Heuristic Based Localization and LeNet Classification vs YOLOv5 for Road Sign Detection

Lewis Koplon
School of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85719 USA
ECE 523: Engineering Applications of Machine Learning and Data Analytics

***Abstract*—This paper evaluates the performance of a single stage detector against a two-stage detector. The single stage detector utilized is the state-of-the-art YOLOv5 algorithm, whereas two-stage detector has been developed to take advantage of image processing and heuristics to localize road signs and a LeNet classifier to label them.**

***Index Terms*—LeNet, Heuristic, YOLOv5, Detection, Localization, Classification, Image Processing, Object Recognition.**

I. Introduction

Road sign detection is an important issue for Autonomous Vehicle (AV) applications, it is necessary to provide a robust algorithm that can correctly detect road signs in varying environmental conditions with high confidence and accuracy to ensure the safety of passengers in said AVs. The task of object detection can be split up into two smaller tasks, image classification and object localization. Image classification can be achieved via an algorithm that takes an image of an object and outputs a label corresponding to what it is. Object localization is the process of locating the objects in the image and outputting their location via bounding box coordinates. Moreover, object detection is the culmination of the two previous tasks, locate the objects and classify them, outputting the correct bounding boxes and their respective labels. This can be achieved through two implementations, two stage or single stage detectors. Two stage detectors are algorithms that localize the object first, then try to classify the image based off that localization, whereas one stage detectors are used to localize and classify in the same step. This paper will look at two methods of detecting road signs, the first approach is a two-stage detector, that uses heuristic based localization in conjunction with a LeNet classifier, while the other is state of the art single stage detector known as the You Only Look Once (YOLOv5) algorithm developed by Glenn Jocher.

II. Topic Flow

In section III, the data-set will be explained in detail regarding the dimensions of the images, and the layout of the respective target vectors, as well as the distribution of classes in the training set. Section IV will briefly explain the concept behind the two stage detector, and will go into further detail about the design of the this particular two stage detector. Section V will explain the YOLO algorithm and its architecture, as well as some of the metrics used to measure its performance such as mAP (mean Average Precision). Section VI interprets the results of both approaches and compares them, highlighting some of the benefits each model may have over the other. Section VII concludes this paper by addressing where the two stage detector can be improved in order to be more competitive with YOLOv5.

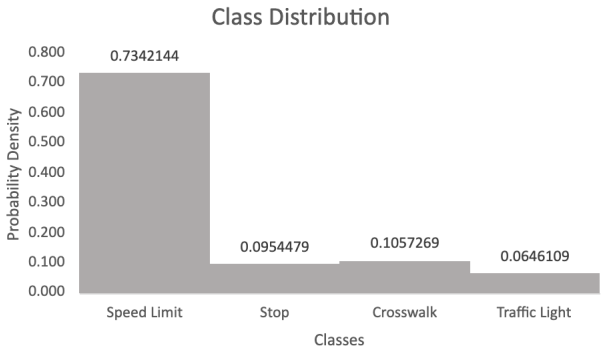


Fig. 1: Distribution of Classes in the Training Data-Set

III. Data-Set

The data-set utilized was retrieved from a website that hosts machine learning competitions, Kaggle.com, this data-set contained images that consisted of either singular road signs, or numerous road signs. The target values for their corresponding training images took the form of $class, x_1, y_1, x_2, y_2$ where class takes the value of an integer defined by this relationship: $\{0 : speedlimit, 1 : stop, 2 : crosswalk, 3 : trafficlight\}$. Note that the values x_1, y_1, x_2, y_2 are the coordinate points for the bounding box, (x_1, y_1) is the top left coordinate, and (x_2, y_2) is the bottom right coordinate of the bounding box. The distribution of this data-set can be viewed in figure 1, the probability density function shows that data-set is imbalanced and a large amount of the images contain speed limit signs. This is corrected through under-sampling the speed sign images in order to have a uniform distribution. These data points were used to train a variation of the LeNet model and were used to fine tune the YOLOv5 model. The way that data was pre-processed and utilized for training was different for each model, and will be explained in the upcoming sections.

IV. Heuristic Localization and LeNet Classification

This procedure makes use of a two stage detection approach. Firstly, the object is localized through some means, in this case

image processing is utilized. The localized shape is cropped from the original image and passed through the classifier to determine the class it belongs to. Each stage of the detector will be explained in the following sections.

A. Heuristic Localization

Some of the defining characteristics of a road sign include but is not limited to: shape, color, size, and location. The idea behind heuristic based localization is to utilize these defining characteristics to put a bounding box around said sign while minimizing any noise in the image.

In this application, the defining characteristic utilized to localize the sign is its shape and its size. This is accomplished by grey scaling the image, removing any influence of color, and adding a Gaussian blur which aims to reduce the amount of noise in the image. Edge detection is utilized via an Open CV function called Canny Edge Detection, which returns an image whose objects' edges are shown if they fall between a high and low threshold values. This edge highlighting allows for the use of two other function, Find Contour and Approx Poly Dp. Find Contour returns a list of contours found in the image, this list is then passed to the Approx Poly Dp which takes the contours and returns a list of vertices. These vertices can be used to determine what shapes exist in the image, they are counted and used to rule out noise. For example, if a shape has four vertices then it is said to be a square, if it has more than 8 than it is said to be a circle or an octagon, any other number of vertices will result in the shape being discarded as noise. The area is then calculated for each shape in the image, if the area is below a threshold, in other words to small to be a sign, it is thrown out. This area was decided by looking at the most extreme cases of road signs in the data-set. Figure 2 shows varying localization results, figure 2.a and 2.b, are instances of very good object localization, whereas figure 2.c is an example of very poor object localization due to the edges of the speed sign not having contours because of the closeness of the sign above it as well as the pole it is attached to.

(a) Very Accurate Localization

(b) Another Instance of Good Localization

(c) Failure to Localize

Fig. 2: Localization Results

Fig. 3: Image processing, show gray scale and contour image

B. LeNet Classification

The following section will discuss the network architecture, the loss function, the optimizer used, as well as some of the tactics utilized to improve the validation via Hyperparameter Tuning.

1) Network Architecture: The first layer is the two dimensional convolutional layer that takes a single channel image of shape 100 pixels by 100 pixels. The first convolutional layer has a kernel size of 12 by 12, and 16 total filters, the activation function is a rectified linear unit, usually used in object detection problems. This layer outputs a feature map of size 84 by 84. The second layer is a max-pooling layer, that is responsible for down sampling the input from the previous layer, this is achieved by calculating the maximum value for patches of a feature map, resulting in a down sampled or pooled feature map that is 21 by 21 in size. The next layer is another convolutional layer that uses a kernel size of 3 by 3 with 16 filters, and the same activation function as the first convolutional layer, and outputs an 18 by 18 dimension feature map to the next pooling layer. This pooling layer takes that output and finds the maximum values in patches of 2 by 2, resulting in a down sampled 9 by 9. This 9 by 9 gets flattened into a one dimensional vector that is then connected to a dense layer of some number of neurons, with an activation of "relu". Finally these neurons are then connected to the output layer which is a dense layer that consists of 4 neurons, one for each corresponding class. At this last layer, the activation function used is the softmax activation function used for classification, which takes the numbers from the logits and converts it into a probability of each class corresponding to the given image.

The reason why the second to last layer is specified as some number of neurons, is due to the fact that the training of the neural network uses hyperparameter tuning to determine these neurons. This network architecture is shown in figure 4.

2) Loss Function: The loss function utilized is categorical cross entropy for one hot encoding. This is the standard loss function for multi-class classification problems. Categorical cross entropy calculates the average difference between the actual and predicted probability distributions for all classes, it ensures that one class can only be correct out of the 4 possible classes for this application.

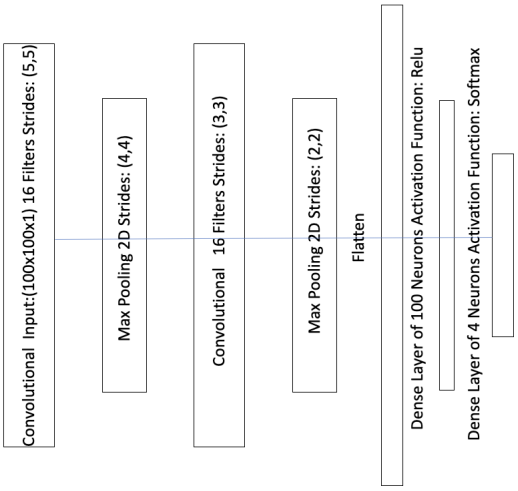


Fig. 4: LeNet classifier used to train and classify the road signs

3) Training and Validation: The model was trained on an under-sampled portion of the data-set due to the heavy imbalance of data-points belonging to speed limit class. The validation data was presented in a fashion where each class was equally represented. These two steps were taken in order to prevent a situation where the model could earn a high precision and accuracy by guessing the class that was represented the most. An important portion of the training process was the use of hyperparameter tuning which optimized the learning rate and the second to last layer's amount of neurons to produce the highest validation precision and binary accuracy. It is important to note that the road signs being trained on and used for validation are perfectly cropped so that the image is of just the road sign itself and contains minimal environmental noise. This later becomes an issue because of the images being sent from the localizer to the classifier, contain noise due to the inaccurate localization.

As a result, model achieved a 99 percent training precision and accuracy, whereas on the validation set the precision was 90.5 percent and the accuracy was 92 percent. Ideally the model would have a 97-95 percent validation precision, and improvements to achieve this will be discussed in the Conclusions and Improvements section.

C. Results

The combination of a heuristic localizer and a LeNet classifier yielded poor results. The classifier itself performs very well at determining what the image is, however, while the localizer does localize the road signs often, the bounding box which is eventually cropped and sent to the neural network for classification often includes too much noise for the classifier to classify properly. This noise takes two forms. The first form is when the bounding box is not "tight", meaning that the image is not just of the sign, but includes too much background or portions of the original image that are not relevant for the classifier. This is shown in figure 5, note how much irrelevant information is still in the cropped image, the whole bottom half of the image includes no sign and all background. The

Fig. 5: Noise in Localized Images

Fig. 6: Heuristic Localization and LeNet Classification Prediction (Good)

second form, is when the localizer just localizes something that is not a road sign, such as a tire, a portion of a car, or portion of the original image that passed the shape and area criteria. That then gets passed to the classifier, which results in the net having to choose one of the classes that "fits" the best.

Some of the localizations are accurate in their bounding boxes and corresponding labels, this is shown in figure 6. While others once again are not so accurate as shown in Figure 7. Figure 7 shows that there is one good localization and classification, while part of the bus gets localized and classified as a speed sign, when alternatively it should be disregarded as noise and the bounding box should be discarded.

Improvements to mitigate these issue are discussed in the Conclusion and Improvements section.

Fig. 7: One Accurate Classification and Once Inaccurate

V. You Only Look Once

YOLOv5 is a real-time object detection framework whose name is earned through the fact that each image is only passed through fully convolutional neural network once. It boasts the ability to quickly and accurately classify and localize objects at high frames rates which is critical in autonomous vehicle applications. Due to there not being a paper written with YOLOv5, the network architecture of YOLOv4 will be explained, since YOLOv5 is a revision and expansion on top of YOLOv4.

A. Network Architecture

For modern detectors, the deep neural network is composed of two portions, the backbone and the head. The backbone is a feature extractor network utilized to find the features in an input image, these networks are usually trained on ImageNet. Some well known nets used as backbones are DarkNet or VGG16. The head can be either a one stage detector or a two-stage detector and is used to predict the classes of the object as well as the bounding boxes [1].

Where the YOLOv4 algorithm differs from other object detectors is its use of a neural network between the head and the backbone, referred to as the neck and takes the form of a Feature Pyramid Network. Feature Pyramid Networks are feature extractors that uses a pyramid like concept, with top down and bottom up pathways. Progression through the bottom up pathway results in images that have greater semantic value, however have lower resolution, progressing from the top down is exactly the opposite relationship [2].

YOLOv4 utilizes CSPDarknet53 for its backbone which employs CSPNet. CSPNet was created to solve the issue of repeated gradient information within the optimization for the network via creating a feature map of the base layer and partitioning it into two parts. The first part passing through a dense layer and a transition layer, while the second part is combined with the result of the first [3].

For its Neck, YOLOv4 utilizes SPPNet which is a spatial pyramid pooling that removes image dimension constraints by outputting images of the same fixed dimension to the next neural network which would be the head of the network. YOLOv3 is the single stage object detector that functions as the head of YOLOv4 [6].

B. Training and Validation

The authors who wrote the YOLOv4 algorithm and its respective paper refer to the methods utilized to improve training performance without a resulting cost in inference as a "Bag of Freebies" (BoF). What is in the BoF will be explained for both the backbone and the detector [1].

1) Bag of Freebies for Backbone: CutMix, Mosaic data augmentation, DropBlock regularization, and Class label smoothing were the freebies utilized to increase training performance of YOLOv4's backbone [1].

CutMix is a data augmentation strategy that uses cropped images to cover the training images, in other words, if you were trying to build a detector that would localize and classify cats and dogs, utilizing CutMix would present your model with an image of a dog with a portion of a cat or another dog overlaid onto the image for training [1].

Mosaic is another data augmentation method that was created by the YOLOv4 team, this method combines and mixes 4 training images by stitching portions of the images together into one image. This results in creating 4 different contexts for the model to train on, which they state helps improve the detection of object outside of its usual context.

DropBlock regularization is a form of regularization for convolutional neural networks, where continuous regions of feature maps are removed in order to prevent under or over-fitting.

2) Bag of Freebies for Detector: CIoU-loss, CmBN, DropBlock regularization, Mosaic data augmentation, Self-Adversarial Training, cosine annealing scheduler and hyper-parameter tuning, are all the freebies utilized to improve the training performance [1].

CIoU-loss is the loss function used for the bounding box regression in the YOLOv4 model, CIoU or Complete Intersection over Union measures the overlap area, the distance and the respective aspect ratio between the prediction the net makes and the ground truth.

CmBN or Cross mini Batch Normalization is a modified Cross Batch Normalization that looks at the statistics between "mini-batches" in a batch, and uses them to improve the speed of the neural network through normalization of the networks inputs.

Self-Adversarial Training is another data augmentation technique created by the YOLOv4 team that works by an Adversary augmenting the original image rather than the weights of the network. The image is manipulated to fool the detector by making it think that there is no object to be detected, the second portion of this process is to have the network train on this image as it normally would [1].

Cosine annealing scheduler is a learning rate scheduler, learning rate schedulers adjust the learning rate between epochs of training to improve the training metrics. Moreover, cosine annealing scheduler uses an aggressive approach to start with learning rate very high, and then dropping it near zero, following by returning it to a very high value, the relationship between epoch and learning rate is sinusoidal.

Hyperparameter tuning is a method that allows for the optimization of network parameters to control the behavior of the neural network and produce optimal results. The YOLOv4 team uses 8 million training steps, with a batch size of 128 and mini-batch size of 32, an initial learning rate of .1, while the momentum is .9 and the decay is .005 for optimization search [1].

3) Training and Validation Results: The training results for the YOLOv4 algorithm trained on the Road Sign data-set specified above are shown in figure 9 through 11 on the last page. These figures display the training and validation losses for bounding boxes and classifications which are then plotted against the epochs. As shown in the figures, the bounding box loss after 100 epochs is near zero, considering that the loss function being utilized is the CIoU, the inference can be made that the bounding boxes are precise and accurate. Furthermore,

the class loss, is also a near zero value instantiating that the predictions on classifications is also precise. These figures show that the bag of freebies chosen by the team who developed YOLO created an effective training routine that allowed for accurate predictions and little to no over fitting shown by the training and validation accuracy being very similar. Figure 11, shows the increase in mean Average Precision as the number of epochs increase reaching values of over 90 percent. Mean Average Precision is the metric used for object detection models, it is the average of the average precision across all classes. MAP is shown in the equation below, note C is the number of classes and c a class from C.

$$mAP = \frac{1}{C}\sum_{c=1}^{C} AP_c$$

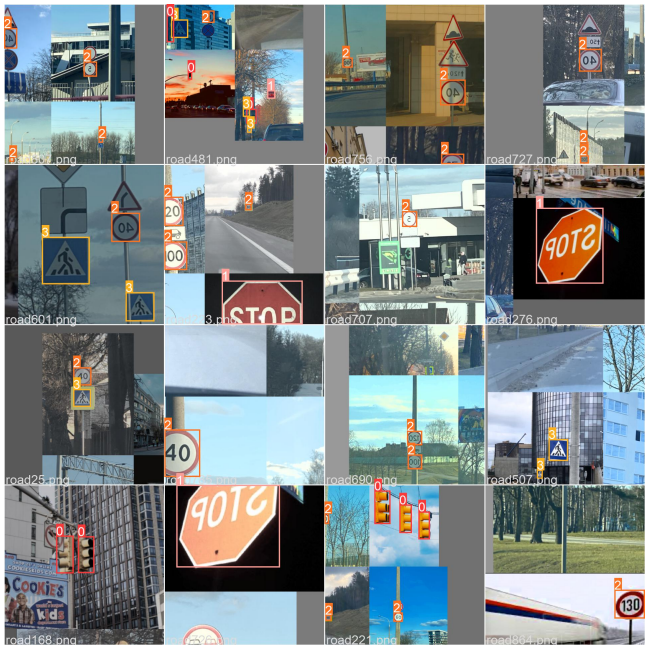
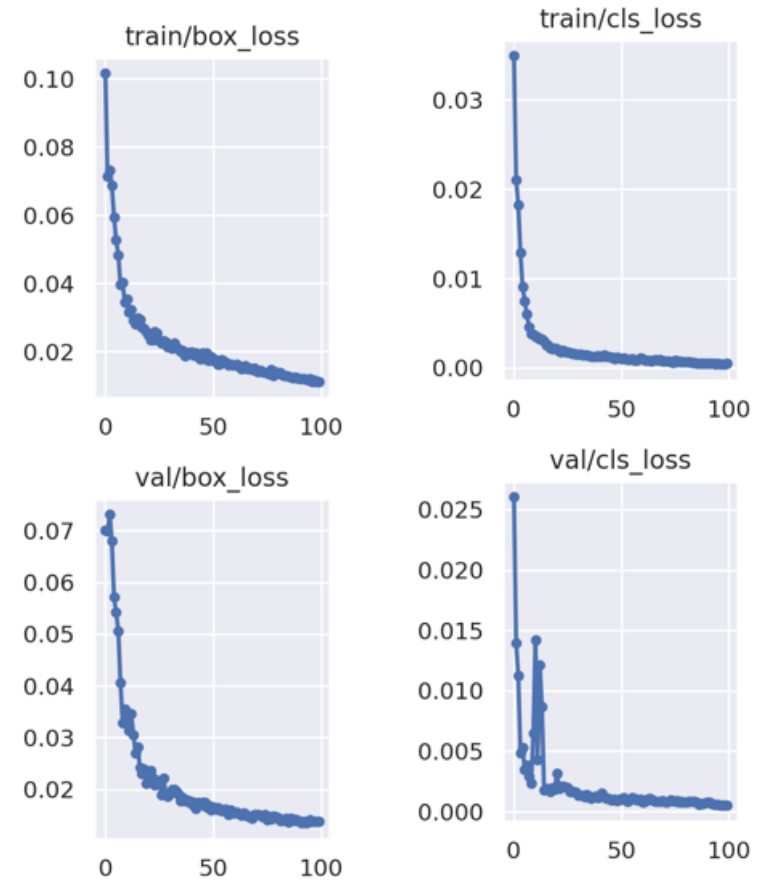


Fig. 8: Batch Training for YOLOv5

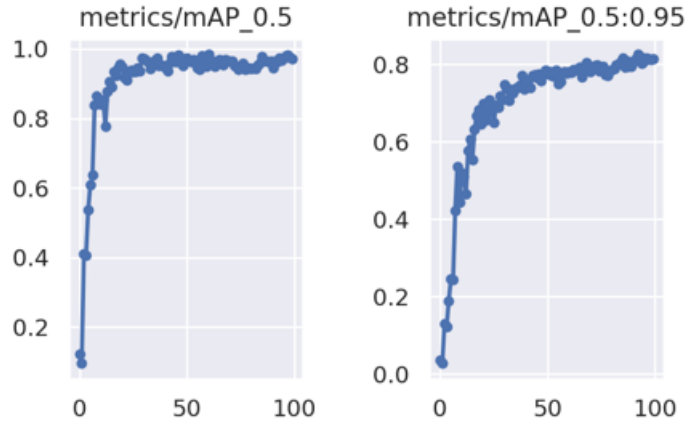
C. Results

The predictions made by the YOLOv5 algorithm showed the accuracy of the model as well as its robustness to varying environments in the images thanks to its expansive training process. Near and far road signs in images were localized with little to no error, and the predictions for these road signs were correct even in hard to distinguish scenarios such as partial obscuring of signs. Figure 12, shows the results of YOLOv5 predictions on images it has not been tested on. Numerous road signs are localized and classified precisely. With respect to sign being obscured, the second image in figure 12 shows the speed limit sign localized and classified even though it is only partially visible due to the CutMix data augmentation used in the training process.



(a) Bounding Box Loss for Training and Validation vs Epochs

(b) Classification Loss for Training and Validation vs Epochs



(c) Mean Average Precision for Training and Validation vs Epochs

Fig. 9: Training and Validation Results

VI. Results Comparison

The YOLOv5 model predicts far more accurately than this. As stated in section 4.C the localizer produces noisy images that the neural network was unable to properly classify which highlighted two issues, the first being the two stage

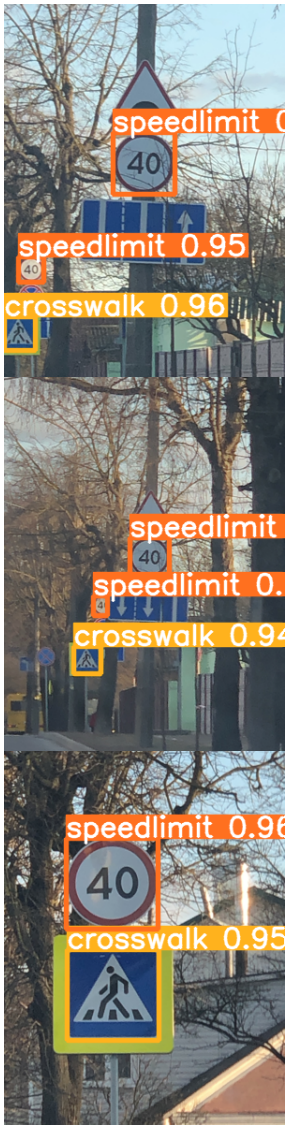


Fig. 10: YOLOv5 Localization and Classification Predictions

detector's inability to properly localize, and the classifier's inability to properly generalize. YOLOv5 does not have these issues, all localizations for the tested images are concise, and all classification are correct. This is possibly due to how YOLOv5 learns to apply bounding boxes, as it learns the underlying relationship between objects in the image, and their classes, whereas the heuristic localizer tries to utilize the intrinsic properties of the signs for localization and then passes a cropped image to the classifier that only knows how to differentiate from different classes of signs, and not the background noise or contextual information.

YOLOv5 has the capability to accurately track road signs in AVs at high frame rates, however, the higher the frame rate and the faster the images are sent to YOLOv5 will result in a lower mAP. The two stage detector was not tested in this aspect due to its poor accuracy in the single image trials, however, the overhead of using heuristics and a small parameter model to predict classes is a small one. Meaning that the two stage detector would be able to quickly localize and predict, however those predictions would not be accurate.

VII. Conclusion and Improvements

The LeNet model is resilient in classifying signs that have been perfectly localized, this is shown in the training process, via the precision and accuracy metric discussed in section 4C. The issue with the two stage detector is its inability to do perfect localization via heuristics. Localization via heuristics is very difficult due to varying complexities: light interference, inconsistent edge detection of objects at far distances as there is no way of generalizing the thresholds for edge detection. Once again the imperfect localization often results in two different types of "noisy" images, the first being too much environmental information, the second being a cropped image of no sign. The neural net will classify both noisy images incorrectly, and this is a result of how the data-set was processed and how the model was trained. Note the model was trained on cropped images of just the signs as its only job was to classify which sign it was. The ways of improving the model all together will be discussed here.

The use of data augmentation on the training and validation data-set may result in a strengthened ability of inferring for the neural network. The amount of noise produced by the image localization is too much interference for the classifier to correctly classify. A potential solution may be reached via adding noise to the data-set in the form of simulating light interference, partial obstruction of the sign, to differing orientations of the sign. These data augmentations aim to strengthen the generalization capability of the model.

Another method is to utilize the confidence the model has in what it is predicting. If the label is below a threshold value of say .6, then ignore the bounding box and do not display the label.

Lastly, the model could be rebuilt to be a convolutional 3d neural network, where color is an extra feature to be trained on, as it currently only trains single channel gray-scaled images, this inclusion of color may help classify signs more accurately.

References

[1] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection" CoRR, vol. 2004.10934, 2020.

[2] J. Hui, "Understanding Feature Pyramid Networks for Object Detection (FPN)," Medium, 30-Apr-2020. [Online]. Available: https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c.

[3] S.-H. Tsang, "Review-CSPnet: A new backbone that can enhance learning capability of CNN," Medium, 04-Sep-2021. [Online]. Available: https://sh-tsang.medium.com/review-cspnet-a-new-backbone-that-can-enhance-learning-capability-of-cnn-da7ca51524bf

[4] J. Brownlee, "Snapshot ensemble Deep Learning Neural Network in python," Machine Learning Mastery, 27-Aug-2020. [Online]. Available: https://machinelearningmastery.com/snapshot-ensemble-deep-learning-neural-network/: :text=The

[5] S. Jia, Y. Bai, and J. Zhang, "Feature comparison based channel attention for fine-grained visual classification," 2020 IEEE International Conference on Image Processing (ICIP), 2020.

[6] P. R. Dedhia, "Understanding SPPNet for Object Detection and Classification," Medium, 26-Mar-2021. [Online]. Available: https://towardsdatascience.com/understanding-sppnet-for-object-detection-and-classification-682d6d2bdfb.