International Journal of Informatics and Communication Technology (IJ-ICT)
Vol. 13, No. 2, August 2024, pp. 285~296
ISSN: 2252-8776, DOI: 10.11591/ijict.v13i2.pp285-296

Efficient traffic signal detection with tiny YOLOv4: enhancing road safety through computer vision

Santhiya[1], Immanuel Johnraja Jebadurai[1], Getzi Jeba Leelipushpam Paulraj[1], Ebenezer Veemaraj[2], Randlin Paul Sharance[1], Rubee Keren[1], Kiruba Karan[1]
[1]Division of Computer Science and Engineering, Karunya Institute of Technology and Science, Coimbatore, India
[2]Division of Data Science and Cyber Security, Karunya Institute of Technology and Science, Coimbatore, India

Article Info

Article history:

Received Jan 30, 2024
Revised Mar 13, 2024
Accepted Apr 30, 2024

Keywords:

Artificial intelligence
Autonomus vehicle
Deep learning
Object detection
Tiny YOLO

ABSTRACT

As decades go by, technology advances and everything around us becomes smarter, such as televisions, mobile phones, robots, and so on. Artificial intelligence (AI) is applied in these technologies where AI assists the computer in making judgments like humans, and this intelligence is artificially fed to the model. The self-driving technique is a developing technology. Autonomous driving has been a broad and fast-expanding technology over the last decade. This model is carried out using the tiny you only look once (YOLO) algorithm. YOLO is mainly used for object detection classification. Tiny YOLO model is explored for the traffic signal detection. ROBI FLOW dataset is used for object detection which contains 2000+ image data to train the tiny YOLO model for traffic signal detection in real time. This model gives an improved accuracy and lightweight implementation compared to other models. Tiny YOLO is fast and accurate model for real-time traffic signal detection.

Corresponding Author:

Ebenezer Veemaraj
Division of Data Science and Cyber Security, Karunya Institute of Technology and Sciences
Coimbatore, Tamil Nadu, India
Email: ebenezerv@karunya.edu

1. INTRODUCTION

The integration of artificial intelligence (AI) has shown a rapid advancement in the era of technologies. There is an inescapable march towards technologies that rely on AI algorithms that mimic and make human decisions. This march towards technologies signifies the desire to gear up machines with cognitive abilities resembling human judgment, where the important role is played by the artificial integration of intelligence into the model. The major goal of the research is to identify traffic lights and make appropriate judgments. This research is closely intertwined with the necessity of overcoming the inherent challenges in autonomous driving, specifically targeting traffic signal detection. Convolutional neural networks (CNN) are an essential component of AI solutions for autonomous driving, as they enable the system to identify traffic signals and make informed decisions. This is especially important when tackling the problem of traffic signal detection.

Convolution neural networks, or CNNs for short, are a subset of machine learning. They are made up of several node layers, including input, output, and one or more hidden layers. These node layers are connected with their associated weight and threshold values. CNN is the best neural network out of many artificial neural networks used for different applications. The convolutional layers take advantage of the inherited properties of the images that are fed by the user [1]. Every node that has an output that is greater than the set threshold activates and starts sending data to the network's next layer. If not, none of the data is sent to the network's next tier. CNN is known for its identification and recognition of objects, which makes them more suitable for computer vision (CV) which plays a vital role in autonomous car and facial recognition systems [2]. Their main applications are as follows: i) image and video recognition, ii) recommender systems, iii) image classification, iv) natural language processing, v) brain-computer interface, vii) financial time series, and viii) medical image analysis.

you only look once (YOLO) is a well-known technique for its rapidness and precision that provides real-time identification of objects using neural networks. YOLO approaches recognizing objects as an estimation issue, in which the model delivers probability estimates of classes and bounding box location predictions for the observed photos [3]. CNNs are used by the YOLO algorithm for recognizing objects in real time. Tiny YOLOv4 is a light weight module compared to any other modules used for traffic signal detection. As promised, the approach just requires one propagating forward via a neural network in order to recognize items. Moreover, autonomous vehicles have been more focused on as technology is widespread. This project will be used in traffic signal detection for autonomous vehicles with perfect speed and accuracy.

2. RELATED WORKS

To address the challenge of achieving accuracy in lightweight networks for traffic sign detection, we present an enhanced algorithm-CDYOLO. Using separable convolution with depth and the convolution block attention module (CBAM) attentive function, this light in weight identification of traffic signs method improves upon the YOLOv4-tiny approach in terms of backbone feature extraction and detection head. Moreover, we present a variation called CDYOLO-SP that is optimized for difficult multi-category identification tasks. Our approach involves utilizing the 'CCTSDB + TT100K' transfer learning mode during training to enhance overall performance. When compared to its initial YOLOv4-tiny, our enhanced method continuously produces better results [4].

An entirely new compacted CNN technology that addresses the issues of speed, scale, and accuracy. ReSTiNet, like SqueezeNet, uses fire modules by analyzing how many fire modules are used and where they are positioned within the model to reduce the total amount of features and, as a result, the model's size. The remaining links between a fireplace unit of ReSTiNet are estimated and delicately designed to enhance the spread of features and ensure the broadest feasible movement of data in the algorithm, with the goal of further improving the proposed ReSTiNet in terms of detection speed and accuracy. The proposed technique shrinks the size of the previously popular tiny-YOLO model and improves its following features: i) quick identifying speed, ii) smaller design dimension, iii) fixing excess fitting, and iv) better performance compared to other lightweight [5].

One key piece of technology for gangue sorting is the quick recognition of waste and coals. The enhanced tiny-YOLO-v3 quick recognition method, which incorporates the dilating convolution modules and the squeeze-and-excitation (SE) modules, and the spatial pyramid pooling (SPP) net, is proposed in this paper. This is because the benefits of tiny YOLOv3 include a simple network, quick operation, and good efficacy. On the SPP net, the source snaps are first processed in order to an appropriate size using a single convolution layer. Later, the RGB picture streams are brought closer together by the SE segment, which makes it possible to precisely collect important information and increase network sensitivity. Finally, to improve even more and accomplish quick recognition of coals and waste [6].

During the COVID-19 pandemic, it is important to prioritize personal hygiene by wearing face masks and taking the necessary sanitary precautions. Social distancing is also necessary. CNNs can be used to detect objects, even though this is difficult to monitor and control precisely and effectively. This can be accomplished in part by utilizing tiny-YOLOv4, an object detection algorithm that, without the need for such hardware resources, offers lightning-fast detection for numerous classes of objects. In order to develop a highly effective and precise face mask detection system that can be readily expanded to include extra features like warning systems, this project intends to train and test a custom data set using this algorithm [7].

Through constant deep learning advancements in accuracy and speed. These developments have helped applications like self-driving cars, medical imaging, and pedestrian detection. This paper aims to give a basic overview of object detection techniques, dividing them into two classes: one-stage detectors (like SSD, YOLO v1, v2, v3) that prioritize speed, and two-stage detectors (like RCNN, fast RCNN, and faster RCNN) that emphasize accuracy. To help readers fully grasp the detection and recognition capabilities of the updated YOLO version, YOLOv3-tiny, the paper includes a graphical comparison with earlier approaches [8]. The goal of this research was to create a model that uses convolution neural networks, a deep learning technique, to identify lung cancer with an exceptionally high degree of accuracy. This approach takes into account and closes important gaps in earlier research. We assess the degree of accuracy and loss characteristics of our model in comparison to VGG16, InceptionV3, and ResNet50. Our model yielded a minimal loss of 0.1% and a 94% accuracy rate [9].

For real-time ship detection, we presented a new SDVI algorithm in this paper called enhanced tiny YOLOv3 system. Using the algorithm, there are six different kinds of ships (fishing boats, cruise ships, bulk freight carriers, mineral carriers, and containers) can be accurately classified and placed in real-time during video surveillance. We have made the following adjustments to the YOLOv3 tiny network, based on its original design [10].

This paper presents a real-time vehicle counting method that uses fast motion estimation for tracking and tiny YOLO for detection. Testing our application on low-cost devices, such as the Jetson Nano, is the next step. It is currently running in Ubuntu with GPU processing [11].

This work suggested an improved YOLOv4 goal recognition technique: it added a novel forecasting layer to yolo_head to improve its performance in small target identification; it clustered the dataset's ground truth box using the k-means algorithm to obtain anchors suitable for fabric defect detection; and it strengthened the algorithm's robustness via an integrated data enhancement technique; in order to accomplish precise defect categorization and localization, it cleverly substituted the CEIOU loss equation for the CIOU loss perform and incorporated a convolutional block attention module into the main feature extraction network [12].

This study presents a YOLO-tiny method for the detection of spinal fracture lesions related to cervical (cfracture), thoracic (tfracture), and lumbar (lfracture). To improve the detection of spinal lesions. We enhance the original YOLO-tiny network with three 1×1 convolution layers [13].

For the convenience of such autonomous vehicles, the tiny YOLOv2 algorithm is used here. Which has higher real-time performance and requires less computational resources than the YOLO method. The model has been tested on the test images that are available for the same and trained using the training images in the aforementioned benchmark [14].

The YOLO tiny V3 technique and the Euclidean algorithm are used to create a human detection system that measures the separation between people. You may use the little YOLOv3 network for both object categorization and recognition. The research process consists of the following stages: physical distance detection, data preprocessing, training, and data collection [15].

In the paper, we propose a one-shot detection approach using an intense CNN known as tiny SSD for live imbedded item recognition. A stack of non-uniform sub-networks with highly optimized SSD-based auxiliary convolutional feature layers are incorporated into the network. These layers are designed with the express purpose of minimizing model size without sacrificing efficient object detection capabilities. A well-optimized, non-uniform fire subnetwork stack is included in the system to support this design, increasing efficiency and enabling real-time processing [16].

A sensor is used by the fire early detection system however many sensors are not resistant to fire. cameras were employed in this study to determine the presence or absence of a fire hotspot. We employ artificial intelligence as supplemental technology to evaluate the CCTV data. We suggest using CCTV footage to identify fire hotspots by applying the YOLO method [17]. As an alternative to hardware implementation, this paper presents a novel design for an analysis element. It is suggested that large MAC units and conventional adder trees be replaced with the WALLACE tree-based addition algorithms and the modified booth encoding (MBE) multiplier, respectively [18].

In the suggested model, a new architecture is added by placing convolution layers in various places to improve the network's capacity for feature extraction. In order to address the issue of gradient disappearance or dispersion brought on by an increase in network depth, residual modules are added concurrently. Moreover, the network's high-level and low-level features are integrated to increase the precision of tiny vehicle object detection [19]. YOLO is a method that provides object recognition in real time using neural networks and it is a well-known algorithm for its rapidity and precision. Identifying objects in YOLO is approached as a development issue, wherein the model predicts bounding box coordinates and provides category estimates for the detected images [20].

The fundamentals of CNN are presented in this overview, along with the reasons why they are especially well-suited for vegetative remote sensing. The majority of the section summarizes the most recent advancements and trends, including topics such as spectral resolution, spatial granularity, various sensor types, methods of generating reference data, sources of currently available reference data, CNN techniques, and architectures. Numerous studies have demonstrated that CNN outperforms shallow machine learning approaches. The literature study demonstrated that CNN may be applied to a variety of challenges, including the identification of individual plants or the pixel-wise segmentation of vegetation types [21].

When it comes to a number of factors, including detection accuracy and speed, YOLOv3 is currently in the forefront. While real-time detection on a typical GPU is achievable with YOLOv3-tiny, the accuracy is reduced. By using ShuffleNet's structure as a guide, SS-YOLO creates a feature extraction network that increases accuracy without sacrificing performance. The concept of SENet is incorporated into the structure in order to modify the level of focus on the channel based on the significance of various channels' attributes [22].

To lessen the image’s unwanted noise. We trained a number of models on the dataset before deciding to use ResNet50 since it yielded the best results, with an accuracy of 92.90%. Numerous tests and comparisons with existing studies demonstrate the effectiveness of the suggested strategy [23].

In order to help identify skin illnesses, this project aims to develop the CNN architecture. We made use of a 3,297-image dataset of cancers of the skin that is accessible to the general public on the Kaggle website. We suggest two CNN designs with different parameter counts. There are 2,797,665 parameters in the second architecture and 6,427,745 in the first. The suggested models had an accuracy of 93% and 74%, respectively [24]. The stage of emotional reactance occurs when users of the persuasive system experience denial, disappointment, and dissatisfaction. In order to obtain a better behavioral enforcement. This study suggests a persuasive agent (PAT) design that restricts an individual’s experience of emotional reactance [25].

3. THE PROPOSED ALGORITHM

Step 1: set up environment
- Install necessary libraries and dependencies such as OpenCV, NumPy, and YOLO.
- Download the pre-trained YOLO weights and configuration files.

Step 2: download YOLO pre-trained weights and configuration
- YOLO has different versions (e.g., YOLOv3, YOLOv4). Download the weights and configuration files for the desired version.

Step 3: configure YOLO
- Modify the YOLO configuration file to suit your specific needs. Adjust parameters such as the number of classes, anchor boxes, and detection thresholds.

Step 4: load YOLO model
- Load the YOLO model using pre-trained weights and configuration files. This can be using libraries like OpenCV or a deep learning framework such as TensorFlow or PyTorch.

Step 5: capture video feed
- Capture video frames from a camera or video file. You can use OpenCV for this purpose.

Step 6: preprocess frames
- Preprocess the frames to match the input format expected by the YOLO model. This typically involves resizing the frames and normalizing pixel values.

Step 7: run YOLO object detection
- Apply the YOLO model to detect objects in each frame. YOLO will output bounding boxes, class labels, and confidence scores for each detected object.

Step 8: traffic signal detection
- Like lane detection, filter out traffic signals from the YOLO detections. Use additional logic to determine the state of each traffic signal (red, green, and yellow).

Step 9: display results
- Draw bounding boxes around detected lane lines and traffic signals on the video frames. Display the processed video with annotations.

Step 10: evaluate and fine-tune
- Evaluate the performance of your system and fine-tune parameters as needed. This may involve adjusting YOLO configuration, lane detection parameters, or traffic signal detection logic.

Step 11: optimization
- Implement optimizations to improve the speed and efficiency of your system. This may include hardware acceleration, model quantization, or other techniques.

Step 12: testing
- Test your system on various video feeds to ensure robust performance under different conditions.

Step 13: deployment
- If needed, deploy the system on the target platform, such as an embedded system or a server, for real-time or offline processing. These steps are shown as a workflow diagram in Figure 1.

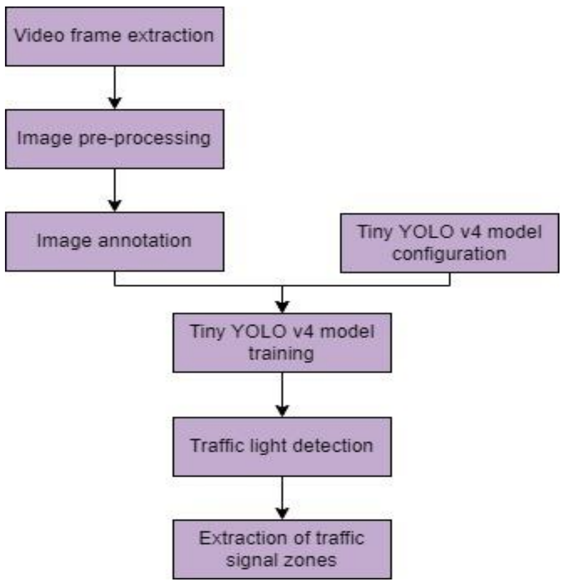
Video frame extraction
Image pre-processing
Image annotation
Tiny YOLO v4 model configuration
Tiny YOLO v4 model training
Traffic light detection
Extraction of traffic signal zones

Figure 1. Workflow diagram

4. METHOD

The YOLO algorithm, in general, is a new approach used for object detection that recommends a integrated model that is capable of speculating bounding boxes and probabilities of classes directly from an input image in real-time. This shift in prototype, moving away from region-based strategies to single-pass, end-to-end neural networks, contributed significant benefits in terms of speed and efficiency. Tiny YOLO is a lightweight implementation of the YOLO object detection technique. YOLO is a popular real-time object identification system that analyzes an image in a single forward pass through a neural network to recognize and locate objects inside the image. The "tiny" version of YOLO is intended to be more computationally efficient, making it appropriate for applications with limited resources, such as real-time embedded systems and mobile devices.

To further fill in the methodology that was used in this research paper firstly we proposed the idea as the main objective of traffic signal detection in real time application. This real time detection of traffic signal is been implemented using the algorithm called tiny YOLOv4. Then we have integrated the ROBIFLOW dataset, the dataset is an opensource which is available for public. This dataset consists various images of traffic signals. After integrating dataset in the tiny YOLOv4 model the data is configured, trained, and the model undergoes the process of evaluation. After this the model does the post-processing techniques and later the real time deployment.

4.1. Tiny YOLO working principle

Built upon the foundational operational principles of the traditional YOLO algorithm, tiny YOLO offers a streamlined and effective method for real-time object recognition. YOLO's core strength is its ability to quickly and thoroughly process an entire image in a single forward pass via a neural network. That distinctive feature enables to predict bounding boxes YOLO [26], which represent an object's spatial coordinates, and class probabilities, which represent the probability that an object will fall within a certain category.

$$confidence\ score = p(object) * \ IoU_{pred}^{truth} \tag{1}$$

Where p(object) is the probability of the object that is present in the model, and IoU $^{truth}{}_{pred}$ indicates the convergence over union with the predicted bounding box with the truth bounding box. Some of the important measures taken to improve the performance of YOLO as in (1).

Incorporates innovation and enhancement aimed at enhancing performance, flexibility, and efficiency. It provides comprehensive support for various vision AI tasks, encompassing detection, segmentation, pose estimation, tracking, and classification. This adaptability enables users to bring YOLOv8 into services across a wide range of applications and domains [27]. YOLO differs from the conventional region-based approaches for object detection, marking an innovation in the field of study. Utilizing YOLO, speed and efficiency are significantly improved as the full input image undergoes processing at once, compared to being divided into sections for independent analysis. This method is very helpful for applications like surveillance systems, autonomous vehicles, and video analysis that need for quick and precise object detection.

Although it is designed for situations with constrained computational resources, tiny YOLO is a compact version of YOLO that maintains to the same operational concept. Tiny YOLO are now able to function on platforms having limited resources, which includes real-time embedded systems and mobile devices, according to this adaption. YOLO and tiny YOLO possess an identical, single-pass, end-to-end neural network methodology that has revolutionized object recognition and found applications in a variety of fields where efficiency and real-time processing are critical.

4.1.1. Input processing

Similarly, to its predecessor YOLO, tiny YOLO utilizes a simplified neural network architecture that operates on the object recognition principle in real-time. The acceptance of images as input is the initial stage. To ensure the most effective possible performance of the model, preprocessing activities are carried out before the model enters the neural network. These phases usually include adjusting aspect ratio variances, normalizing the values of pixels to a specified range, and diminishing the image to a predefined size. The objective is to develop a consistent input format that enables effective object detection and feature extraction. After the preprocessing completes, the altered image is put into the neural network, which does a thorough analysis of the image using layers of convolutional processes and feature extraction algorithms.

4.1.2. Single forward pass

The input image can be classified into a grid via YOLO in a single forward movement through the network. Predicting bounding boxes and class probabilities for items inside its spatial domain is the responsibility of each grid cell. Because of the parallel processing made possible by this grid-based method, YOLO is extremely effective at real-time object detection. Through the assignment of specified regions to individual cells, YOLO maximizes computing performance and ensures accurate representation of the entire image in a single pass by streamlining the localization and classification processes within each grid cell. This methodology adds to the efficiency of YOLO in detecting objects effortlessly and precisely. Tiny YOLOv4 architecture is shown in Figure 2.

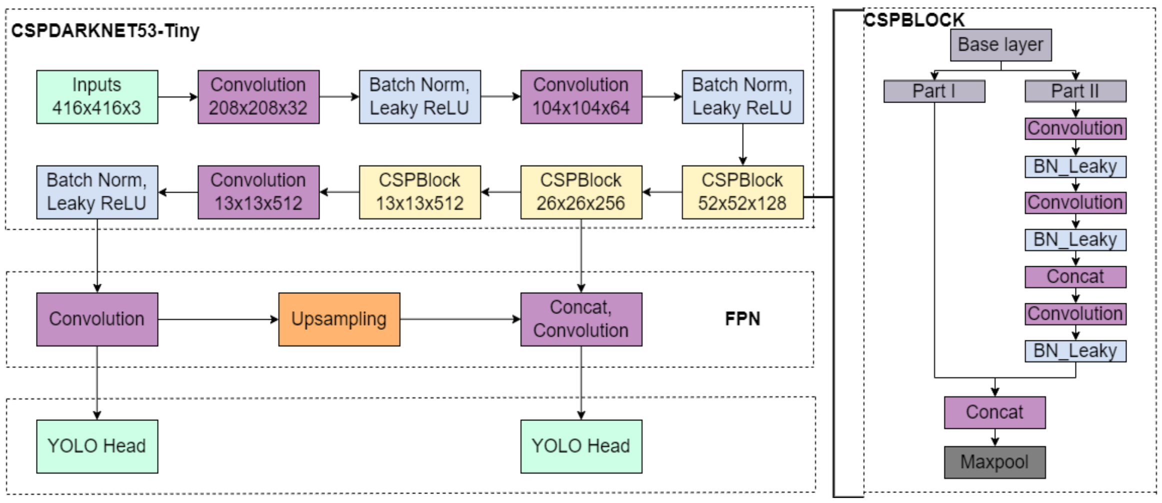

Figure 2. Tiny YOLOv4 architecture

4.1.3. Bounding box prediction

The YOLO algorithm's reduced version, tiny YOLO, predicts multiple bounding boxes for every grid cell. Essential parameters for any bounding box are its width, height, and center coordinates (x, y). The projection of these parameters is based on the grid cell's size. Tiny YOLO improves accuracy and object localization by achieving a fine-grained representation of objects in the image by distributing the number of bounding box predictions over the grid. The model can be utilized for real-time object detection in contexts with limited resources since it can manage a wide range of item sizes and configurations due to the application of several bounding boxes per cell.

4.1.4. Class prediction

Tiny YOLO enhances its ability to predict ability beyond bounding boxes by projecting class probabilities for every bounding box that is found. For each object detected by the system, a probability distribution over predefined classes, like “person,” “car,” or “dog,” is assigned. This makes it achievable for the model to accurately identify things as well as categorize them. Through the assignment of class probabilities to every bounding box, tiny YOLO enhances its capability to identify and categorize an array of objects in the image, thereby offering a thorough and in-depth comprehension of the scene in real-time object identification.

4.1.5. Post processing

A confidence threshold is vital to eliminate low-confidence detections in tiny YOLO predictions. Only predictions with an appropriate level of certainty will be taken into account due to this threshold. Following that, a non-maximum suppression method is used to get rid of bounding boxes that are duplicate or overlap. Non-maximum suppression enhances item localization and precision in classification by maintaining only the bounding box with the highest confidence in overlapping regions. By ensuring that the final output is composed of precise, non-redundant predictions, this post-processing phase improves the model’s capacity to deliver correct and compact information in real-time object identification tasks are explained by the architecture diagram shown in Figure 1.

4.1.6. Traffic signal detection

The initial stage of employing YOLO for traffic signal detection is to gather and annotate a dataset which includes images of traffic signals. Select an appropriate variant and configure YOLO by adjusting the number of traffic light classes. Verify that the model learns to recognize traffic signals accurately by learning it with the annotated dataset. Utilize real-time video streams or images to apply the trained YOLO model for inference. Integrate the model to perform automatic traffic signal detection into your system following post-processing the data to improve detections. YOLO can effectively recognize and categorize traffic signals in a variety of settings for real-time applications because to this methodical procedure.

4.2. Dataset used

The YOLOv4-tiny object detection model, a compressed version of YOLOv4 made to run on computers with less processing power, undergoes training utilizing the ROBIFLOW dataset. This model is among the fastest for object identification considering its weight totals about 16 MB. When computational resources are limited and some detection performance can be sacrificed for speed, YOLOv4-tiny can be extremely beneficial.

One may select the preprocessing and augmentation procedures to perform after uploading the dataset to Robiflow. We could opt to auto-orient and resize to 416×416, for instance. To identify particular items, the YOLOv4-tiny model can be trained on customized data. The methodology is particularly well-suited to people aiming to attain faster object detection and training.

5. RESULT AND DISCUSSION

Some of the previously used YOLO algorithms like the E-YOLOv4-tiny algorithm which has acquired the mean average precision index as 54.37%. In accordance to previous studies the algorithm called MoblieNetv3-YOLOv4 used in traffic lights detection, has performed negatively in terms of both predicted confidence and loss ratio. But tiny YOLOv4has shown beneficial confidence and loss ratio. While some study has shown the enhanced traffic signal detection and recognition. The tiny YOLOv4 architecture scored impressively, achieving an accuracy of 95.8%. This result outperforms the findings of multiple other methods and demonstrates the efficiency with which tiny YOLOv4 performs to solve traffic signal recognition challenges. The robustness of the tiny YOLOv4 model in accurately identifying and locating traffic signals within the provided dataset has been demonstrated by its high accuracy of 95.8%. This level of accuracy is especially remarkable when compared to other modern approaches, demonstrating the model’s accuracy in handling intricate real-world circumstances frequently seen in traffic signal recognition applications.

Tiny YOLOv4 outperforms competitors in terms of accuracy while maintaining a compact architecture, according to a comparison with existing approaches. This is particularly advantageous for deployment on edge devices, where real-time processing is crucial but computational resources are constrained. In addition, the tiny YOLOv4 model demonstrates its flexibility by adapting to changes in illumination, occlusions, and various traffic situations. In real-world situations, when external influences may affect the performance of less resilient detection approaches, the robustness of the model contributes to its reliability. It was found that traffic signal detection using tiny YOLOv4, revealed a correlation between certain features and outcomes. Especially, the module proposed demonstrated a notably higher prevalence of a particular characteristic in relation to traffic signal detection. This suggests a potential effectiveness or advantage of the method in detecting and recognizing traffic signals accurately. Tiny-YOLOv4 has been implemented in various application using different algorithms which have been proved to have more computational power.

A comprehensive performance analysis of the tiny YOLOv4 model in relation to traffic signal recognition is displayed in Figure 3. The graphical representation offers a comprehensive understanding of the model's capacity to strike a balance between precision and recall by encapsulating crucial parameters like the precision-recall curve. The significant rise in precision that occurs as recall increases indicates that the model is effective at reducing false positives, which is a crucial component of reliable traffic signal recognition.

We therefore shift our focus to the real-time results generated by the tiny YOLOv4 model in Figure 4 and Figure 5. The illustration below demonstrates how well the model performs in real-time image processing, highlighting its applicability for scenarios where responsiveness with minimal latency is crucial. The real-time findings highlight the model's practical application in dynamic contexts by graphically demonstrating how effective it is at quickly and reliably recognizing traffic signals.

Figure 6 illustrates the tiny YOLOv4 model's loss curve, helping to improve the analysis. Understanding the training dynamics and the model's convergence throughout the training process depend heavily on the loss curve. Effective learning is indicated by a loss curve that decreases gradually; abnormalities, on the other hand, might point to areas that need more tuning. As a result, Figure 4 and Figure 5 offers insightful information on the tiny YOLOv4 model's training stability and convergence, which helps to fully understand the model's overall performance characteristics.

Our research findings in the context of traffic signal detection using tiny YOLOv4 indicate that increased levels of a specific variable do not necessarily lead to decreased performance in a particular aspect. Moreover, our proposed method shows promise in potentially enhancing certain aspects without causing negative effects on others. This suggests a potential advantage or optimization opportunity for the proposed method in the domain of traffic signal detection, highlighting its ability to improve performance without compromising other critical factors.

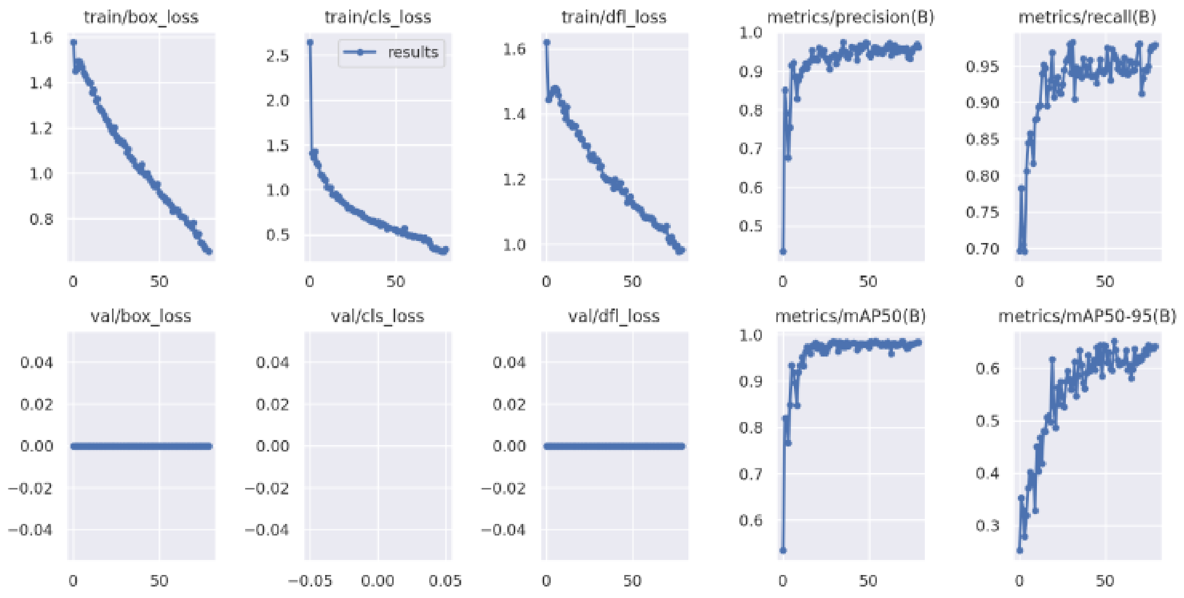

Figure 3. Performance analysis of tiny YOLOv4

While our study delved into a wide range of factors related to traffic signal detection using tiny YOLOv4, it is important to acknowledge certain limitations that could influence the results. Further extensive research is essential to validate the findings, particularly concerning specific aspects such as the robustness of the proposed method in varying conditions or scenarios. By addressing these limitations through additional studies, we can enhance the credibility and applicability of our research findings in the field of traffic signal detection.

Building upon our study’s findings that certain factors are more resilient than others in the context of traffic signal detection using YOLOv4, future research could focus on exploring ways to enhance the resilience of the proposed method. This could involve investigating feasible approaches to potentially improve the overall performance and reliability of the system. By continuing to explore these aspects, we can further advance the field of traffic signal detection and contribute to the development of more efficient and reliable systems.

In summary, our study on traffic signal detection using YOLOv4 has revealed that recent observations of a particular phenomenon are not solely attributed to increased numbers of a specific factor. Instead, our findings provide conclusive evidence that this phenomenon is linked to a change in another variable, which is not directly related to the number of factors in question. This highlights the importance of considering multiple factors and their interactions when analyzing complex systems, such as traffic signal detection systems.

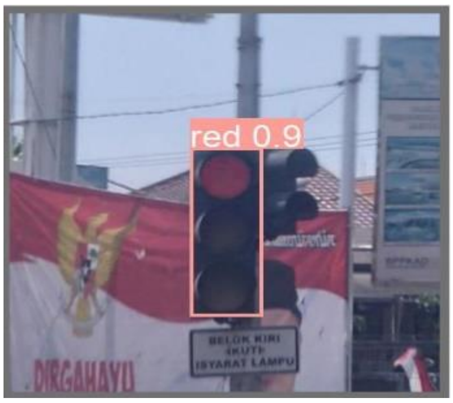

Figure 4. Real time result of the model for red signal

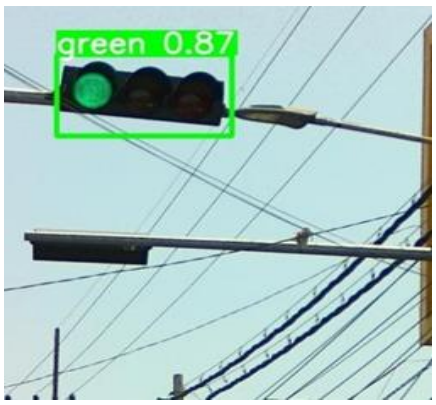

Figure 5. Real time result of the model for green signal

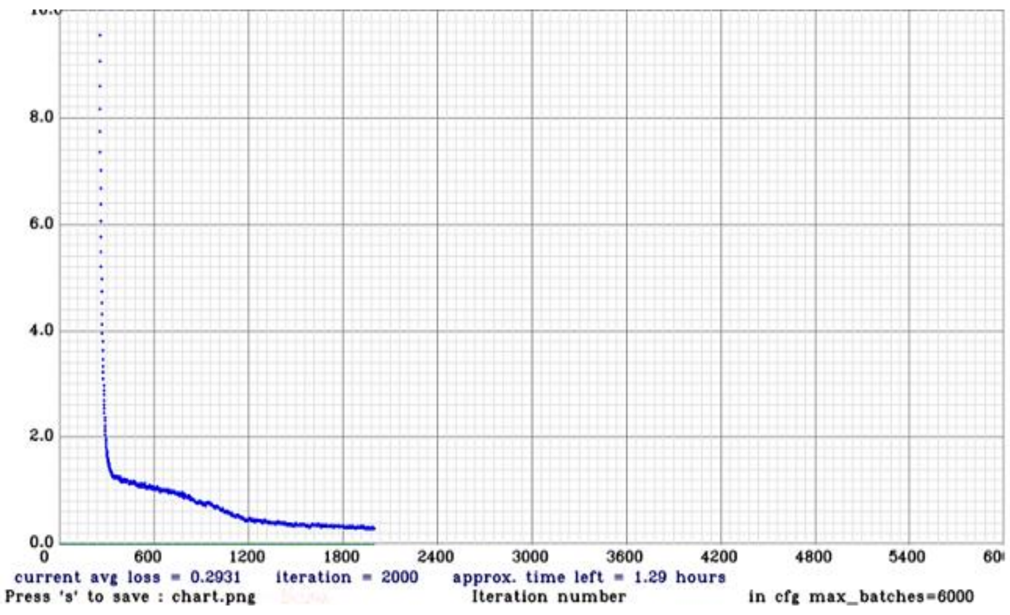

Figure 6. Loss curve of tiny YOLOv4

6. CONCLUSION

To summarize the implementation and evaluation of tiny YOLOv4 has exhibited its efficiency and effectiveness in real-time object detection of traffic signal. The adaptation of tiny YOLOv4 in the implementation of traffic signal has shown remarkable results in the domain of intelligent transportation systems. tiny YOLOv4 is proven to have its lightweight architecture and also shown impressive accuracy, speed and holds assurance for real-time traffic signal detection through various situations. Experimentation of tiny YOLOv4 has shown an effective result, it also exhibits the restrictions associated with traffic signal detection in resource-constrained environments. This model has proven to have the ability to maintain the accuracy in detection of object through crucial real-time applications like autonomous vehicle, traffic management system, and pedestrian safety. This further pays attention to state of art deep learning techniques like YOLO for real-time traffic management systems. Incorporating tiny YOLOv4 to intelligent transportation system helps the flow of traffic, enhanced road safety, and reduced traffic congestion. Current average loss in traffic signal detection using tiny YOLOv4 is derived as 0.2931. To conclude, the integration of tiny YOLOv4 application in intelligent transportation and deep learning models has shown optimistic results.

REFERENCES

[1] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/s13244-018-0639-9.

[2] M. Krichen, "Convolutional neural networks: a survey," *Computers*, vol. 12, no. 8, p. 151, 2023, doi: 10.3390/computers12080151.

[3] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022, doi: 10.1016/j.procs.2022.01.135.

[4] J. Shen, H. Liao, and L. Zheng, "A lightweight method for small scale traffic sign detection based on YOLOv4-Tiny," *Multimedia Tools and Applications*, 2023, doi: 10.1007/s11042-023-17146-3.

[5] S. S. Sumit, D. R. Awang Rambli, S. Mirjalili, M. M. Ejaz, and M. S. U. Miah, "ReSTiNet: on improving the performance of Tiny-YOLO-based CNN architecture for applications in human detection," *Applied Sciences*, vol. 12, no. 18, p. 9331, 2022, doi: 10.3390/app12189331.

[6] H. Pan, Y. Shi, X. Lei, Z. Wang, and F. Xin, "Fast identification model for coal and gangue based on the improved tiny YOLO v3," *Journal of Real-Time Image Processing*, vol. 19, no. 3, pp. 687–701, 2022, doi: 10.1007/s11554-022-01215-1.

[7] K. V. Sathyamurthy, A. R. Shri Rajmohan, A. Ram Tejaswar, K. V, and G. Manimala, "Realtime face mask detection using TINY-YOLO V4," *2021 4th International Conference on Computing and Communications Technologies (ICCCT)*. IEEE, 2021. doi: 10.1109/iccct53315.2021.9711838.

[8] P. Adarsh, P. Rathi, and M. Kumar, "YOLO v3-Tiny: object detection and recognition using one stage improved model," *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 2020. doi: 10.1109/icaccs48705.2020.9074315.

[9] M. Sohaib and M. Adewunmi, "Artificial intelligence based prediction on lung cancer risk factors using deep learning," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 12, no. 2, p. 188, 2023, doi: 10.11591/ijict.v12i2.pp188-194.

[10] H. Li, L. Deng, C. Yang, J. Liu, and Z. Gu, "Enhanced YOLO v3 tiny network for real-time ship detection from visual image," *IEEE Access*, vol. 9, pp. 16692–16706, 2021, doi: 10.1109/access.2021.3053956.

[11] G. Oltean, C. Florea, R. Orghidan, and V. Oltean, "Towards real time vehicle counting using YOLO-tiny and fast motion estimation," *2019 IEEE 25th International Symposium for Design and Technology in Electronic Packaging (SIITME)*. IEEE, 2019. doi: 10.1109/siitme47687.2019.8990708.

[12] X. Yue, Q. Wang, L. He, Y. Li, and D. Tang, "Research on tiny target detection technology of fabric defects based on improved YOLO," *Applied Sciences*, vol. 12, no. 13, p. 6823, 2022, doi: 10.3390/app12136823.

[13] G. Sha, J. Wu, and B. Yu, "Detection of Spinal fracture lesions based on improved YOLO-tiny," *2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications(AEECA)*. IEEE, 2020. doi: 10.1109/aeeca49918.2020.9213684.

[14] K. C. Saranya, A. Thangavelu, A. Chidambaram, S. Arumugam, and S. Govindraj, "Cyclist detection using tiny YOLOv2," *Advances in Intelligent Systems and Computing*. Springer Singapore, pp. 969–979, 2019. doi: 10.1007/978-981-15-0184-5_82.

[15] M. C. Hasani, F. Milenasari, and N. Setyawan, "Monitoring physical distance in public areas using YOLO tiny V3 (in Indonesian)," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 6, no. 1, pp. 146–152, 2022, doi: 10.29207/resti.v6i1.3808.

[16] A. Womg, M. J. Shafiee, F. Li, and B. Chwyl, "Tiny SSD: a tiny single-shot detection deep convolutional neural network for real-time embedded object detection," *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE, 2018. doi: 10.1109/crv.2018.00023.

[17] D. P. Lestari, R. Kosasih, T. Handhika, Murni, I. Sari, and A. Fahrurozi, "Fire hotspots detection system on cctv videos using you only look once (YOLO) method and tiny YOLO model for high buildings evacuation," *2019 2nd International Conference of Computer and Informatics Engineering (IC2IE)*. IEEE, 2019. doi: 10.1109/ic2ie47452.2019.8940842.

[18] F. U. D. Farrukh *et al.*, "Power efficient tiny YOLO CNN using reduced hardware resources based on booth multiplier and WALLACE tree adders," *IEEE Open Journal of Circuits and Systems*, vol. 1, pp. 76–87, 2020, doi: 10.1109/ojcas.2020.3007334.

[19] P. Deng, K. Wang, and X. Han, "Real-time object detection based on YOLO-v2 for tiny vehicle object," *SN Computer Science*, vol. 3, no. 4, 2022, doi: 10.1007/s42979-022-01229-3.

[20] S. Valladares, M. Toscano, R. Tufiño, P. Morillo, and D. V. Huanga, "Performance evaluation of the nvidia jetson nano through a real-time machine learning application," *Advances in Intelligent Systems and Computing*. Springer International Publishing, pp. 343–349, 2021. doi: 10.1007/978-3-030-68017-6_51.

[21] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on convolutional neural networks (CNN) in vegetation remote sensing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 173, pp. 24–49, 2021, doi: 10.1016/j.isprsjprs.2020.12.010.

[22] Y. Li and C. Lv, "SS-YOLO: an object detection algorithm based on YOLOv3 and ShuffleNet," *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. IEEE, 2020. doi: 10.1109/itnec48623.2020.9085091.

[23] M. Dhouibi, A. K. B. Salem, A. Saidi, and S. B. Saoud, “Acceleration of convolutional neural network based diabetic retinopathy diagnosis system on field programmable gate array,” *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 12, no. 3, p. 214, 2023, doi: 10.11591/ijict.v12i3.pp214-224.
[24] N. Nafi’iyah and A. Yuniarti, “A convolutional neural network for skin cancer classification,” *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 11, no. 1, p. 76, 2022, doi: 10.11591/ijict.v11i1.pp76-84.
[25] A. Ojeniyi, A. Khalid, and O. C. Kehinde, “A persuasive agent architecture for behavior change intervention,” *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 11, no. 2, p. 128, 2022, doi: 10.11591/ijict.v11i2.pp128-139.
[26] P. Wang, J. Qiao, and N. Liu, “An Improved convolutional neural network-based scene image recognition method,” *Computational intelligence and neuroscience*, vol. 2022, p. 3464984, Jun. 2022, doi: 10.1155/2022/3464984.
[27] G. Oreski, “YOLO*C — adding context improves YOLO performance,” *Neurocomputing*, vol. 555, p. 126655, 2023, doi: 10.1016/j.neucom.2023.126655.

BIOGRAPHIES OF AUTHORS

Santhiya received the B.Tech. degree in Information Technology from Anna University, Tamil Nadu, India, in 2010 and the M.E. degree in computer science and engineering from Anna University, Tamil Nadu, India, in 2014. Currently, she is pursuing Ph.D. in Compueter Science and Engineering in karunya institute of technology and science. Her research interests include IoT, networking, and intelligent transportation system. She can be contacted at email: pvsanthiya89@gmail.com.

Immanuel Johnraja Jebadurai received his B.E. degree in Computer Science and Engineering from M.S. University, Tirunelveli, India. in the year 2003. He received his M.E. degree in Computer Science and Engineering from Anna University, Chennai India in the year 2005. He received his Ph.D. in Computer Science and Engineering from Karunya Institute of Technology and Sciences, Coimbatore, India in the year 2017. His areas of interest include network security, vehicular Ad-Hoc networking, and IoT. He can be contacted at email: immanueljohnraja@gmail.com.

Getzi Jeba Leelipushpam Paulraj received her B.E. degree in Electronics and Communication Engineering from M.S. University, Tirunelveli, India in the year 2004. She received her M.Tech. degree in Network and Internet Engineering from Karunya University, Coimbatore, India in the year 2009. She received her Ph.D. from Karunya Institute of Technology and Sciences, Coimbatore, India in the year 2018. Her areas of interest include IoT, fog computing, and data analytics. She can be contacted at email: getzi@karunya.edu.

Ebenezer Veemaraj received his B.Tech. degree in Information Technology and M.E degree in Computer Science and Engineering from Anna University, Chennai in the years 2009 and 2012. He also received his Ph.D. in Information and Communication Engineering from Anna University, Chennai in the year 2020. He is currently working as an Assistant professor in the Computer Science and Engineering department, Karunya Institute of Technology and Sciences, Coimbatore Tamil Nadu, India. He has published many research papers in various International/National Conferences and Journals. His area of interest includes the IoT, cloud computing, body area networks, data structures, and distributed systems. He can be contacted at email: ebenezerv@karunya.edu.

Randlin Paul Sharance pursuing the B.Tech. degree in Computer Science and Engineering from Karunya Institute of Technology and Science, Tamil Nadu, India. His interest includes Intelligent transportation and IoT. He can be contacted at email: randlinsharance@gmail.com.

Rubee Keren pursuing the B.Tech. degree in Computer Science and Engineering from Karunya Institute of Technology and Science, Tamil Nadu, India. Her interest includes machine learning and explainable AI. She can be contacted at email: rubeekeren.07@gmail.com.

Kiruba Karan pursuing the B.Tech. degree in Computer Science and Engineering from Karunya Institute of Technology and Science, Tamil Nadu, India. His interest includes deep learning, machine learning, and artificial intelligence. He can be contacted at email: kirubakarandev19@gmail.com.