Real-Time Detection And Classification of Traffic Signs

Team Members - Aryan Garg & Kartik Aggarwal (IIT Jodhpur)

Model's Working-

The presented model has trained on the **YOLO(Version 5)** object detection algorithm, which was released by **Ultralytics** in June 2020 and currently being state of the art among other algorithms.
YOLO(You only look once) is a creative convolutional neural network (CNN) for doing object detection in real-time with high accuracy. This algorithm applies a single neural network to the full image, and then divides the image into regions and predicts bounding boxes and probabilities for each part. The predicted probabilities weight these bounding boxes. The algorithm "you only look once" at the image in the sense that it requires only one forward propagation pass through the neural network to make predictions. After non-max suppression (which makes sure that the object detection algorithm only detects each object once), it then outputs recognized objects.

Capabilities-

1. Real-Time detection and classification upto 45 fps
2. Accuracy as high as 98%
3. Robust model - detects in all weather conditions
4. Trained on 61 classes
5. Able to detect a wide variety of signs (for ex. Indian, European, American traffic signs)

200m, 50-100m, Ahead-Left, Ahead-Right, Axle-load-limit, Barrier Ahead, Cattle, Compulsory Ahead, Compulsory Keep Left, Compulsory Left Turn, Compulsory Right Turn, Cross Road, Cycle Crossing, Compulsory Cycle Track, Cycle Prohibited, Dangerous Dip, Falling Rocks, Gap in median, Give way, Horn prohibited, Humpy Road, Left hair pin bend, Left hand curve, Left Reverse Bend, Left turn prohibited, Major road ahead, Men at work, Motor vehicles prohibited, Narrow bridge, Narrow road ahead, Straight prohibited, No parking, No stopping, One way sign, Overtaking prohibited, Pedestrian crossing, Pedestrian prohibited, Restriction ends sign, Right hair pin bend, Right hand curve, Right Reverse Bend, Right turn prohibited, Road wideness ahead, Roundabout, School ahead, Slippery road, Compulsory sound horn, Speed limit, Staggered intersection, Steep ascent, Steep descent, Stop, Truck prohibited, Compulsory turn left ahead, Compulsory right turn ahead, T-intersection, U-turn prohibited, Y-intersection, No entry, Compulsory Keep Right, Parking

Specifications-

YOLOv5 is released in four different versions namely- YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x. We have implemented our model using **YOLOv5s** algorithm because it is fast as well accurate enough to perform real time classification. We have collected our dataset from sources like Kaggle, it comprises a total of **2796 images** in which **61 different classes of traffic signs** are present. **2269** of these images are used for training purposes and the remaining **527** images are used for testing and

validation purposes. All the Traffic Signs are labelled by hand using the software named LabelImg. Firstly, all the images are converted into **640x640x3 pixels**(3 represents the RGB channels) and after performing some normalization and pre-processing tasks these images are fed into the model for training. Model is trained for **800 epochs** and achieves **mAP@0.5**(Mean Average Precision) equal to **91.75%** & **mAP@0.5:0.95** equal to **74.08%** .Some other parameters like Precision, Recall, Objectness are also shown in **Fig.1.**

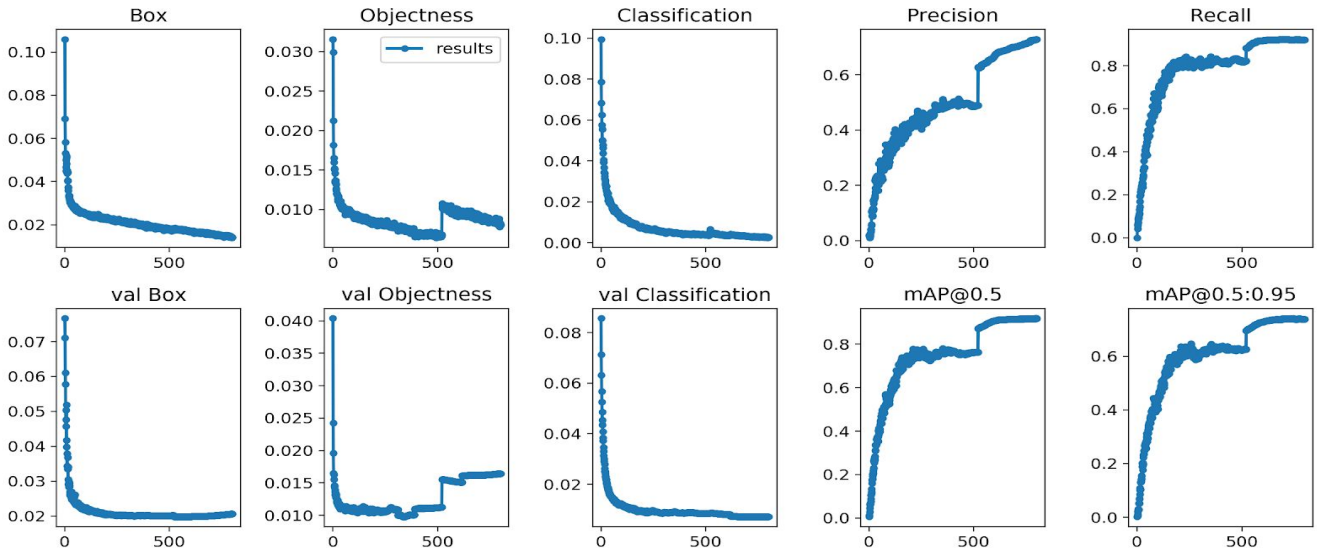

Fig.1- Showing the graphs for mAP, Precision, Recall, etc

Directions to Run The Code-

1) Open the main directory named RecogniSign in your command prompt.
2) Firstly install all the required dependencies using the requirements.txt file-
   If you are using windows machine:
   Type - pip install -r requirements.txt in your cmd.
   Or if you are using anaconda prompt-
   Type - conda install --file requirements.txt
3) Then go to the directory named Codes using the command-
   Type - cd Codes
4) Before running the next command, put your test file in the directory named Test.
   Type - python detect.py --source ../Test/{name of your file}

   For eg- If your image name is test.jpg, then put your image in the directory named Test and then run python detect.py --source ../Test/test.jpg
5) Above same step can also be used to test the model on video files.
6) For running on WebCam run python detect.py --source 0

7) A pop up window appears as shown below in **Fig.2**, showing the bounding boxes of all Traffic signs present in that frame with their label and accuracy score. FPS for each frame is also shown in the top-left corner of that window.
8) Also, the output results are saved in the directory named Results.

Directory Structure-

```
RecogniSign
├──Codes
│   ├──models
│   │   ├──hub
│   │   └──__pycache__
│   └──utils
│       ├──google_app_engine
│       └──__pycache__
├──Model
│   └──weights
├──Results
├──Sample Dataset
└──Test
```

Experimental Setups-

The training and testing of the proposed classification & detection system for RecogniSign challenge was implemented using Python 3.8 programming language with a processor of IntelR Core i5- 10210U CPU @ 2.30GHz × 8 and RAM of 8GB running on Windows 10 with NVIDIA Geforce MX 250 with 2GB Graphics.

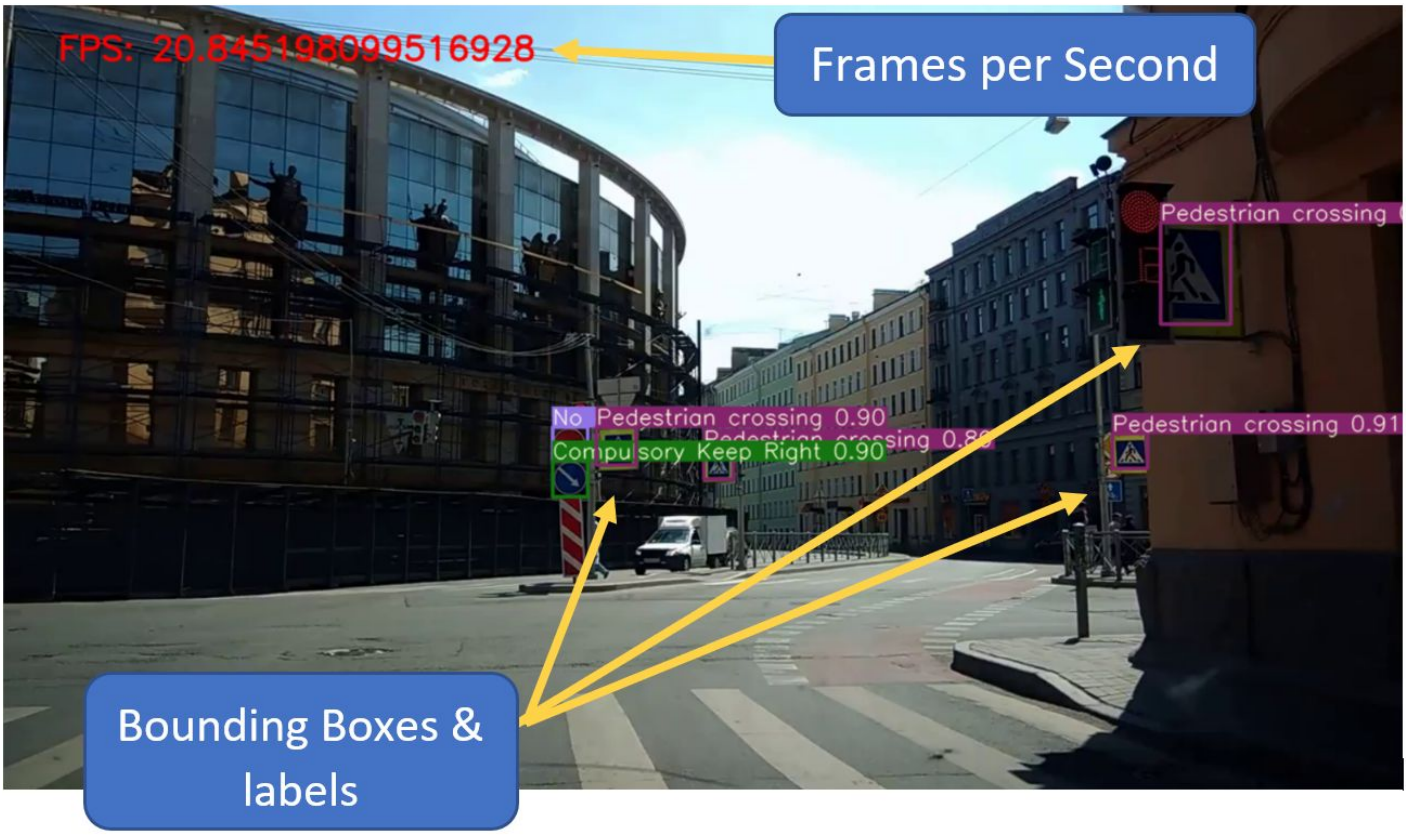

Fig.2: Showing the output results of a frame

References-

1) Github link to the Ultralytics repository- https://github.com/ultralytics/yolov5
2) Kaggle link for some images of the dataset- https://www.kaggle.com/andrewmvd/road-sign-detection
3) Original paper of YOLO released in May, 2016- https://arxiv.org/pdf/1506.02640.pdf
4) Github link to the LabelImg repository- https://github.com/tzutalin/labelImg