International Research Journal of Computer Science (IRJCS) ISSN: 2393-9842
Issue 06, Volume 6 (June 2019) www.irjcs.com
SPECIAL ISSUE - 5th International Conference - "ACCE – 2019"

OBJECT DETECTION USING DEEP NEURAL NETWORKS

Mahantesh N [1], Divya K G [2], Deepika B [3]
1,2, 3. UG student, Department of CSE, Vemana Institute of Technology, Bangalore.
1. gmahanteshn@gmail.com, 2. kgdivya11@gmail.com, 3. deepikadepu2012@gmail.com

Manuscript History
Number: IRJCS/RS/Vol.06/Issue06/JNCS10114
Received: 29, May 2019
Final Correction: 30, May 2019
Final Accepted: 02, June 2019
Published: June 2019
doi://10.26562/IRJCS.2019.JNCS10114
Editor: Dr.A.Arul L.S, Chief Editor, IRJCS, AM Publications, India

ABSTRACT - A new approach for object detection is presented. Prior work on object detection repurposes classifiers to perform detection. Instead, framing object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance. Proposed model processes images in real-time at 45 frames per second. It has been defined as a multi-scale inference procedure which is able to produce high-resolution object detections at a low cost by a few network applications.

KEYWORDS – Machine Learning, Real time Object Detection, Neural Network, Image Classifiers, Real-Time Video Analysis

I. INTRODUCTION

Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. The human visual system is fast and accurate, allowing us to perform complex tasks like driving with little conscious thought. General purpose object detection should be fast, accurate, and able to recognize a wide variety of objects. There's no shortage of interesting problems in computer vision, from simple image classification to 3Dpose estimation. Manually engineered representations in conjunction with shallow discriminatively trained models have been among the best performing paradigms for the related problem of object classification disregard to the computational intensity and human intervention for overseeing the process But, since the introduction of machine learning, detection frameworks have become increasingly fast and accurate. Algorithms for object detection would allow computers to drive cars in any weather without specialized sensors, enable assistive devices to convey real-time scene information to human users, and unlock the potential for general purpose, responsive robotic systems.

Approaches like R-CNN use region proposal methods to first generate potential bounding boxes in an image and then run a classifier on these proposed boxes. After classification, post-processing is used to refine the bounding box, eliminate duplicate detections, and rescore the box based on other objects. These complex pipelines are slow and hard to optimize because each individual component must be trained separately.

In the last years, however Deep Neural Networks (DNNs) have emerged as a powerful machine learning model. Deep Neural Networks (DNNs) exhibit major differences from traditional approaches for classification. First, they are deep architectures which have the capacity to learn more complex models than shallow ones. This expressivity and robust training algorithms allow for learning powerful object representations without the need to hand design features.

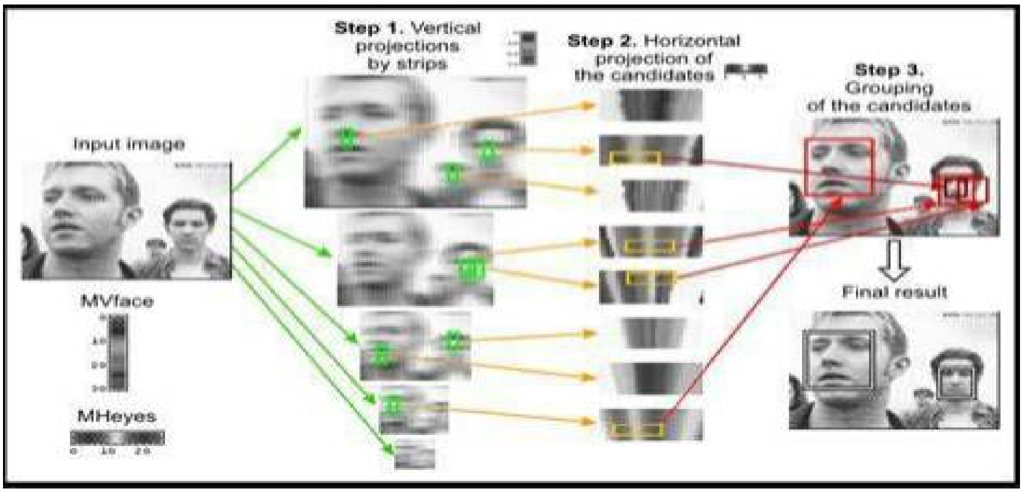


Fig 1: Viola-Jones Object Detection Framework

II. DESIGN

SYSTEM ARCHITECTURE:

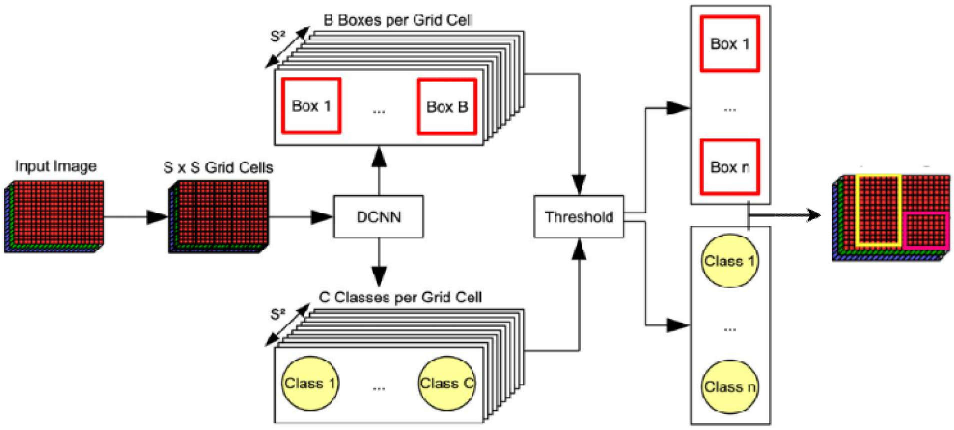


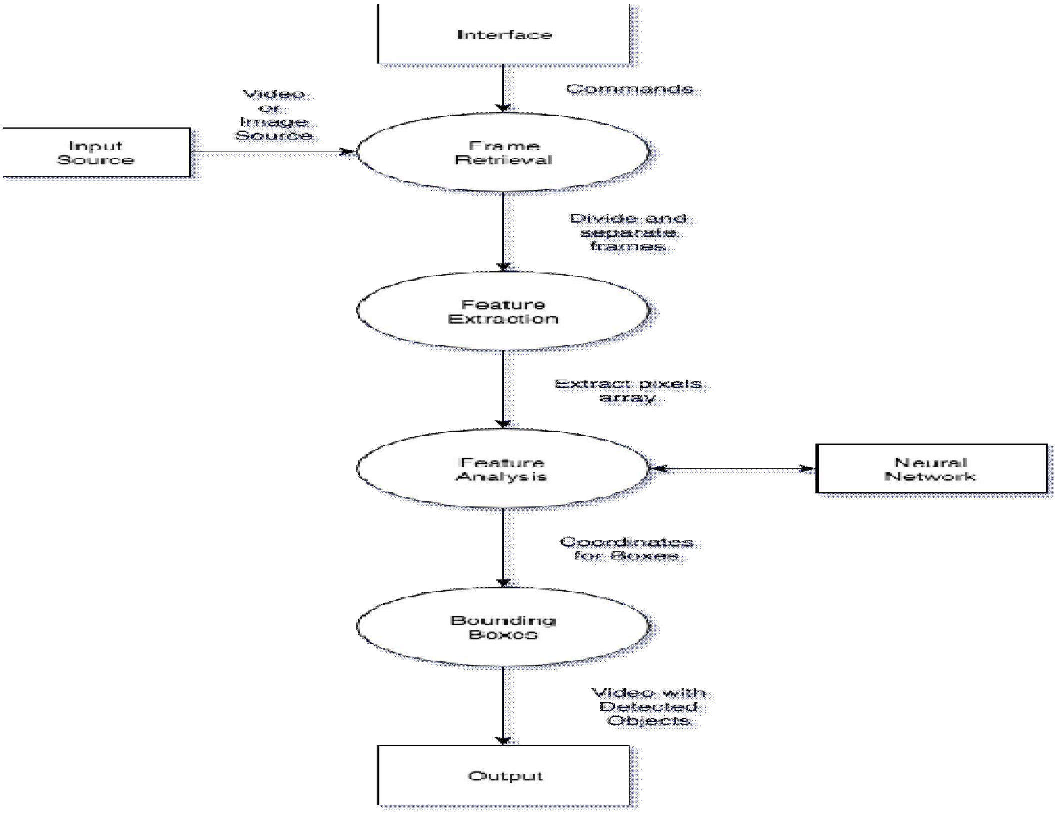
Fig 2: System Architecture

The above architecture describes the work structure of the system.

- The input image is first divided into SxS grid cells. Here S denotes the number of pixels we want the picture divided in.
- This grid celled picture is then passed on to the deep convolution neural network, where each object in the image is bounded by boxes with probabilities and classified into a class.
- The next step will eliminate all the bounding boxes with the probability less than that of the threshold.
- The final image with the class name and the probability of that bounding box is given as output.

DATAFLOW DIAGRAM:

- The graphical user interface is a type of user interface that allows users to interact with electronic devices using images rather than text commands.
- Frame retrieval the video or an input source is either given by a camera, a picture in a folder, or a screen recorded video. These videos are divided into frames per second, these frames are extracted.
- Using these framers we are computing the frame features like dimensions, pixel values etc
- The computed values are sent for further analysis where it sends it to a neural network to analyze.
- The co-ordinates for the bounding boxes are given by the neural network and they are put here.
- The object in the video is detected and the output is shown.



III. IMPLEMENTATION

The implementation phase involves the actual materialization of the ideas, which are expressed in the analysis document and developed in the design phase. Implementation should be perfect mapping of the design document in a suitable programming language in order to achieve the necessary final product. Often the product is ruined due to incorrect programming language chosen for implementation or unsuitable method of programming. It is better for the coding phase to be directly linked to the design phase in the sense if the design is in terms of object oriented terms then implementation should be preferably carried out in a object oriented way.

The implementation involves:

- Careful planning.
- Investigation of the current system and the constraints on implementation.
- Training of staff in the newly developed system.

Implementation of any software is always preceded by important decisions regarding selection of the platform, the language used, etc. these decisions are often influenced by several factors such as real environment in which the system works, the speed that is required, the security concerns, and other implementation specific details. There are three major implementation decisions that have been made before the implementation of this project.

They are as follows:

- Selection of the platform (Operating System).
- Selection of the programming language for development of the application.
- Coding guideline to be followed.

IV. PROGRAMMING LANGUAGE SELECTION

Python is an interpreter, object-oriented **programming language** similar to PERL that has gained popularity because of its clear syntax and readability. It is a scripting language like PHP, Perl, Ruby and so much more. It can be used for web programming (**django, Zope, Google App Engine, and much more).** But it also can be used for desktop applications (Blender 3D, or even for games pygame). Python can also be translated into binary code like java.

Python is a general purpose and high level programming language. You can use Python for developing desktop GUI applications, websites and web applications. Also, Python, as a high level programming language, allows you to focus on core functionality of the application by taking care of common programming tasks. The simple syntax rules of the programming language further makes it easier for you to keep the code base readable and application maintainable. While writing a software application, you must focus on the quality of its source code to simplify maintenance and updates. The syntax rules of Python allow you to express concepts without writing additional code. At the same time, Python, unlike other programming languages, emphasizes on code readability, and allows you to use English keywords instead of punctuations. Python also supports several programming paradigm. It supports object oriented and structured programming fully. Also, its language features support various concepts in functional and aspect-oriented programming. At the same time, Python also features a dynamic type system and automatic memory management. The programming paradigms and language features help you to use Python for developing large and complex software applications. Its large and robust standard library makes Python score over other programming languages. The standard library allows you to choose from a wide range of modules according to your precise needs. Each module further enables you to add functionality to the Python application without writing additional code. Python can be used to create prototype of the software application rapidly. Also, you can build the software application directly from the prototype simply by refactoring the Python code. Python even makes it easier for you to perform coding and testing simultaneously by adopting test driven development (TDD) approach. Due to these major functionalities it's safe to say, Python can be used to develop the application in today's environment.

V. PLATFORM SELECTION

A platform is the hardware or software environment in which a program runs. As 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 Python platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms. We've already been introduced to the Python Programming Language. It's the base for the Python based platform and is ported onto various hardware-based platforms. The Python's API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Python's API is grouped into libraries of related functionality and interfaces; these libraries are known as modules. Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Python 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.

Input Design

The info outline is the connection between the client and data framework and creating particular and strategies for information arrangement and those strides are important to put exchange information into a usable structure for handling can be accomplished by investigating the PC to peruse information from a composed or printed archive or it can happen by having individuals entering the information specifically into the framework.

Output Design

A quality output is one, which meets the necessities of the end client and presents the data obviously. In any framework consequences of preparing are conveyed to the clients and to other framework through yields. In yield outline it is resolved how the data is to be dislodged for prompt need furthermore the printed version yield. It is the most essential and direct source data to the client. Productive and insightful yield outline enhances the framework's relationship to help client choice making.

GRAPHICAL USER INTERFACE DESIGN

Graphical user interface is a type of user interface that allows users to interact with electronic devices using images rather than text commands. A GUI represents the information & action available to a user through graphical icons & visual indicators such as secondary notation, as opposed to text-based interfaces, type command labels or text navigation. The actions are usually performed through direct manipulation of the graphical elements. A GUI uses a combination of technologies & devices to provide a platform that the user can interact with, for the tasks of gathering & producing information.

A series of elements confirming a visual language have evolved to represent information stored in computers. This makes it easier for people with a few computer skills to work with and use computer software. Designing the visual composition & temporal behavior of a GUI is an important part of software application programming in the area of human-computer interaction. Its goal is to enhance the efficiency & ease of use for the underlying logical design of a stored program, a design discipline known as usability. Methods of user-centered design are used to ensure that the visual language introduced in the design is well tailored to the tasks.

But the GUI is also an extra addition to the application development, the functionalities of the application can be used with a minimal to no GUI.

VI. MODULES

VIDEO DATA PROCESSING:

Video Data Processing is the solution to obtain the fundamental block of videos for further processing. For real-time detection on videos, the detection algorithm has to be applied to the fundamental or to the smallest part of the video element. Video works with playing a bunch of images with continuous relativity one after another in series. The usual videos that all the consumer applications use are at 24fps (frames per second) but applications like gaming and filmmakers use higher frame rates of up to 60fps and sometimes even 120 to 240fps. The videos are split into each of its frames and passed on to the next module

ANALYSIS:

The features of each frame have to be extracted and computed for neural network compatibility. The frame is first divided to its individual smallest value i.e, it's to its individual pixels. Pixels values are varied based on quality and the resolution of the video. The pixel value is converted into mathematical function value. This value is represented by an array. The array, to on which classification, localization functions are performed. Classification is obtained by performing K Means clustering on the image data to obtain anchor points for grouping the image. Using these anchor points as a means to classify pixel clusters, the localization process begins where all the similar anchor points are reduced to accurate ones. All these computed data is passed on to the neural network which compares the localized cluster pack with the trained data to deduce a number of output classes that the frame might contain. The computer data might be iterated inside the neural network for accuracy and to eliminate the redundancy. The coordinates of the class object are detected is passed further on.

DETECTION:

The output classes from the neural network may vary in the accuracy of the objects that were detected; the threshold is set for the accuracy of the classes. Which if it meets, the bounding boxes are drawn with the class object names on the frame for the analyzed object coordinates. This is print on each frame of the video in real time of the video capture.

VII.RESULTS

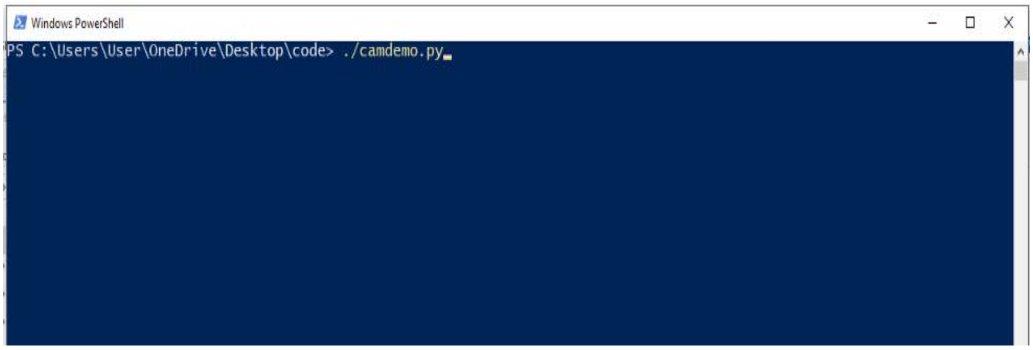


Fig 7.1: The command to open Web Camera

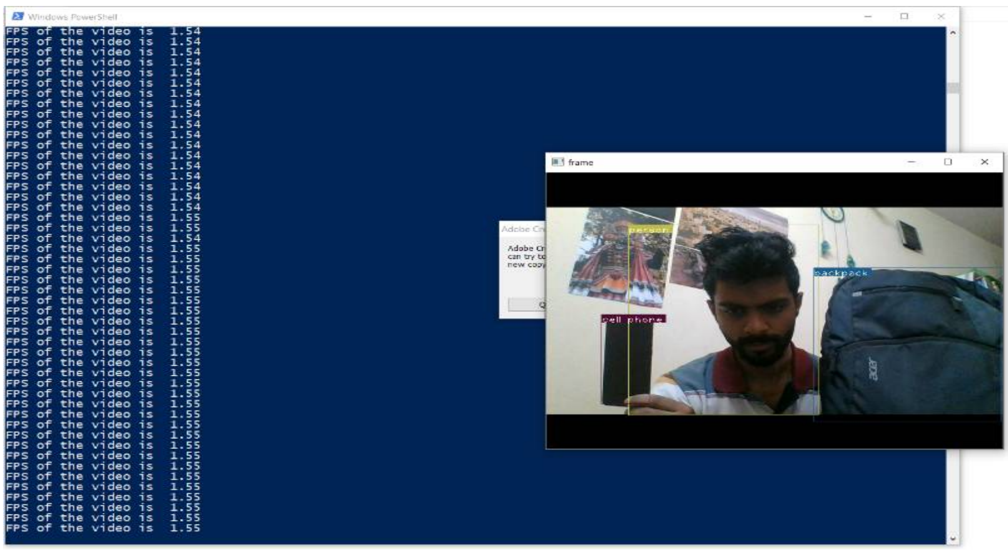


Fig 7.2: The output for the web camera command

The above figures show the command to execute the problem using the web camera and the output of it as we can see in the second figure. The second figure shows the objects along with the label and the bounding boxes.

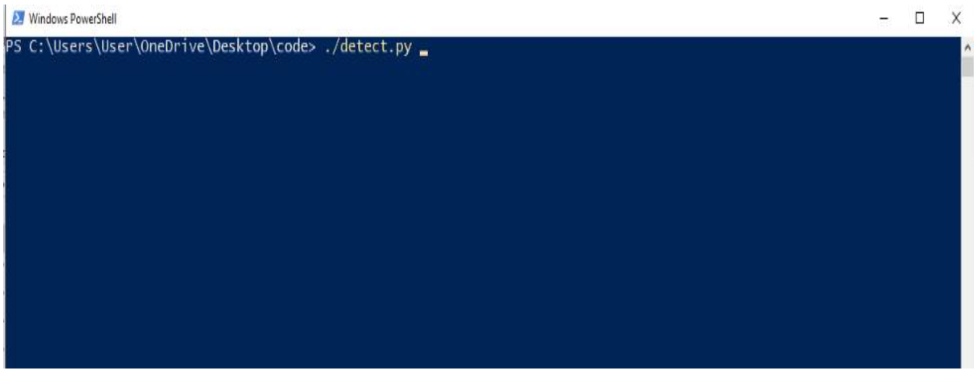


Fig 7.3: The command to open the folder that has the images

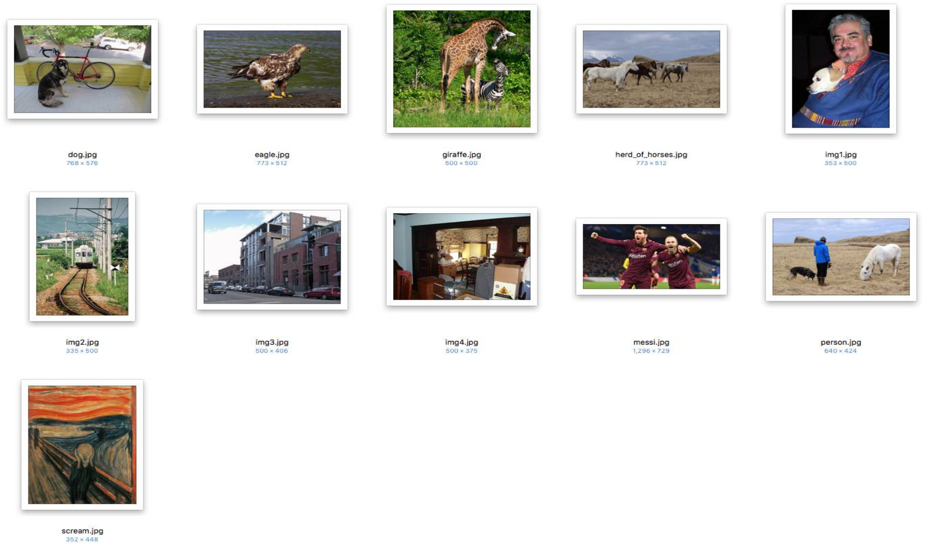


Fig 7.4: The folder that has all the images to detect

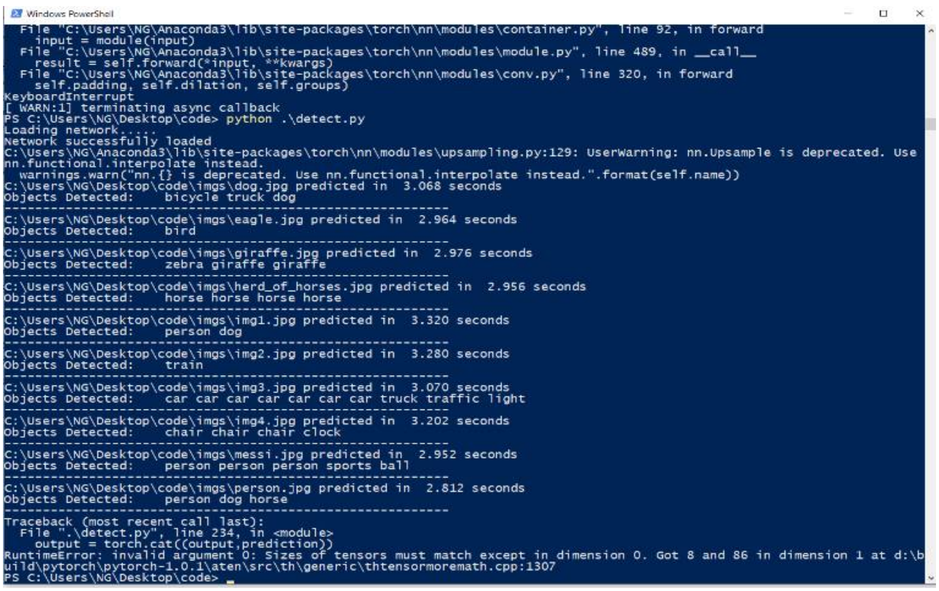


Fig 7.5: The output for the pervious command to detect objects in images

The above figures should the command to detect objects in still pictures, the folder with images to detect the objects and the output with all the objects in each image.

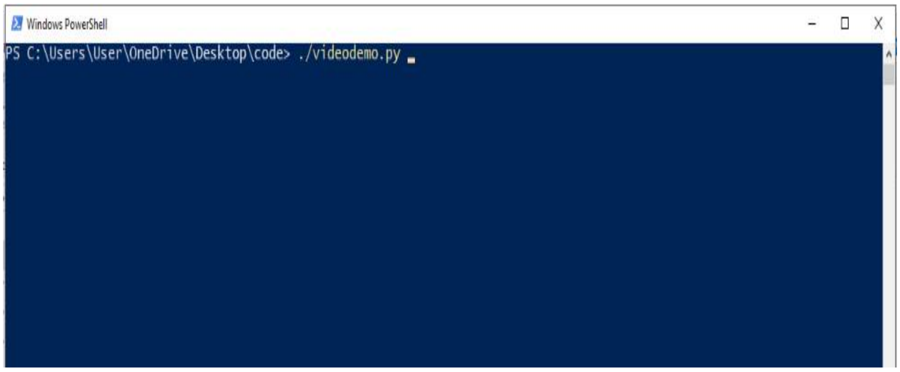


Fig 7.6: The command to open recorded videos

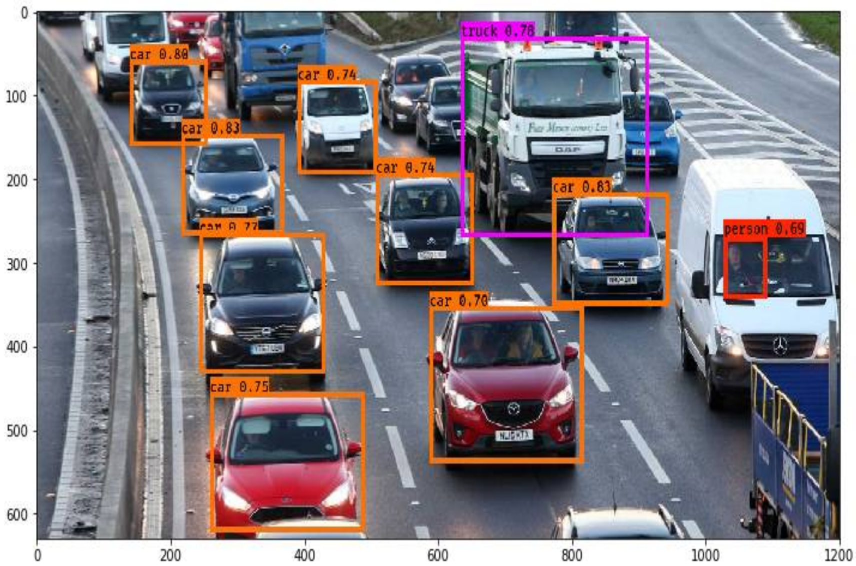


Fig 7.7: The output from the video A

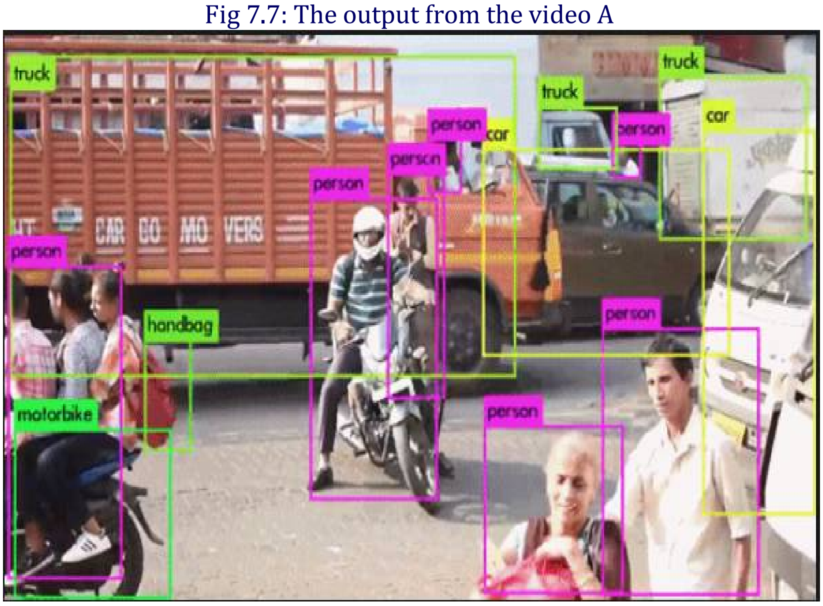


Fig 7.8: The output from video B

The above figures show the command to detect objects when a recorded videos are feed as input and the output for two different videos are shown.

VIII. CONCLUSION

An accurate and efficient object detection system has to be developed which achieves comparable metrics with existing state-of-the-art system results without any computing intensity rigs specially made for the purpose. Using recent techniques in the field of computer vision and deep learning using neural networking to attain state-of-the-art results within the boundary of lower end computing machines. This can be used in real-time applications which require object detection for pre-processing in their pipeline. This work has leverage the expressivity of DNNs for object detector. It shows that the simple formulation of detection as DNN-base object mask regression can yield strong results when applied using a multi-scale course-to-fine procedure. These results come at some computational cost at training time–one needs to train a network per object type and mask type. An important scope would be to train the system on a video sequence for usage in tracking applications. Addition of a temporally consistent network would enable smooth detection and more optimal than per-frame detection

REFERENCES

1. Integrated Recognition, Localization and Detection using Convolutional Networks. By Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun
2. Very deep convolutional networks for large-scale image recognition. By Karen Simonyan & Andrew Zisserman
3. Object Detection Using Deep Neural Network. By Christian Szegedy, Dumitru Erhan, Alexander Toshkov Toshev
4. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations. By Honglak Lee, Roger Grosse, Rajesh Ranganath, Andrew Y. Ng
5. Deep Learning in Neural Networks: An Overview. By Jurgen Schmidhuber
6. Multi-view Face Detection Using Deep Convolutional Neural Networks. By Sachin Sudhakar Farfade, Mohammad Saberian, Li-Jia Li
7. Faster R-CNN - Towards Real-Time Object Detection with Region Proposal Networks. By Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun