IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⁂ Impact Factor 7.047 ⁂ Vol. 10, Issue 5, May 2022

DOI: 10.17148/IJIREEICE.2022.10542

Object Detection and Image Segmentation

Mrs. Harsha Sarode, Mrs.Nutan Patil, Mr. Sagar Gaikwad,

Mr. Anurag Wagh, Mr. Aditya Kumar

EnTC Department, NMIET College, India

Abstract*:* Object detection is one of the most basic and central task in computer vision. Its task is to find all the interested objects in the image, and determine the category and location of the objects. Object detection is widely used and has strong practical value and research prospects. Applications include face detection, pedestrian detection and vehicle detection. In recent years, with the development of convolutional neural network, significant breakthroughs have been made in object detection. This paper describes in detail the classification of object detection algorithms based on deep learning. The algorithms are mainly divided into one-stage object algorithm and two-stage object algorithm, and the general data sets and performance indicators of object detection.
Image segmentation plays an important role in a pre-processing phase of images having as objective a partition of the image into components or regions of interest for a more detailed analysis of one or more of these regions. Image segmentation may also be used as a pre-processing phase for a better image de-noising or de-blurring that will be done in a separate image processing phase. In this article we study mostly theoretical concepts and some experimental results for evaluation of some image segmentation techniques and their role for a better analysis of image details.

INTRODUCTION

Image processing is among the rapidly growing technologies today. It helps to improve images for human interception. here the required specific toolkit is used for the analysis and recognition of image and ensure the effective
Development of application through it.
Object recognition is collection of related computer vision task that involves activities like identification object in digital Photographs. Here the two task Object localization and image classification are combined to predict the class of an object in an image and to draw bounding box around the detection object here the basic process is
Input: An image which consist od one or more object such as photograph
Output: One or more bounding boxes.
One of the further extensions to this breakdown of computer vision tasks is object segmentation, also called "object instance segmentation" or "semantic segmentation," where instances of recognized objects are indicated by highlighting the specific pixels of the object instead of a coarse bounding box. From this breakdown, we can understand that object recognition refers to a suite of challenging computer vision tasks.
Humans can detect and identify objects present in an image. The human visual system is fast and accurate and can also perform complex tasks like identifying multiple objects and detect obstacles with little conscious thought. The availability of large sets of data, faster GPUs, and better algorithms, we can now easily train computers to detect and classify multiple objects within an image with high accuracy. We need to understand terms such as object detection, object localization, loss function for object detection and localization, and finally explore an object detection algorithm known as "You only look once" (YOLO).
Object recognition refers to a collection of related tasks for identifying objects in digital photographs. Region-based Convolutional Neural Networks, or R-CNNs, is a family of techniques for addressing object localization and recognition tasks, designed for model performance. You Only Look Once, or YOLO is known as the second family of techniques for object recognition designed for speed and real-time use.

BACKGROUND

The aim of object detection is to detect all instances of objects from a known class, such as people, cars or faces in an image.
Generally, only a small number of instances of the object are present in the image, but there is a very large number of possible locations and scales at which they can occur and that need to somehow be explored.
Each detection of the image is reported with some form of pose information. This is as simple as the location of the object, a location and scale, or the extent of the object defined in terms of a bounding box. In some other situations, the pose information is more detailed and contains the parameters of a linear or non-linear transformation.
For example, for face detection in a face detector may compute the locations of the eyes, nose and mouth, in addition to the bounding box of the face.

An example of a bicycle detection in an image that specifies the locations of certain parts is shown in Figure 1. The pose can also be defined by a three-dimensional transformation specifying the location of the object relative to the camera. Object detection systems always construct a model for an object class from a set of training examples. In the case of a fixed rigid object in an image, only one example may be needed, but more generally multiple training examples are necessary to capture certain aspects of class variability

Convolutional implementation of the sliding windows Before we discuss the implementation of the sliding window using convents, let us analyze how we can convert the fully connected layers of the network into convolutional layers. Fig. 2 shows a simple convolutional network with two fully connected layers each of shape.

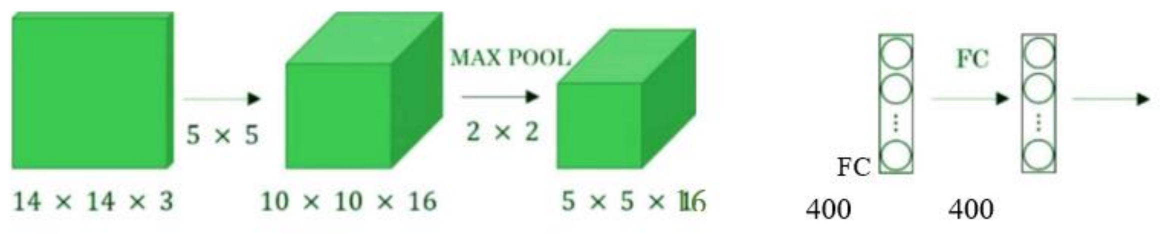


Fig1:

A fully connected layer can be converted to a convolutional layer with the help of an ID convolutional layer. The width and height of this layer is equal to one and the number of filters is equal to the shape of the fully connected layer.

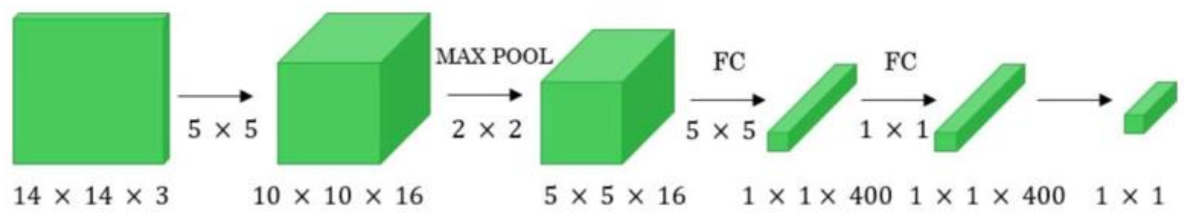


Fig2:

We can apply the concept of conversion of a fully connected layer into a convolutional layer to the model by replacing the fully connected layer with a I-D convolutional layer. The number of filters of the ID convolutional layer is equal to the shape of the fully connected layer. This representation is shown in Fig 1. Also, the output SoftMax layer is also a convolutional layer of shape (1, 1, 4), where 4 is the number of classes to predict

OBJECT DETECTION

Object detection is an important task, yet challenging vision task. It is a critical part of many applications such as image search, image auto-annotation and scene understanding, object tracking.

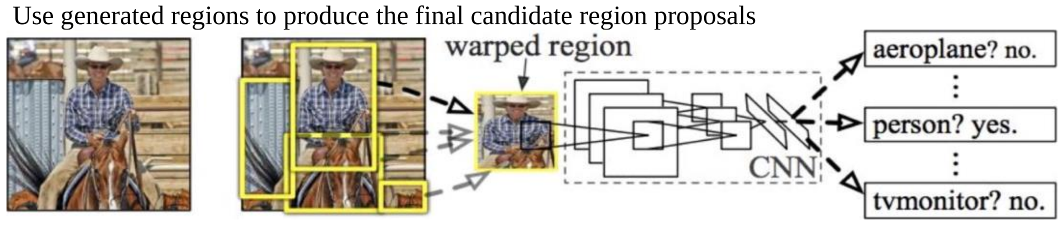
DOI: 10.17148/IJIREEICE.2022.10542

Moving object tracking of video image sequences was one of the most important subjects in computer vision. It had already been applied in many computers vision fields, such as smart video surveillance artificial intelligence, military guidance, safety detection and robot navigation, medical and biological application.
In recent years, a number of successful single-object tracking system appeared, but in the presence of several objects, object detection becomes difficult and when objects are fully or partially occluded, they are obtruded from the human vision which further increases the problem of detection. Decreasing illumination and acquisition angle.

The proposed MLP based object tracking system is made robust by an optimum selection of unique features and also by implementing the Ad boost strong classification method.

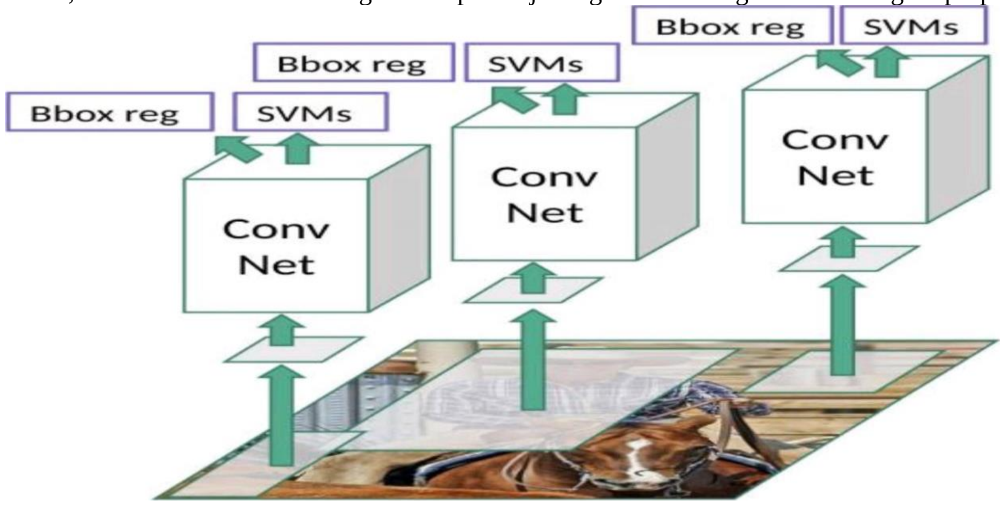
RCNN

To circumvent the problem of selecting a huge number of regions, Ross Girshick et al. proposed a method where we use the selective search for extract just 2000 regions from the image and he called them region proposals. Therefore, instead of trying to classify the huge number of regions, you can just work with 2000 regions. These 2000 region proposals are generated by using the selective search algorithm which is written below.
Selective Search:
Generate the initial sub-segmentation, we generate many candidates' regions
Use the greedy algorithm to recursively combine similar regions into larger ones
Use generated regions to produce the final candidate region proposals



1.Input image 2. Extract region proposals (—2k) 3. Compute CNNfeatures 4. Classify regions
e

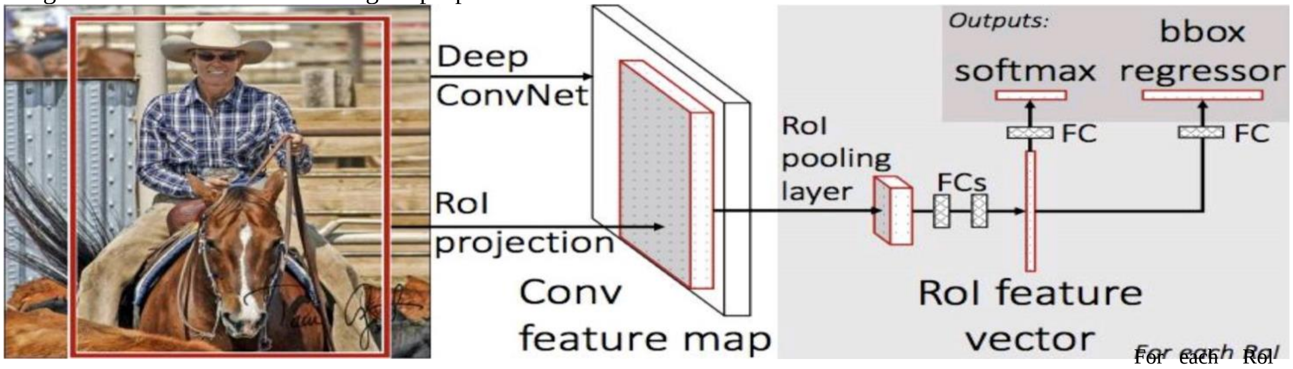
These 2000 candidate regions which are proposals are warped into a square and fed into a convolutional neural network that produces a 4096-dimensional feature vector as output. The CNN plays a role of feature extractor and the output dense layer consists of the features extracted from the image and the extracted features are fed into an SVM for the classify the presence of the object within that candidate region proposal. In addition to predicting the presence of an object within the region proposals, the algorithm also predicts four values which are offset values for increasing the precision of the bounding box. For example, given the region proposal, the algorithm might have predicted the presence of a person but the face of that person within that region proposal could have been cut in half.
Therefore, the offset values which is given help in adjusting the bounding box of the region proposal.



Problems with R-CNN
It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image.
It cannot be implemented real time as it takes around 47 seconds for each test image.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526
International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
ISO 3297:2007 Certified ⁒ Impact Factor 7.047 ⁒ Vol. 10, Issue 5, May 2022
DOI: 10.17148/IJIREEICE.2022.10542

The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals

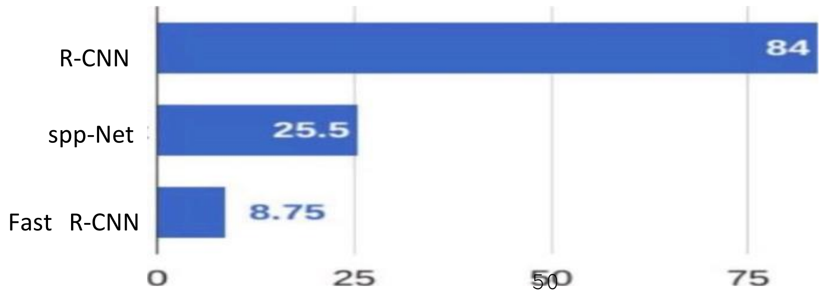


III. Fast R-CNN

The same author of the previous paper(R-CNN) solved some of the drawbacks of R-CNN to build a faster object detection algorithm and it was called Fast R-CNN. The approach is similar to the R-CNN algorithm.

But, instead of feeding the region proposals to the CNN, we feed the input image to the CNN to generate a convolutional feature map. From the convolutional feature map, we can identify the region of the proposals and warp them into the squares and by using a Roll pooling layer we reshape them into the fixed size so that it can be fed into a fully connected layer. From the Roll feature vector, we can use a SoftMax layer to predict the class of the proposed region and also the offset values for the bounding box.

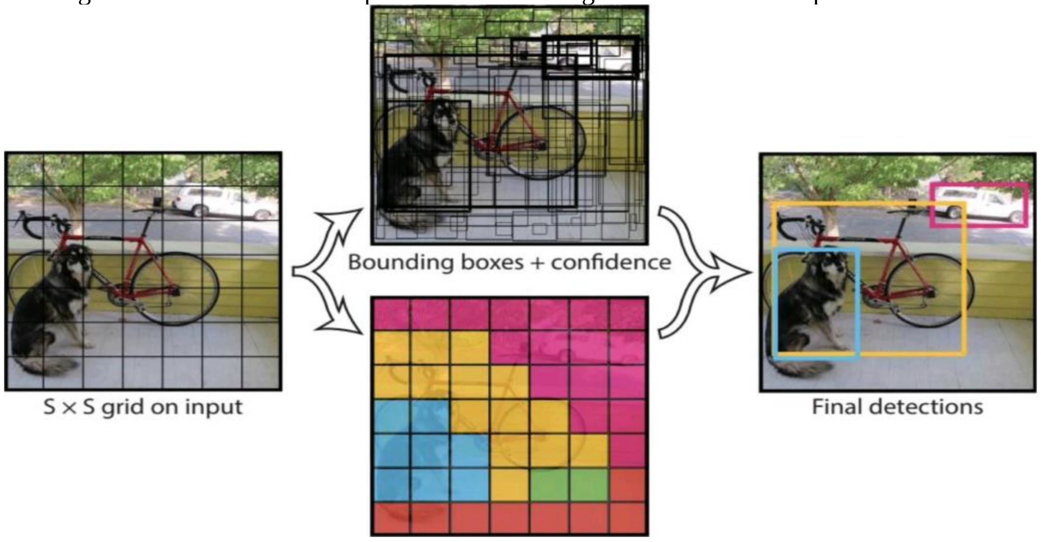
The reason "Fast R-CNN" is faster than R-CNN is because you don't have to feed 2000 region proposals to the convolutional neural network every time. Instead, the convolution operation is always done only once per image and a feature map is generated from it.



YOLO

All the previous object detection algorithms have used regions to localize the object within the image. The network does not look at the complete image.

In YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes.



.Figure 6

YOLO works by taking an image and split it into an SxS grid, within each of the grid we take m bounding boxes. For each of the bounding box, the network gives an output a class probability and offset values for the bounding box. The bounding boxes have the class probability above a threshold value is selected and used to locate the object within the image.

YOLO is orders of magnitude faster (45 frames per second) than any other object detection algorithms. The limitation of YOLO algorithm is that it struggles with the small objects within the image, for example, it might have difficulties in identifying a flock of birds. This is due to the spatial constraints of the algorithm.

. SYSTEM REQUIREMENT:

Install Python on your computer system
Install Image AI and its dependencies like tensorflow, NumPy, OpenCV, etc.
Download the Object Detection model file (Retinanet)
Steps to be followed: -
l) Download and install Python version 3 from official Python Language website
https://python.org
2) Install the following dependencies via pip:
Tensorflow:
Tensorflow is an open-source software library for dataflow and differentiable programming across a range of tasks. It is an symbolic math library, and is also used for machine learning application such as neural networks,etc.. It is used for both research and production by Google.
Tensorflow is developed by the Google Brain team for internal Google use. It is released under the Apache License 2.0 on November 9,2015.
Tensorflow is Google Brain's second-generation system. 1st Version of tensorflow was released on February I l, 2017.While the reference implementation runs on single devices, Tensorflow can run on multiple CPU's and GPU (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow is available on various platforms such as64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.
The architecture of tensorflow allows the easy deployment of computation across a variety of platforms (CPU's, GPU's, TPU's), and from desktops - clusters of servers to mobile and edge devices.
Tensorflow computations are expressed as stateful dataflow graphs. The name Tensorflow derives from operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.
pip install tensorflow -command
Numpy:
NumPy is library of Python programming language, adding support for large, multi-dimensional array and matrice, along with large collection of high-level mathematical function to operate over these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several developers. In 2005 Travis Olphant created NumPy by incorporating features of computing Numarray into Numeric, with extension modifications. NumPy is open-source software and has many contributors.
pip install NumPy -command
SciPy:
SciPy contain modules for many optimizations, linear algebra, integration, interpolation, special fumction, FFT, signal and image processing, ODE solvers and other tasks common in engineering. SciPy abstracts majorly on NumPy array object,and is the part of the NumPy stack which include tools like Matplotlib, pandas and SymPy,etc., and an expanding set of scientific computing libraries. This NumPy stack has similar uses to other applications such as MATLAB,Octave, and Scilab. The NumPy stack is also sometimes referred as the SciPy stack.
The SciPy library is currently distributed under BSDlicense, and its development is sponsored and supported by an open community of developers. It is also supported by NumFOCUS, community foundation for supporting reproducible and accessible science.
pip install scipy -command
OpenCV:
OpenCV is a library of programming functions mainly aimed on real time computer vision.
originally developed by Intel, it is later supported by Willow Garage then Itseez. The library is a cross-platform and free to use under the open-source BSD license.
pip install opencv-python -command
Pillow:
Python Imaging Library is a free Python programming language library that provides support to open, edit and save several different formats of image files. Windows, Mac OS X and Linux are available for this.

DOI: 10.17148/IJIREEICE.2022.10542

pip install pillow -command
Matplotlib:
Matplotlib is a Python programming language plotting library and its NumPy numerical math extension. It provides an object-oriented API to use general-purpose GUI toolkits such as Tkinter, wxPython, Qt, or GTK+ to embed plots into applications.
pip install matplotlib - command
H5py:
The software h5py includes a high-level and low-level interface for Python's HDF5 library. The low interface expected to be complete wrapping of the HDF5 API, while the high-level component uses established Python and NumPy concepts to support access to HDF5 files, datasets and groups.
A strong emphasis on automatic conversion between Python (NumPy) datatypes and data structures and their HDF5 equivalents vastly simplifies the process of reading and writing data from Python.
pip install h5py
Kera's
Kera's is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.
pip install karas
Image AI:
Image AI provides API to recognize 1000 different objects in a picture using pre-trained models that were trained on the ImageNet-1000 dataset. The model implementations provided are Squeeze Net,
ResNet, InceptionV3 and DenseNet.
pip3 install imageai –upgrade

3) Download the RetinaNet model file that will be used for object detection using following link
https://github.com/OlafenwaMoses/ImageAI/releases/download/l.0/resnet50_coco_best_v2.0. I. h5
Copy the RetinaNet model file and the image you want to detect to the folder that contains the python file.

Process of image detection and segmentation

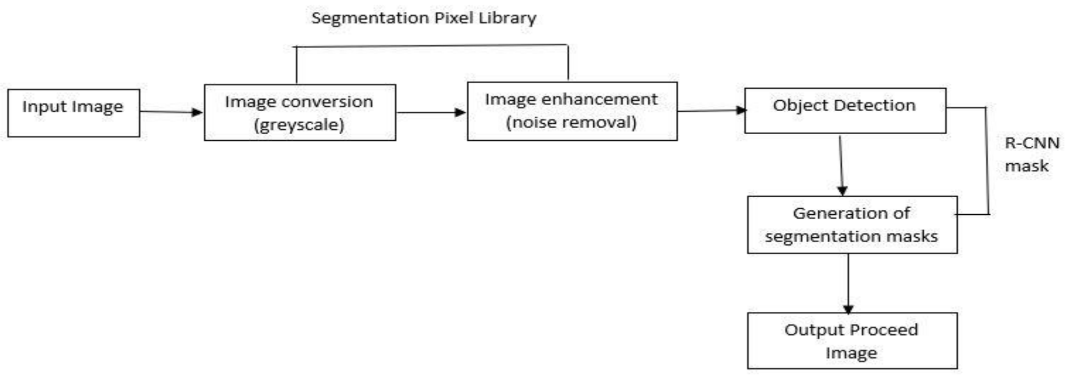

Segmentation Pixel Library
Input Image
Image conversion (greyscale)
Image enhancement (noise removal)
Object Detection
R-CNN mask
Generation of segmentation masks
Output Proceed Image

Fig. 1 A sample figure

Image conversion (Gray Scale):
An intuitive way to convert a color image 3D array to a grayscale 2D array is,for each pixel, take the average of the red, green, and blue pixel values to get the grayscale value.
This combines the lightness or luminance contributed by each color band into a reasonable gray approximation.
In this step, we convert the RGB image into gray scale image for easier segmentation process.This process is done using Pixellib library.

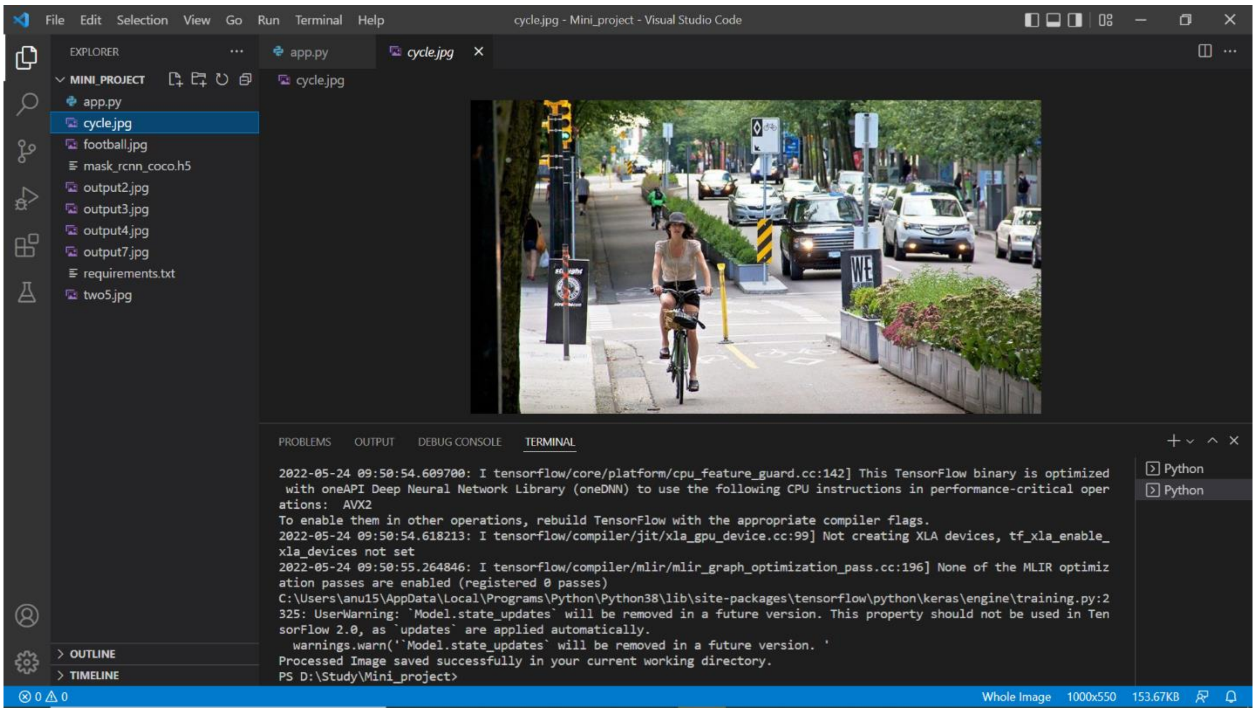
Image Enhancement (Noice Removal):
Image enhancement refers to the process of highlighting certain information of an image, as well as weakening or removing any unnecessary information according to specific needs.
For example, eliminating noise, revealing blurred details, and adjusting levels to highlight features of an image.

DOI: 10.17148/IJIREEICE.2022.10542

The purpose of the image enhancement is to improve the visual interpretability of an image by increasing the apparent distinction between the features in the scene.
This process is done using Pixellib library.

Object Detections:
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.
Object detection is completely inter-linked with other similar computer vision techniques such as image segmentation and image recognition that assist us to understand and analyze the scenes in videos and images.
Here we used RCNN for object detection. Mask R-CNN is a state-of-the-art model for instance segmentation, developed on top of Faster R-CNN.
Generation of Segmentation Masks:
Masking is an image processing method in which we define a small 'image piece' and use it to modify a larger image. Masking is the process that is underneath many types of image processing, including edge detection, motion detection, and noise reduction.
Mask R-CNN is a state-of-the-art model for instance segmentation, developed on top of Faster R-CNN.
A mask is a binary image consisting of zero- and non-zero values. If a mask is applied to another binary or to a grayscale image of the same size, all pixels which are zero in the mask are set to zero in the output image.

RESULTS



```
2022-05-24 09:50:54.609700: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized
 with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical oper
ations:  AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-05-24 09:50:54.618213: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_
xla_devices not set
2022-05-24 09:50:55.264846: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:196] None of the MLIR optimiz
ation passes are enabled (registered 0 passes)
C:\Users\anu15\AppData\Local\Programs\Python\Python38\lib\site-packages\tensorflow\python\keras\engine\training.py:2
325: UserWarning: `Model.state_updates` will be removed in a future version. This property should not be used in Ten
sorFlow 2.0, as `updates` are applied automatically.
  warnings.warn('`Model.state_updates` will be removed in a future version. '
Processed Image saved successfully in your current working directory.
PS D:\Study\Mini_project>
```

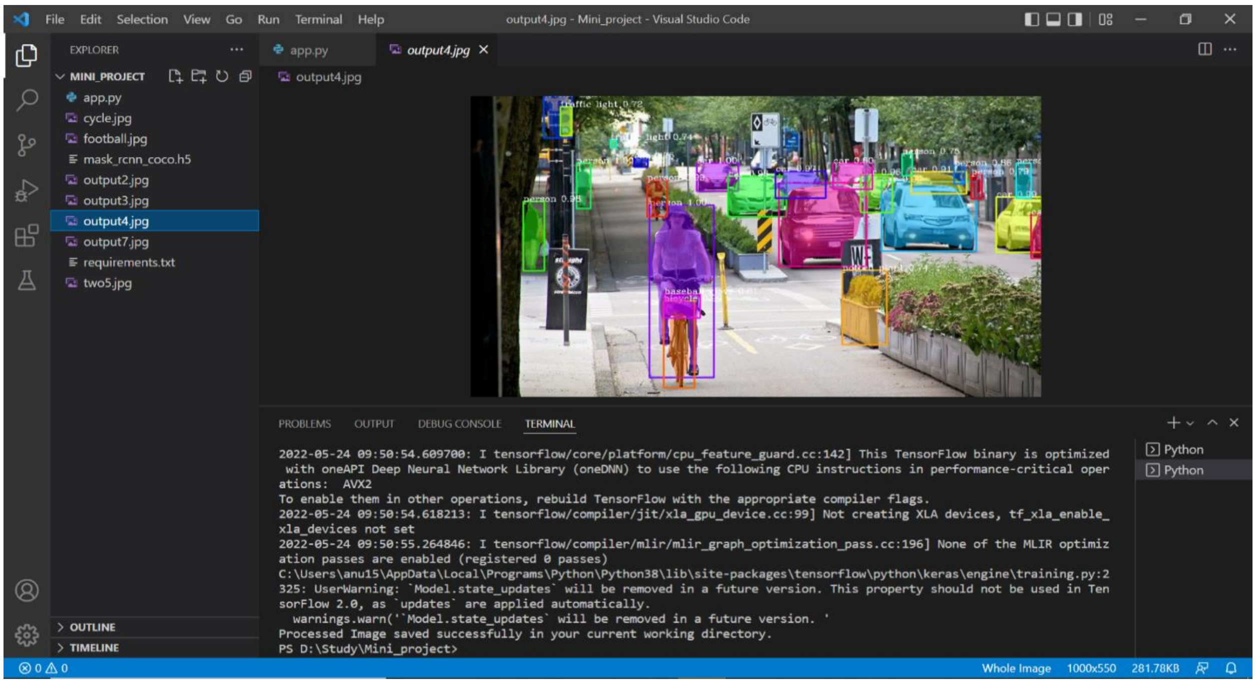
This is a sample image that identify object and segmentize them with use of algorithm

ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⁜ Impact Factor 7.047 ⁜ Vol. 10, Issue 5, May 2022

DOI: 10.17148/IJIREEICE.2022.10542



```
2022-05-24 09:50:54.609700: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized
 with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical oper
ations:  AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-05-24 09:50:54.618213: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_
xla_devices not set
2022-05-24 09:50:55.264846: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:196] None of the MLIR optimiz
ation passes are enabled (registered 0 passes)
C:\Users\anu15\AppData\Local\Programs\Python\Python38\lib\site-packages\tensorflow\python\keras\engine\training.py:2
325: UserWarning: `Model.state_updates` will be removed in a future version. This property should not be used in Ten
sorFlow 2.0, as `updates` are applied automatically.
  warnings.warn('`Model.state_updates` will be removed in a future version. '
Processed Image saved successfully in your current working directory.
PS D:\Study\Mini_project>
```

CONCLUSION

These projects tackle the basic functionality of image processing and Al learning. It focuses on features like segmentation, object identification. By using this thesis and based on experimental results we are able to detect object more precisely and identify the objects individually with exact location of an object in the picture in x, y axis

REFERENCES

Agarwal, S. , Awan, A. , and Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. IEEE Trans. Pattern Anal. Mach. Intell. 26,1475-1490. doi: 10.1109/TPAM1.2004.108
Alexe, B. , Deselaers, T., and Ferrari, V. (2010). "What is an object?," in ComputerVision and Pattern Recognition (CVPR), 2010 IEEE Conference on (San Francisco,CA: IEEE), 73-80. doi: 10.1109/CVPR.2010.5540226
Aloimonos, J. , Weiss, 1., and Bandyopadhyay, A. (1988). Active vision. Int. J.Comput. Vis. 1,
333-356. doi:10.1007/BF00133571
Andreopoulos, A. , and Tsotsos, J. K. (2013). 50 years of object recognition: direc-tions forward.
Comput. Vis. Image Underst. 117, 827—891. doi:10.1016/j.cviu.2013.04.005
Azizpour, H., and Laptev, I. (2012). "Object detection using strongly-superviseddeformable part models," in Computer Vision-ECCV 2012 (Florence: Springer),836—849.
Azzopardi, G., and Petkov, N. (2013). Trainable cosfire filters for keypoint detectionand pattern recognition. IEEE Trans. Pattern Anal. Mach. Intell. 35, 490-503 .doi: 10.1109/TPAM1.2012.106
Azzopardi, G. , and Petkov, N. (2014). Ventral-stream-like shape representation:from pixel intensity values to trainable object-selective cosfire models. Front.Comput. Neurosci. 8:80. doi:10.3389/fncom.2014.00080
Benbouzid, D., Busa-Fekete, R., and Kegl, B. (2012). "Fast classification using sparsedecision dags," in Proceedings of the 29th International Conference on MachineLearning (ICML-12), ICML ' 12, eds
J. Langford and J. Pineau (New York, NY:Omnipress), 951—958.
Bengio, Y. (2012). "Deep learning of representations for unsupervised and transferlearning," in ICML Unsupervised and Transfer Learning, Volume 27 of JMLRProceedings, eds I. Guyon, G. Dror,
V. Lemaire, G. W. Taylor, and D. L. Silver(Bellevue: JMLR.Org), 17—36