RESEARCH ARTICLE OPEN ACCESS

Classification and Identification of Traffic Signals, Symbols and Text Using Deep Learning and AI

Kirpalsinh Jaymalsinh Raj
(Master of Engineering in Computer Engineering [Software Engineering], Gujarat Technological University, Ahmedabad
Ipcowala Institute of Engineering & Technology, Dharmaj)
Email: rajkirpalsinh HYPERLINK "mailto:rajkirpalsinh@gmail.com"@gmail.com
Contact No. : +91 8160190160
Supported By : Kajal Patel, Premal J. Patel

-------------------------------*************************-----------------------------

Abstract:

To travel at any place one needs to be aware of the route. Route is an essential part for any journey. The traveler has to be familiar with the route. The start of route, exit of route and the traffic signals coming along the way all are playing a vital role in any journey. Whether on the route road the traffic signals and symbols are appropriately installed on not. At some turning point the warning sign boards need to be installed. In the majority of cases when we travel to a known place we ignore the traffic signs and warnings. But, whenever we have to travel to an unknown place then the traffic sign and warnings help us a lot to reach our destination. Traffic signs and symbols are very essential. In today's rapidly developed era of technologies the roads are also well prepared with the appropriate traffic signals and symbols at the required places. Here we have concentrated on development of a system which detects traffic signs, symbols and text.

Keywords — **YOLO, YOLO7, CNN, DARKNET, Traffic Sign Detection, Python, Google Colab**

--------------------------------*************************----------------------------

1. INTRODUCTION

Technology growth is very rapid nowadays. And we come in use of many new technologies on a regular basis. Artificial Intelligence is widely used in the area of Computer Science. Nowadays Artificial Intelligence has grown so high that it is very helpful in building an Intelligent Machine. AI is getting used in many different fields. Such as Self-Driving Cars, Games, Social Media, Music, Home appliances, Restaurants,Drawings etc. Automobiles that drive themselves and do not require human drivers to control them. “Autonomous” or “Driverless” cars are made with the installation of appropriate sensors devices and prerequisite softwares to navigate and control the vehicles. Driverless vehicles (Cars) are performing many automatic jobs. But as per my thesis, Self Driving Car or Driverless cars Detects and Classifies Traffic Symbols, Signs and Text. And here the Classification operation and Detection operation is getting applied using AI and Deep Learning. In the classification process the categorization of the things are done based on the specific properties. Discovering something essential comes in the process of Detection. Traffic symbols / Road Sign are on the road side that have information for vehicles drivers and Deep Learning is the technique based on Machine Learning that tries to mimic the human mind working system. So, based on this deep learning approach my topic signifies that I will detect the traffic symbol and classify according to their respective properties. Properties like their shape and color.

Recognition of traffic symbols and text is an essential task in Automated Cars. It has two stages : First is based on real traffic sce, finding the traffic symbol location & the second is categorization of detected traffic symbols and text into their categories. Categories like, Guide sign, Construction related Signals or

Symbols, Recreation Signals or Symbols, Regulatory Signs, Services Symbols or Signs, Warning Sign, sign, Incident Management sign, School Zone Signs etc

(1.1) Overview of Traffic signal, symbol and Text detection and classification based on deep learning technique and artificial intelligence:

In road transportation systems, traffic signs provide useful and vital information and instructions to road users. Depending on the traffic situation and environmental conditions, speed limits vary, informative symbols, signs, and signs or directions are placed on the road sides to regulate traffic safety. In order to monitor and manage traffic signs, it is essential for transportation agencies to update them rapidly. The most common method for detecting and recognizing traffic signs is through the use of digital images and videos. A complex environment can easily be distinguished from traffic signs that are created by highly contrasting colors (e.g., white, yellow, red, blue, white). The major traffic signs getting used in transport systems have circular, triangular,, rectangular, square, octagonal and pentagonal types shapes. Categories of traffic-sign language, such as prohibition, danger, obligation, and warning, are represented by the combination of forms and colors. Therefore, traffic signs are typically described by a variety of feature descriptors, including the following, based on picture features generated from color and shape information.: wavelet decomposition, local binary pattern, scaling-invariant feature transform, and directed gradient histograms, along with associated Fourier descriptors. The attributes have typically been applied to a number of classifiers and object-classification algorithms to create video or image based traffic symbols detection and recognition systems, such as the ones listed below.: neural network, support vector machine, random forest, decision fusion and reasoning module, convolutional neural network (CNN) and template-matching-based methods. However, the following restrictions apply to these image or video-based systems: 1) Fog and rain, for example, might make it difficult to see traffic signals. 2) shadows, brought on by nearby objects or varying levels of illumination, 3) light condition, and 4) color fading, text distortion, and sign similarity are all examples of signal and text condition. The task of detecting traffic signals and symbols has shown to be dependable and efficient.

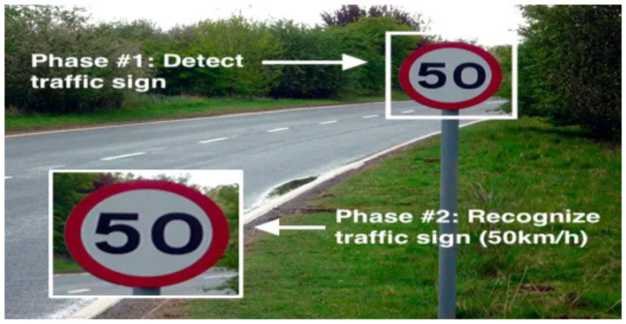

Phase #1: Detect traffic sign
50
50
Phase #2: Recognize traffic sign (50km/h)

Figure-1.1: Detected Traffic Symbols

Traffic sign classification and detection are carried out in this image. Here, we can see that first the traffic signs have been identified and in the phase2 it detects the internal information of the traffic sign. Images of the traffic signs labeled with Numeric value, Speed Limit are shown here. The traffic sign's circle is enclosed by a digit. This boundary of the circle box both locates and detects the location of the specific traffic sign.

(1.2) Deep Learning Introduction:

Computer science subfields known as machine learning (ML), natural language processing (NLP), and artificial intelligence (AI) enable computers to automatically understand intricate patterns from empirical data. The technologies enable intelligent decision-making in assistance of human specialists, particularly in fields where there is a high degree of ambiguity, based on learned behavior.

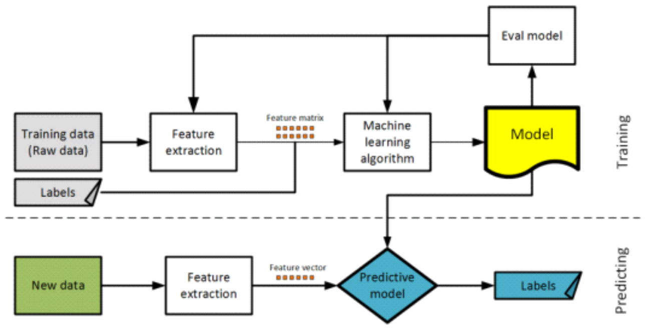


Figure-1.2: Deep Learning Process

(1.3): Objectives

Objective Finding the positions and dimensions of traffic signs in landscape photos is the goal of traffic sign detection.

The two primary elements for traffic sign detection are colors and forms.

As a result, there are two sorts of detection techniques: shape-based and color-based. Classifying observed traffic signs into their distinct subcategories is the goal of traffic sign categorization.

Automatically identifying roadside traffic indicators, such as yield, merge, and speed limit signs, is known as traffic sign classification. Independent of how they appear, traffic signs in the image should be detectable by the system.

Because of this, it ought to be required that distortion of perspective, the lighting modifies Barricades in parts and shadows.

Additionally, it offers details regarding the existence of potential issues: Bad placement, poor condition, and lack of visibility.

We can create a "smart car" if it has the ability to recognise traffic signs automatically

(1.4): Motivation

The mechanism or system or combination of words that rejuvenate the thoughts inside one individual that inspires a human to start, direct, and sustain a goal-oriented activity is called motivation.

Automatic Systems are popular right now. For instance, automatic systems like smart security, locks, and intelligent lighting are employed in homes. Automatic systems come under the process of “AUTOMATION”.

A technology known as "AUTOMATION" is utilized to execute jobs with little to no interference from humans.

This method is used to automate the operation of devices, systems, or processes. And the latest craze is AUTOMATION, from homes, smartphones to vehicles and more.

The primary mechanism that transports travelers to their destinations is called the ROUTE.

We might disregard the traffic signs when we travel to a familiar location. But Suddenly Traffic Sign turns into our closest companions when we wish to visit somewhere fresh.

A future without traffic signs is difficult to envision.
Finding boards that have been allocated with traffic signs is the main objective.

I want to develop a traffic sign recognizer that automatically alerts motorists to approaching traffic signs so that they can obey them.

2. WRITING STUDY [LITERATURE SURVEY]

PAPER-1: Real-Time Detection Method for Small Traffic Signs Based on Yolov3

Publication Year: 2020

Author: Huibing Zhang , Longfei Qin, Jun Li, Yunchuan Guo, Ya Zhou, Jingwei Zhang

Journal Name: IEEE

Summary: With respect to driver-assistance systems for driving automobiles and autonomous systems, it is particularly difficult to recognise traffic signals, Symbols and text using a high-precision real-time technique in realistic settings. In order to successfully localize and classify minor traffic signals in real time, a new detection approach (called MSA YOLOv3) is suggested in this study to meet this difficulty. First, picture mixup technology is used to enhance data. The Darknet53 network is then given a multi-scale spatial pyramid pooling block to help it more thoroughly learn object properties. At Last a bottom up augmented path is created to for the improvement of the feature pyramid in YOLOv3 and obtain precise object localisation by making efficient use of fine-grained features in the lower levels. The proposed MSA YOLOv3 performs better than YOLOv3 in detecting minor traffic signs, As per the testing based on the TT100K dataset (a dataset for traffic sign detection). The (mean Average Precision)mAP of MSA YOLOv3 may reach up to 86%, and the detection speed is 23.81 FPS.

PAPER-2: Machine Vision Based Traffic Sign Detection Methods

Publication Year: 2020

Author: Chunsheng Liu, Faliang Chang, Shuang Li, Yinhai Wang

Journal Name: IEEE

Summary: An Important component of the various advanced driver assistance systems (ADASs) and auto driving systems is traffic sign recognition (TSR) (ADSs). Traffic sign detection (TSD), the first crucial phase of TSR, is difficult and very challenging due to a variety of types, small sizes, complex driving scenarios, and occlusions. There have been many TSD algorithms based on machine vision and pattern recognition in recent years. This report presents a thorough analysis of the TSD literature. We categorize the reviewed detection techniques into five primary groups: LIDAR-based, color-based, shape-based, color-and-shape-based, and machine-learning-based. To comprehend and summarize the mechanisms of various approaches, the methods in each category are further divided into many subcategories. We reimplemented a portion of several of the methods we studied in order to compare them with those that have comparisons on open datasets. In the reported performance and in our reimplemented approaches, experimental comparisons scenarios and analyses are provided. In order to advance the development of the TSD, future directions and suggestions for TSD research are also provided.

PAPER-3: Robust Classification of Traffic Signs using MRMR Feature Reduction Technique and Optical Character Recognition

Publication Year: 2021

Author: Manisha Vashisht, Brijesh Kumar

Journal Name: IEEE

Summary: Driver safety has become a major worry for humanity as a result of increasing motor traffic on the roads and a growing worldwide population. The study of traffic sign recognition (TSR), which is important for both the idea of autonomous vehicles and road safety, has seen an increase in interest. The development of IOT and related artificial intelligence technologies has given research the much-needed impetus. Modern automobiles are equipped with sophisticated technologies that can read the roadside traffic signs and either indicate to the driver the best course of action to take, or even be set up to carry out the action automatically. Despite the tremendous efforts of experts in the past, most automakers and drivers still find it difficult to predict traffic signs accurately, especially in poor weather conditions. The authors of this study have put forth a novel method for recognising traffic signs from images via dimensionality reduction utilizing the maximum relevance and minimum redundancy (MRMR) algorithm in conjunction with artificial neural networks (ANNs) for classification. When compared to earlier studies in the field, the results obtained were deemed to be positive.

PAPER-4: Real-Time Traffic Sign Detection and Classification Using Machine Learning

Publication Year: 2020

Author: Victor Ciuntu, Hasan Ferdowsi

Journal Name: IEEE

Summary: Traffic sign identification and classification is playing a significant part in the rapid advancement of autonomous vehicle technology. This study examines a few potential methods for carrying out this operation in real-time using a portable machine. The final approach combines an optical character recognition technique for speed limit signs with a convolutional neural network for detection and classification. The Belgian Dataset, German Dataset, and photos captured while driving in Illinois, United States, serve as the foundation for the training and testing datasets.

PAPER-5: Autonomous Traffic Signs (ATSR) Detection and Recognition using Deep CNN

Publication Year : 2020

Author: Ghazanfar, Danyah A., Alghmghama, Latifab, Jaafar Alghazoa, Loay Alzubaidi

Journal Name: Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Malaysia

Summary: Automatic traffic sign detection and recognition is crucial and has the potential to be employed in driver assistance systems to prevent accidents as well as eventually driverless cars. In this study, an autonomous traffic and road sign (ATRS) detection and identification system is created using deep convolutional neural networks (CNN). The suggested system detects and recognises photos of traffic signs in real time. The addition of this article also includes a freshly created database of 24 different traffic signs that were gathered from Saudi Arabian roadside randomness. The images were taken from different perspectives and included additional conditions and circumstances. To create the database, which we called Saudi Arabian Traffic and Road Signs, a total of 2718 photos were gathered (SA-TRS-2018). To get the highest recognition rates, different parameters were applied to the CNN architecture. Experimental results demonstrate that the suggested CNN architecture attained 100% accuracy, exceeding that of similar earlier studies.

PAPER-6 : Traffic Sign Detection And Classification Using DL

Publication Year : 2020

Author : Shanthi.S, Sunitha.

Journal Name: International Journal Of Innovative Research In Technology

Summary: The primary objective of the research is to investigate the classifier algorithms for identifying and categorizing traffic indications signs and Text. As a categorization and recognition method, either the information about the color or shape of the traffic sign is utilized. The image quality in actual traffic situations is often poor due to the low resolution, condition of wheather, varying illumination, motion blur, occlusion, rotation, scale and other respective issues. Additionally, traffic signs frequently have a variety of appearances, a high degree of similarity, and intricate backgrounds. In order to appropriately incorporate shape and color information, both the detection and classification phases must pay much more attention. A CNN-SVM-based recognition and classification approach has been proposed in this study. Using the YCbCr color scheme, CNN extracts the deep image features during training. SVM is connected to the last layer of CNN for further classification, which enhances training results. However, during testing, a variety of image preprocessing techniques are utilized to eliminate those negative impacts, including poor illumination, partial occlusion, and considerable deformation. Experimental comparison with other state-of-the-art methods shows that our model outperforms the competition in terms of training speed and accuracy. Additionally, we have found that some of the traffic symbols and text are misidentified in the autonomous ground vehicles when we employ this method. In the near future, we plan to increase the quantity of traffic sign photographs in our data collection, especially those shot at night. The algorithm will subsequently be improved for usage in real-time in autos, which will speed up the process.

PAPER-7:Traffic Sign Detection and Classification for Indian Traffic Signals Using DL

Publication Year: 2021

Author: Manjiri Bichkar, Sonal Chaudhari, Suyasha Bobhate,

Journal Name: International Journal of Scientific Research in Computer Science, Engineering and Information Technology Summary: In order to detect traffic signs on roads, this paper first classifies images of traffic signs using a convolutional neural network (CNN) on the German Traffic Sign Recognition Benchmark (GTSRB) before detecting images of Indian traffic signs using the Indian Dataset, which will be used as a testing dataset when developing a classification model. Therefore, this method aids in the accurate and efficient recognition of traffic signs by electric or self-driving vehicles. The system consists of two components: the first detects traffic signals in the environment, and the second classifies them using CNN to identify them. The classification process entails creating a CNN model with various filters in the dimensions 3 by 3, 5 by 5, 9 by 9, 13 by 15, 15 by 19, 23 by 23, 25 by 25, and 31 by 31, from which the most effective filter is selected to classify the picture discovered further. YOLO v3–v4 and BLOB detection are used in the detection to find the traffic sign. The trained model is used to identify photos of Indian traffic signs by transfer learning

PAPER-8: Indian traffic sign detection and recognition using deep learning

Publication Year: 2021

Author: Konda reddy Thanigundala, Sreevatsava Reddy Musani, Rajesh Kannan Megalingam , Hemanth Nidamanuru, Lokesh Gadde

Journal Name: International Journal of Transportation Science and Technology Summary: Traffic signs are essential for controlling traffic, enforcing driving behavior, and preventing accidents, injuries, and fatalities. Any Intelligent Transportation System must have automatic detection and recognition of traffic signs (ITS). The need for automatic traffic sign detection and recognition in the age of self-driving cars cannot be emphasized. A deep-learning-based autonomous strategy for traffic sign cognizance in India is presented in this research. The idea for automatic traffic sign detection and recognition came from end-to-end learning using a Convolutional Neural Network (CNN) and Refined Mask R-CNN (RM R-CNN). The proposed idea was evaluated using a cutting-edge dataset made up of 6480 photos representing 7056 instances of Indian traffic signs categorized into 87 categories. We describe the number of architectures and data augmentation improvements to the Mask R-CNN model. We have taken into account extremely difficult Indian traffic sign types that haven't been discussed in other publications. Real-time photographs taken on Indian highways provide the dataset for the proposed model's training and testing. The evaluation's findings show an error rate of less than 3%. The effectiveness of RM R-CNN was also evaluated in comparison to other well-known deep neural network architectures as Mask R-CNN and Fast R-CNN Our suggested model outperformed Mask R-CNN and Faster RCNN models in terms of accuracy, achieving a precision of 97.08%

PAPER-9: Traffic Sign Classification Using Cnn And Detection Using Faster-Rcnn And YOLOv4

Publication Year: 2021

Author: Njayou Youssouf

Summary: Autonomous driving cars are becoming popular everywhere and the need for a robust traffic sign recognition system that ensures safety by recognizing traffic signs accurately and fast is increasing. In this paper, we build a CNN that can classify 43 different traffic signs from the German Traffic Sign Recognition benchmark dataset. The dataset is made up of 39,186 images for training and 12,630 for testing. Our CNN for classification is light and reached an accuracy of 99.20% with only 0.8 M parameters. It is tested also under severe conditions to prove its generalization ability. We also used Faster R–CNN and YOLOv4 networks to implement a recognition system for traffic signs. The German Traffic Sign Detection benchmark dataset was used. Faster R–CNN obtained a mean average precision (mAP) of 43.26% at 6 Frames Per Second (FPS), which is not suitable for real-time application. YOLOv4 achieved an mAP of 59.88% at 35 FPS, which is the preferred model for real-time traffic sign detection. These mAPs are obtained using Intersection Over Union of 50%. A comparative analysis is also presented between these models.

Paper-10 : Real Time Detection And Classification Of Traffic Signs Based On YOLOv3
Publication Year: 2020

Author: Sergey A. Kolyubin, Valentyn Sichkar.

Journal Name: Scientific and Technical Journal of IT Mechanics and Optics

Summary: This study discusses the problem of traffic signals and Text identification for various categories. Because there are a lot of categories and few images in the available training related dataset, it was advised to partition the detection and classification processes into distinct models. A deep neural YOLO version 3 model was trained on GTSDB to predict the locations of traffic signs among 4 categories. A one-layer convolutional model that was trained on GTSRB was stacked in order to use final classification among one of the 43 classifications. According to studies, training a deep network to recognise traffic signals with only 4 categories yields a high mAP@0.5 accuracy of 97.22%, which is greater than previous approaches using a significant number of categories. An additional convolutional layer is stacked to achieve classification, resulting in a rapid and efficient system that can be used in real-time applications. One can compare the suggested method to earlier CNN implementations. The Swedish Traffic Signs dataset (STSD) was used by the authors of, who used six categories and achieved average precision accuracy of 97.69% and average recall accuracy of 92.9%. 10 STSD categories were employed by the authors to attain a 95.2% mAP@0.5 accuracy. The 200-category DFG dataset, which had a 95.5% mAP@0.5 accuracy rate and was given by the Slovene company DFG Consulting d.o.o., was also used by the authors in.

PAPER-11: Vision-Based Traffic Sign Detection and Recognition Systems

Publication Year: 2019

Author: Safat B. Wali, Majid A. Abdullah, Mahammad A. Hannan 2, Aini Hussain Shammari, Xianfang Sun
Journal Name: Centre for Integrated Systems Engineering and Advanced Technologies, Universiti Kebangsaan Malaysia

Summary: The development of advanced driver assistance systems depends heavily on research into the automatic traffic sign detection and recognition (TSDR) system (ADAS). The scientific community has shown a strong interest in studies on vision-based TSDR, which is primarily driven by three factors: detection, tracking, and classification. A sizable number of TSDR-related techniques have been reported over the past ten years. This work offers a thorough analysis of the detection, tracking, and categorization of traffic signs. In the tables, along with the pertinent key references, are the details of the algorithms, methodologies, and their specifications for detection, tracking, and classification. Each area has undergone a comparative analysis to assess the TSDR data, performance indicators, and their accessibility. With few discussion and suggestions of how driver assistance system research may develop in the future, current concerns and challenges of the available technologies are highlighted. Hopefully, this research and evaluation will encourage more work to be put into creating a vision-based TSDR system in future.

PAPER-12: Traffic Sign Detection And Classification Based On Combination Of MSER Features And Multi-Language OCR

Publication Year: 2020

Author: Ali Retha Hasoon Khayeat, Ashwan A. Abdulmunem, Rafeef Fauzi Najim AlShammari, Xianfang Sun Journal Name : Webology

Summary: The novel multi-language OCR approach for traffic sign identification and recognition proposed in this study is based on MSER. The produced highest-level grayscale 2D image contains the main characteristics of the input image. The experimental work demonstrates that compared to using a standard grayscale image, employing the MSER with an image with the highest intensity can produce results that are more improved and superior. Additionally, by applying the following two techniques, the required computation time has been cut down and the results have improved: First, instead of employing a time-consuming machine learning method, MSER can be used to train a classifier to recognise and understand the traffic sign (s). We will consider the fact that combination of the two methodologies frequently will result in superior outputs in our future work. Second, the removal of non-text areas that fall under simple thresholds using a variety of text's geometrical properties. With this technique, the non-text (non-sign) region(s) are deleted while the interest area is preserved (s) (s). Following the thinning process, a text line's characters are identified and merged together (s). Consider how a multilingual OCR could help a driver in another nation understand a traffic sign.

(2.1) DIFFERENTIAL ANALYSIS (LITERATURE SURVEY)

Sr#	Method	Advantage	Limitation
1	MSA_YOLOv3	Real-time localization Classification of small traffic signs	Low precision Fuzzy objects detection is low
2	TSR, LIDAR	Detects complete driving scenes. Color-based methods.	Lack comparisons on public datasets
3	MRMR ANN	Detection in tough wheather condition. Compared to the old result, the new result is encouraging	Takes more time
4	ML, Mask R-CNN	Recognition of Character. Natural Image. Find Edges.	Cost is High More memory usage
5	CNN	Give precise output. Real time detection	The way a neural network operates must be trained.
6	CNN-SVM	Superior over the competition in terms of	Single Classification for Single Patch.

		training accuracy and training speed.	Execution Time is very high
7	CNN, YOLO v3-v4 and BLOB	Works properly on increased dataset It detects on Real time.	Processing time is more. Low precision in cloudy wheather
8	CNN, R-CNN	Low error rate High Precision	Expensive and takes more time for accurate result
9	YOLOv4, CNN	High FPS. High accuracy Find Edges, Corners	Lower mean average precision under severe conditions
10	YOLO, GTSRB	High accuracy Character Recognition, Natural Images	The speech quality might be degraded with more data
11	TSDR	Research community is highly interested in TSDR.	Takes more time • Very expensive. Not synchronized with the expected output

3. EXISTING WORK PROCESS
(3.1) LAYOUT OF EXISTING WORK

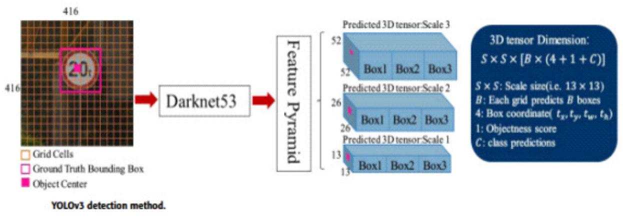
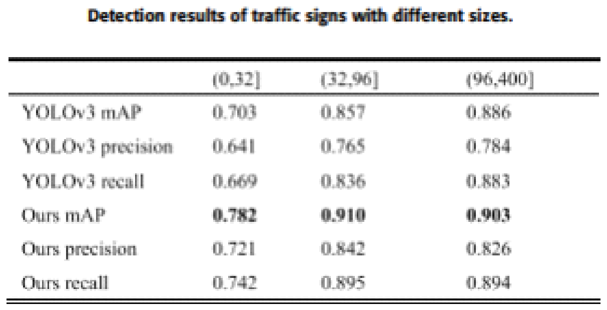

416
416
Grid Cells
Ground Truth Bounding Box
Object Center
Darknet53
Feature Pyramid
Predicted 3D tensor:Scale 3
Predicted 3D tensor:Scale 2
Predicted 3D tensor:Scale 1
52
26
13
Box1 Box2 Box3
3D tensor Dimension:
$S \times S \times [B \times (4+1+C)]$
$S \times S$: Scale size(i.e. 13×13)
B: Each grid predicts B boxes
4: Box coordinate(t_x, t_y, t_w, t_h)
1: Objectness score
C: class predictions
YOLOv3 detection method.

Figure-3.1 : YOLOv3 detection method diagram


Detection results of traffic signs with different sizes.

	(0,32]	(32,96]	(96,400]
YOLOv3 mAP	0.703	0.857	0.886
YOLOv3 precision	0.641	0.765	0.784
YOLOv3 recall	0.669	0.836	0.883
Ours mAP	**0.782**	**0.910**	**0.903**
Ours precision	0.721	0.842	0.826
Ours recall	0.742	0.895	0.894

Table 3.1: Output result of traffic signs

(3.2): Base Work Explanation

Our method is based on the YOLOv3 and MSA_YOLOv3 structure. In contrast to YOLOv3 and YOLOv4 the MSA_YOLOv3 generates better results on the small traffic signs and text. The deep learning scheme for detection of traffic signs which is based on CNN can be divided into two categories: two stage approach and one stage approach. In one stage approach regression is used with YOLOv3 for real time detection but its performance is very low while detecting small objects. In a two stage approach a MSA_YOLOv3 is used for detection of traffic signs but it uses high computing power and is very complex. The GTSDB data set is the most common dataset in the traffic sign detection field. The GTSDB divides the dataset into three main categories. Mandatory signs, prohibitory signs and danger signs. The three types of sign category is not enough. So a new dataset Tsinghua-Tencent 100K (TT100K) data set was introduced. It has more practical signs. Compared to YOLOv3 the MSA_YOLOv3 improves the efficiency of detection of small traffic signs

In MSA_YOLOv3 the main contributions are as given below: 1. Data Processing stage : In this stage the mixed technology is applied for data augmentation. In this stage the couple of images are selected randomly and mixed pixelwise. The mixed images are then used between the training image pair. The MSA_YOLOv3 is trained on such kinds of mixed pairs of traffic images data. 2. The SPP (spatial pyramid pooling) is used in the convolution layer at the end of Darknet53. The SPP performs the pooling operation on the input feature map. Then it connects the input feature map and the three feature maps and based on that MSA_YOLOv3 learns about the object's features more comprehensively. For accurately utilizing the localization signals and signs at the lower layer the augmented path is designed from bottom to top to enhance the feature pyramid in YOLOv3. It improves the accuracy of locating small objects features.
In CNN there are two types of schemes for detection of objects. 1. Two stage scheme (R-CNN): In two stage scheme it combines the Region Proposals with the CNN to detect and identify the objects. Two stages used the heuristic algorithm to generate a large set of region proposals. Then it classifies and regresses the candidate regions. For example the R-CNN takes 2k region proposals and it extracts the features of region proposals using the CNN and then determines the classes of the detected objects using the multiple SVMs. The SPP net convolves the complete image at once to extract features and avoids the problems of having unnecessary computation. 2. One stage scheme: while into one stage scheme the object detection is converted into a problem of regression for end to end detection of objects. Both the R-CNN and faster R-CNN used the region's proposal + CNN and feature extraction + SVM or the softmax classification techniques to detect objects. The two stage schemes are slightly not sufficient for real time performance.

3.2.1: Simplified Structure of MSA_YOLOv3

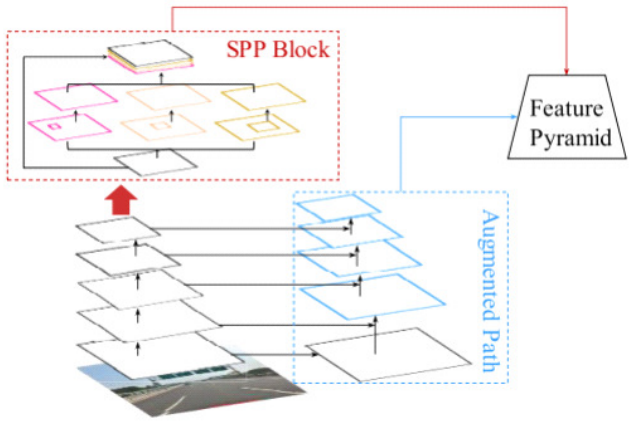


Figure-3.2 : Simple Structure of MSA_YOLOv3

To detect and categorize the traffic signs, there are numerous datasets available, including TT100k [37], GTSDB [17], GTSRB [32], and others. No matter the country, the distribution of traffic signs is terribly inconsistent. For instance, more than 1000 different signs for No Parking, Speed Limit and No Entry in the TT100k, whereas warnings related to rocks falling and mountain danger are quite uncommon.

The existing non-uniform arrangement of traffic signs makes it challenging to recognise traffic signs. The variances in the identical traffic signs, according to our detailed examination of the available data sets, are mostly reflected in the following characteristics: viewpoint, background, size, color, illumination, contrast, occlusion, and pollution. A special data augmentation technology based on traffic Signs, Symbols, Text and logos is suggested in order to obtain enough and uniformly distributed photos and replicate as many scenarios as feasible in the actual world. Figure 4 depicts the augmentation data pipeline based on the logo. First, the traffic sign's logo is acquired jointly, and the sign's mask is created automatically by a clever operator. The perspective transformation is made to the traffic sign logo to mimic the shift in viewpoint in the actual scene. Through perspective transformation, a two-dimensional image is changed into another plane.

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} .x' = \frac{a}{c}, y' = \frac{b}{c}$$

Mij is a transformation matrix with 9 parameters. The original coordinates are (x, y), and the transformed coordinates are (x, y). Due to the influence of lighting, pollution, manufacturing technique, and other factors, traffic signs' forms and thickness vary in real-world traffic situations. The main morphological operations of the image are erosion and dilation. The results of these two actions are diametrically opposed, which can increase sample variation even more. The merging of the background and the sign is the following stage.

where Is stands for the traffic sign logo, mask for the sign's mask, and Ib for the image that has been taken from the real traffic scenes and cropped. Finally, the merged image is given color jitter. Additionally, we blur the image using Gaussian noise with various kernel sizes, such as {1×1, 3×3, 5×5, 7×7} , and we apply cut-out [9, 35] to the created image to improve the network's generalization performance

We execute appropriate studies to verify the efficacy of our data augmentation strategy. In the figure below some traffic related images are displayed .

Figure-3.3 Some synthetic images

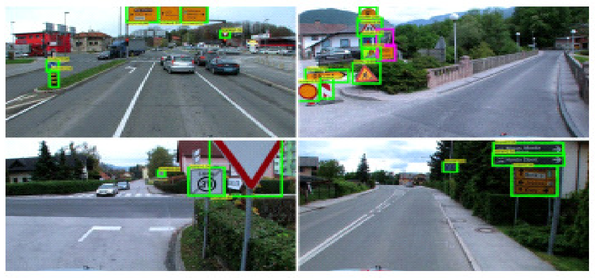


Figure-3.4 Detected Traffic Symbols

4: Proposed Work

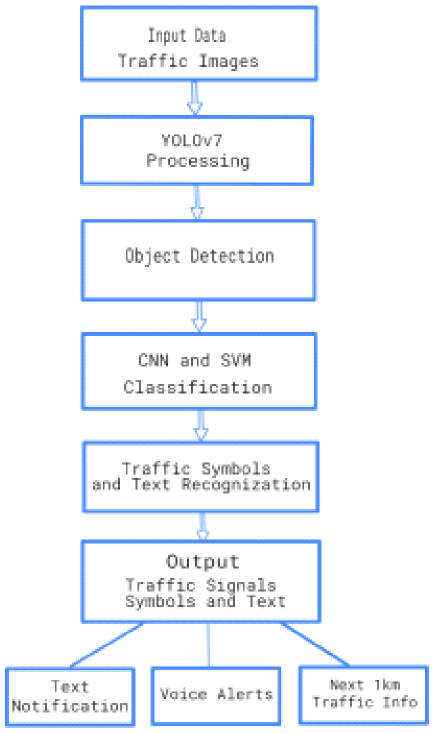


Figure-3.5: Proposed Work Layout

(4.1): Proposed Work Explanation

Though I employ some new methodologies, my proposal is the same as my basic work. They have explored YOLOv3, YOLOv4 and YOLOv7 for detection and Darknet19 for classification in my foundational paper. YOLOv7 is a better, more accurate version of Yolov4 and is what I want to use in this situation instead of YOLOv4. And because certain traffic signs have text associated with the traffic sign board, I want to use SVM for Text classification and CNN for image classification for classification purposes. We would obtain a better accurate result with the use of SVM and CNN.

Acquiring the Dataset

We first acquire Traffic related images in Input DataSet. On Colab or Jupyter, the first cell clones the required resources of the traffic symbols and their respective labels from the data set repository. This is followed by importing the libraries such as matplotlib, numpy, and keras.

Analyzing the Dataset Analyze the dataset for the dimension and its categories. Then, using the bounding box tool, we can determine the bounding box's x, y, height, and weight. Additionally, these data are automatically recorded in a text file.

Preprocessing Images Convert the images into required other scales. Applying histogram equalization will evenly distribute each pixel density and standardize lighting which will make sure the images look less messy and more composed. Pixels Normalization. **Data Augmentation**

It is possible to augment the dataset by making a few modifications on each image. This will help generate more images to train, thus, more accurate learning rates will be achieved. Apply the YOLOv4 or YOLOv7 The characteristics of the bounding box value must be converted into YOLOv4 or YOLOv7 format. Then coding, configuration and integration for training and testing of the DataSet using YOLOv4 or YOLOv7.

(4.2) PROPOSED WORK ALGORITHM

1: Collect images of traffic signs from datasets.
2: Group the images based on category.
3: Rename the images based on its category
4: Applying labels to all images using LabelImg/
Bounding Box tool.
5: It creates the respective text file for each image. 6: Gather all the images and respective text files
into one folder.
7: Using google colab I run the code.
8: I mount Google Drive and have to use the GPU. 9: Download YOLOV7 regarding repositories.
10:I Change config file as per the number of classes. Then also change files for training, validation and testing.
11: Put the images into the training images and labels folder and into the validation images and labels folder.

We need a configuration file for it. The file needs to be processed after that. The training and testing files follow. Finally, a traffic sign with a bounding box appears. After obtaining the bounding box, classification is carried out, with text classification using SVM and picture classification using CNN. After doing that, we have traffic sign labeling. That will give us the desired result in output.

5. DATASET, IMPLEMENTATION AND RESULT

(5.1) DATASET

(5.1.1) TT100K dataset. TT100K means Tsinghua- Tencent 100k. This dataset is used for detecting and classifying the traffic signs. TT100K is a country-specific traffic sign dataset with images collected in China. That contains 10,000 images with traffic signs and 90,000 background images without any traffic signs.

(5.1.2) GTSDB dataset. GTSDB means German Traffic Sign Detection Benchmark GTSDB contains 3 categories namely mandatory, prohibitory and danger. GTSDB data set is split into a training set of 600 images and a test set of 300 images and covers natural traffic scenes of various roads (road, rural, urban) recorded during the day and dusk.

(5.2) USED TOOLS & TECHNOLOGY

Python: Python is a general-purpose, high-level, interpreted dynamic programming language that places a strong emphasis on code readability. Compared to Java and C, it requires less steps. Guido Van Rossum, a developer, started it in 1991. Due to its compatibility for several programming paradigms, it is utilized by numerous organizations. Additionally, it automatically manages memory. In contrast to other programming languages, which frequently employ semicolons or parentheses, Python uses new lines to finish a command. Python 2 and Python 3 are the two most popular versions. Both are really different.

NumPy: NumPy, short for "Numerical Python," is a Python library used for working with arrays. In the year 2005, Travis Oliphant developed NumPy. You can use it for free because it is an open-source project. A set of multi-dimensional array objects and routines for manipulating arrays make up this library. Additionally, it has matrices, Fourier transform, and functions for working in the area of linear algebra. It is a library made up of multidimensional arrays and stands for Numerical Python.

Kera's: The open-source machine libraries TensorFlow, Theano, and Cognitive Toolkit are built on top of Kera's (CNTK). Python's Theano module is utilized for quick numerical computing jobs. Kesars is a Python-based API for sophisticated neural networks that runs on top of TensorFlow, CNTK, or Theano. It makes use of standalone machine learning toolkits, C#, Python, and C++ libraries. It was created with the intention of facilitating speedy experimentation. A clean and simple method for building TensorFlow or Theano-based deep learning models is provided by Kera's basic structure. Deep learning models may be quickly defined with Keras. Well, Kera's is the best option for applications requiring deep learning
Doing high-quality research requires being able to move quickly from concept to conclusion. Python is the language utilized by the SRCNN model.

Tensor Flow: The Google team created the software library or framework TensorFlow to make it simple to use machine learning and deep learning techniques.For the quick calculation of numerous mathematical equations, it combines computational algebra and optimization techniques.
It is a foundation library that can be used to generate Deep Learning models either openly or through the usage of wrapper libraries that make the process more straightforward and are built on top of TensorFlow. It contains a range of deep learning and machine learning algorithms. the version that the SRCNN model uses. It has a capability that uses multi-dimensional arrays known as tensors to quickly create, optimize, and calculate mathematical expressions.

Google Colab:
Collaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education

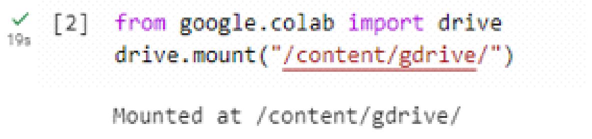
GPU:
GPU stands for Graphics processing unit. Using the Google Colab environment, we have free access to the “NVIDIA Tesla K80” GPU. But keep in mind that you are limited to use it for 12 hours continuously, after that you may not be able to access it for a particular duration of time unless you purchase Colab pro.

Anaconda:
Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

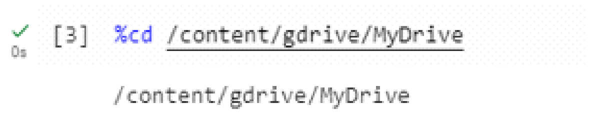
(5.3) PROPOSED SYSTEM IMPLEMENTATION

Step-1: I need to mount Google drive.

Give appropriate permissing
Press shift+Enter (Allow the permissions)



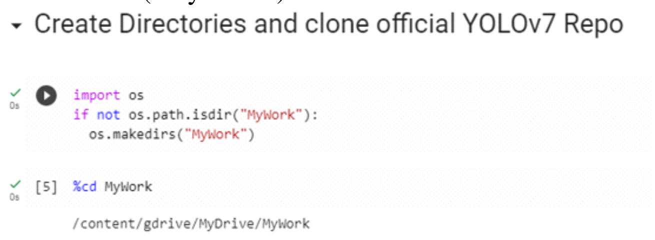
```
[2] from google.colab import drive
    drive.mount("/content/gdrive/")

    Mounted at /content/gdrive/
```



```
[3] %cd /content/gdrive/MyDrive

    /content/gdrive/MyDrive
```

Step-2: Create Directories

import os
if not os.path.isdir("MyWork"):
os.makedirs("MyWork")



Create Directories and clone official YOLOv7 Repo

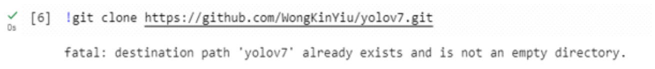
```
import os
if not os.path.isdir("MyWork"):
  os.makedirs("MyWork")
```

```
[5] %cd MyWork

    /content/gdrive/MyDrive/MyWork
```

Step-3: Clone YOLOv7 Repo

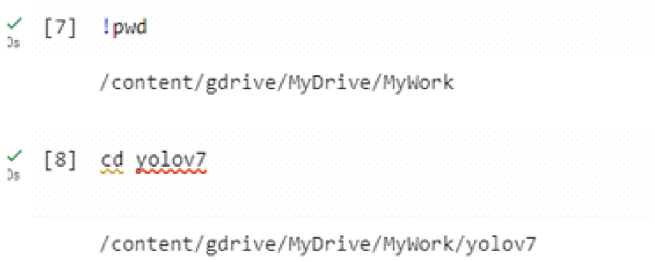
%cd MyWork

!git clone https://github.com/WongKinYiu/yolov7.git



```
[6] !git clone https://github.com/WongKinYiu/yolov7.git

    fatal: destination path 'yolov7' already exists and is not an empty directory.
```

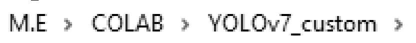
Step-4: Change current directory to YOLO7

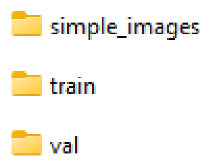


```
[7] !pwd

    /content/gdrive/MyDrive/MyWork
```

```
[8] cd yolov7

    /content/gdrive/MyDrive/MyWork/yolov7
```

Step-5: Create Train and val folder inside the YOLOV7_custom folder

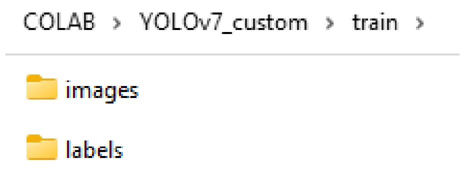



simple_images

train

val

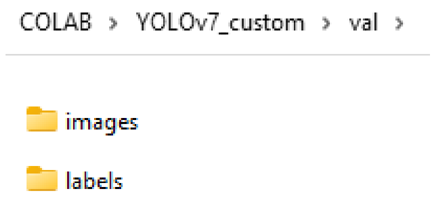
Step-6: Put the appropriate images inside the train/images and train/labels folders


COLAB › YOLOv7_custom › train ›

images

labels

Step-7: Put appropriate images inside the val/images and val/labels folders


COLAB › YOLOv7_custom › val ›

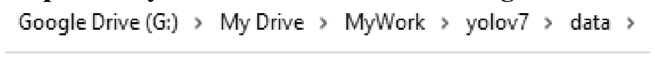
images

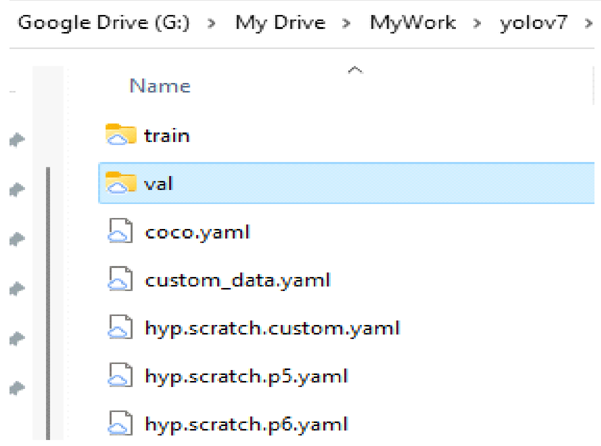
labels

Step-8: using the labelImage command open the labelling the Bounding box tool.

Open Anaconda Powershell
Execute following commandss
1.pyrcc5 -o libs/resources.py resources.qrc
2.python labelImg.py
and apply the bounding box on the respective images.
It will generate the respective .txt files for the image in the labels folder.

Step-8: Put your train and validation images in the mounted google drive YOLO7/data folder


Google Drive (G:) › My Drive › MyWork › yolov7 › data ›


Google Drive (G:) › My Drive › MyWork › yolov7 ›

Name

train

val

coco.yaml

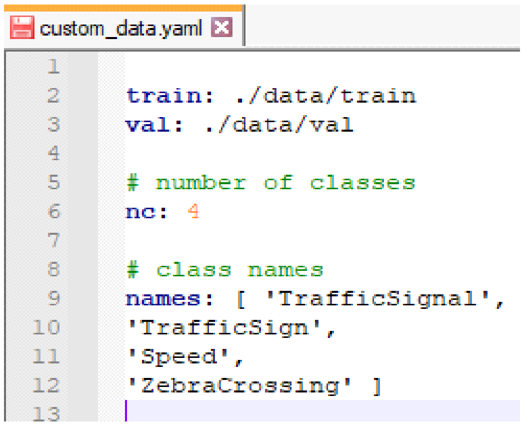
custom_data.yaml

hyp.scratch.custom.yaml

hyp.scratch.p5.yaml

hyp.scratch.p6.yaml

Step-9: Create custom_data.yaml file with the provided classes images and also mention the train and val path

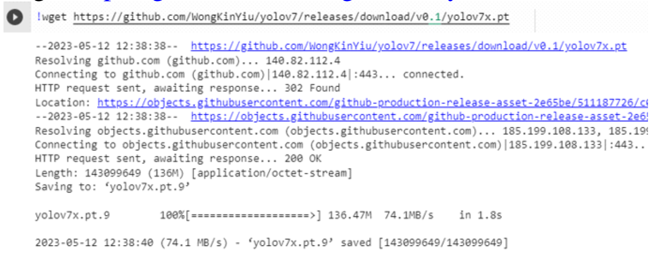


custom_data.yaml

```

train: ./data/train
val: ./data/val

# number of classes
nc: 4

# class names
names: [ 'TrafficSignal',
'TrafficSign',
'Speed',
'ZebraCrossing' ]

```

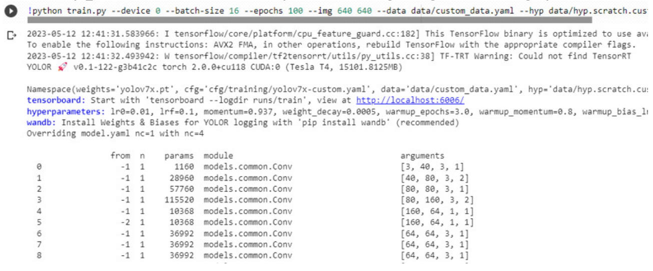
Step-9: create the yolov7x-custom.yaml file in the following directory
G:\My Drive\MyWork\yolov7\cfg\training

Step-10: Execute following two commands in the google colab
!wget https://github.com/WongKinYiu/yolov7/releases/download/v0.1/yolov7x.pt

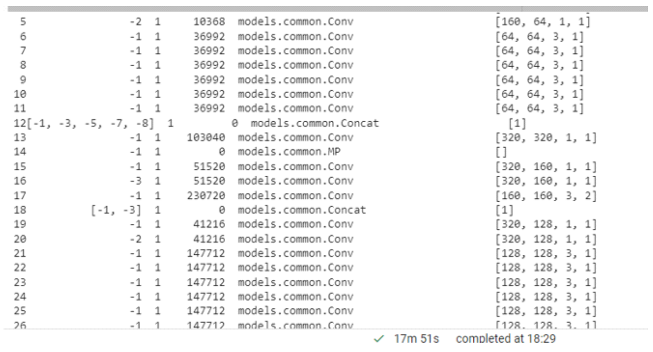


```
!wget https://github.com/WongKinYiu/yolov7/releases/download/v0.1/yolov7x.pt

--2023-05-12 12:38:38--  https://github.com/WongKinYiu/yolov7/releases/download/v0.1/yolov7x.pt
Resolving github.com (github.com)... 140.82.112.4
Connecting to github.com (github.com)|140.82.112.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/511187726/c
--2023-05-12 12:38:38--  https://objects.githubusercontent.com/github-production-release-asset-2e6
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.19
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443..
HTTP request sent, awaiting response... 200 OK
Length: 143099649 (136M) [application/octet-stream]
Saving to: ‘yolov7x.pt.9’

yolov7x.pt.9        100%[===================>] 136.47M  74.1MB/s    in 1.8s

2023-05-12 12:38:40 (74.1 MB/s) - ‘yolov7x.pt.9’ saved [143099649/143099649]
```

!python train.py --device 0 --batch-size 16 --epochs 100 --img 640 640 --data data/custom_data.yaml --hyp data/hyp.scratch.custom.yaml --cfg cfg/training/yolov7x-custom.yaml --weights yolov7x.pt --name yolov7x-custom

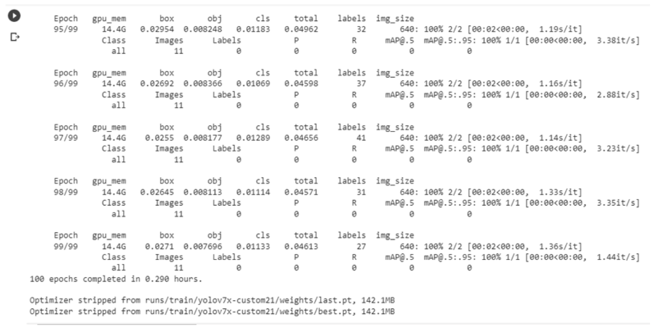


```
!python train.py --device 0 --batch-size 16 --epochs 100 --img 640 640 --data data/custom_data.yaml --hyp data/hyp.scratch.cus

2023-05-12 12:41:31.583966: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use av
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-05-12 12:41:32.493942: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
YOLOR 🚀 v0.1-122-g3b41c2c torch 2.0.0+cu118 CUDA:0 (Tesla T4, 15101.8125MB)

Namespace(weights='yolov7x.pt', cfg='cfg/training/yolov7x-custom.yaml', data='data/custom_data.yaml', hyp='data/hyp.scratch.cu
tensorboard: Start with 'tensorboard --logdir runs/train', view at http://localhost:6006/
hyperparameters: lr0=0.01, lrf=0.1, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_l
wandb: Install Weights & Biases for YOLOR logging with 'pip install wandb' (recommended)
Overriding model.yaml nc=1 with nc=4

                 from  n    params  module                                  arguments
  0                -1  1      1160  models.common.Conv                      [3, 40, 3, 1]
  1                -1  1     28960  models.common.Conv                      [40, 80, 3, 2]
  2                -1  1     57760  models.common.Conv                      [80, 80, 3, 1]
  3                -1  1    115520  models.common.Conv                      [80, 160, 3, 2]
  4                -1  1     10368  models.common.Conv                      [160, 64, 1, 1]
  5                -2  1     10368  models.common.Conv                      [160, 64, 1, 1]
  6                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
  7                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
  8                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
```



```
  5                -2  1     10368  models.common.Conv                      [160, 64, 1, 1]
  6                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
  7                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
  8                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
  9                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
 10                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
 11                -1  1     36992  models.common.Conv                      [64, 64, 3, 1]
 12[-1, -3, -5, -7, -8]  1         0  models.common.Concat                    [1]
 13                -1  1    103040  models.common.Conv                      [320, 320, 1, 1]
 14                -1  1         0  models.common.MP                        []
 15                -1  1     51520  models.common.Conv                      [320, 160, 1, 1]
 16                -3  1     51520  models.common.Conv                      [320, 160, 1, 1]
 17                -1  1    230720  models.common.Conv                      [160, 160, 3, 2]
 18          [-1, -3]  1         0  models.common.Concat                    [1]
 19                -1  1     41216  models.common.Conv                      [320, 128, 1, 1]
 20                -2  1     41216  models.common.Conv                      [320, 128, 1, 1]
 21                -1  1    147712  models.common.Conv                      [128, 128, 3, 1]
 22                -1  1    147712  models.common.Conv                      [128, 128, 3, 1]
 23                -1  1    147712  models.common.Conv                      [128, 128, 3, 1]
 24                -1  1    147712  models.common.Conv                      [128, 128, 3, 1]
 25                -1  1    147712  models.common.Conv                      [128, 128, 3, 1]
 26                -1  1    147712  models.common.Conv                      [128, 128, 3, 1]
```

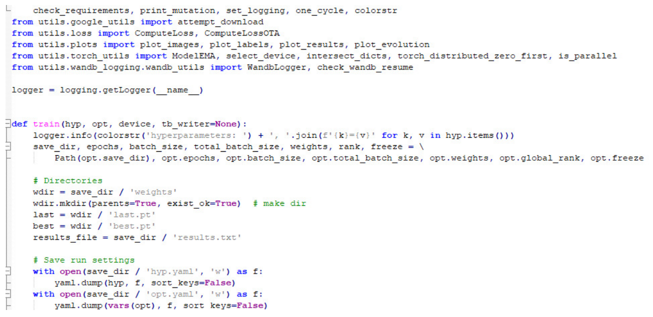
✓ 17m 51s   completed at 18:29



```
     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     95/99     14.4G   0.02954  0.008248   0.01183   0.04962        32       640: 100% 2/2 [00:02<00:00,  1.19s/it]
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100% 1/1 [00:00<00:00,  3.38it/s]
                 all        11         0         0         0         0         0

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     96/99     14.4G   0.02692  0.008366   0.01069   0.04598        37       640: 100% 2/2 [00:02<00:00,  1.16s/it]
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100% 1/1 [00:00<00:00,  2.88it/s]
                 all        11         0         0         0         0         0

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     97/99     14.4G    0.0255  0.008177   0.01289   0.04656        41       640: 100% 2/2 [00:02<00:00,  1.14s/it]
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100% 1/1 [00:00<00:00,  3.23it/s]
                 all        11         0         0         0         0         0

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     98/99     14.4G   0.02645  0.008113   0.01114   0.04571        31       640: 100% 2/2 [00:02<00:00,  1.33s/it]
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100% 1/1 [00:00<00:00,  3.35it/s]
                 all        11         0         0         0         0         0

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     99/99     14.4G    0.0271  0.007696   0.01133   0.04613        27       640: 100% 2/2 [00:02<00:00,  1.36s/it]
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100% 1/1 [00:00<00:00,  1.44it/s]
                 all        11         0         0         0         0         0
100 epochs completed in 0.290 hours.

Optimizer stripped from runs/train/yolov7x-custom21/weights/last.pt, 142.1MB
Optimizer stripped from runs/train/yolov7x-custom21/weights/best.pt, 142.1MB
```

Step-11 : Now train and test files.
So I started the training process. During / after training I get weights files. And we can test model using that weights files

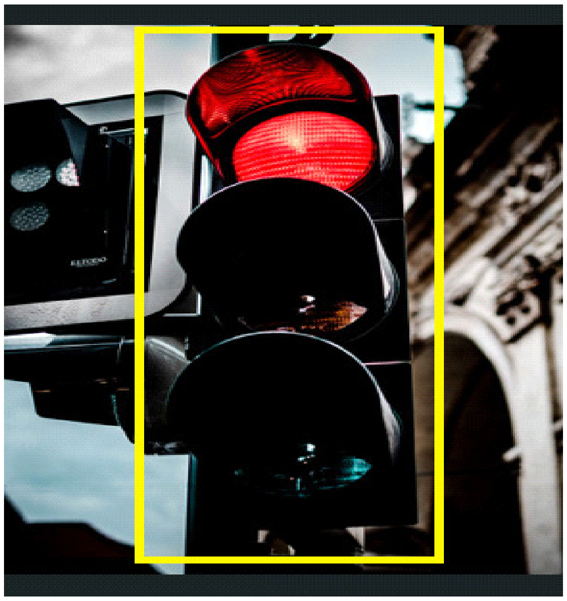
Using the following code I can get a chart which contains information about the training process.



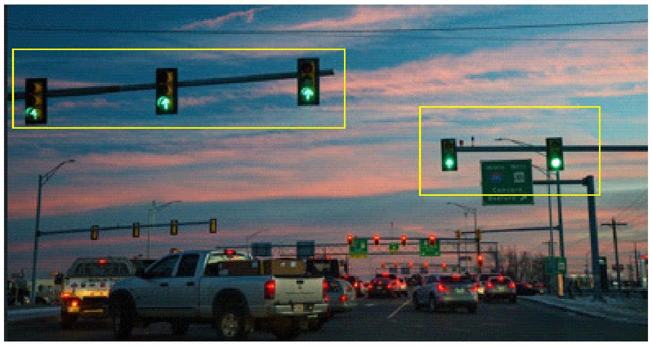
```python
    check_requirements, print_mutation, set_logging, one_cycle, colorstr
from utils.google_utils import attempt_download
from utils.loss import ComputeLoss, ComputeLossOTA
from utils.plots import plot_images, plot_labels, plot_results, plot_evolution
from utils.torch_utils import ModelEMA, select_device, intersect_dicts, torch_distributed_zero_first, is_parallel
from utils.wandb_logging.wandb_utils import WandbLogger, check_wandb_resume

logger = logging.getLogger(__name__)


def train(hyp, opt, device, tb_writer=None):
    logger.info(colorstr('hyperparameters: ') + ', '.join(f'{k}={v}' for k, v in hyp.items()))
    save_dir, epochs, batch_size, total_batch_size, weights, rank, freeze = \
        Path(opt.save_dir), opt.epochs, opt.batch_size, opt.total_batch_size, opt.weights, opt.global_rank, opt.freeze

    # Directories
    wdir = save_dir / 'weights'
    wdir.mkdir(parents=True, exist_ok=True)  # make dir
    last = wdir / 'last.pt'
    best = wdir / 'best.pt'
    results_file = save_dir / 'results.txt'

    # Save run settings
    with open(save_dir / 'hyp.yaml', 'w') as f:
        yaml.dump(hyp, f, sort_keys=False)
    with open(save_dir / 'opt.yaml', 'w') as f:
        yaml.dump(vars(opt), f, sort keys=False)
```

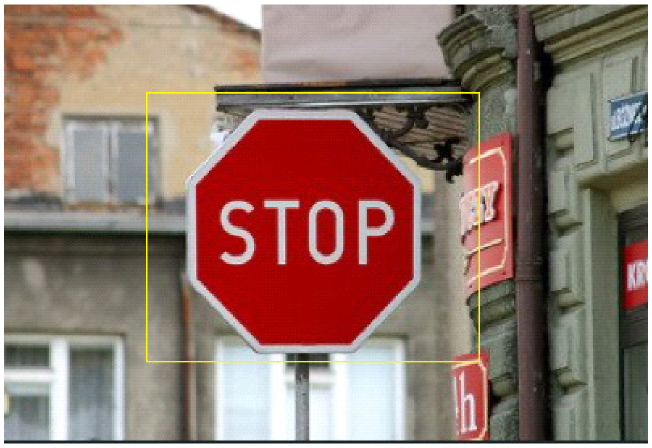
Step-12: During and after training I can get weights in the backup folder. So I can test my model using the trained weights data.

Step-13 : Now I can show the prediction which I tested in the previous step and this prediction will give us a bounding box with their labelling. And this way I performed a testing operation on other images and that image will show my results.



Traffic sign detection


Traffic sign detection



Warning sign detection

Zebra crossing detection

Speed limit detection

(5.4) FUTURE WORK

In future I want to implement a classifier using which one can recognize text also. I would take more dataset for testing and training. Also I would like to implement voice alerts. Further I would implement an algorithm which would notify upcoming hurdles in the next 1km path.

6. CONCLUSION

Here, I conclude that I learned a lot of stuff during this research work. During the research work I faced many challenges. I kept studying and exploring the appropriate papers and tried to implement them step by step.

Initially I tried to use my local machine for the implementation work. but later I decided to sue the google colab for the testing and training. Also I installed Linux Opensue myself and used it in a virtual box.

I was not aware about Google colab, YOLO, Google Jupyter Notebook, etc before.

But now I know how to use each of the technologies.. I studied many research papers in which YOLO is used for object detection. So I decided to continue my research according to YOLO. When I practiced this, I found out that YOLOV4 and YOLOV3 are the same and YOLO7 is the latest one. During this Research I studied and discussed many methods for traffic sign detection and classification. In my base paper, traffic sign detection was completed using YOLOV7 and classification was completed using Darknet19. Here, I propose my work in the Proposed paper as the same as base work but I include some other new methods.

7. REFERENCES

1. Ali Retha Hasoon Khayeat, Ashwan A. Abdulmunem, Rafeef Fauzi Najim AlShammari, Xianfang Sun
"Traffic Sign Detection And Classification Based On Combination Of MSER Features And Multi-Language OCR" 2020 Webology

2. Safat B. Wali, Majid A. Abdullah, Mahammad A. Hannan 2, Aini Hussain Shammari, Xianfang Sun
"Vision-Based Traffic Sign Detection and Recognition System" 2019 Centre for Integrated Systems Engineering and Advanced Technologies, Universiti Kebangsaan Malaysia

3. Sergey A. Kolyubin, Valentyn Sichkar. "Real Time Detection And Classification Of Traffic Signs Based On YOLOv3" 2020 Scientific and Technical Journal of IT Mechanics and Optics

4. Njayou Youssouf " Traffic Sign Classification Using Cnn And Detection Using Faster-Rcnn And YOLOv4 " 2021

5. Konda reddy Thanigundala, Sreevatsava Reddy Musani, Rajesh Kannan Megalingam , Hemanth Nidamanuru, Lokesh Gadde "Indian traffic sign detection and recognition using deep learning" 2021 "International Journal of Transportation Science and Technology "

6. Manjiri Bichkar, Sonal Chaudhari, Suyasha Bobhate,

"Traffic Sign Detection and Classification for Indian Traffic Signals Using DL" 2021 International Journal of Scientific Research in Computer Science, Engineering and Information Technology

7.Shanthi.S, Sunitha.A " Traffic Sign Detection And Classification Using DL" 2020 :International Journal Of Innovative Research In Technology

8.Ghazanfar, Danyah A., Alghmghama, Latifab, Jaafar Alghazoa, Loay Alzubaidi "Autonomous Traffic Signs (ATSR) Detection and Recognition using Deep CNN" 2021 Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Malaysia

9. Victor Ciuntu, Hasan Ferdowsi "Real-Time Traffic Sign Detection and Classification Using Machine Learning" 2020 IEEE

10 Manisha Vashisht, Brijesh Kumar "Robust Classification of Traffic Signs using MRMR Feature Reduction Technique and Optical Character Recognition" 2021 IEEE

11. Chunsheng Liu, Faliang Chang, Shuang Li, Yinhai Wang "Machine Vision Based Traffic Sign Detection Methods" 2020 IEEE

12. Huibing Zhang , Longfei Qin, Jun Li, Yunchuan Guo, Ya Zhou, Jingwei Zhang "Real-Time Detection Method for Small Traffic Signs Based on Yolov3" 2020 IEEE