Exploring Transfer Learning Approaches for Traffic Light Classification

Landon Zhang

Abstract

Accurate traffic light detection is critical for autonomous driving, but achieving high performance in real-time systems remains challenging due to constraints in processing speed and data volume. This study examines the impact of bounding boxes and customized dense layers on a VGG16-based transfer learning model for traffic light classification. Using a dataset of 2,600 traffic scene images from Beijing, we evaluated four model variants: base VGG16, base with bounding boxes, base with added dense layers, and base with both modifications. Results demonstrate that incorporating bounding boxes substantially improves performance, achieving 97–98% accuracy compared to approximately 75% for models without bounding boxes, while also reducing loss. Adding small dense layers had minimal effect, indicating that the VGG16 base is already proficient for extracting relevant features. The confusion matrices show that bounding boxes also enhance recognition of less frequent yellow lights. These findings highlight the importance of input localization with bounding boxes over minor network modifications with dense layers for efficient and accurate traffic light detection. The study provides insights for designing transfer learning strategies in real-time autonomous driving applications.

1. Intro

Artificial intelligence has been at the forefront of technology for several years. Its capability to complete seemingly impossible tasks and incredible potential has led to extensive work done by researchers and developers. Autonomous driving has been one of the biggest areas of focus relating to artificial intelligence, with its need to constantly identify traffic lights, signs, and other objects calling for high-level image detection AI models. However, a prevalent problem regarding AI in autonomous driving has been its ability to perform at high speeds while maintaining accuracy and effectiveness. Hawlader et al. [1] stated that the large amount of real-time sensory data needed to maintain a safe driving vehicle such as camera data, LiDAR, and radar is constrained in both cost and power consumption. As a result, a trade-off seems inevitable: running detections locally require sacrifices on perception quality, in favor of real-time operation. Although it is very hard to come up with a one-size-fits-all solution, developers have come up with many unique ways to tackle this problem.

Transfer learning has been one of the most effective solutions, which involves taking a much larger pre-trained model and combining it with smaller, fine-tuned models to perform tasks much more efficiently than with one or the other. Common ways to apply transfer learning to the task of traffic light classification include separating the original image into smaller sections and then classifying from the smaller sections. Gokul et al. [2] used transfer learning with Faster R-CNN and YOLOv4, two state-of-the-art detectors that use this approach and discovered that R-CNN performs better with precision while YOLOv4 performs better with real-time applications. Both of

these models are convolutional neural networks (or CNNs), which use convolutional layers, activation functions, pooling layers, and fully connected layers to achieve an output [3]. Niu and Liu [4] proposed using both YOLOv5 and AlexNet together, with YOLO to detect the traffic light location and then using AlexNet to accurately classify it, achieving an average accuracy of 87.75%.

Here, I present a comprehensive understanding of the impact bounding boxes and dense layers have regarding accuracy and transfer learning on top of a large base image detection model. I discuss the details of our approach in the Methods section and present results on the VGG model with and without both bounding boxes and dense layers for the problem of traffic light detection. I discuss some implications of these results for real-time self-driving applications.

2. Methods

This work investigates the impact of implementing VGG16 with bounding box features and customized dense layers, aiming to determine which approaches work best when paired with transfer learning. I chose VGG16 as my model because, although VGG16 is not the most recent architecture for image recognition and object detection tasks, its simplicity and wide availability make it the perfect baseline model for studying transfer learning strategies for traffic light detection. Like a typical Convolutional Neural Network, VGG16 contains five convolutional layers, three dense layers, and a final softmax activation function for a total of 138 million parameters[5]. For my task of traffic light classification, I first imported a pre-made VGG16 model from keras, then replaced the original VGG16 top layer with my own customized layer. Since the original VGG16 top layer consists of excessively detailed dense layers which is overkill for a relatively simple task like classifying three colors, I dramatically reduced the units for my dense layers, changing it from 4096, 4096, 1000 to 50, 20, and 3 respectively. This allows me to dramatically speed up the time while only sacrificing a small bit of accuracy.

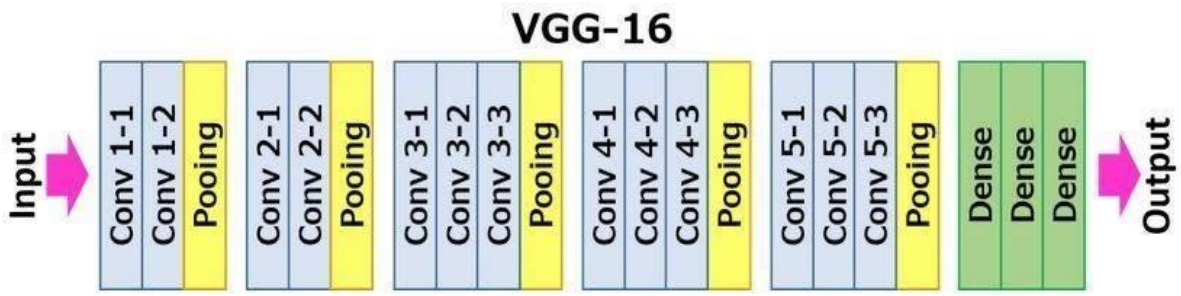

Fig.1: VGG16 architecture [5]

The dataset I chose includes 2,600 images of ordinary traffic scenes in Beijing, China. The traffic lights are tiny and hard to see, but the dataset comes with a JSON file that lists each image's color along with coordinates for the bounding box. Each image comes with three classes for the colors: class 0 as red, class 1 as green, and class 2 as yellow. Since this model already comes with bounding boxes, it allows me to analyze the effects of bounding boxes without having to create an entire new model for that purpose. Something interesting to note in

the dataset is that there were far fewer yellow light images compared with the red and green light ones. The train dataset only contained 137 yellow lights.

In order to clearly see the impacts bounding boxes and customized dense layers have on the model, I designed four different models using the VGG16 architecture. These include the base model alone, base with bounding box inputs, base with two additional dense layers, and a full combination of both modifications. Both dense layers used the ReLU activation function, with the first one containing 50 units and the second one containing 30. This allows a controlled evaluation of each component and makes it easy to see the variations between the models. When implementing the dataset in my code, I chose to split off 20% of the images to use as testing, allowing me to determine true classes compared with predicted classes.

Listed below are the accuracy and loss graphs for each four models, listed in order as: base only, base and dense, base and bounding box, and base with bounding box and dense. An analysis of these graphs shows that the accuracy and loss graphs tend to grow logarithmically/exponentially, with lots of change in the first ten epochs and far less in the final fifteen.

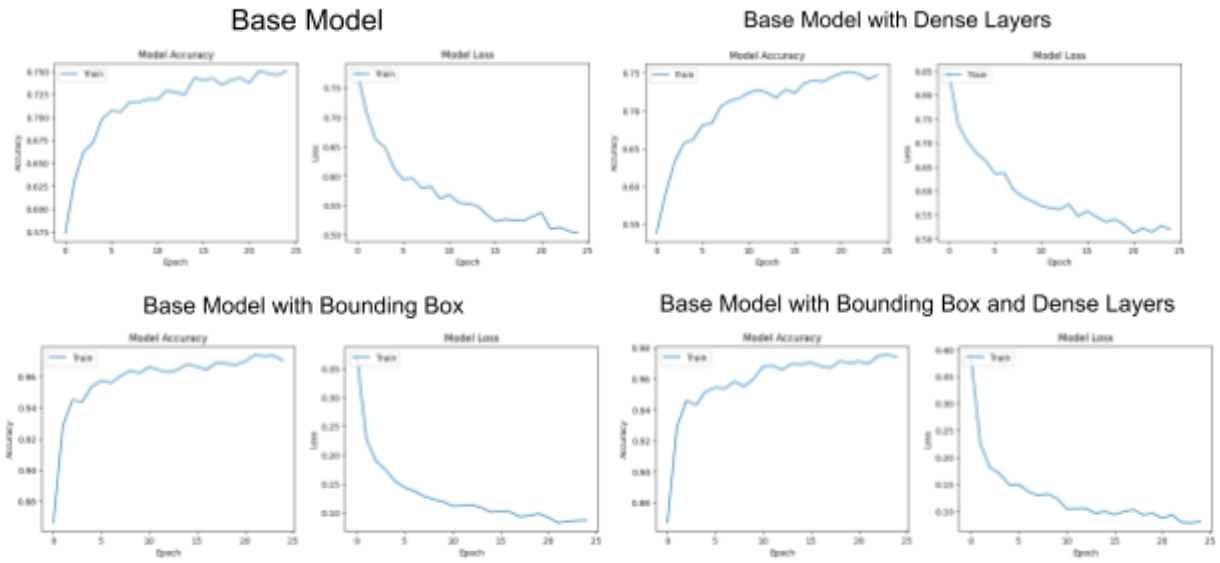

Fig.2: Accuracy and Loss Plots

3. Results

Based on the training and loss curves, it is clear that restricting the images to the bounding boxes significantly improved the metrics compared to the dense layers. For the first two models without bounding boxes, each model only reached around a 75% accuracy, with little difference between the two. Loss was also higher, peaking around the 0.50-0.55 range for both. However, the two models that included bounding boxes performed much better, with accuracies in the 97-98% range and losses in the 0.1 range. Again, those two models had little difference in terms

of accuracy and loss, which is largely because VGG16 already performs well for color recognition, so adding small dense layers would have minimal effect on the performance.

Below, I can see the difference between the normal images and the images cropped to bounding box coordinates.

Fig.3: Sample Images

Below, I take a look at the confusion matrices of the four models. The diagonal boxes each show true positives, which are predictions the model makes that match the actual color value assigned in the dataset. Each other box shows, based on the axes, which color (0 for red, 1 for green, and 2 for yellow) was wrongly detected. For example, the upper middle box shows how many red lights were falsely identified as green lights.

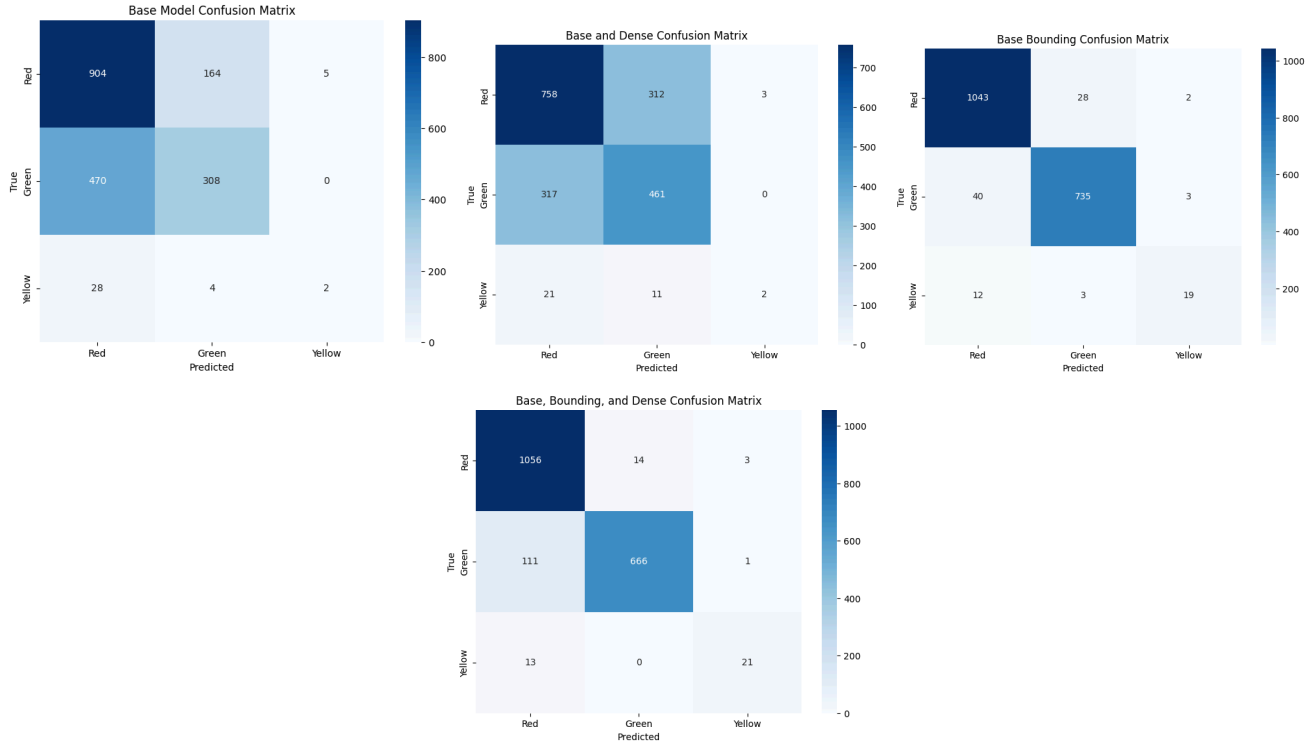

Fig.4: Confusion Matrices

4. Discussion

Based on both the accuracy/loss plots and the confusion matrix, I can see that using bounding boxes resulted in much better results, and using dense layers had little significant impact. It is pretty obvious why using bounding boxes would have such a big impact, as the lights in the full image are barely noticeable, with the color covering only a couple of hundred pixels at best. However, after cropping the image to the bounding box, the light and color pretty much fill the whole screen, making it substantially easier for the model, thus resulting in higher accuracies and lower loss values. Based on our accuracy/loss plots, the models with bounding boxes had higher overall accuracies and lower overall loss, with almost 20% more accuracy and 0.4 less loss. Our confusion matrices also support this claim, as the models without bounding boxes performed much worse at guessing green lights than those with bounding boxes compared with those with bounding boxes.

When looking at models with and without dense layers, I can see that it had little effect on the overall effectiveness of the models. For the two models without bounding boxes, the confusion matrices show that the model with dense layers had more true positives for green lights, but it also had more false positives for both red and green. As a result, I cannot determine any significant differences between the two. The main cause of this is because the VGG16 base

model is already very strong, with a complex architecture and millions of parameters, so adding the two small dense layers would have minimal impact on the overall outcome.

Another interesting thing to look at is the model's inability to accurately predict yellow lights. The base and dense models simply could not tell and almost always treated them as red lights. The models with bounding boxes performed better, accurately predicting yellow around 60% of the time. The main reason for this was because the dataset I used simply lacked images that had yellow lights, with only 137 training images out of the 7547 and only 34 test images out of the total 1885. A lack of data makes the model more prone to experimental noise, making it much harder for the model to accurately predict.

5. Conclusions

In this paper, I used transfer learning to train four different models, each using a combination of bounding boxes and/or customized dense layers to analyze their impacts on a VGG16 base model. After training and testing all four models, I determined that using bounding boxes tend to have a much more significant effect on the model's performance and in turn would benefit in a real-time scenario, while the customized dense layers have less of an impact. This tells us that models that can accurately identify the traffic light will have a much better chance of predicting the correct light color.

Although the current model is able to correctly identify traffic lights for the most part, the current dataset's lack of yellow lights restricts the model to only be able to identify red and green lights, which is not feasible for real-life scenarios. Furthermore, the base model used, VGG16, is a pretty outdated model which could definitely be improved upon by using newer models tailored to more specific tasks like traffic light recognition.

References

1. Faisal Hawlader, François Robinet, Raphaël Frank, "Leveraging the edge and cloud for V2X-based real-time object detection in autonomous driving", *Computer Communications*, vol. 213, pp. 372-381, Jan. 2024, https://doi.org/10.1016/j.comcom.2023.11.025.

2. R. Gokul, A. Nirmal, K. Bharath, M. Pranesh, and R. Karthika, "A Comparative Study between State-of-the-Art Object Detectors for Traffic Light Detection," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Vellore, India, 2020, pp. 1-6, https://doi.org/10.1109/ic-ETITE47903.2020.449.

3. S. Ren, K. He, R. Girshick, and J. Sun, "The Fundamental Guide to Faster R-CNN", *Viso Suite*, October 5, 2024. [Online]. Available: https://viso.ai/deep-learning/faster-r-cnn-2/. [Accessed: June 5, 2025].

4. C. Niu and K. Li, "Traffic Light Detection and Recognition Method Based on YOLOv5s and AlexNet", *Applied Sciences*, vol. 12, no.21, Art. no. 10808, Oct. 2022, https://doi.org/10.3390/app122110808.

5. R. Gupta, R. Angelou, and K. Pandya, “Everything you need to know about VGG16 | by Great Learning”, *Medium*, September 3, 2021. [Online]. Available: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918. [Accessed: October 26, 2025].