COMP9444 Project Summary

<Traffic sign recognition >

<z5286363 Haoyu Sun | z5111402 Yuchen Zhang |

z5271151 Yaxin Guan | z5356803 Sichen Zhou | z5336045 Chengcheng Zhu>

I. Introduction

With the development of AI intelligence, many breakthroughs have been made in autonomous driving technology. For example, the intelligent vision system has made a qualitative leap in the recognition effect in just a few years. With the breakthrough of technology, more and more prototype intelligent cars can go out of the laboratory, be tested in a natural road environment, and step into practical application. The traffic signs are used to convey the indication information to the vehicles and pedestrians, which is a critical factor in ensuring the smoothness and safety of the traffic line. Therefore, accurate and rapid identification of traffic signs is the key to achieving the security of automatic driving.

Our project aims to find a model with high performance in recognising traffic signs. After our research, we found that the model of Yolov5 has an excellent performance on this problem. Still, the object detection accuracy of Yolov5 is low, so we found that the model of RCNN could lead to higher accuracy. From this report, you can see how RCNN recognise the traffic signs and the comparison with Yolov5.

II. Methods

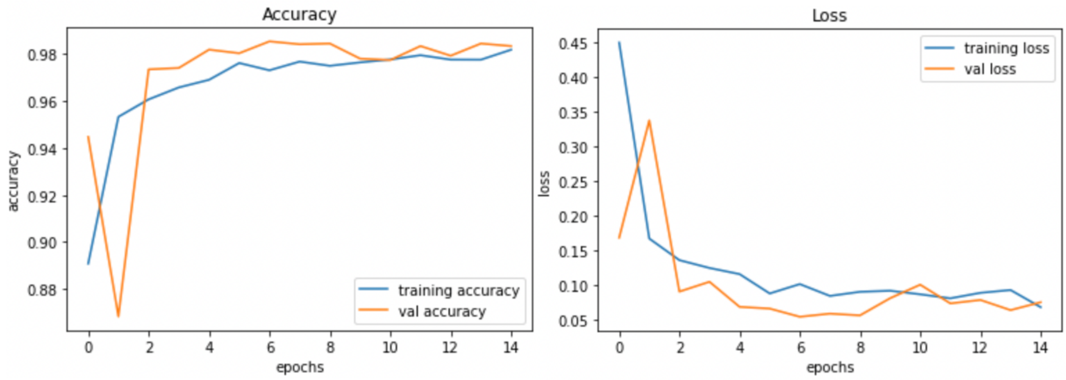
The dataset we found from Kaggle is displayed by XML. We transformed the data type from XML into TXT for pre-processing for both models. For the method of Yolo, two folders must exist, "image" and "labels", and each contains the data of training, testing and validation.

For both RCNN and Yolov5, after labelling the dataset, we need to find all bounding boxes that do not contain the target area in the image. For this step, we calculate the property of dividing t the union of the ground truth box and bounding box into the intersection of these two boxes. If the property is bigger than 0.7, then this bounding box will be added to the training set. Otherwise, we will label this bounding box as 4 ("other"), which does not contain the target. However, we also need to train the bounding box with the label "other". If not training the "other" label, every proposed bounding box will be forced a class label, resulting in misclassification. Training "other" helps the model distinguish the target traffic sign from the general background.

III. Experimental Setup

For RCNN, the dataset is from Kaggle (https://makeml.app/datasets/road-signs). The number of images in this dataset is 877. However, the actual number of training data is 32697, which could be seen by the *train_set.shape* because we capture multiple parts of an image during pre-training. In addition, the dataset is divided into four classes: traffic lights, stop signs, speed limit signs and crosswalk signs, and we labelled them from 0 to 3. For model visualisation, we could predict whether the image contains a traffic sign by supposing that one bounding box must have the target traffic sign for the first 1000 data. Moreover, if the label of the image is not 'other' and the *maxItem* which indicates the probabilities of the predicted class is more than 0.95, it means this bounding box contains the target from one of four classes.

IV. Results



For the problems of high accuracy, multi-scale and small objects, the model of RCNN has more obvious advantages. The plots above show the performance of accuracy and loss of RCNN. The accuracy is around 0.98, and the loss tends to be low. We import the package of accuracy_score from sklearn.metrics to evaluate the accuracy (0.98) of RCNN which is shown in the below graph.

```
23 print(accuracy_score(ylabel, predLabel))

205/205 [==============================] - 1s 4ms/step
0.9834862385321101
```

V. Conclusions

The main problem is capturing the traffic sign from the images. The preprocessing can be realised through two steps, the transformation of data type and pre-training for bounding boxes containing the target and without targets. Trials of two methods have been applied on the way to solve this problem. Accuracy_score is the metric that has been utilised to measure the performance of RCNN, and it reaches 0.98.

Compared with Yolov5, RCNN is good at detecting small objects and the automatic extraction of image features is realized. However, RCNN still has a limitation in that the amount of calculation is large, which means that it will take more time. Most of the region proposals extracted overlapped each other, and the overlapped parts will be extracted many times.

For further improvement, we could record the coordinates for each bounding box, and compare the value of IOU (intersection of the union) between the new bounding box and existing boxes. Save the coordinate of the bounding box with a higher IOU. After this step, we could shorten the runtime.

VI. References

Github, accessed 26 July 2022, <https://github.com/ultralytics/yolov5>

Girshick R, Donahue J, Darrell T, et al. 'Rich feature hierarchies for accurate object detection and semantic segmentation', CVPR, 2014.