Object Detection using YOLOv5

(Custom Trained)

Jaskaran Singh Walia

While the Complete working is Explained into the Colab Notebook File, here I will provide a brief about it.

I have created a transfer learning based custom Object Detection using the model Yolov5. Train and Validation data is created by dividing the number of Images and Creating Annotation Text files for YOLO.

For testing the model, just copy the Given Data folder Directly into Google Drive and mount google drive with Google Colab. And run the Custom Image Testing Code provided at the End. Rest Everything is Mentioned in the Colab File.

Model name: Yolov5 (Person-Car-Detection Custom Trained)

Links to dataset : https://drive.google.com/drive/folders/1ldxNfSH2-Ncjsd3Ky1FZXgBMNwbNeRMr?usp=sharing
Data - Images and Annotation in Json format
train - Images and Labels for training
val - Images and Labels for validation
test - Images for testing
yolov5 - Model Repository
data.yaml - "Yaml" file containing the path to data

Link to Yolov5 Framework: https://github.com/ultralytics/yolov5

About the model: YOLO refers to "**You Only Look Once**" as one of the most versatile and famous object detection models. It is state of the art and newest version of the YOLO object detection series, and with the continuous effort and 58 open source contributors, YOLOv5 set the benchmark for object detection models very high.
The YOLO network consists of three main pieces.
1) **Backbone** - A convolutional neural network that aggregates and forms image features at different granularities.
2) **Neck -** A series of layers to mix and combine image features to pass them forward to prediction.
3) **Head** - Consumes features from the neck and takes box and class prediction steps.

Primary Analysis: For Object Detection there are various options but YOLO is State of the Art. So I thought of implementing a custom training on that. Our Annotation was not as per the YOLOv5 standard, it needed conversion. Data also needed to divide into Training and Validation Set.

Assumptions: All the bounding boxes available for training are correctly positioned. Images are having the same distribution. All the instances of objects within an Image are taken into consideration.

Inference:
I have taken Minimum Average Precision as Evaluation Metrics.
60 epochs completed in 0.3 hours.

Class	Val-Images	Labels	Precision	Recall	mAP@.5	mAP@.5:.95: 100%
all	448	3399	0.709	0.494	0.552	0.259
person	448	2177	0.64	0.467	0.508	0.201
car	448	1222	0.777	0.521	0.596	0.316

Conclusion: Created a Custom Training based Object Detection Model using State Of The Art YOLOv5 with 2239 Images (1791-Training, 448-Validation) Running for 20 mins at 60 epochs. The model got trained in 20 mins as it used the Tesla T4 GPU. Model is performing well, tested on unseen data, objects are detected well but the confidence score is little less.

False Positive: There could be some very less False Positive Detection as Precision is quite high per class. But seeing some Examples in the data (where some sign boards are considered as Human), cleaning the data further could help in removing any false positives.

Recommendations:
Deep Learning is mostly experimenting and to achieve better results, there are multiple experiments still left. It could be training for longer period, fine-tuning at different level of layers unfrozen, removing the false data, retraining, trying different other models like YOLO-X, YOLO-R, SSD, EfficientDet, RCNN, Detectron, Changing Anchor box sizes, image-sizes, Data-Augmentation Techniques and many more to go.