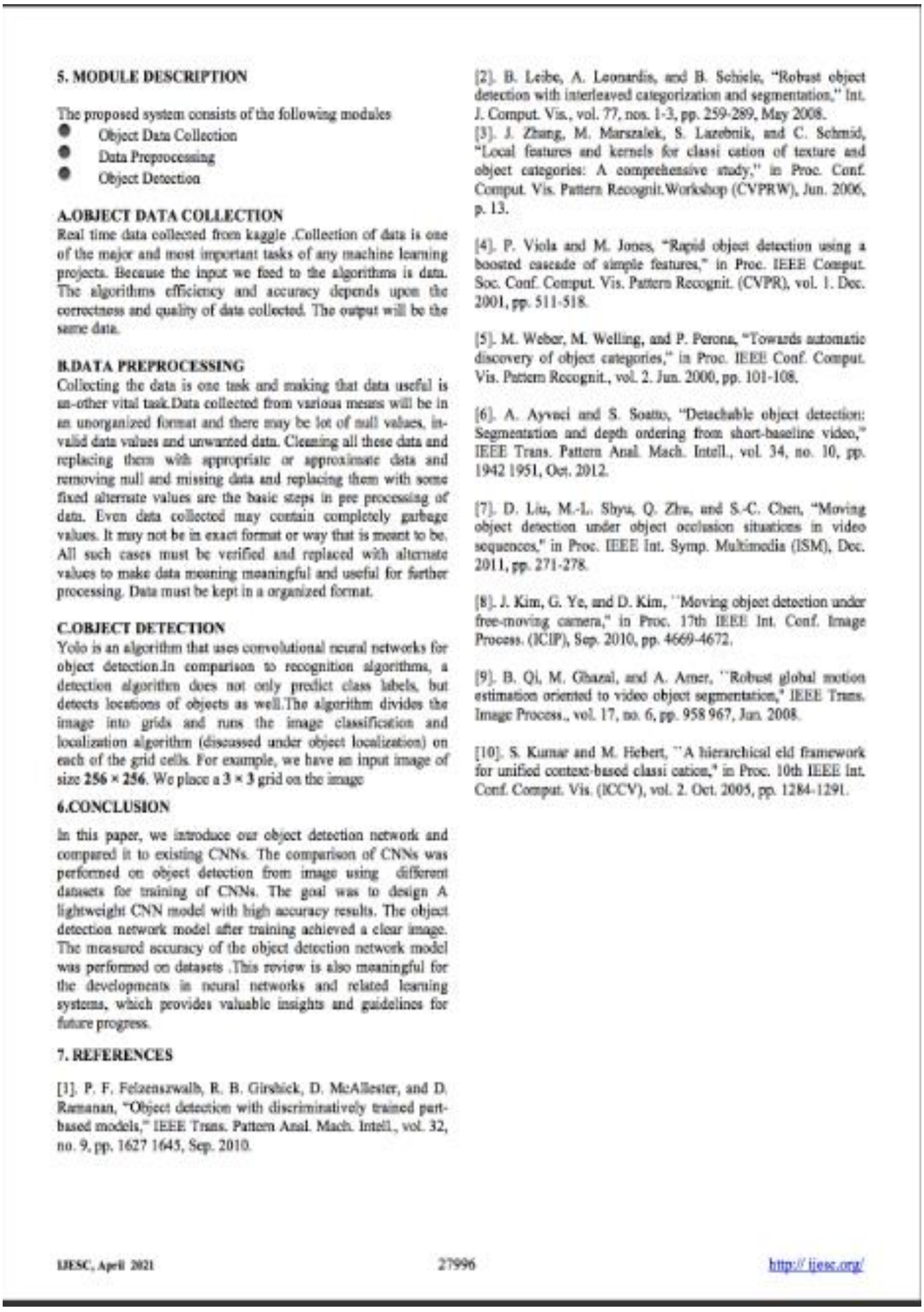
2. EXISTING SYSTEM

Over the past two decades, an extremely rich set of object detection methods have been developed. The majority of object detection methods are based on hand-crafted local features, global features, or both. A complete survey of these methods is beyond the scope of this paper and we refer the readers to recent survey papers for details. Here, we mainly focus on discussing recent object detection methods based on deep learning architectures.

3.PROPOSED SYSTEM

Deep CNNs have been extensively used for object detection. CNN is a type of feed-forward neural network and works on principle of weight sharing. Convolution is an integration showing how one function overlaps with other function and is a blend of two functions being multiplied. Image is convolved with activation function to get feature maps. To reduce spatial complexity of the network, feature maps are treated with pooling layers to get abstracted feature maps. This process is repeated for the desired number of filters and accordingly feature maps are created. Eventually, these feature maps are processed with fully connected layers to get output of image recognition showing confidence score for the predicted class labels.

4. SYSTEM ARCHITECTURE



5. MODULE DESCRIPTION

The proposed system consists of the following modules

- Object Data Collection
- Data Preprocessing
- Object Detection

A.OBJECT DATA COLLECTION

Real time data collected from kaggle .Collection of data is one of the major and most important tasks of any machine learning projects. Because the input we feed to the algorithms is data. The algorithms efficiency and accuracy depends upon the correctness and quality of data collected. The output will be the same data.

B.DATA PREPROCESSING

Collecting the data is one task and making that data useful is an-other vital task.Data collected from various means will be in an unorganized format and there may be lot of null values, in-valid data values and unwanted data. Cleaning all these data and replacing them with appropriate or approximate data and removing null and missing data and replacing them with some fixed alternate values are the basic steps in pre processing of data. Even data collected may contain completely garbage values. It may not be in exact format or way that is meant to be. All such cases must be verified and replaced with alternate values to make data meaning meaningful and useful for further processing. Data must be kept in a organized format.

C.OBJECT DETECTION

Yolo is an algorithm that uses convolutional neural networks for object detection.In comparison to recognition algorithms, a detection algorithm does not only predict class labels, but detects locations of objects as well.The algorithm divides the image into grids and runs the image classification and localization algorithm (discussed under object localization) on each of the grid cells. For example, we have an input image of size **256 × 256**. We place a **3 × 3** grid on the image

6.CONCLUSION

In this paper, we introduce our object detection network and compared it to existing CNNs. The comparison of CNNs was performed on object detection from image using different datasets for training of CNNs. The goal was to design A lightweight CNN model with high accuracy results. The object detection network model after training achieved a clear image. The measured accuracy of the object detection network model was performed on datasets .This review is also meaningful for the developments in neural networks and related learning systems, which provides valuable insights and guidelines for future progress.

7. REFERENCES

[1]. P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627 1645, Sep. 2010.

[2]. B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," Int. J. Comput. Vis., vol. 77, nos. 1-3, pp. 259-289, May 2008.

[3]. J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classi cation of texture and object categories: A comprehensive study," in Proc. Conf. Comput. Vis. Pattern Recognit.Workshop (CVPRW), Jun. 2006, p. 13.

[4]. P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), vol. 1. Dec. 2001, pp. 511-518.

[5]. M. Weber, M. Welling, and P. Perona, "Towards automatic discovery of object categories," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., vol. 2. Jun. 2000, pp. 101-108.

[6]. A. Ayvaci and S. Soatto, "Detachable object detection: Segmentation and depth ordering from short-baseline video," IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 10, pp. 1942 1951, Oct. 2012.

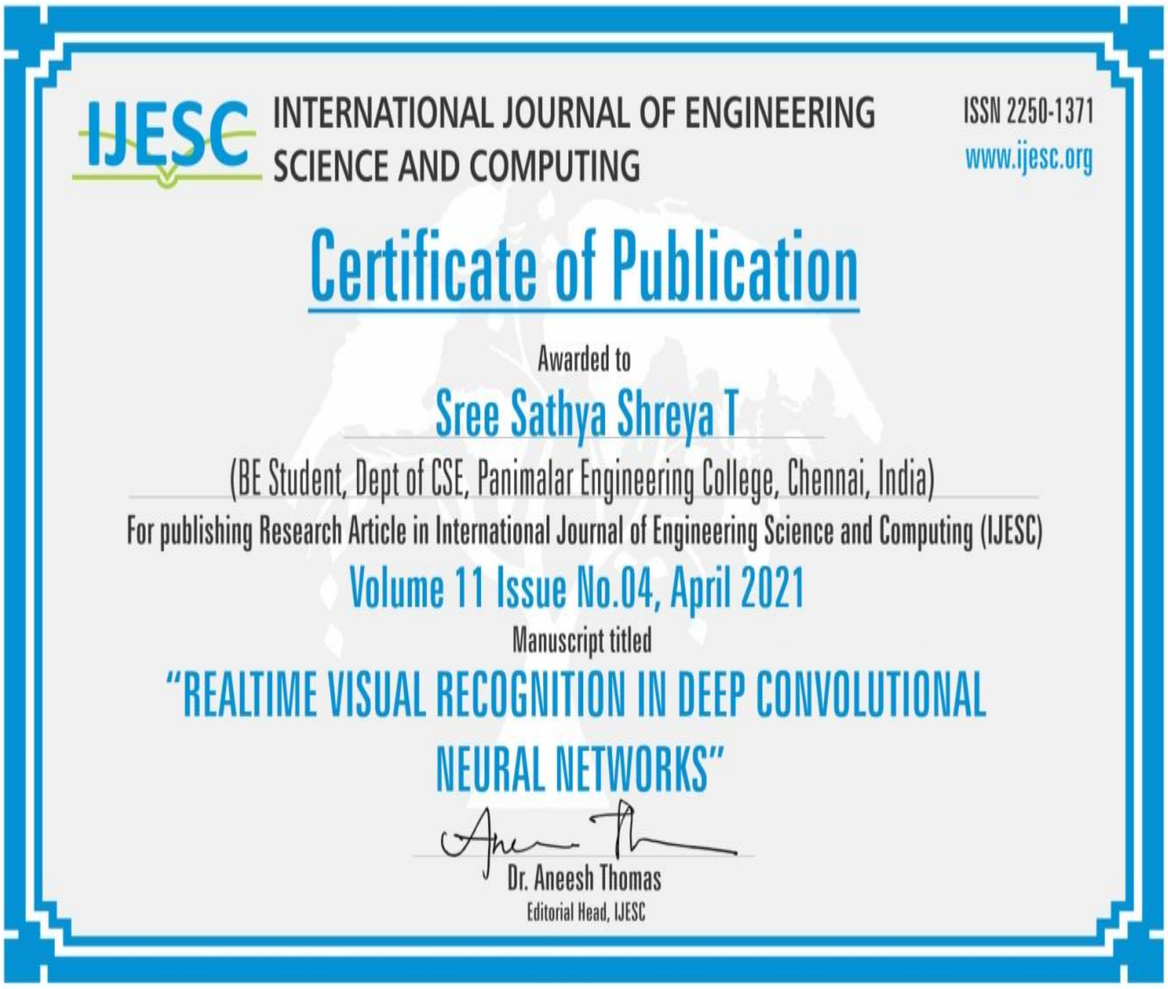
[7]. D. Liu, M.-L. Shyu, Q. Zhu, and S.-C. Chen, "Moving object detection under object occlusion situations in video sequences," in Proc. IEEE Int. Symp. Multimedia (ISM), Dec. 2011, pp. 271-278.

[8]. J. Kim, G. Ye, and D. Kim, ''Moving object detection under free-moving camera," in Proc. 17th IEEE Int. Conf. Image Process. (ICIP), Sep. 2010, pp. 4669-4672.

[9]. B. Qi, M. Ghazal, and A. Amer, ''Robust global motion estimation oriented to video object segmentation," IEEE Trans. Image Process., vol. 17, no. 6, pp. 958 967, Jun. 2008.

[10]. S. Kumar and M. Hebert, ''A hierarchical eld framework for unified context-based classi cation," in Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV), vol. 2. Oct. 2005, pp. 1284-1291.

A2.2 PUBLISHED CERTIFICATES



IJESC INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE AND COMPUTING

ISSN 2250-1371
www.ijesc.org

Certificate of Publication

Awarded to

Sree Sathya Shreya T

(BE Student, Dept of CSE, Panimalar Engineering College, Chennai, India)

For publishing Research Article in International Journal of Engineering Science and Computing (IJESC)

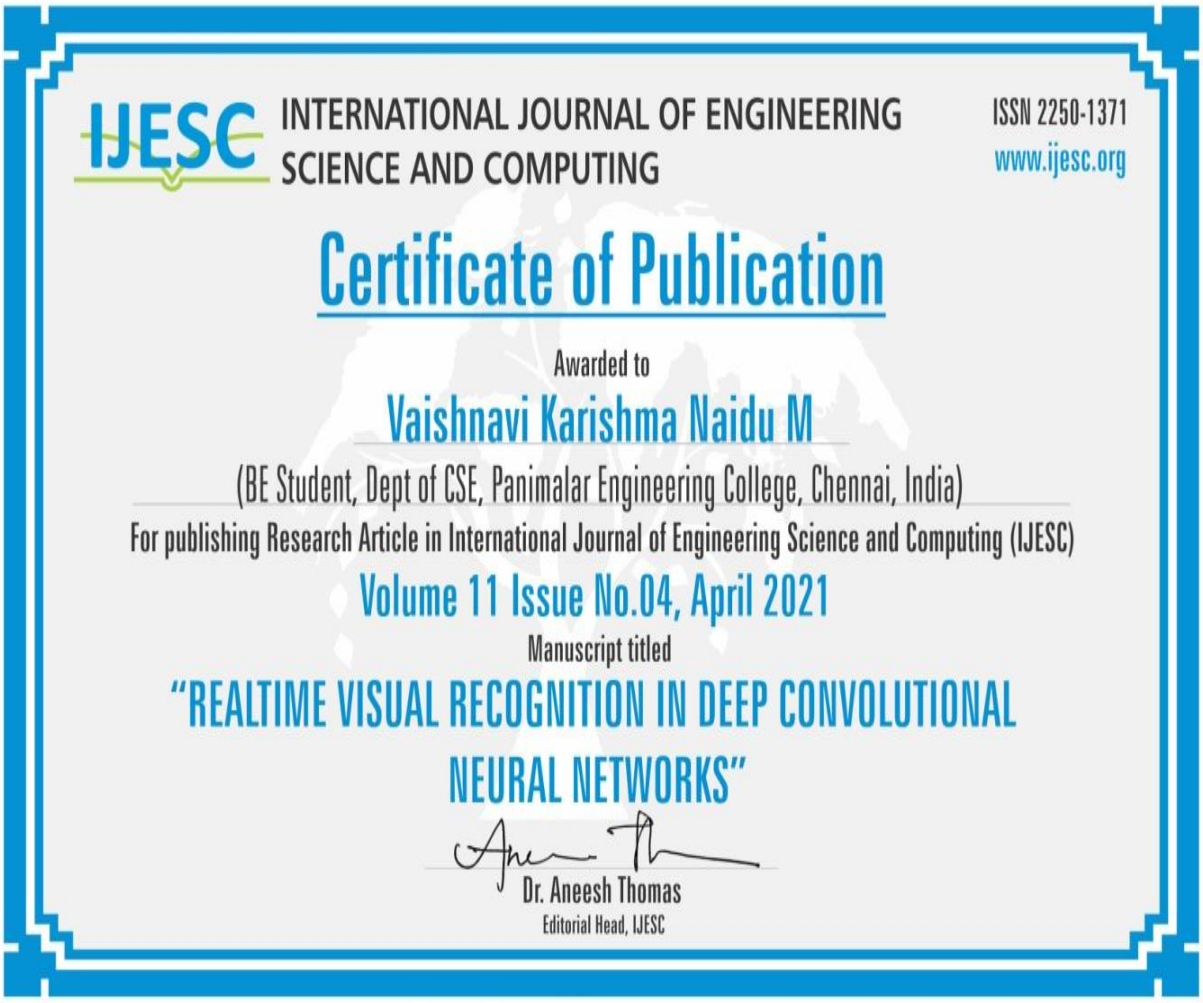
Volume 11 Issue No.04, April 2021

Manuscript titled

“REALTIME VISUAL RECOGNITION IN DEEP CONVOLUTIONAL NEURAL NETWORKS”

[illegible]

Dr. Aneesh Thomas
Editorial Head, IJESC



IJESC INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE AND COMPUTING

ISSN 2250-1371
www.ijesc.org

Certificate of Publication

Awarded to

Vaishnavi Karishma Naidu M

(BE Student, Dept of CSE, Panimalar Engineering College, Chennai, India)

For publishing Research Article in International Journal of Engineering Science and Computing (IJESC)

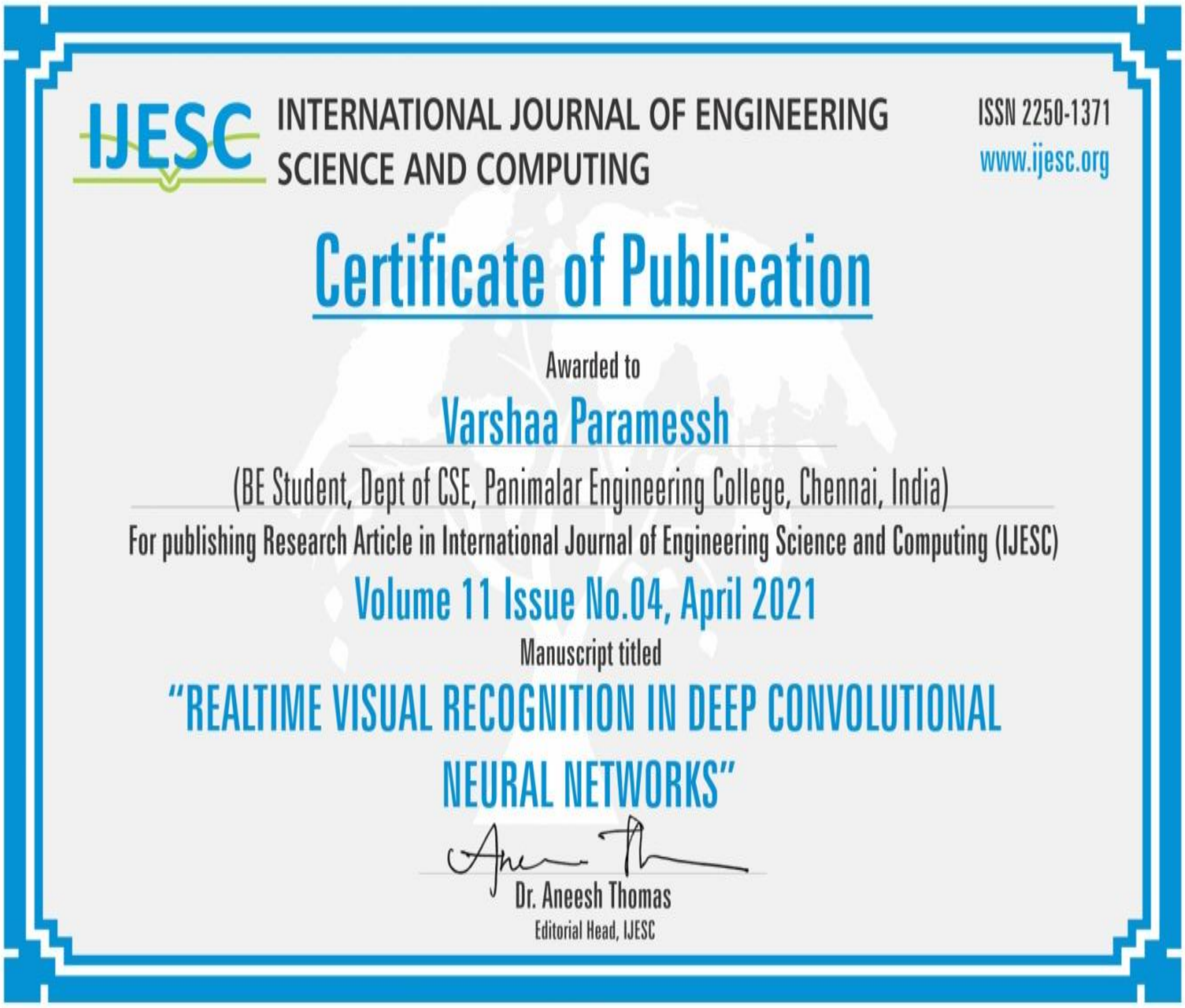
Volume 11 Issue No.04, April 2021

Manuscript titled

"REALTIME VISUAL RECOGNITION IN DEEP CONVOLUTIONAL NEURAL NETWORKS"

[illegible]

Dr. Aneesh Thomas
Editorial Head, IJESC



IJESC INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE AND COMPUTING

ISSN 2250-1371
www.ijesc.org

Certificate of Publication

Awarded to

Varshaa Paramessh

(BE Student, Dept of CSE, Panimalar Engineering College, Chennai, India)

For publishing Research Article in International Journal of Engineering Science and Computing (IJESC)

Volume 11 Issue No.04, April 2021

Manuscript titled

"REALTIME VISUAL RECOGNITION IN DEEP CONVOLUTIONAL NEURAL NETWORKS"

[illegible]

Dr. Aneesh Thomas
Editorial Head, IJESC

REFERENCES

1.P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627 1645, Sep. 2010.

2. B. Leibe, A. Leonardis, and B. Schiele, “Robust object detection with interleaved categorization and segmentation,” Int. J. Comput. Vis., vol. 77, nos. 1-3, pp. 259-289, May2008.

3. J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, “Local features and kernels for classi cation of texture and object categories: A comprehensive study,” in Proc. Conf. Comput. Vis. Pattern Recognit.Workshop (CVPRW), Jun. 2006, p.13.

4. P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), vol. 1. Dec. 2001, pp.511-518.

5. M. Weber, M. Welling, and P. Perona, “Towards automatic discovery of object categories,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., vol. 2. Jun. 2000, pp.101-108.

6. A. Ayvaci and S. Soatto, “Detachable object detection: Segmentation and depth ordering from short-baseline video,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 10, pp. 1942 1951, Oct.2012.

7. D. Liu, M.-L. Shyu, Q. Zhu, and S.-C. Chen, “Moving object detection under object occlusion situations in video sequences," in Proc. IEEE Int. Symp. Multimedia (ISM), Dec. 2011, pp.271-278.

8. J. Kim, G. Ye, and D. Kim, ``Moving object detection under free-moving camera," in Proc. 17th IEEE Int. Conf. Image Process. (ICIP), Sep. 2010, pp. 4669-4672.

9.B. Qi, M. Ghazal, and A. Amer, ``Robust global motion estimation oriented to video object segmentation," IEEE Trans. Image Process., vol. 17, no. 6, pp. 958 967, Jun.2008.

10. S. Kumar and M. Hebert, ``A hierarchical eld framework for unified context-based classification," in Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV), vol. 2. Oct. 2005, pp.1284-1291.