Object Detection by Image Processing

Archana Amoriya
Department of MCA, Computer Science, Jain (Deemed-to-be) University, Banglore, Karnataka, India

***Abstract* - In this world of information for human beings. Earlier it was impossible to achieve but due to the development of new technologies it has been made possible. Image processing has its impact on communication devices also. By digital image processing we can enhance the image extract the text from image, edges of images can be detected, and we can apply other effects also in object detector. We can get any details about the images. There are many applications of digital image processing. Almost this technique is use in every field medical field, robotics., neural networking, also useful in Crime branch for investigation. There text detector face detector technology be use in detection part. We can use in education fields like schools as well college children mind can fast understand if using object detection for identify its must they can use and found the object with the help of image. In digital generation we can improve our system based on detection way we can also detect text as well and many things helping with OpenCV. There are many different datasets there are millions of varieties of image which can use in this technology.**

***Index Terms* - Object Detection, OpenCV, CoCo Dataset, Yolov3, Python GUI programming (tkinter).**

1.INTRODUCTION

Image processing is concerned with processing of an image. Image processing is a method to perform operations on images like enhancing images, extracting text from image, detecting edge of image and many other operations. Object Detection is the process of finding real-world object instances like car, bike, TV, flowers, and humans in still images or Videos. It allows for the recognition, localization, and detection of multiple objects within an image which provides us with a much better understanding of an image as a whole. It is commonly used in applications such as image retrieval, security, surveillance, and advanced driver assistance systems.

To effectively manage all this data, we need to have some idea about its contents. Automated processing of image contents is useful for a wide variety of image-related tasks. For computer systems, this means crossing the so-called semantic gap between the pixel level information stored in the image files and the human understanding of the same images. Object recognition is to describe a collection of related computer vision tasks that involve activities like identifying objects in digital photographs. Image classification involves activities such as predicting the class of one object in an image. Also detect drawing of which object have the image height width are mentioned in the existing program the image good pixels are also important for object detection.

An image which consists of one or more objects, such as a photograph in output One or more bounding boxes identify the object This is done through, Locate the presence of objects with a bounding box and types or classes of the located objects in an image. 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.

The aim main is detect all the data easily can be detect we can use in online or offline mood object Detection has a wide range of applications. We can deploy model in future we can deployed another different model. Once we are done with downloading the necessary files, we will load the pre-trained model. It is a real-time scenario.

Import the necessary libraries in a python file. Then by using the OpenCV image capture (mostly jpg recommended) and load the class files, model configuration file, and model weight file. Then define the confidence threshold for detecting an image with the desired accuracy and non-max suppression to reduce the overlapping of the bounding boxes.

2.METHOLOGY

Object detection: The model detecting object for recognizing it with database, image of the object or person gets detected and checks whether there are any

predefined patterns available for that image searching for digital images.

Yolo takes an input image first and the input image is divided into grids(say 3 X 3 grid) as show in fig 1

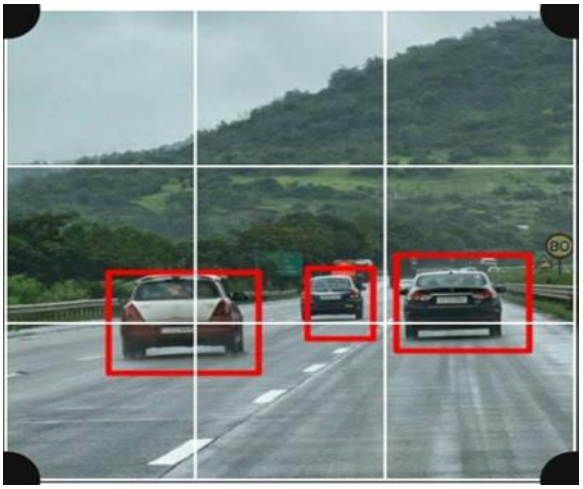


Figure 1

In each grid, image classification and localization are applied. The bounding boxes and their equivalent class probabilities for objects are then predicted by YOLO. In order to train it, the labelled data needs to be transferred to the model. Suppose the image is divided into a 3 X 3 grid and there is an aggregate of 3 classes in which the objects need to be categorized. Suppose that the classes are people, cars, and trucks, the y label is an 8-dimensional vector for each grid cell as shown in Table 1

y =	1
	bx
	by
	bh
	bw
	0
	1
	0

Table 1: 8-dimensional Y label 1

Figure 2 without object grid

- The object being present in the grid is described by pc.
- Bx, by, bh, bw coordinates if an entity is present.
- The classes are reflected by c1, c2, c3. Consider the first grid from fig 1 as shown in Fig 2
- Fig 1.2: Grid having no object
- Since there is no object in this grid as shown in Fig 1.2 pc will be zero and all the other entries in the y lable will be ? i.e., As there o entity in the grids, its doesn't matter what the bx, by, bh, bw, c1, c2, and c3 values are

Since in this grid there is an entity, pc will be to 1 and bx, by, bh, bw will be determined according to the same grid cell with which we are dealing. Therefore, since the car is the 2nd class, c2 = 1 and c1 and c3 = 0. An 8-dimensional output vector for each of the 9 grids is outputted. This performance will have a dimension to the (3 X 3 X 8). The object could be assigned to a solitary grid where its mid-point is found, regardless of whether an entity spreads to more than one grid. By increasing the number of grids, we can reduce the odds of different objects occurring in a similar grid cell.

There is three objects (3cars) in Fig 1; YOLO takes the mid-point of these objects to the grid containing the mid-point of these objects. The Y label for the left-cantered grid with the cars is as shown in Table 2.

y =	pc
	bx
	by
	bh
	bw
	c1
	c2
	c3

Table 2: Y label of left centred grid

3. SYSTEM ARCHITECTURE YOLOV3

In a single glance, take the entire image and predicts for these boxes the bounding box coordinates and class probabilities. YOLOs greatest advantage is its outstanding pace, it's extremely fast, and it can handle 45 frames per second . Amongst the three versions of YOLO, version-3 is fastest and more accurate in terms of detecting small objects. The proposed algorithm, YOLO version-3 consists of total 106 layers. The architecture is made up of 3 distinct layer forms.

Firstly, the residual layer which is formed when activation is easily forwarded to a deeper layer in the neural network. In a residual setup, outputs of layer 1 are added to the outputs of layer 2. Second is the detection layer which performs detection at 3 different scales or stages. Size of the grids is increased for detection. Third is the up-sampling layer which increases the spatial resolution of an image. Here image is up sampled before it is scaled. Also, concatenation operation is used, to concatenate the outputs of previous layer to the present layer. Addition operation is used to add previous layers. In the Fig 3, the pink colored blocks are the residual layers, orange ones are the detection layers and the green are the up-sampling layers. Detection at three different scales is as shown Fig 3.

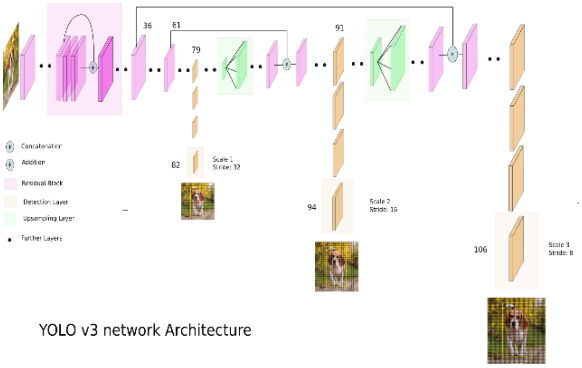


Figure 3 Architecture of yolov3
Image credit by:https://cdn-images-

YOLO v3 predicts 3 different scales of prediction. The detection layer is used to detect feature maps of three different sizes, with strides 32, 16, 8 respectively. This means that detections are made on scales of 13 x 13, 26 x 26 and 52 x 52 with an input of 416 x 416.
To understand the YOLO algorithm, it is necessary to establish what is actually being predicted. Ultimately, we aim to predict a class of an object and the bounding box specifying object location. Each bounding box can be described using four descriptors:

1. center of a bounding box (bxby)
2. width (bw)
3. height (bh)
4. value cis corresponding to a class of an object (such as: car, table, etc.).

Bounding Box Prediction:
Following YOLO9000 our system predicts bounding boxes using dimension clusters as anchor boxes. The network predicts 4 coordinates for each bounding box, tx, ty, tw, th. If the cell is offset from the top left corner of the image by (cx, cy) and the bounding box prior has width and height pw, ph, then the predictions correspond to:
bx = σ(tx)+ cx
by = σ(ty) + cy
bw = pwe tw
bh = phe th
During training we use sum of squared error loss. If the ground truth for some coordinate prediction is tˆ * our gradient is the ground truth value (computed from the ground truth box) minus our prediction: tˆ * − t*. This ground truth value can be easily computed by inverting the equations above. YOLOv3 predicts an objectness score for each bounding box using logistic regression. This should be 1 if the bounding box prior overlaps a ground truth object by more than any other bounding box prior. If the bounding box prior

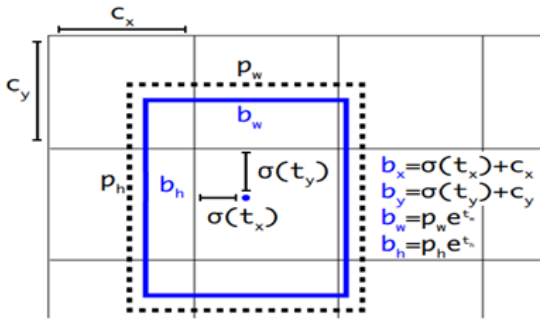


Figure 4 Bounding box dimension
Figure 2.2 Bounding boxes with dimension priors and location prediction. We predict the width and height of the box as offsets from cluster centroids. We predict the center coordinates of the box relative to the location of filter application using a sigmoid function. This figure blatantly self-plagiarized from is not the best but does overlap a ground truth object by more than some threshold we ignore the prediction, following . We use the threshold of Unlike our system only assigns one bounding box prior for each ground truth object. If a bounding box prior is not assigned to a ground truth object it incurs no loss for coordinate or class predictions, only object.

Implementation of YOLO

- Darknet: This algorithm is implemented using an open-source neural network framework i.e., Darknet which was developed in C Language and

CUDA technology to render speedy calculations on a GPU necessary for real-time predictions.

- DNModel.py: Darknet Model file is a computer vision code used for building the model using the configuration file and it appends each layer.
- Util.py: Contains all the formulas used.
- imageprocees.py: Required to perform the image processing task. It takes all the input images to resize them and perform Up-sampling, also performs transpose function.
- detect.py: The main code which is run to perform object detection. This code uses all the above-mentioned files to perform object detection. Performs all the functions according to the YOLO concept.

4. EXPLAINING OBJECT DETECTION

1. Object Detection: A object detection computer technology which can be use for detect the object in real time. Its deep learning machine there are such type of detection text detect vehicle detection mostly are used this technology in the world. The object detections are one type of image processing it is edge to detect the object there are some model are use in detection like opencv, threshold etc using yolov3 algorithm its version 3 of the yolo series. In the detector we use coco dataset.
2. Object Detection Workflow: Every Object Detection Algorithm has a different way of working, but they all work on the same principle.
3. Feature Extraction: They extract features from the input images at hands and use these features to determine the class of the image. Be it through Open CV.
4. Object detection flowchart: In the object detector first Image classification aims at assigning an image to one of a number of different categories (e.g. car, dog, cat, human, etc.) Object localization then allows us to locate our object in the image Object detection provides the tools for doing just that finding all the objects in an image and drawing the bounding boxes around them.
5. Load The Pre-Trained Model: Once we are done with downloading the necessary files we will load the pre-trained model. Since Yolov3 model has been trained on the MS COCO dataset for thousands of epochs and hours of training, so there is no need for explicit training rather we can just use the pretrained model and its weights provided by yolo Next step is to load the Yolov3 weights and configuration file so that it can perform the multi-class object detection.

TESTING THE MODEL

Once we have loaded all the configuration files and the model and its weights. Now we just have to test our model with test images that we downloaded earlier. The model performed well recognizing most of the objects in the image and also drew a bounding box around the objects.

6.

Figure 5 Process of object detection

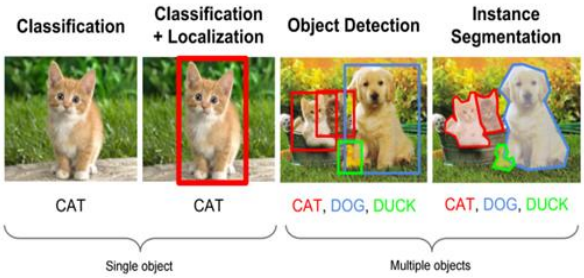


Figure 6 Process of multiple object detection

Image credit by: https://appsilon.com

7.COCO DATASET

COCO (Common Objects in Context), element of deep learning and machine learning at large is dataset. A good dataset will contribute to a model with good precision and recall being one of the most popular image datasets out there, with applications like object detection, segmentation, and captioning. COCO provides multi-object labeling, segmentation mask annotations, image captioning, key-point detection and panoptic segmentation annotations with a total of 81 categories, making it a very versatile and multi-purpose dataset. object detection model should get bounding boxes for objects, i. e. return list of object

classes and coordinates of rectangles around them; objects (also called "things") are discrete, separate objects, often with parts, like humans and cars; the official dataset for this task also contains additional data for object segmentation.

Image Segmentation with a python library called pycoco. This library eases the handling of the COCO dataset, which otherwise would have been very difficult to code yourself.

COCO is a large-scale object detection segmentation, and captioning dataset.

figure 7

COCO provides multi-object labeling, segmentation mask annotations, image captioning, key-point detection and panoptic segmentation annotations with a total of 81 categories, making it a very versatile and multi-purpose dataset.

Objects in COCO:

There are 91 object categories in COCO. However, only 80 object categories of labeled and segmented images were released in the first publication in 2014. Currently there are two releases of COCO dataset for labeled and segmented images. After the 2014 release, the subsequent release was in 2017. To compare and confirm the available object categories in COCO dataset, we can run a simple Python script that will output the list of the object categories.

- Manual labeling and modeling: Objects are labeled using bounding box or segmentation technique and neural network for object recognition.
- Transfer learning: Existing pre-trained model is adapted when performing object recognition in a new domain. A prevalent technique is by reusing the hidden layers of the pre-trained model to extract features of objects and replacing the final / output layer with classification that is specific to the new domain.

Class	AP	Class	AP
Aeroplan	0.7540	Dining table	0.2885
bicycle	0.5928	dog	0.6976
bird	0.1813	horse	0.8702
boat	0.5299	motorbike	0.7213
bottle	0.4654	person	0.7176
bus	0.9681	potted plant	0.5069
car	0.8584	sheep	0.7767
cat	0.8568	sofa	0.7894
chair	0.7546	train	0.8623
cow	0.7546	TV monitor	0.7670

Table 3 . AP (Average Precision) for different classes of the pretrained Yolo model on COCO 2014 dataset

The following JSON shows 2 different annotations.

The first annotation:

Has a segmentation list of vertices (x, y pixel positions)

Has an area of 702 pixels (pretty small) and a bounding box of [473.07,395.93,38.65,28.67]

Is not a crowd (meaning it's a single object)

Is category id of 18 (which is a dog)

Corresponds with an image with id 289343 (which is a person on a strange bicycle and a tiny dog)

The second annotation:

Has a Run-Length-Encoding style segmentation

Has an area of 220834 pixels (much larger) and a bounding box of [0,34,639,388]

Is a crowd (meaning it's a group of objects)

Is a category id of 1 (which is a person)

8.PYTHON TKINTER

The tkinter package ("Tk interface") is the standard Python interface to the Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, as well as on Windows systems. Most of the time, tkinter is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named _tkinter. This module contains the low-level interface to Tk and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

In addition to the Tk interface module, tkinter includes a number of Python modules, tkinter.constants being one of the most important. Importing tkinter will

automatically import tkinter.constants, so, usually, to use Tkinter all you need is a simple import statement.

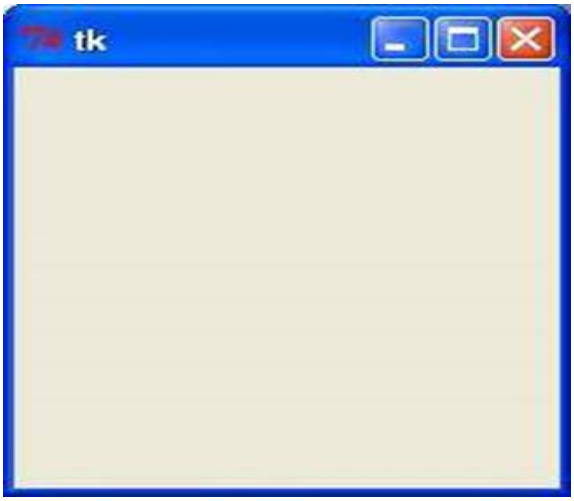
Tkinter Programming

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps

- Import the Tkinter module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

This would create a following window –



a. Tkinter Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets. There are currently 15 types of widgets in Tkinter.

9.LITERATURE REVIVEW

- A proposed object detection is a well-known computer technology connected with computer vision and image processing that focuses on detecting objects or its instances of a certain class (such as humans, flowers, animals) in digital images and videos. There are various applications of object detection that have been well researched including face detection, character recognition, and vehicle calculator. Object detection can be used for various purposes including retrieval and surveillance. In this study, various basic concepts used in object detection while making use of OpenCV library of python 3.7, improving the efficiency and accuracy of object detection are presented.
- Detection highly use in present world like we can use in medical fields to scan human body to easy detect the problem of human defect part its also use crime branch. At the time corona virus crises the education system in digital way we can use object detection like attendance its easy detect student through the video . In opencv we can use different dataset for object detection as well algorithm .
- Objection detection application are also in school for study like they easily put a picture and they found identify the object name its help teachers and there parents for the knowledge of the children they can learn most of the object to identify.

10.PROBLEM DEFINITION

- The goal of this project is to make the detection techniques to the production level. Because the loading time of the trained weights are so long before the predictions, in the production level, trained weights should be ready all the time in the server and if the testing data comes, the prediction of this data should be committed immediately without loading the weights.
- The application not prediction always right the problem with the animals birds shape and color are not find correctly through library of dataset so prediction also wrong detect in the application. If one image has so many object its difficult to proper detect.
- There also problem if Image size are small so highlighted rectangular box not proper show output must not clear show text and object box are mixing.
- The loading time are slow to finding object in the library its difficult like in animal image.

11.RESULT AND ANALYSIS

With the help of object detection base on yolov3 weight yolov3 cfg algorithm detect the real time object such as car, table, dog, monitor etc. The image processing detect edges through texture color or shape. The real time object detection there are one and multiple object detect at a time

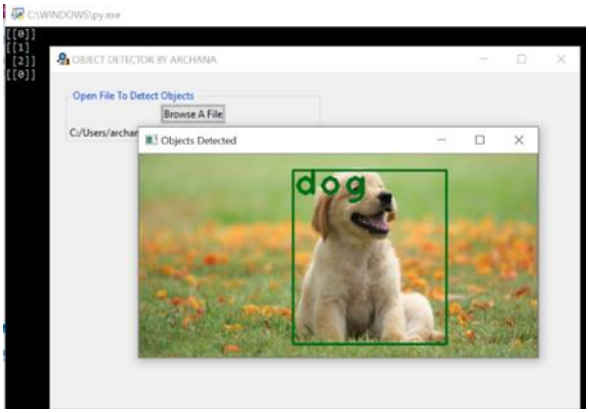


Figure 8 Result single object detect

In this fig 5.1 object detection application the object can be detect in the picture the object is dog the bounding box will be appearance in the object located. Object localization then allows us to locate our object in the image Object detection provides the tools for doing just that finding all the objects in an image and drawing the bounding boxes around them.

Figure 9 Result of multiple object detect

The real time object detect fig 9 image multiple object detect with bounding box each object have separate bounding box and it detect all the object with can clearly visible in the image.

10 CONCLUSIONS

This article was able to show you the ease of creating beautiful apps using the Python language Tkinter. that making apps in Python is easy. So, some cool ideas, and turn them into reality with.

The different model helps different detection technology there are many object detection application in future we will create next level technology with the use of the dataset and OpenCV. It is crucial to get the right resources to increase your knowledge.

Object detection is an application which can powerful to use identify the object by image processing 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.

The object detection is real time object detected and identify with the help yolo algorithm. It will be different way we can use detection. We use multiple libraries mainly library for this project like OpenCV TensorFlow for object detection.

In future the technology has must more advance compare the at the time. The good to learn how will be using different way very helpful and very good practices.

REFERENCES

[1] Bodla, N., Singh, B., Chellappa, R., & Davis L. S. (2017). SoftNMS improving object detection with one line of code. In ICCV (pp. 5562–5570).

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: unified, real-time object detection,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788, Las Vegan, NV, USA, 2016.

[3] Apoorva Raghunandan, Mohana, Pakala Raghav and H. V. Ravish Aradhya, Object Detection Algorithms for video surveillance applications international conference on communication and signal processing (ICCSP), India, 2018, pp. 0570-0575.

[4] Joseph Redmon, Ali Farhadi, YOLOv3: An Incremental Improvement, University of Washington.

[5] Karlijn Alderliesten, YOLOv3 Real-time object detection, May 28, 2020.

[6] Arka Prava Jana, Abhiraj Biswas, Mohana, YOLO based Detection and Classification of Objects in video records 2018 IEEE International Conference on Recent Trends in Electronics Information Communication Technology, (RTEICT) 2018, India.

[7] Viraf, Master the COCO Dataset for Semantic Image Segmentation, May 2020.
[8] J. Redmon. Darknet: Open-source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.
[9] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. arXiv preprint arXiv:1612.08242, 2016.
[10] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.
[11] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. Yfcc100m: The new data in multimedia research. Communications of the ACM, 59(2):64–73, 2016
[12] Bhumika Gupta, Ashish Chaube, Ashish Negi, Umang Goel, "Study on Object Detection using Open CV Python", International Journal of Computer Applications Foundation of Computer Science (FCS), NY, USA, Volume 162, Number 8, 2017.
[13] Geethapriya. S, N. Duraimurugan, S.P. Chokkalingam, "Real-Time Object Detection with Yolo", International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-8, Issue-3S, February 2019.
[14] https://blog.paperspace.com/how-to-implement-a-yolo-object-detector-in-pytorch/
[15] Create COCO Annotations from Scratch — Immersive Limit
[16] YOLO v3 Object Detection with Keras | by Christie Natashia Archie | Towards Data Science
[17] Nikhil Yadav, Utkarsh, "Comparative Study of Object Detection Algorithms", IRJET, 2017.
[18] Viraf, "Master the COCO Dataset for Semantic Image Segmentation", May 2020. [
[19] Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement", University of Washington.
[20] Karlijn Alderliesten, "YOLOv3 — Real-time object detection", May 28, 2020.