

PAPER • OPEN ACCESS

## Traffic sign detection based on classic visual recognition models

To cite this article: Jirui Yang 2023 *J. Phys.: Conf. Ser.* **2646** 012024

View the [article online](#) for updates and enhancements.

### You may also like

- [Investigation on the Effect of the Feature Extraction Backbone for Small Object Segmentation using Fully Convolutional Neural Network in Traffic Signs Application](#)  
H M Elhawary, M I Shapiaib and A Elfakharany
- [Attention feature fusion network for small traffic sign detection](#)  
Miaozhi Wu, Jingmin Yang, Wenjie Zhang et al.
- [Traffic sign detection method based on Faster R-CNN](#)  
Linxiu Wu, Houjie Li, Jianjun He et al.



**ECS** The Electrochemical Society  
Advancing solid state & electrochemical science & technology

**247th ECS Meeting**  
Montréal, Canada  
May 18-22, 2025  
*Palais des Congrès de Montréal*

**Showcase your science!**

**Abstracts due December 6th**

# Traffic sign detection based on classic visual recognition models

**Jirui Yang**

College of ENGINEERING, COMPUTING AND CYBERNETICS, The Australian National University, Canberra, Australian

u6596617@anu.edu.au

**Abstract.** This paper aims to assess the effectiveness of various object detection high-level architectures, including Faster R-CNN, R-FCN, SSD, and YOLO, in recognizing traffic signs. Since traffic sign recognition is a critical element of driver-assistance and autonomous driving systems, this study explores how five distinct architectures for feature extracting (ResNet V1 50, ResNet V1 101, Inception V2, Inception-ResNet-V2, and Darknet-19) can enhance the performance of these meta-architectures. To evaluate the models, the authors fine-tuned a pre-trained object recognition algorithm on the German Traffic Sign Detection Benchmark (GTSDb) dataset. While the Faster R-CNN Inception Resnet v2 proved to be the most accurate among the models, the unbalanced distribution of mandatory, prohibitory, and danger signs in the GTSDb dataset may have caused biased results in object detection. Overall, the result obtained from this study can help improve the accuracy of driver-assistance and autonomous driving systems by providing insights into the performance of various object detection models in recognizing traffic signs.

**Keywords:** Object detection, Traffic sign detection.

## 1. Introduction

"The TSR (traffic sign recognition) system plays a crucial role in advanced driver-assistance systems (ADASs) and auto-driving systems (ADSs), typically comprising two components: traffic sign detection (TSD) and traffic sign recognition" [1]. TSD aims to locate the position of the traffic sign, while TSR classifies the category of the object in the position. Therefore, the accuracy of TSD is critical to the overall accuracy of TSR.

Traffic signs are an essential form of instruction equipment on roads that offer drivers and pedestrians guidance under different conditions, including day and night [2]. They are crucial in providing information about road conditions and restrictions, particularly in vision-based vehicle guidance systems. Due to their unique colors and shapes that stand out from the environment, traffic signs are easy for humans to understand. The RGB camera and LIDAR are the two primary sensors used to perceive the road environment. While the RGB camera observes the color, structure, and visual characteristics of traffic signs, it does not provide 3D geometric information, which the LIDAR sensor can offer. Combining the two sensors enables accurate detecting and classifying traffic sign even in complex environments [3].

There exist multiple classical leading algorithms that leverage convolutional neural networks for



feature extraction. These algorithms include Faster R-CNN [4], R-FCN [5], SSD [6], and YOLO V2 [7]. Two categories of deep learning-based object detection detectors exist: the two-stage detector and the one-stage detector. The two-stage detector filters the bounding box at a coarse level. With the filtered bounding boxes, detectors with two stages, for example, Faster R-CNN and R-FCN are more accurate at the expense of speed, while the one-stage detectors like SSD and YOLO V2 attempt to retrieve the object at once [8]. When assessing the suitability of these detectors for certain tasks, engineers not only consider accuracy metrics but also other metrics like computational complexity and memory consumption. In the case of autonomous vehicles, decision speed is crucial, and the trade-off between accuracy and decision speed must be evaluated carefully. In autonomous vehicle scenarios, the emphasis is more on decision speed.

Object detection models are typically evaluated using different challenges, including Microsoft COCO [9] and VOC Pascal [10], which comprise a variety of entities, such as humans, trains, and vehicles, for the models to recognize. This paper examines four models that utilize distinct feature extractors pre-learned on the Microsoft COCO dataset. Our focus is on evaluating the models' performance by examining their meta-architecture and feature extractor. The paper presents four key contributions: (1) a summary of the five different models and their meta-architecture and feature extractors, (2) an assessment of the models' mean average precision (mAP) on the Microsoft COCO and VOC PASCAL databases, as well as their ability to detect traffic signs at different scales, (3) a comparison of the models' accuracy, revealing an unbalanced distribution problem in the GTSDb dataset that causes biased detection, and (4) a comparison of the models' accuracy, demonstrating that the Faster R-CNN Inception Resnet v2 achieves the highest level of accuracy.

The rest is separated into several sections. Section 2 introduces the dataset used for model pretraining, while Section 3 presents details on previously leading models about traffic sign detection and classification. Section 4 shows and analyzes the results generated from the Microsoft COCO and VOC PASCAL databases. Finally, Section 5 draws conclusions from the experiment and discusses the implications of the results regarding detecting and recognizing traffic signs.

## 2. Dataset

This study utilized the German Traffic Sign Detection Benchmark (GTSDb) dataset [11] for conducting the experiments described in this paper. The dataset comprises 900 images containing a total of 1206 traffic signs, which were split into training and testing set. The training set consists of 600 images containing 846 traffic signs, while the testing set consists of 300 images containing 360 traffic signs. The dataset images were captured from diverse traffic scenes with various road conditions, weather conditions, and lighting conditions. The traffic signs has four groups: prohibitory, mandatory, danger, and other. The "other" category was not taken into account during the experiments because of its small number. The distribution of the remaining three categories was uneven. The training set had 396 prohibitory signs (59.5%), 156 danger signs (23.4%), and 114 mandatory signs (17.1%). Similarly, the test set included 161 images of prohibitory signs, 63 images of danger signs, and 49 images of mandatory signs.

## 3. Diverse traffic sign detection model

This section provides a detailed introduction to four CNN-based model architectures and six feature extractors. Models are pre-learned on the Microsoft COCO database [9] and are used for direct evaluation, without considering computational complexity. The pre-trained models classify traffic signs into three classes: prohibitory, mandatory, and danger. Table 1 shows the combination between meta-architecture and feature extraction. The reason for not pairing the meta-architecture with the feature extractor in a pairwise manner is that each feature extractor requires customization and adaptation before being integrated into a meta-architecture. Due to its computational complexity, only this pre-trained model is selected for evaluation.

**Table 1.** Combination of Meta-architecture and Feature extractor.

	<i>Faster R-CNN</i>	<i>R-FCN</i>	<i>SSD</i>	<i>YOLO</i>
<i>Resnet v1 101</i>		√		
<i>Resnet v1 50</i>	√			
<i>Inception v2</i>			√	
<i>Inception Resnet v2</i>	√			
<i>Darknet-19</i>				√

### 3.1. Model architecture

This section will present and compare the characteristics of every meta-architecture, namely Faster R-CNN, R-FCN, SSD, and YOLO V2.

**3.1.1. Faster R-CNN.** Faster R-CNN is an exemplar of a object detector with two stages that utilizes a "coarse to fine" strategy. The method initially filters bounding boxes in a coarse manner, followed by extracting classification features from these filtered boxes to achieve greater accuracy. The Region Proposal Network (RPN) was proposed by Faster R-CNN as a fully convolutional network (FCN) [12] capable of end-to-end learning for producing detection proposals. This approach is novel and not present in previous two-stage object detection models. To generate detection proposals, the RPN slides windows on the convolutional feature map, which can be a network like ResNet or Inception. At each window position, the RPN generates a group of object candidates, each having an objectness score that measured the likelihood of the proposal containing the object of interest or simply being a background.

The Region Proposal Network (RPN) generates multiple region proposals for each anchor box. An anchor box refers to a set of reference boxes, and the RPN predicts proposals by considering the position of each anchor box. The quantity of detection proposals generated for every location is determined by the number of these anchors. To accommodate objects of different sizes and shapes, anchor boxes have multiple scales and aspect ratios. These boxes form a pyramid of anchors that serve as a reference for bounding box regression. To avoid redundancy in detections, proposed regions undergo non-maximum suppression (NMS) based on their objectness scores, which filters out lower-scoring detections when multiple detections overlap. The detection network then receives proposal regions ranked in the top-N for object categorization and the estimation of bounding box positions.

**3.1.2. R-FCN.** The Region-Based Fully Convolutional Network (R-FCN) is an object detection model that operates in two stages and has a comparable architecture to Faster R-CNN. R-FCN, like Faster R-CNN, incorporates proposal of regions and classification of regions. The feature map produced by the fully convolutional neural network (CNN) of Faster R-CNN is shared with both RPN and R-FCN in R-FCN's architecture, and is processed by the region proposal network to produce candidate regions. In R-FCN, parameterized layers are trained based on the entire image, which reduces computations on each region. Additionally, the author of the paper demonstrates the trade-off between transition invariance and variance through their experiments.

**3.1.3. SSD.** Apart from the detectors like Faster R-CNN and R-FCN with two stages, there is a detector named the single-shot multi-box detector (SSD) with one stage. Unlike its two-stage counterparts, SSD employs a feed-forward network to generate a set of fix-size bounding boxes along with their respective objectness scores directly. For each detected object, SSD only requires a ground truth bounding box and generates a feature map pyramid, with a predetermined set of bounding boxes assigned to each feature cell, which are then used to calculate the confidence scores and shape offset for all object categories.

The SSD model adheres to a conventional design for producing top-notch image classification results. However, it omits the classification layer and incorporates extra layers for capturing crucial features, which will be elaborated on in subsequent discussion.

The SSD model comprises multiple convolutional feature layers, which have the ability to generate a range of feature maps at different scales, forming a pyramid of features. To obtain a classification score or an offset in shape associated with the coordinates of the default box., a small  $3 \times 3 \times P$  kernel is applied to each feature layer, which has an input size of  $m \times n$  with  $p$  channels. Each feature map cell is assigned a group of default detecting boxes, which are akin to anchor boxes, and their positions are fixed to their corresponding cells. These default boxes are selected carefully to handle various tasks that involve different target object scales. In summary, the SSD architecture is capable of predicting at multiple scales.

**3.1.4. YOLO V2.** Firstly, the original YOLO algorithm [13] will be discussed, which takes  $448 \times 448$  fixed input size. The input image is separated into  $13 \times 13$  grid cells and estimate whether the cell contains an object with a confidence score. Bounding box regression is then applied to generate bounding boxes around the objects with a metric measuring their position and objectness score. The Intersection over Union (IOU) filter out bounding boxes that only partially contain an object. Last, non-maximum suppression (NMS) merges all bounding boxes containing the same object into a single bounding box with the highest confidence score.

Next, the updates of YOLO v2 on the original YOLO will be introduced. YOLOv2 improved the original YOLO algorithm by introducing anchor boxes for better object localization and adding batch normalization for faster training. Batch normalization is significantly effective in improving convergence [14]. In order to obtain accurate bounding boxes, a specially designed distance metric is employed to utilize k-means clustering on the detection rectangles of the training set, as opposed to depending on manually chosen priors. (Eq. (1))

$$d(\text{box}, \text{centroid}) = 1 - \text{IOU}(\text{box}, \text{centroid}) \quad (1)$$

To avoid model instability, which can lead to anchor boxes being placed at arbitrary positions in the image irrespective of the location where the box is predicted, the model predicts location coordinates respect to the grid cell's location apart from predicting offsets.

The Darknet-19 which will be discussed in Section 3.2.4 is a classification model recently created to serve as the basis for YOLOv2. During the training process, the model is customized and refined for object detection. To enhance detection accuracy, conventional augmentation methods like random crops, rotations, hue and saturation adjustments, and exposure shifts are utilized [7]. Additionally, the network incorporates training across multiple scales by selecting a different image size every 10 batches. This approach ensures that the network learns features from various resolutions.

### 3.2. Feature extractor

In this section, some classic convolutional neural networks (Resnet V1 50 / Resnet V1 101, Inception V2, Inception Resnet V2 and Darknet-19) are adopted as feature extractors for high-level features.

**3.2.1. Resnet V1 50 / Resnet v1 101.** ResNet V1 50 and ResNet V1 101 are deep neural networks that achieved success in the ILSVRC and COCO 2015 competitions, including tasks such as segmentation, detection, and classification [15]. In the Faster R-CNN and R-FCN frameworks, ResNet is used as a feature generator, which is split into two parts. The first part extracts features for the Region Proposal Network (RPN), while the latter segment extracts features for the box classifier.

Both two Resnets are in the structure of four blocks. The first three blocks exact a feature map that is shared by RPN and region of interest (RoI) layer in both Faster R-CNN and R-FCN while the last layer in third block predicts region proposal. Last, the box classifier feature is extracted by the last block.

**3.2.2. Inception V2.** Inception V2 [13] set the state-of-the-art performance in both the ILSVRC2014 classification and detection challenges. One of the key features of Inception V2 is the use of inception units, which are composed of a group of parallel convolutional sorts with different kernel sizes, and a

pooling layer. These units allow the network to increase its width and depth without increasing computational complexity, which can be beneficial for training and inference efficiency.

As discussed in Section 3.1.3, SSD adopts a one-stage detection approach where region proposals are not needed. Therefore, instead of splitting the Inception V2 model into distinct modules for PRN feature extraction and box classifier feature extraction, the entire model is employed as the feature extractor.

**3.2.3. Inception Resnet V2d.** For the Inception Resnet V2 [16], in the experiment, only the Faster R-CNN high-level architecture is combined with this feature encoder. Faster R-CNN, being a detector with two stages is partitioned into two components. The computational efficiency of the inception unit is combined with the optimization effect by residual connect.

The Inception-ResNet-V2 consists of multiple modules, including the Inception-ResNet-B and Inception-ResNet-C modules, which are used for feature extraction for the Region Proposal Network (RPN) and box classifier, respectively. The Inception-ResNet-B module operates on a 17x17 grid, while the Inception-ResNet-C operates on an 8x8 grid.

**3.2.4. Darknet-19.** As explained in Section 3.1.4, Darknet-19 is a deep learning model initially designed for object classification, but it has been adapted to perform object detection through modifications.

The Darknet-19 model follows a similar architecture as the VGG models, where the pooling increases the channel's number by twice, and a 3x3 kernel is utilized. Then the global average pooling is applied along with 1x1 kernels to compress the space dimension of the feature map produced by the 3x3 convolutions. Also, Batch normalization is applied to improve the model regularization, stabilize training and accelerate convergence.

Initially, the model is trained with the 224x224 input image size on ImageNet. Then the model is fine-tuned in the larger dataset for a few epochs with a relatively low learning rate which gives the model better performance on the higher-resolution images.

Finally, the last convolutional layer in the pre-trained feature extractor is replaced by several fully connected layers to generate region proposals, and then those proposals are fed into a separate head network which consists of three 3x3 convolutional layers with 1024 filters each, with two layers afterwards: one component for identifying object categories and another component for predicting object locations through bounding boxes. The the final convolutional layer's output size is decided by the quantity of object categories that the model is designed to recognize.

## 4. Result

Here, the performance of each meta-architecture with diverse feature extractors that are introduced in Section 3 will be analyzed. Measurement on mean average precision (mAP) and Intersection of Union (IoU) in the experiment will be compared. Before comparing each metric, definitions are introduced in the following.

### 4.1. Accuracy measurement

**4.1.1. Mean Average Precision (mAP).** The performance of object detection systems is evaluated using the mean average precision (mAP) obtained from PASCAL VOC 2010 [10]. To introduce mAP, average precision (AP) will be conducted. AP summarises the precision/recall curve to one scalar value which it is the area under the precision/recall curve (Eq (2)).

$$\text{Average Precision (AP)} = \int_{r=0}^1 p(r) dr \quad (2)$$

where  $p(r)$  denotes the precision measured at a given recall level  $r$ .

The function of precision corresponding to recall  $r$  is defined as the highest precision attainable at or beyond a recall level of  $r$  ( $h \geq r$ ).

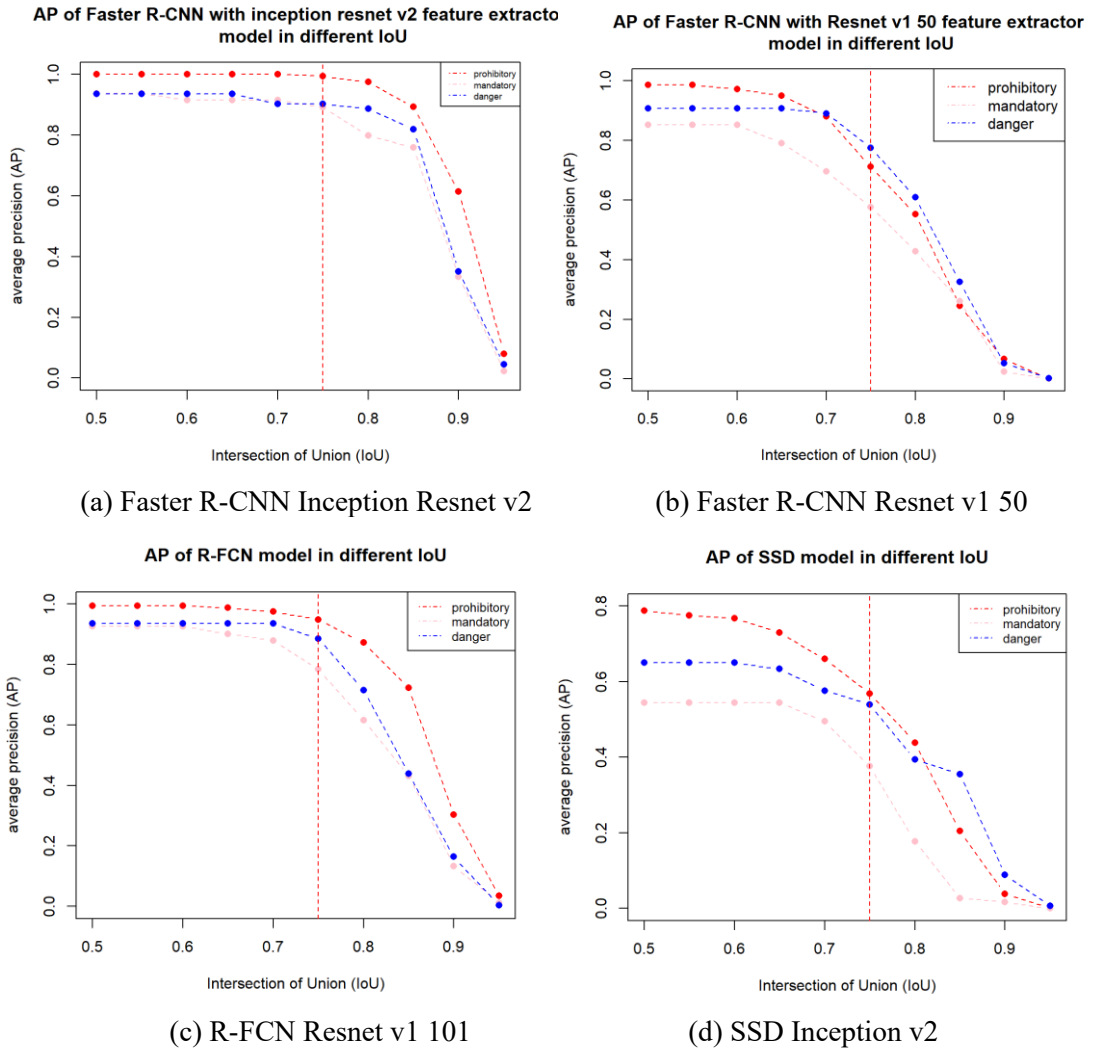
$$p(r) = \max_{h:h>r} p(h) \quad (3)$$

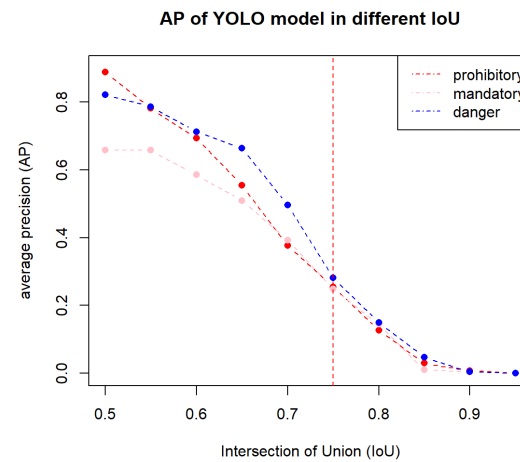
Finally, mAP is obtained by taking the mean value of each category's average precision.

**4.1.2. Intersection of Union (IoU).** The accuracy of predictions is assessed using a metric denoted as Intersection of Union (IoU). The IoU metric computes the percentage of the overlap between the predicted ( $B_p$ ) and the ground truth bounding box ( $B_{gt}$ ) over the union area of both boxes ( $B_{gt} \cup B_p$ ) (Eq 4.). A threshold of 0.5 is set to determine the correctness of a prediction. If the Intersection over Union (IoU) score is higher than 0.5, the prediction is deemed accurate, while those with an IoU less than 0.5 are false positives. Objects in the ground truth data that do not have matching predicted bounding boxes are considered false negatives, while redundant object detections are recognized as the false positives.

$$IoU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \quad (4)$$

#### 4.2. Evaluation on AP over different IoU





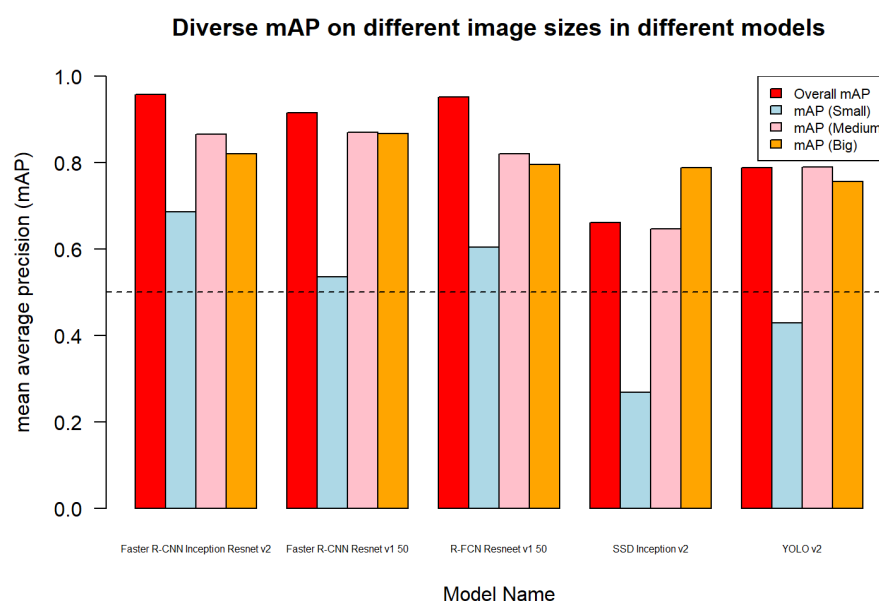
(e) YOLO V2

**Figure 1.** (a-e) Average precision of different model of different IoU threshold.

Average precision is a good estimator of the accuracy of the model prediction and the IoU detects the bias between prediction and ground truth. In Fig. 1., there are five scatter plots on average precision with different IoU thresholds in different categories and one vertical line at  $\text{IoU} = 0.75$  is drawn to compare different model precision at high accuracy standards. The average precision is obtained through the pre-learned and fine-tuned model on the GTSDb dataset that is tested on the Microsoft COCO dataset.

From the five plots above, the average precision on mandatory traffic signs is significantly lower than the other two categories while the prohibitory has the highest AP at  $\text{IoU} = 0.5$  which is the threshold used for training. With the Inception Resnet v2 as feature encoder, in Faster R-CNN the average precision for prohibitory even reaches 1.00. This result is highly correlated with the bias distribution of three categories of traffic signs described in Section 2.

#### 4.3. Evalution on mAP over diverse model



**Figure 2.** Accuracy in different size on 5 different detectors.



To be noticed, the size of the traffic sign influence negatively on the accuracy. The ground truth traffic images are split into three parts according to the width of the image. There are two thresholds to split the images. The first threshold is a width of 32 pixels, which separates the small group of 89 images from the other two groups. Any image in the dataset with a width smaller than 32 pixels will be assigned to the small group. The second threshold is a width of 45 pixels, which separates the medium group of 93 images from the large group of 91 images. Any image in the dataset with a width between 32 pixels and 45 pixels (inclusive) will be assigned to the medium group, while any image with a width greater than 45 pixels will be assigned to the large group. Since the size of the three groups are relatively large and the difference between each other is small. The effect of the group size on the mAP can be ignored.

From Fig. 2., all the detectors perform worse in the small group than in other groups. This may be caused by the feature extractor that is pre-trained by the Microsoft COCO dataset which the traffic sign size is large and located in the middle while in reality the traffic sign usually appears at the edge of the image and small. Then one horizontal line is drawn to compare the mAP. It is clear that the SSD Inception v2 and YOLO v2 perform significantly worse than the other three models while their performance on medium or large image sizes is similar to the other three models. This drawback can be observed in the original paper [6,7]. For instance, the SSD detector, trained on Microsoft COCO and finetuned on PASCAL VOC2012 and PASCAL VOC2007 dataset, has a lower performance on the small object like plants (57.2%AP) and bottles (60.8% AP) while on the larger objects, the performance is better, such as aero (90.7%AP) and horse (89.0%AP).

Finally, regardless of the recognizing speed, on the performance of accuracy, the Faster R-CNN with Inception Resnet v2 and R-FCN with Resnet v1 50 have similar performance and work best within the five detectors.

## 5. Conclusion

In this paper, an experiment on comparison of five traffic sign deep-learning-based detectors with different feature extractors is present. The accuracy is the main aspect compared in this paper. The models compared were pre-learned from the Microsoft COCO dataset and then optimized using the GTSDb dataset with the aim of traffic sign detection and classification. The traffic signs were categorized as three groups: mandatory, prohibitory, and danger signs.

Accuracy which is measured by mAP is calculated through the dataset PASCAL VOC 2010. With Inception Resnet V2 feature extractor, the Faster R-CNN gain the mAP of 95.77% at top level and the R-FCN with Resnet v1 101 extractor has a similar mAP of 95.17% as the first rank detector. Then, the SSD with Resnet v1 50 and YOLO v2 have poor performance in detecting small objects which is the drawback shown in the original paper. Finally, the bias of the GTSDb dataset is observed that causes all five detectors to perform worse in mandatory traffic signs than the other two.

In future work, other deep learning-based architectures in recent years that have good performance in object detection can be researched such as YOLO v7 [17] and adjustment to the traffic sign detection will be applied for analysis.

## References

- [1] Chunsheng Liu, Shuang Li, Faliang Chang, & Yinhai Wang. (2019). Machine Vision Based Traffic Sign Detection Methods: Review, Analyses and Perspectives. *IEEE Access*, 7, 86578–86596. <https://doi.org/10.1109/ACCESS.2019.2924947>
- [2] de la Escalera, Moreno, L. E., Salichs, M. A., & Armingol, J. M. (1997). Road traffic sign detection and classification. *IEEE Transactions on Industrial Electronics* (1982), 44(6), 848–859. <https://doi.org/10.1109/41.649946>
- [3] Zhou, & Deng, Z. (2014). LIDAR and vision-based real-time traffic sign detection and recognition algorithm for intelligent vehicle. 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), 578–583. <https://doi.org/10.1109/ITSC.2014.6957752>
- [4] Shaoqing Ren, Kaiming He, Girshick, R., & Jian Sun. (2017). Faster R-CNN: Towards

- Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137–1149. <https://doi.org/10.1109/TPAMI.2016.2577031>
- [5] Dai, Li, Y., He, K., & Sun, J. (2016). R-FCN: Object Detection via Region-based Fully Convolutional Networks. *arXiv.org*.
- [6] Liu, Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. *Computer Vision – ECCV 2016*, 21–37. [https://doi.org/10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2)
- [7] Redmon, & Farhadi, A. (2016). YOLO9000: Better, Faster, Stronger. *arXiv.org*.
- [8] Zou, Chen, K., Shi, Z., Guo, Y., & Ye, J. (2023). Object Detection in 20 Years: A Survey. *Proceedings of the IEEE*, 1–20. <https://doi.org/10.1109/JPROC.2023.3238524>
- [9] Lin, Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (n.d.). Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014* (pp. 740–755). Springer International Publishing. [https://doi.org/10.1007/978-3-319-10602-1\\_48](https://doi.org/10.1007/978-3-319-10602-1_48)
- [10] Everingham, Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2), 303–338. <https://doi.org/10.1007/s11263-009-0275-4>
- [11] Houben, Stallkamp, J., Salmen, J., Schlipsing, M., & Igel, C. (2013). Detection of traffic signs in real-world images: The German traffic sign detection benchmark. *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 1–8. <https://doi.org/10.1109/IJCNN.2013.6706807>
- [12] Shelhamer, Long, J., & Darrell, T. (2017). Fully Convolutional Networks for Semantic Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 640–651. <https://doi.org/10.1109/TPAMI.2016.2572683>
- [13] Redmon, Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788. <https://doi.org/10.1109/CVPR.2016.91>
- [14] Ioffe, & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv.org*.
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- [16] Szegedy, Ioffe, S., Vanhoucke, V., & Alemi, A. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv.org*.
- [17] Chien-Yao, Bochkovskiy, A., & Hong-Yuan, M. L. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv.org*.