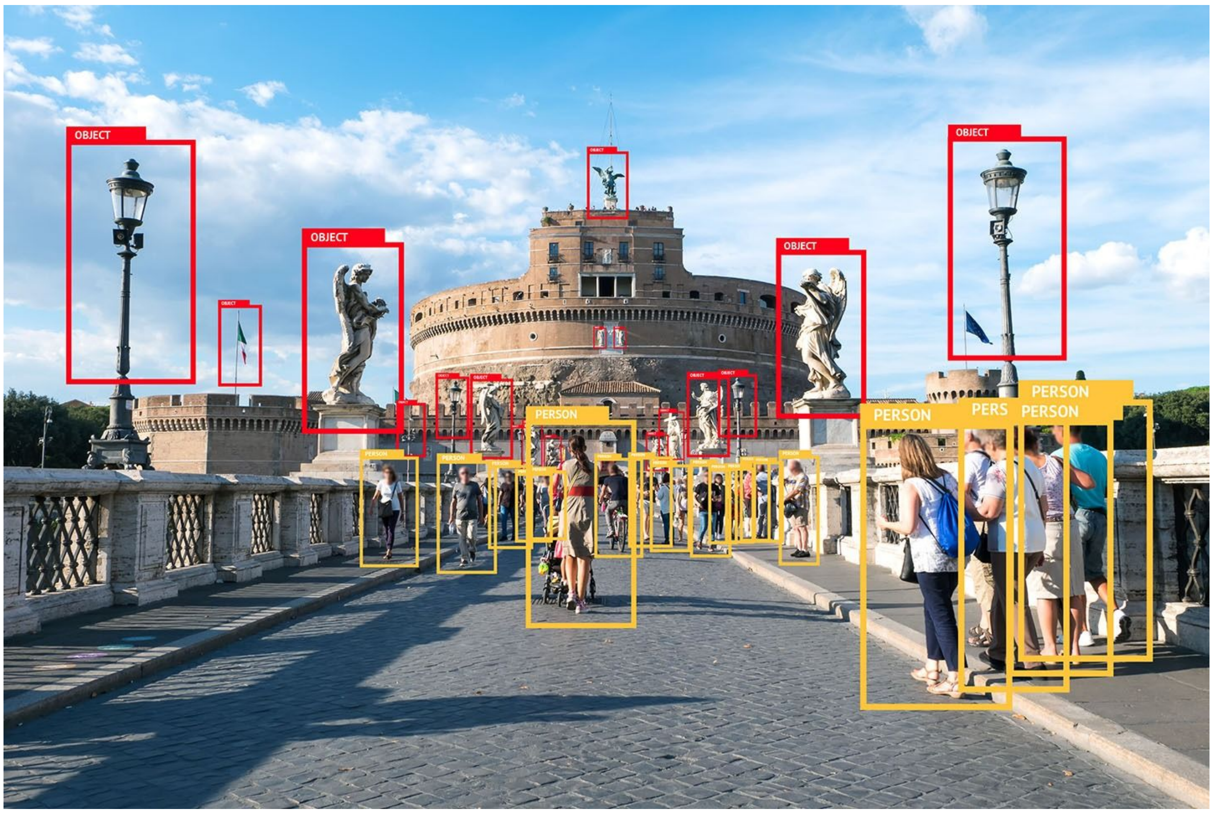
Analysis of Algorithm Final Project

Real time Object Recognition with mobilenetSSD



Submitted to Mr. Tri Asih Budiono

Submitted by

Christensen Mario Frans - 2301963316

Jeconiah Richard - 2301947905

Muhammad Lukman Ismail Hanafi - 2301929493

Sunny Jovita - 2301939046

Odd Semester 2020/2021

1.1. **Introduction**

As we are moving forward in time, technology advancement is growing rapidly. One of the huge breakthroughs in the technology world occurred when the Neural Network arose. It was firstly initiated by **Bernard Windrow** and **Marcian Hoff** of Stanford developed models they called **ADALINE** and **MADALINE**. MADALINE (Multiple ADAptive LINear Elements) was the first neural network to be applied to real world problems. It is an adaptive filter which eliminates echoes on phone lines and is still in commercial use.

Neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. With python, it is possible for us to make a computer see what humans can see and apply Neural Network to make the computer recognize the objects inside the frame. Further explanation on how the program works will happen in the next section.

1.2. **Problem description**

Computers have a lot of magnificent functions that makes human life and work easier. With the additional vision ability given to computers, it will enable the computer to reach a wider scope of access. In our project, we use computer vision to detect objects that appear in the given frame by using OpenCV (Python library) and Caffe implementation of Google MobileNet Single Shot multibox Detection (SSD) to recognize them.

With this program, computers are now able to assist surveillance systems to detect humans in each location or can be advanced further to look up a person's identity from a scene with TensorFlow and Keras. We have come up to the extent where the computer is able to detect humans in the frame and other 21 objects listed in the Caffe Model. This program is very possible to be further advanced, however, more computer power is required to do so. Here is the architecture of Single Shot Multibox that will be used in the program.

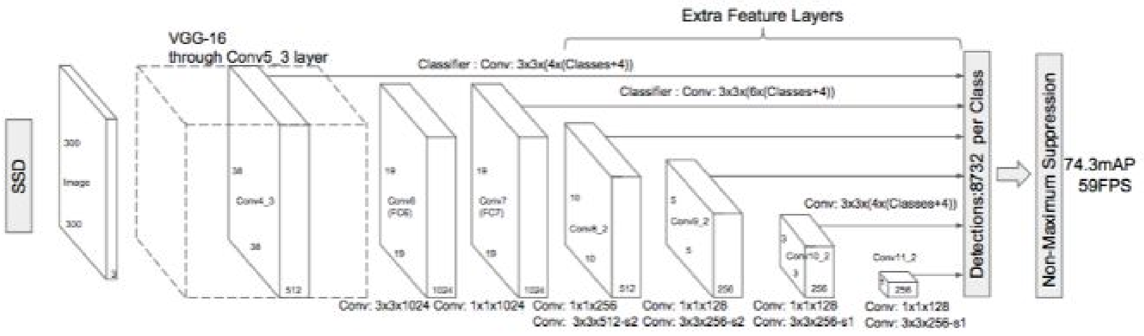


Figure 1.1. Single Shot Multibox architecture

2.1. **Related Work**

In the process of making this project, there are several libraries that are used to make this program work. As mentioned in Section 1.2., OpenCV and MobileNet SSD are part of them. Here is the list of libraries that are used in the program:

a. OpenCV library is commonly known as opencv-python and is imported to the code as cv2. cv2 is the latest version of opencv and can be accessed by downloading the opencv-python package.
b. Numpy library is commonly used in python to make 2d arrays and matrices. This library is crucial in resizing the frame of the input and array making in the program.

NOTE: Each library is needed for the program to run and must be set as project interpreter. Python 3+ is needed.

3.1. **Formal description of Computational Problem**

This Analysis of Algorithm project has shown us that there is a difference between human-like images and computer-like images. In other words, how humans see images/scenery with our eyes is not preferable for computers to see them. Computer sees and processes images/scenes better through binary numbers in a 2D array, therefore, in our project we convert the frames from the video into Binary Large OBject (BLOB) to make the computer process it better.

Binary Large OBject is a collection of binary data stored in a single entity in a database management system. BLOB are typically images, audio or other multimedia objects. The image in this case needs to be converted into BLOB since the input for MobileNet SSD is in a BLOB format. After turning the image into BLOB, then it will be passed through the model and will return the prediction both on the frame and on the console.

3.2. **Design of Algorithm**

This program mainly uses python as the programming language with two additional Python libraries, OpenCV and Numpy.

OpenCV is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez.

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

In this project, we firstly take a video input with openCV from the device's webcam (if any) or any camera connected to the computer. The video will be broken down into frames that will be converted into Binary Large Object (BLOB) before the computer processes it further. Then, for each frame that has been converted into BLOB will be forwarded to MobileNet SSD as an input and will return some values like, Object name index, Detection confidence and X and Y location of the image in the frame. These values will act as the output of this program. The object name index will be displayed on the terminal console and on the frame (in real time) with its confidence next to it. The X and Y coordinates will be used to draw the bounding box of the image on the frame (in real time). A while loop will be inserted in the code to ensure the program repeats until a certain key is pressed.

3.3. **Complexity analysis for the algorithms and data structures involved**

For the complexity of this program, we as a sophomore would classify this project as a quite complex program. This project exposes us to a new data structure conversion from a media/image file into a Binary Large Object (BLOB). Furthermore, exposure of OpenCV libraries pushes us to learn new python functions like VideoCapture and Imshow. Even though this program uses many open-source libraries and models, connecting the dots to make the program work is quite a challenge.

4.1. **Implementation details**

To make this program work, users are required to have the main two python libraries which are OpenCV and Numpy.

After importing both packages, the user must have other files consisting of the model and its weight. The files are named *mobilenetssd.prototype* and *mobilenetssd.caffemodel*. In the program (line 25), the relative path of both files is mentioned. These relative paths must be replaced with the user's path to the files mentioned before.

NOTE: Both relative path and absolute path works fine.

The input is modifiable. Users may use a real-time video by using camera or webcam from the device or pre-saved video from the device. The input can be changed in *line 6* as follows.

- cv2.VideoCapture(0) is to set in-build webcam as input. If a user has more than 1 camera input in the device, the user may consider replacing number 0 to 1.

- cv2.VideoCapture("path") is to set the pre-saved video as the input. This works completely the same with the real-time video in code.

After changing these things, users may run the program with no error.

4.2. **Dataset**

In this project, we mainly use MobileNet SSD as the model to recognize the images. Every model is needed to be trained in order to have the ability to recognize images, differentiating one with another. The Pre-trained MobileNet SSD model was trained using a VOC0712 dataset with 11530 samples of images. These labeled images are used to further create a model that can recognize 20 objects which are; aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, sofa, train, tv monitor and one additional object, background.

4.3. **Experimental Analysis and Result**

This is the result of running the program in real time. Since the program is only capable of recognizing 21 objects that were mentioned before, the other objects may not be surrounded by any bounding box nor object prediction label.

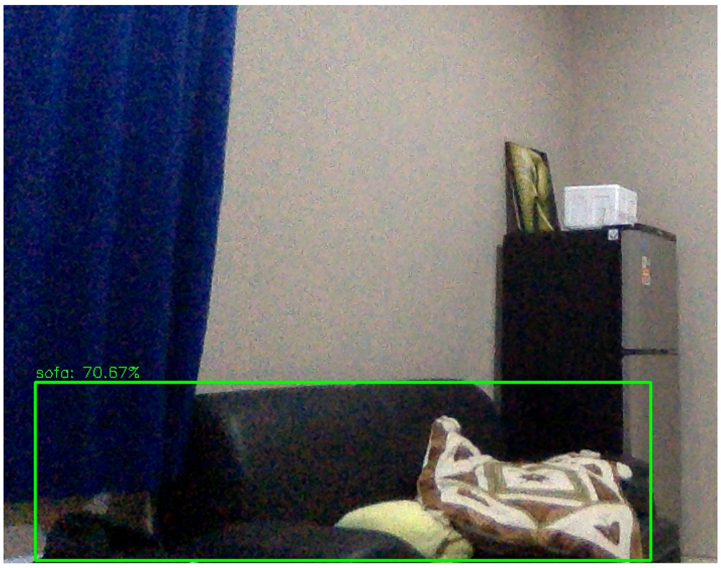


Figure 2.1. Detected Sofa

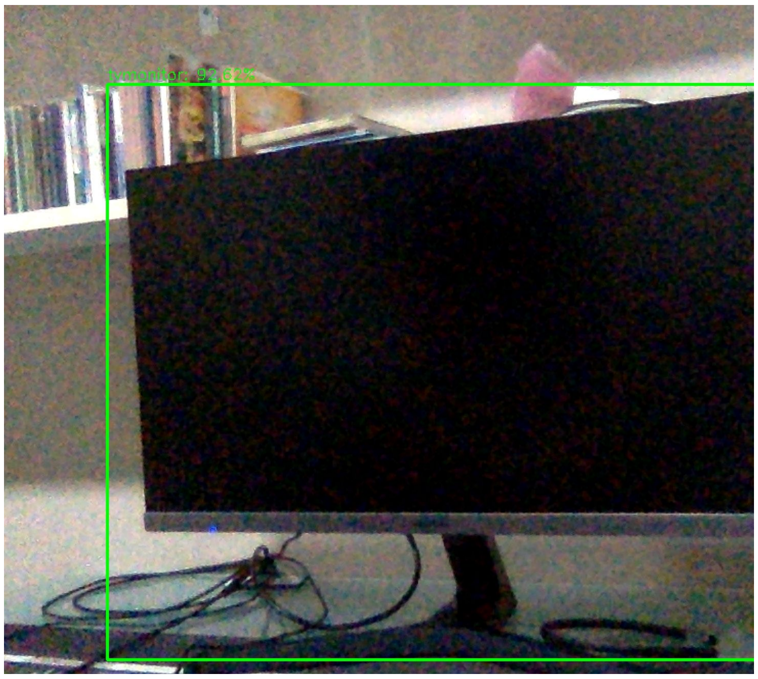


Figure 2.2. Detected TV Monitor

5.1. **Discussion (FAQ)**

Q: Why do you pick this topic over the other?

A: We as a team believe that the Neural Network as a big part of Artificial Intelligence and Machine Learning will involve a big role in the future. We would like to expose ourselves to this technology to keep ourselves up with technology advancements.

Q: Can any device run this program?

A: Yes. The program is runnable in any device as long as it has Python 3+ as its language and a decent CPU (Intel i3+ or Ryzen 3+) and GPU. Note that the better the CPU and GPU, the better and faster the program will run, and vice versa. Webcam or any camera as input is required.

Q: Is there a better object recognition library or program you did not pick?

A: Yes. Caffe implementation of MobileNet SSD is one of the earliest in the market. Other models like You Only Look Once (YOLO) and Mask RCNN are available off the shelf, however it requires much more computing power to run it. As a quick comparison to run this program with a different model on MacBook Pro 2017,

- MobileNet SSD takes 59 FPS
- YOLO takes 2 seconds/frame
- MaskRCNN takes 4 seconds/frame

These differences occurred due to the number of objects one model can identify.

- MobileNet SSD is able to recognize 21 Objects
- YOLO is able to recognize 80+ Objects
- MaskRCNN is able to recognize 91+ Objects

Training a model would require much time to do so. Users have to label every image from a dataset and train with Deep Learning. A decent number of images will be required to make the computer able to detect such objects. As a quick example, it takes 2000 images of each dog and cat to train a machine to differ between dogs and cats with 50% accuracy (with TensorFlow and Keras).

6.1. **Conclusion and recommendation**

After finishing this project, we conclude that programming revolves around data input and output. We firstly must understand how to get the data input, process it and have a valuable and meaningful output through the programming algorithm. Our Computer Vision program in this case relies on OpenCV as both the data input and output. VideoCapture and Imshow are both functions of cv2 and are used to take the input and show the output. In the middle part of the programming, cv2 works as the function to create bounding boxes in the frame to show the recognized objects.

We have learned that understanding how the program works is the most important thing to be worked on. Other than that, the algorithm is used in any program, yet its complexity differs. This leaves us with the question whether our algorithm in the program is efficient or not, and that is why Analysis of Algorithm class is important for a Computer Science student.

7.1. **Program manual**

Disclaimer: In order to run this program, you need Python 3+ with the following libraries: Numpy and OpenCV (opencv-python) and camera or webcam to run real-time video recognition.

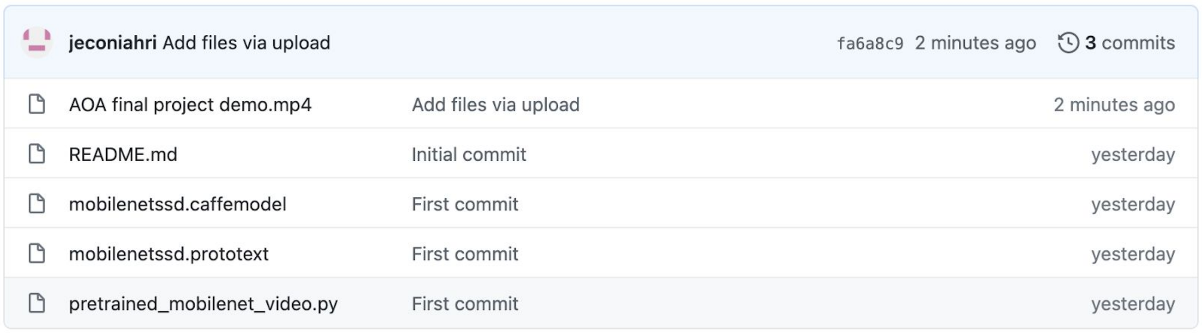
First step, you have to have the following files that can be found in Github.

- Pretrained_mobilenetssd_video.py
- Mobilenetssd.prototext
- Mobilenetssd.caffemodel
- Project demo video

Github link: https://www.github.com/jeconiahri/mobilenet

Note: If in any case, the link to the video, or the video itself does not work as expected, please kindly contact one of our group members for assistance

NOTE: Users are recommended to put these three (3) files in the same directory in order to have a simpler file path to link one file to the other.

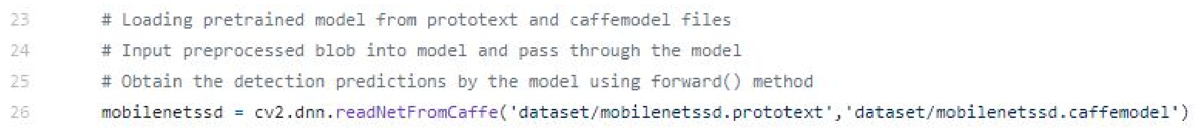


jeconiahri Add files via upload fa6a8c9 2 minutes ago **3** commits

AOA final project demo.mp4	Add files via upload	2 minutes ago
README.md	Initial commit	yesterday
mobilenetssd.caffemodel	First commit	yesterday
mobilenetssd.prototext	First commit	yesterday
pretrained_mobilenet_video.py	First commit	yesterday

Figure 3.1. Files on github

Next step, the user must change the directory of the *mobilenetssd.caffemodel* and *mobilenetssd.prototext* in **line 26** with the user's file path to both files.



```
23    # Loading pretrained model from prototext and caffemodel files
24    # Input preprocessed blob into model and pass through the model
25    # Obtain the detection predictions by the model using forward() method
26    mobilenetssd = cv2.dnn.readNetFromCaffe('dataset/mobilenetssd.prototext','dataset/mobilenetssd.caffemodel')
```

Figure 3.2. Line 23 to Line 26 of pretrained_mobilenet_video.py

The final step before running the program, the user has to put Numpy and OpenCV as Project interpreter. If user does not have the library, both Numpy and OpenCV is downloadable through pip with the following command:

- Pip install numpy
- Pip install opencv-python

Sources:

http://www2.psych.utoronto.ca/users/reingold/courses/ai/cache/neural4.html#:~:text=In%201959%2C%20Bernard%20Widrow%20and,to%20a%20real%20world%20problem.

https://www.investopedia.com/terms/n/neuralnetwork.asp

https://jonathan-hui.medium.com/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359

https://pjreddie.com/darknet/yolov2/

https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11