Comparing the performance of yolov10s and ssd300 models in the problem of automatic fruit identification and classification

Bui Xuan Tung[1,*], Trinh Quang Minh[1], Ngo Thi Lan[1], Dang Thi Dung[2] and Quach Dai Vinh[2]

[1] Tay Do University
[2] Can Tho University of Technology
*Corresponding author: bxtung@tdu.edu.vn

■ Received: 14/01/2025 ■ Revised: 12/02/2025 ■ Accepted: 02/03/2025

ABSTRACT

The research focuses on applying deep learning to automate the fruit recognition and classification process, meeting the development needs of modern agriculture. Applying this technology helps improve efficiency and classification quality and reduces labor costs, resulting in lower product prices. The research team used two deep learning models, SSD300 and YOLOv10s, to recognize and classify six types of fruits: apples, bananas, kiwis, lemons, oranges, and strawberries. The dataset consists of 2575 images divided into Train, Validation, and Test sets with a ratio of 87%-8%-4%. The images were resized to 300x300 pixels for SSD300 and 640x640 pixels for YOLOv10s. The experimental results show that the YOLOv10s model achieved higher precision at 96% compared to 93% for SSD300. The research also proposes future improvements to enhance the system's accuracy and applicability.

Keywords: *YOLOv8, YOLO-NAS, Vehicle License Plate Detection, Machine Learning, Deep Learning.*

1. INTRODUCTION

Currently, with the strong development of artificial intelligence, the Internet of Things, cloud computing, and big data - the main pillars of the Industrial Revolution 4.0, many new application models have been created in production. These advances have promoted many activities and strongly impacted the digital economy, politics, and social life. Artificial intelligence is quickly becoming one of the most anticipated fields of science, thanks to its ability to benefit many industries and fields such as industry, agriculture, medicine, and education. In the field of precision agriculture, artificial intelligence applications have been widely used and brought about many great results, such as drones, self-driving tractors, harvesting support robots, and soil moisture measurement systems for agricultural irrigation. The application of artificial intelligence in different areas of life and society has brought great benefits to the national economy. In this context, the application of advanced technologies in agriculture not only helps improve production efficiency but also contributes to modernizing the agricultural sector, towards smart and sustainable agriculture.

1.1. Research objectives

This study focuses on: Applying deep learning models to fruit recognition, classification, and counting. Understanding knowledge about data collection, data preprocessing, building deep learning models, and evaluating deep learning models. Mastering the knowledge base of libraries used for data processing, and model training such as libraries: Padans, Numpy, Tensorflow, OpenCV,... Comparison shows the effectiveness of different network architectures between CSPNet in YOLOv10s [1][2], and MobileNetV2 [3][4] in SSD300 [5]. From there, it shows the performance

and accuracy between the two models in the problem of fruit classification, recognition, and counting.

1.2. Overview of research situation

The typical domestic research situation is the research paper of the group of authors Truong Quoc Bao and colleagues [6], with the research topic of detecting and identifying defects in mango peel, the research group has proposed an approach solution by proposing a new algorithm to detect and identify defects on mango peel using image segmentation techniques and browsing connected regions to separate the mango image from the background and detect candidate regions containing defects. Next, the candidate regions will be analyzed to extract image features and classify them to identify defects using neural networks. With this approach, the research group has achieved positive results such as an accuracy of 92.79% and a recognition time is less than 7s for a mango image. The group of authors Trinh Trung Hai and colleagues [7] has the research topic of using the improved faster r-cnn model for the solution to identify and detect ripe pineapples. The team came up with a solution by using a CNN network built for image classification using deep learning techniques. Building a CNN model is easier, along with using the Keras library with Python language. The model uses CNN layers such as Conv2D & MaxPooling2D. Conv2D This layer generates dozens of outputs by creating a convolutional kernel with the input layer. MaxPooling 2D is used for max pooling operations for spatial data. Spatial data can be defined as representing information about a physical object using numeric values. Selecting the maximum element from the region of the electronic map covered by the filter is the operand performed by the max pooling layer. To reduce the size of the feature map, pooling layers are used. The team has achieved results such as maximum accuracy ranging from 90% to 95%.

A typical foreign research situation is the research paper of Harmandeep S.Gill1, Osamah I.Khalaf, and colleagues [8] presented the research topic Fruit Image Classification Using Deep Learning, the research group approached the topic by using CNN to extract image features, then using RNN to select the optimal extracted features and LSTM was used to classify fruits based on the image features extracted and selected by CNN and RNN. The group achieved positive results such as an accuracy of up to 97%. The research topic Efficient Fruits Classification Using Convolutional Neural Network by Adnan Abidin, et al [9] approached the topic by using the CNN model with the results achieved of 97% accuracy for Granny apples and 97% for red apples.

2. PROPOSED METHOD

2.1. Model

Figure 2.1 shows the problem model of recognizing, classifying, and counting fruits of the YOLOv10s and SSD300 models. The data to be tested will be collected from many different sources, which can be images or videos. The image and video data will be processed with an image size of 640x640px for the YOLOv10s model, and 300x300px for the SSD300 model. After being processed, the data will be sent to the models to recognize, classify, and count the number of fruits in the frame, by using the Bounding Box technique to block the object area and then count the number of each object in the frame, which will give the result as shown in Fig. 1.1.

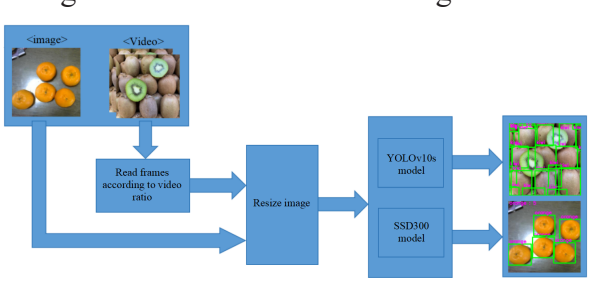

Fig. 1.1. Model architecture

2.2. Dataset

In this topic, our group used the Roboflow

dataset [10] which includes 1077 original images, then the dataset was preprocessed and data augmentation was applied to create other images from the original data. After the images were processed, 2575 images containing fruits such as apples, bananas, kiwis, lemons, oranges, and strawberries were created. The dataset was divided into the following subsets: The training set accounted for 87% of the total number of images, including 2247 images, used for the model training process. The validation set accounted for 8% of the total number of images, corresponding to 216 images, used to evaluate the accuracy of the model during training. The test set accounted for the remaining 4%, equivalent to 112 images, used to evaluate the final performance of the model after training. The image files are identified with an extension of ".jpg" and the labeled images are saved with a file format of ".csv" which includes image information such as image file name, image size, coordinates of bounding boxes, and fruit type names. Below are some images extracted from the dataset.

Table 1.1. Data set description

Fruit name	Apple	Banana	Kiwi	Lemon	Orange	Strawberry
Quantity	212	294	137	124	223	114
Total	1077					

2.3. Data preprocessing

Preprocessing is necessary before feeding data into the model to ensure high quality input. The dataset includes 1077 original images including 6 different types of fruits. To facilitate comparison between models, we have used some data preprocessing techniques and applied some data enhancement techniques, aiming to bring the image data to the same uniform form before being fed into the training model, this helps to compare the effectiveness of different models on the same dataset, thereby giving more objective results. The data preprocessing and data augmentation steps include: randomly flipping the image horizontally or vertically, rotating the image at coordinate angles in the range (-14^o to 14^o), changing the image brightness in the range (-15% to 15%), blurring the image up to 1.4px, and noise up to 1.51% per px, resizing the images to the same size of 320x320. However, for each original image, the data preprocessing and data augmentation steps will be applied to create 3 new images in the training set, but during the creation process, duplicate images will be removed, or some images will not apply the above processing steps, but only the original image will be kept and resized. In addition, the fixed size of 320x320 will be resized by the model to match the input size of the network architecture. For the SSD300 model, the model will resize the image size back to 300x300, and 640x640 for the Yolov10s model.

2.4. YOLOv10s Model and SSD300 Model

YOLOv10 model

This is a recently improved version with many breakthroughs such as speed improvement through one-to-one head and one-to-many head networks and improved feature extraction through a PAN network. Helps improve speed and accuracy in object recognition and classification problems, with the ability to detect quickly, accurately and effectively process complex data sets. YOLOv10 not only inherits the advantages of previous versions but also integrates improvements in structure and new optimization algorithms. YOLOv10 is capable of processing quickly, with high accuracy with data sets of diverse sizes and types of fruit.

SSD300 model

SSD300 is a single-stage object detection model that does not require as much processing time as previous models. SSD is a single-stage object detection method that decomposes the output space of bounding boxes into a set

of default boxes on different aspect ratios and scales for each feature map location. At prediction time, the network generates a score for the presence of each type of object in each default box and makes adjustments to the box to better match the object shape. Additionally, the network combines predictions from multiple feature maps at different resolutions to naturally handle objects of various sizes. The SSD300 model is lightweight and only recognizes objects through a single stage. It achieves high speed and accuracy suitable for low-resource devices.

Performance and results:

With the same dataset, it is shown that the YOLOv10s model achieves higher Precision, Recall, and F1-score than the SSD300. This shows that the model performs better than the SSD300 model.

Table 1.2. Comparing YOLOv10s model with SSD300

Model	YOLOv10s				SSD300			
Batch_size	22	22	32	32	16	16	22	22
Epoch	100	120	100	120	18000	20000	18000	20000
Precision	95%	94%	96%	93%	93%	94%	93%	94%
Recall	93%	95%	95%	95%	91%	92%	90%	92%
F1-score	94%	94%	95%	94%	92%	93%	91%	93%

2.5. Model training

We use Python with libraries such as OpenCV, Numpy, and Tensorflow.... In addition, there is the Ultralytics library that supports loading, training, and optimizing models for YOLOv10s. For the SSD300 model, we will use the TensorFlow API to train the model as well as load the available model. Other models are used to support data processing and will be bounding boxes.

Model training is the process of adjusting the parameters to help the model achieve optimal performance, avoiding overfitting or underfitting. The model optimization process includes choosing hyperparameters such as batch size, learning rate, and number of epochs, which help to adjust the parameters and the convergence speed of the model quickly and accurately.

2.6. Training configuration

The model training process takes place on the Google Colab platform with the T4 GPU used to speed up training and prediction. This GPU supports the intensive computation process in neural network training, which significantly reduces execution time compared to using only the CPU. And also supports storage up to 112GB and 15GB of RAM. Using Python 3 with supporting libraries such as Ultralytics and Tensorflow API to deploy YOLOv10s and SSD300, and other supporting libraries.

The training process of the Yolov10s model is performed 4 times with the following parameters: 1st Epoch = 100, Batch_size = 22, 2nd Epoch = 120, Batch_size = 22, 3rd Epoch = 100, Batch_size = 32, 4th Epoch = 120, Batch_size = 32. The training times achieving accuracy are: 95%, 94%, 96%, 93%.

Table 1.3. YOLOv10s model training parameters

YOLOv10s				
	1st	2nd	3rd	4th
Epoch	100	120	100	120
Batch_size	22	22	32	32
Precision	95%	94%	96%	93%

The training process of the SSD300 model is performed 4 times similar to the Yolov10s model with the following parameters: 1st Epoch = 18000, Batch_size = 16, 2nd Epoch = 20000, Batch_size = 16, 3rd Epoch = 18000, Batch_size = 22, 4th Epoch = 20000, Batch_size = 22. The training times achieved accuracy: 93%, 94%, 93%, 94%.

Table 1.4. SSD300 model training parameters

SSD300				
	1st	2nd	3rd	4th
Epoch	18000	20000	18000	20000
Batch_size	16	16	22	22
Precision	93%	94%	93%	94%

2.7. Model testing and evaluation

After training, both SSD300 and YOLOv10s are evaluated on a test set, which includes fruit images that were not used during training. Evaluation of the test set will help to evaluate the model more objectively. The model performance evaluation metrics include: **Precision**: The proportion of correct predictions over all predictions. **Recall**: The proportion of objects that are actually correctly recognized. **F1-score**: The harmonic mean of precision and recall.

3. RESULTS AND DISCUSSION

3.1. *YOLOv10s model*

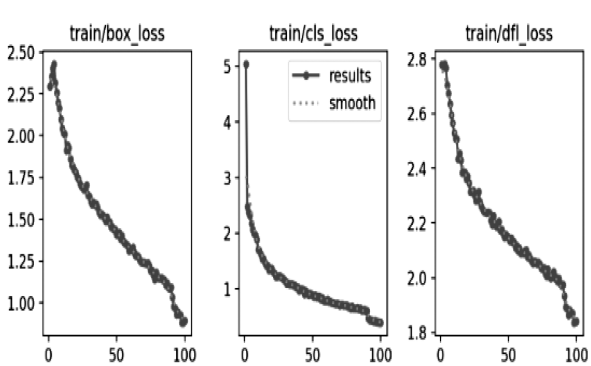

Fig. 3.1. Parameters batch_size=22 and epoch=100

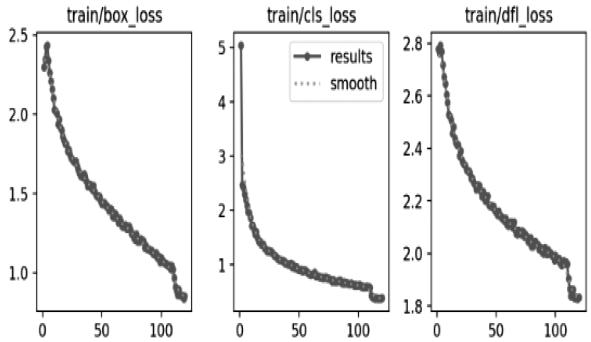

Fig. 3.2. Parameters batch_size=22 and epoch=120

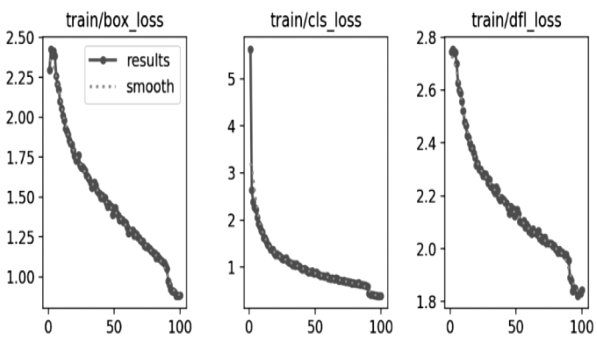

Fig. 3.3. Parameters batch_size=32 and epoch=100

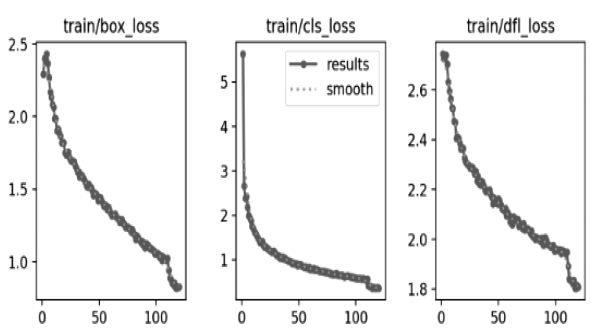

Fig. 3.4. Parameters batch_size=32 and epoch=120

Figures 3.1, 3.2, 3.3, and 3.4, show that the model has a loss function that decreases gradually through each epoch, which shows that the model is learning and improving its parameters actively through each epoch. In which Box_loss tells us the error in predicting the coordinates of the predicted Bounding box and the actual Bounding box, the more Box_loss decreases, the more it will help improve the accuracy when the deviation of the predicted Bounding box coordinates is reduced, thereby helping to localize the objects more accurately. Cls_loss tells us the error in classifying the object, the more

Cls_box decreases, the proof that the model is improving the classification becomes better, when the deviation in predicting the objects of the model is reduced, helping to classify the objects accurately. Dfl_loss will help increase the accuracy of locating the coordinates of the Bounding boxes.

In general, through the above figures, all three Loss functions decrease gradually through each epoch, however, when looking at Cls_loss, we see that it decreases faster than the other two Loss functions, showing that the classification model learns very quickly. Next, the two Box_loss and Dfl_loss functions also decrease but quite slowly, this tells us that optimizing these two functions takes longer than the Cls_loss function.

3.2. *SSD300 model*

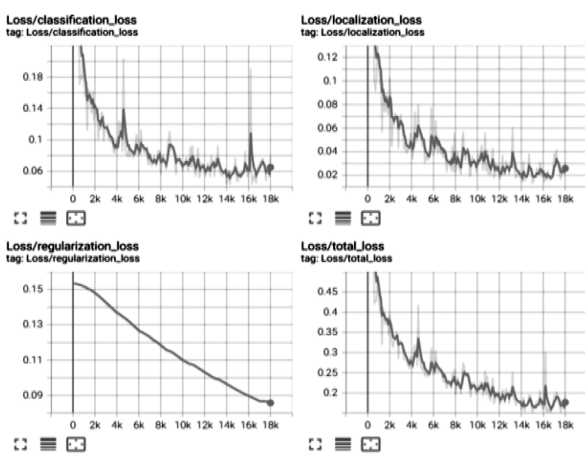

Fig. 3.5. Parameters batch_size=16 and epoch=18000

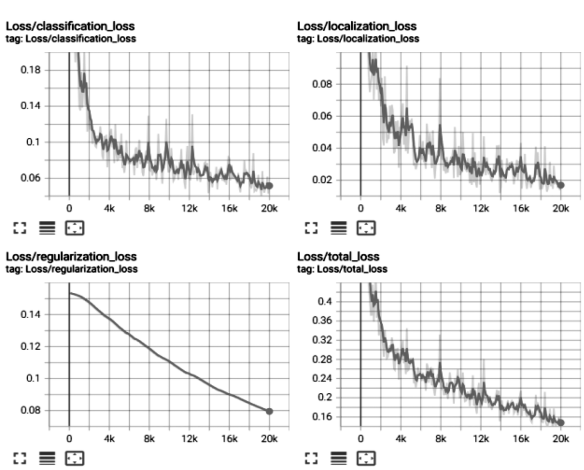

Fig. 3.6. Parameters batch_size=16 and epoch=20000

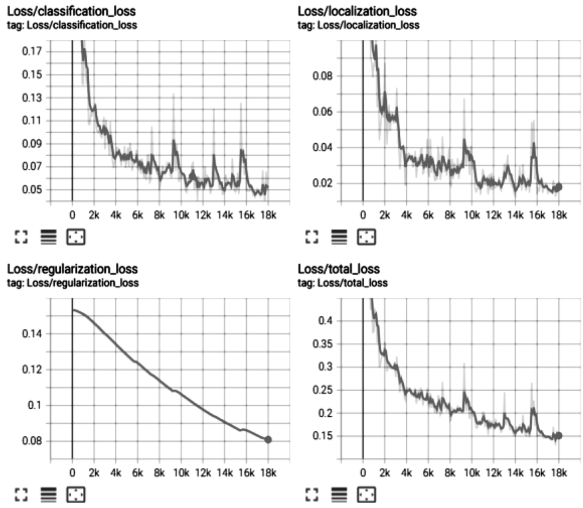

Fig. 3.7. Parameters batch_size=22 and epoch=18000

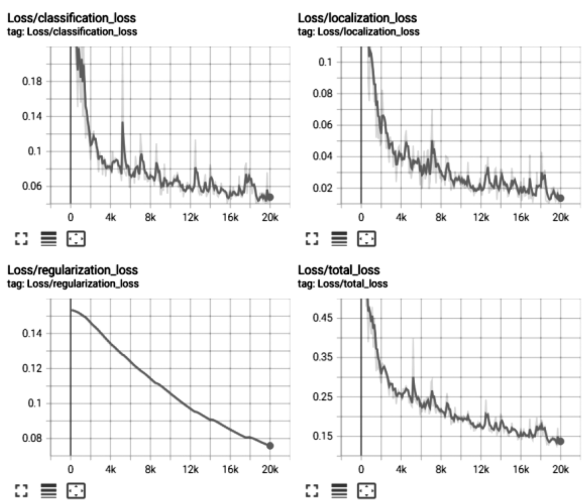

Fig. 3.8. Parameters batch_size=22 and epoch=20000

Figures 3.6, 3.7, 3.8, and 3.9, show that the model has a loss function that decreases gradually through each epoch, but not evenly when there are some epochs with a loss value higher than the loss value of the epoch. This does not affect much the learning process and improve the parameters of the model, but it shows that the convergence speed of the model is quite slow, and has many fluctuations. In which Classification_loss tells us the error in object classification, the more Classification_loss decreases, this proves that the model is improving the classification becomes better when the error in predicting the objects of the model is reduced, helping to classify the objects accurately. Localization_ loss is the deviation of the coordinates of the Bounding box and the size of the Bounding

box, the model learning and optimizing the Localization_loss function will make the model localize the object better, reducing the situation of wrong object localization. Regularization_loss helps the model learn more stable weights, reducing Overfitting. Total_loss is a function that synthesizes the loss functions Classification_loss, Localization_loss, and Regularization_loss to give a general value of the model's loss.

In general, the figures presented above show that the Loss functions decrease over the epochs, however, there is a strong fluctuation in the first 10,000 epochs and a gradual decrease in the following epochs, indicating that the model is gradually stabilizing and converging. In addition, the figure also shows that the Localization_loss function has a faster convergence speed and fewer fluctuations than the Classification_loss function, indicating that the model takes less time to learn and improve when the Localization_loss function fluctuates around 0.03 to below 0.02 while the Classification_loss function has a larger deviation when fluctuating around 0.06.

3.3. *Experimental results*

YOLOv10s model

Fig. 3.9. Experimental results of YOLOv10s model

SSD300 model

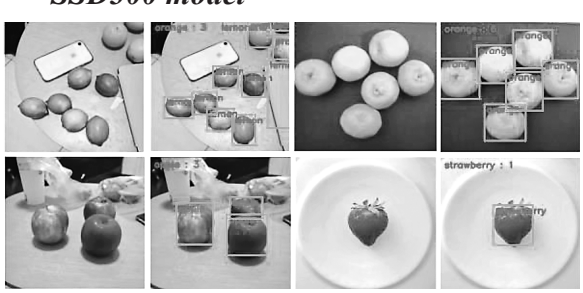

Fig. 3.10. Experimental results of SSD300 model

Through the experimental results of the two models YOLOv10s and SSD300 presented in Figure 3.9 and Figure 3.10, it can be seen that the YOLOv10s model performs better than the SSD300 model. We can observe through Figure 3.11 and Figure 3.12 that with the same input image including 6 lemons and 3 oranges, the YOLOv10s model gives accurate results when correctly identifying 6 lemons and 3 oranges, while the SSD300 model has less accuracy when identifying 7 lemons and 3 oranges because the model mistakenly identifies the Background as a lemon. In addition, the comparison in the same frame of the video shown in Figure 3.13 and Figure 3.14 also shows that the YOLOv10s model gives more accurate results than the SSD300 model, however, because some fruits are hidden or overlapped by leaves, but in general, the YOLOv10s model gives more positive results. From that, we can conclude that the YOLOv10s model gives better results than the SSD300 model.

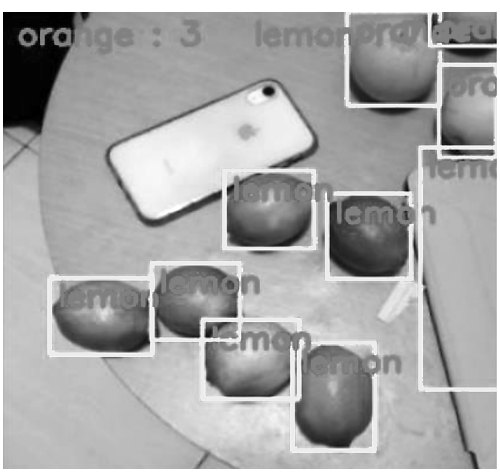

Fig. 3.11. Mô hình SSD300 model

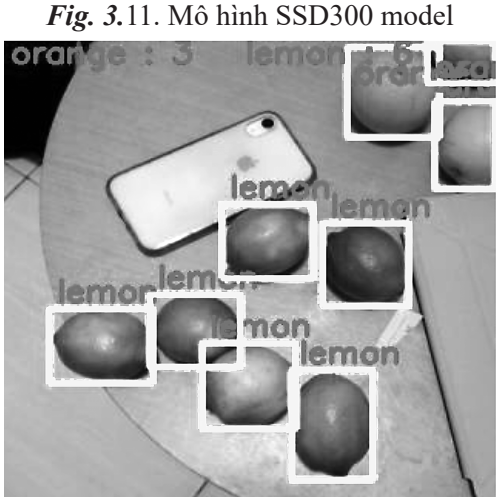

Fig. 3.12. YOLOv10s model

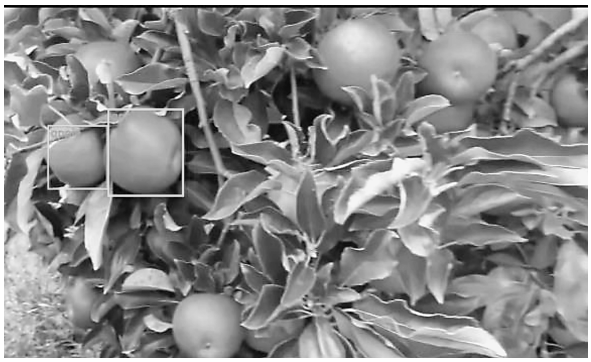

Fig. 3.13. SSD300 model

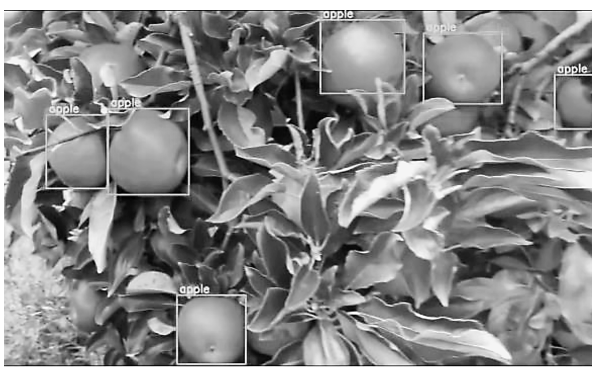

Fig. 3.14. YOLOv10s model

4. CONCLUSION

4.1. *Results achieved*

The research has succeeded in: Applying Deep Learning models to fruit recognition, classification, and counting. Comparing the SSD300 model with the YOLOv10s model. Understanding how deep learning models work and gaining more knowledge about deep learning training libraries and image data processing.

4.2. *Limitations*

There are inevitably some limitations in a topic, so the research inevitably has some limitations. First of all, the scope of the topic focuses on the application of deep learning models to fruit recognition, classification and counting. Although there have been studies on basic knowledge before, there may still be limitations in terms of in-depth knowledge in researching complex and advanced topics in this field of deep learning. In addition, there are limitations in terms of equipment for model training, which can be mentioned here as the time spent using GPU to train the model on Google Colab. Experiments on small datasets may not fully reflect the reality when working with real and complex data.

4.3. *Development direction*

Increasing the data set and using other advanced models to increase productivity, as well as the efficiency of the model to produce good results.

4.4. *Summary*

This study shows that the YOLOv10s model, is capable of recognizing and classifying fruits with higher efficiency than the SSD300 model. Suitable to meet the requirements of real-time applications in the agriculture and food industry.

However, the model still has many weaknesses that need to be improved, such as recognition speed and accuracy. Continuing to find new developments and optimize the model will bring a powerful and effective solution for automating the process of sorting and checking the quality of fruits in the food industry.

REFERENCES

[1]. A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, “YOLOv10: Real-Time End-to-End Object Detection”, arXiv preprint arXiv: 2405.14458, 2024. DOI: 10.48550/ arXiv. 2405.14458

[2]. M. A. R. Alif and M. Hussain, “YOLOv1 to YOLOv10: A Comprehensive Review of YOLO Variants and Their Application in the Agricultural Domain”, arXiv preprint arXiv: 2406.10139, 2024. DOI: 10.48550/ arXiv.2406. 10139

[3]. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. -C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks”, 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, USA, 2018, 4510-4520, DOI: 10.1109/ CVPR.2018.00474.

[4]. C. Cheng, “Real-Time Mask Detection Based on SSD-MobileNetV2”, arXiv: 2208. 13333, 2022. DOI: 10.48550/arXiv.2208. 13333

[5]. J. Leng and Y. Liu, "An enhanced SSD with feature fusion and visual reasoning for object detection" Neural Computing & Applications, 31(10), 6549-6558, 2018. DOI: 10.1007/ s0052 1-018-3486-1

[6]. T.Q. Bao, T.Q. Dinh and N.V. Vung, "Detect and identify defects on mango", Proceedings of the 9th National Conference on Science and Technology, 2016.

[7]. T.T. Hai, N.H.H. Cuong, and N.K. Duy. "The Improved Fast R-CNN Model for Recognition and Detection of Pineapple Ripening". The University of Danang - Journal of Science and Technology, 20(7), 2022, 94-98.

[8]. Gill, H.S., Khalaf, O.I., Alotaibi, Y., Alghamdi, S., Alassery, F. "Fruit image classification using deep learning". Computers, Materials & Continua, 71(3), 2022, 5135–5150. DOI: 10.32604/cmc. 2022.022809

[9]. A. Abidin, Hamzah, M.E. Hiswati, "Efficient Fruits Classification Using Convolutional Neural Network", International Journal of Informatics and Computation, 3(1), 2021. DOI: 10.35842/ ijicom.v3i1.31

[10]. Roboflow, Dataset Versions, https:// universe.roboflow.com/school-qmdcx/fru it-detection-aubry/dataset/24