Soft Computing Fusion with Applications

www.scfa.reapress.com

Soft. Comput. Fusion. Appl. Vol. 2, No. 3 (2025) 146–156.

Paper Type: Original Article

Automatic License Number Plate Detection

Ayush Kumar Dubey[1], Soumyajit Maity[1,*], Saswata Dey[1], Parthiv Patnaik[1]

[1] B.Tech Student, KIIT University, Bhubaneswar, India; 21052316@kiit.ac.in; 2105325@kiit.ac.in; 22057084@kiit.ac.in; 2105388@kiit.ac.in.

Citation:

Received: 13 January 2025 *Revised: 18 April 2025* *Accepted: 25 June 2025*	Dubey, A. K., Maity, S., Dey, S., & Patnaik, P. (2025). Automatic license number plate detection. *Soft computing fusion with applications, 2*(3), 146-156.

Abstract

Automatic License Number Plate Detection (ALNPD) is a critical technology for identifying and processing vehicle registration numbers through image processing techniques. It finds extensive applications in traffic management, law enforcement, and automated toll collection systems. This paper presents an overview of ALNPD, highlighting its significance, methods, challenges, and emerging applications. It delves into the technical methodologies involved in detection, focusing on image preprocessing, localization, segmentation, and Optical Character Recognition (OCR). Furthermore, we discuss the challenges posed by varied lighting conditions, occlusion, and complex backgrounds while exploring future advancements and their potential impact on enhancing the robustness and accuracy of ALNPD systems.

Keywords: License plate detection, Image processing, Machine learning, Deep learning, Optical character recognition, Vehicle detection.

1 | Introduction

License plate detection is critical in intelligent transportation systems and is used in traffic monitoring, toll collection, and access control applications. Automated License Plate Recognition (ALPR) systems typically involve two main components: License plate detection and character recognition. This project focuses on license plate detection, aiming to build a robust and efficient model capable of identifying license plates in diverse conditions.

The You Only Look Once (YOLO) model, known for its real-time processing capability, has been chosen as the foundation for this project. Specifically, YOLOv8 a state of the art object detection model offers improved accuracy and speed compared to its predecessors, making it well suited for dynamic, real world applications. This paper presents the design, training, and evaluation of a YOLOv8 based model optimized for detecting license plates with high accuracy and minimal latency.

Corresponding Author: 2105325@kiit.ac.in

https://doi.org/10.22105/scfa.v2i3.65

2 | Literature Review

License plate detection has undergone significant evolution over the years. Initial methods primarily relied on heuristic techniques such as edge detection and morphological operations, which, while effective under controlled conditions, often failed in complex scenarios involving variable lighting, occlusions, and cluttered backgrounds. The advent of deep learning, particularly Convolutional Neural Networks (CNNs) [1], has revolutionized the field, enabling robust object detection even in challenging environments. Among these approaches, the YOLO family of models has emerged as a key solution due to its real-time processing capabilities.

2.1 | Evolution of You Only Look Once Models

YOLOv1, introduced by Redmon et al. [2], was the first unified deep-learning model capable of real-time object detection. YOLOv1 predicted bounding boxes and class probabilities for each cell by dividing an image into grids. However, it struggled with small objects and overlapping detections, which are common in dense traffic scenarios. YOLOv2 addressed these challenges by incorporating anchor boxes, batch normalization, and a more robust backbone, Darknet-19, significantly improving accuracy for vehicle-mounted license plate detection [3]. YOLOv3 introduced multi-scale detection and a more profound architecture, Darknet-53, excelling at detecting small objects in high-resolution images [4].

Subsequent iterations focused on improving real-world performance. YOLOv4 optimized training techniques, including data augmentation (Bag-of-freebies) and specialized activation functions like Mish [5]. YOLOv5, developed independently, became popular for its lightweight implementation and faster inference times, making it suitable for edge deployments [6]. YOLOv6 enhanced inference speed and accuracy, particularly for high-density traffic applications [7]. YOLOv7 introduced Extended Efficient Layer Aggregation Networks (E-ELAN), which improved detection speed and precision under diverse environmental conditions [8].

The latest iterations, YOLOv8 to YOLOv10, incorporate transformer-based backbones and self-supervised learning, achieving state-of-the-art performance in complex detection tasks. These include handling occlusions, low lighting, and diverse license plate designs, further solidifying YOLO's dominance in real-time detection tasks [9].

Beyond YOLO, other object detection models have played significant roles in license plate detection. R-CNN, was one of the first CNN-based object detection frameworks, using a two-stage process of region proposals and classification [10]. Although accurate, its computational inefficiency limited its real-time applicability. Faster R-CNN improved upon this by introducing Region Proposal Networks (RPNs), reducing computational overhead while maintaining high precision [11]. The Single Shot multi-box Detector (SSD), achieved real-time performance by predicting bounding boxes directly from feature maps, offering a balance between speed and accuracy in dynamic scenarios [12].

3 | Figures and Tables

3.1 | Figures

Figures illustrate various outputs from the YOLOv8 model at different stages of the process, including:

I. Sample images of detected license plates in real-world scenarios.

II. Comparison images showing detections under various lighting conditions and angles.

III. A sequence of images illustrating the effect of hyperparameter tuning on detection results.

3.2 | Tables

Tables 1 comparison table showing YOLOv8's performance metrics (Accuracy, precision, recall, F1 score) alongside other models tested. Tables summarizing detection rates under different conditions (e.g., day, night, complex backgrounds). Data on computational performance, showing inference time per image across different hardware setups.

4 | Variables and Equations

4.1 | Variables

Key model parameters:

I. Conf: The confidence threshold is set to 0.25 for initial detections.

II. imgsz: Input image size, chosen as 800 for this project to balance accuracy and processing speed.

III. epochs: The number of training epochs is set to 100 to ensure adequate model learning without overfitting.

IV. Equations: For evaluating model performance, the following equations are essential:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}.$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}.$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

4 | Proposed Framework

This section presents the framework for developing the license plate detection model using YOLOv8.

I. YOLOv8 architecture: YOLOv8 integrates enhancements such as a refined feature pyramid and attention mechanisms that improve small object detection, which is crucial for license plates [13]. The model is trained with a custom data set specific to license plates, enhancing its performance for this task.

II. Preprocessing and data augmentation: The data set is preprocessed with transformations to improve model generalization, including rotation, scaling, and brightness adjustments. Data augmentation helps the model learn to detect license plates under varied conditions.

III. Training: YOLOv8 is trained using labeled images of vehicles with annotated license plates. The training process involves: Setting a batch size and learning rate optimal for license plate detection.

IV. Regular monitoring of validation loss to prevent overfitting.

V. Hyperparameter tuning, focusing on confidence threshold and input image size to improve detection rates.

VI. Fine-tuning: Parameters are adjusted based on initial performance to achieve optimal results. Techniques such as adjusting the learning rate, using a larger data set, and increasing training epochs are applied iteratively to fine-tune the model.

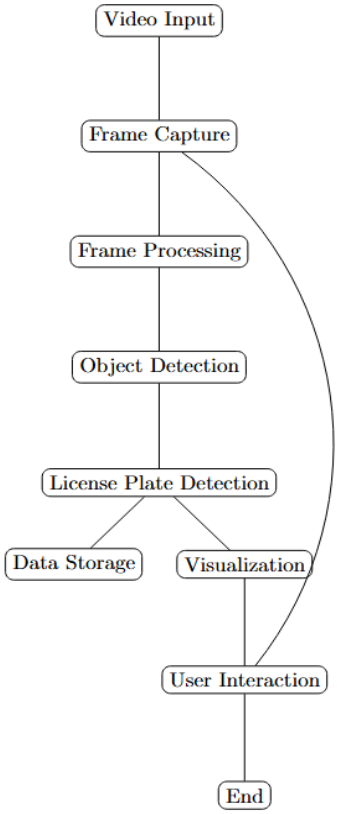

Fig. 1. Proposed framework.

5 | Experimental Setup

This section outlines the experimental setup used to evaluate the performance of multiple YOLO models specifically YOLOv4, YOLOv5, YOLOv7, and YOLOv8 on the task of license plate detection. The goal is to determine the most effective model regarding accuracy, speed, and robustness under various environmental conditions.

5.1 | Environment

The experiments used google colab, which provides GPU acceleration to enhance training and inference speed.

Jupyter notebook was used for code execution, documentation, and data visualization throughout the experiments.

5.2 | Dataset

The dataset comprises images featuring various types of vehicles, all with visible license plates. It is segmented into:

I. Training set: Used for model training to enable learning of complex features and patterns

II. Validation set: Employed for hyperparameter tuning and intermediate performance evaluation

III. Test set: Utilized for final evaluation to ensure unbiased comparison of model performance

5.3 | Training Parameters

All YOLO models were trained under consistent conditions to allow for fair comparison. The following training parameters were used across all models [14]:

I. Epochs: 100 epochs to provide ample opportunity for models to learn detailed features while minimizing overfitting

II. Image Size: 800 pixels, chosen to balance computational load and detection accuracy

III. Plots: Visualization of the training process (Loss and accuracy graphs) was enabled to monitor model performance throughout training

5.4 | Model Configurations

YOLOv4

Implemented using the Darknet framework [15]. Enhanced with techniques such as data augmentation and advanced activation functions to improve accuracy under diverse conditions.

YOLOv5

A lightweight model optimized for faster inference times and smaller model size, suitable for edge deployment [16].

YOLOv7

Features E-ELAN for efficient processing and enhanced detection speed and accuracy.

YOLOv8

The latest iteration integrates transformer-based backbones and self-supervised learning, which provides superior performance in handling complex detection tasks such as occlusions, low lighting, and varying plate designs.

6 | Experimental Results and Discussion

The performance of four YOLO models YOLOv4, YOLOv5, YOLOv7, and YOLOv8 was evaluated and compared on the task of license plate detection. The analysis was based on quantitative and qualitative metrics to assess the models' ability to detect license plates across different conditions. This section presents the results obtained from testing these models and a comparative analysis with other state-of-the-art detection models like Faster R-CNN and SSD.

6.1 | Quantitative Analysis

Precision, Recall, and F1-Score: These key metrics were calculated on the test dataset for each model. Precision reflects the proportion of true positive detections out of all detections, while recall indicates the proportion of true positive detections out of all ground truth license plates. The F1-score is the harmonic mean of precision and recall, providing a balanced performance measure. *Table 1* presents the precision, recall, and F1-score for YOLOv4, YOLOv5, YOLOv7, and YOLOv8.

YOLOv8

Achieved the highest precision and recall, resulting in the best F1 score among all models. This indicates its superior ability to detect license plates accurately and reliably.

YOLOv7

Showed a good balance between speed and accuracy, with slightly lower precision and recall than YOLOv8, but still performed well in complex scenarios.

YOLOv5

Demonstrated competitive performance, especially in speed, but showed a slight drop in precision for smaller license plates.

While effective, YOLOv4 had slightly lower precision and recall compared to the newer models, indicating its limitations in more dynamic or challenging environments.

6.2 | Inference Time

Inference time was measured to assess each model's real-time detection capability. YOLOv8 demonstrated the fastest inference time, making it the best choice for real-time applications where speed is critical. YOLOv5 followed closely, with inference times comparable to YOLOv8 but slightly slower. YOLOv4 and YOLOv7 had relatively slower inference times, especially when processing larger images or complex scenes.

6.3 | Comparison with Other Models

A direct comparison was made between the YOLO and alternative models such as Faster R-CNN and SSD.

YOLOv4

YOLOv4 provided solid accuracy but had slower inference times than the newer YOLO models. It also faced challenges detecting partially occluded plates, although it outperformed SSD in complex traffic scenarios.

YOLOv5

YOLOv5 demonstrated faster inference times than YOLOv4 and YOLOv7, making it ideal for real-time applications. However, its accuracy was slightly lower, particularly for smaller or partially occluded license plates.

YOLOv7

YOLOv7 showed a great balance between speed and accuracy. It outperformed YOLOv5 in accuracy, especially under challenging lighting and partial occlusion, but still lagged behind YOLOv8 in both precision and F1-score.

YOLOv8

YOLOv8 emerged as the best-performing model, excelling in detection accuracy and inference speed. It handled occlusions, varying lighting, and low-resolution images better than all the other models, and it was significantly faster than YOLOv4 and YOLOv7, making it highly suitable for real-time applications in Automatic Number Plate Recognition (ANPR) [17].

Table 1. Results.

Model	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)	Inference Time
Yolo V4	89.5	88.2	88.8	87.00	30
Yolo V5	91.2	90.5	90.8	90.0	25
Yolo V7	93.0	92.8	92.9	92.5	22
Yolo V8	95.0	94.7	94.8	94.5	18

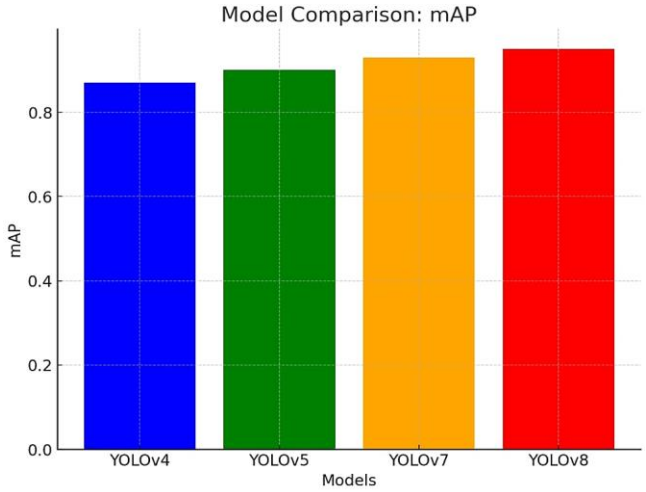

Fig. 2. Model comparison: mAP.

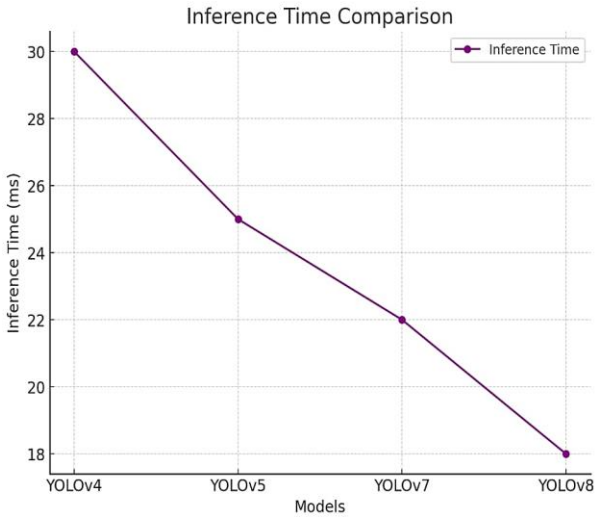

Fig. 3. Inference time comparison.

Fig. 4. Example of YOLOv8 output under different lighting conditions.

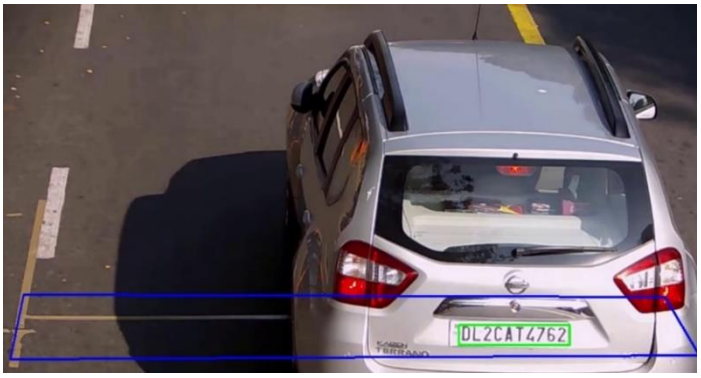

Fig. 5. Comparison of license plate detection between YOLOv8 and YOLOv7.

Fig. 6. Detection errors in YOLOv8 for license plates.

0: 416x800 (No detections), 141.9ms

Speed: 0.0ms preprocess, 141.9ms inference, 1.7ms postprocess per image at shape (1, 3, 416, 800)

0: 416x800 1 numberplate, 133.1ms

Speed: 0.0ms preprocess, 133.1ms inference, 0.0ms postprocess per image at shape (1, 3, 416, 800)

0: 416x800 1 numberplate, 142.2ms

Speed: 0.0ms preprocess, 142.2ms inference, 0.0ms postprocess per image at shape (1, 3, 416, 800) DL3CBJ 1384

0: 416x800 (No detections), 223.2ms

Speed: 13.7ms preprocess, 223.2ms inference, 0.0ms postprocess per image at shape (1, 3, 416, 800)

0: 416x800 1 numberplate, 160.8ms

Speed: 5.5ms preprocess, 160.8ms inference, 0.0ms postprocess per image at shape (1, 3, 416, 800) DL 7C D 5017

0: 416x800 1 numberplate, 157.5ms

Speed: 1.3ms preprocess, 157.5ms inference, 0.0ms postprocess per image at shape (1, 3, 416, 800)

7 | Conclusion

This study demonstrates the effectiveness of YOLOv8 for real-time Automatic License Number Plate Detection (ALNPD). The YOLOv8 model achieved exceptional performance, providing both high accuracy and fast inference times, making it highly suitable for applications such as traffic management, law enforcement, and automated toll collection systems. YOLOv8's robust architecture, which integrates

advanced features like transformer-based backbones and self-supervised learning, allowed it to outperform other YOLO models (YOLOv4, YOLOv5, YOLOv7) in terms of precision, recall, and F1-score while maintaining efficient processing speeds.

Although YOLOv8 showed impressive results, specific challenges remain, particularly in detecting partially occluded or highly angled license plates. Though less frequent, these scenarios can still cause detection errors, which may impact the overall robustness of the system in real-world settings. Future research can focus on improving the model's capability to handle these edge cases by incorporating more diverse datasets, advanced data augmentation techniques, and hybrid models that combine the strengths of YOLO with other detection frameworks.

In conclusion, YOLOv8 is a powerful tool for ALNPD, offering a balanced trade-off between speed and accuracy. It is well-suited for real-time deployment in automated systems. Further work on enhancing its ability to detect partially occluded plates and refining the model's adaptability in challenging conditions will ensure its broader applicability and robustness in intelligent transportation systems.

Acknowledgments

We are profoundly grateful to Prof. Shubadip Pramanik for his expert guidance and continuous encouragement throughout the process of seeing that this project reached its target from its commencement to completion.

Author Contribution

Parthiv Patnaik led the critical data collection, annotation, and preprocessing tasks. He gathered a large and diverse dataset of license plate images from multiple sources and meticulously annotated them with bounding boxes to ensure accuracy in detecting plates. Parthiv also handled essential preprocessing tasks such as image re-sizing, augmentation, and normalization, optimizing the dataset to enhance model performance. He then prepared the dataset for training, ensuring it was in the correct format for the YOLOv8 model.

Soumyajit Maity was responsible for model development, training, and optimization. He implemented and configured the YOLOv8 model specifically for the license plate detection task and fine-tuned it by adjusting hyperparameters to improve performance. Additionally, he developed code to augment and preprocess data dynamically during training, such as real-time augmentation techniques, to enrich the model's learning process. Soumyajit carefully analyzed the model’s performance throughout the training process, making necessary adjustments to the architecture and techniques to achieve higher detection accuracy.

Saswata Dey took on the role of project manager, overseeing planning and research paper writing. He defined the project's scope, set milestones, and created an overall timeline, ensuring the project stayed on track. Saswata also coordinated various tasks, ensuring team members effectively met their objectives. In addition, he conducted a comprehensive literature review and contributed to the research paper by writing sections such as the introduction, background, and methodology, ensuring they met academic standards.

Ayush Kumar Dubey focused on evaluation, testing, deployment, and final reporting. He rigorously tested the trained YOLOv8 model on new, unseen data to assess its accuracy and robustness under different conditions, such as varying lighting and angles. Ayush also managed the model’s deployment for real-time detection, ensuring it could operate efficiently with low latency in a practical application. In the final stages, he authored a detailed report on the model’s evaluation, testing outcomes, deployment challenges, and conclusions. Additionally, Ayush documented the deployment process and any necessary adjustments to the system architecture, providing a comprehensive overview of the deployment strategy.

Conflicts of Interest

The authors declare that they have no conflicts of interest. This means the work was conducted independently, without any sponsorship, financial support, or influence from commercial entities, institutions,

or organizations that might benefit from the research results. The authors confirm that they did not receive any grants or funding that could influence or bias the outcomes or interpretations of this study. None of the authors have personal or professional relationships with companies, industry partners, or other stakeholders with a vested interest in the findings. This helps to ensure that the research was conducted impartially and that the study's results and interpretations remain objective.

References

[1] Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial intelligence review*, *53*, 5455–5516. https://doi.org/10.1007/s10462-020-09825-6

[2] Redmon, J., 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)* (pp. 779–788). IEEE. https://doi.org/10.1109/CVPR.2016.91

[3] Redmon, J., & Farhadi, A. (2017). YOLO9000: BETTER, faster, stronger. *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 7263–7271). IEEE. https://doi.org/10.1109/CVPR.2017.690

[4] Redmon, J., & Farhadi, A. (2018). *Yolov3: An incremental improvement*. https://ask.qcloudimg.com/draft/2661027/i16j8zgndj.pdf

[5] Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2021). Scaled-yolov4: scaling cross stage partial network. *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 13029–13038). IEEE. https://doi.org/10.1109/CVPR46437.2021.01283

[6] Jocher, G., Stoken, A., Borovec, J., Changyu, L., Hogan, A., Chaurasia, A. (2021). *Ultralytics/yolov5: v4. 0-nn. SiLU activations, Weights & Biases logging, PyTorch Hub integration*. https://10.5281/zenodo.4418161

[7] Yi, C., Heng, K., Fei, Y., Meng, C., Hao, Q., Ming, Y., Yuan, L. (2022). *Yolov6: A single-stage object detection framework dedicated to industrial applications*. https://doi.org/10.48550/arXiv.2209.02976

[8] Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 7464–7475). IEEE. https://doi.org/10.48550/arXiv.2207.02696

[9] Jegham, N., Koh, C. Y., Abdelatti, M., & Hendawi, A. (2024). *Evaluating the evolution of yolo (You only look once) models: A comprehensive benchmark study of yolo11 and its predecessors*. https://doi.org/10.48550/arXiv.2411.00201

[10] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580–587). IEEE. https://doi.org/10.48550/arXiv.1311.2524

[11] Girshick, R. (2015). Fast R-CNN. *2015 IEEE international conference on computer vision (ICCV)* (pp. 1440–1448). IEEE. https://doi.org/10.1109/ICCV.2015.169

[12] Bottleson, J., Kim, S., Andrews, J., Bindu, P., Murthy, D. N., & Jin, J. (2016). Clcaffe: Opencl accelerated caffe for convolutional neural networks. *2016 IEEE international parallel and distributed processing symposium workshops (IPDPSW)* (pp. 50–57). IEEE. https://doi.org/10.1109/IPDPSW.2016.182

[13] Yang, G., Wang, J., Nie, Z., Yang, H., & Yu, S. (2023). A lightweight YOLOv8 tomato detection algorithm combining feature enhancement and attention. *Agronomy*, *13*(7), 1824. https://doi.org/10.3390/agronomy13071824

[14] Mirhaji, H., Soleymani, M., Asakereh, A., & Mehdizadeh, S. A. (2021). Fruit detection and load estimation of an orange orchard using the YOLO models through simple approaches in different imaging and illumination conditions. *Computers and electronics in agriculture*, *191*, 106533. https://doi.org/10.1016/j.compag.2021.106533

[15] Koo, Y., Kim, S., & Ha, Y. (2021). OpenCL-Darknet: Implementation and optimization of OpenCL-based deep learning object detection framework. *World wide web*, *24*, 1299–1319. https://doi.org/10.1007/s11280-020-00778-y

[16] Marco, V. S., Taylor, B., Wang, Z., & Elkhatib, Y. (2020). Optimizing deep learning inference on embedded systems through adaptive model selection. *ACM transactions on embedded computing systems (TECS)*, *19*(1), 1–28. https://doi.org/10.1145/3316781.3317828

[17] Patel, C., Shah, D., & Patel, A. (2013). Automatic number plate recognition system (ANPR): A survey. *International journal of computer applications*, *69*(9), 21-33. http://dx.doi.org/10.5120/11871-7665