Journal of Applied Engineering and Technological Science
Vol 5(1) 2023: 305-320

MULTI-OBJECT DETECTION USING YOLOV7 OBJECT DETECTION ALGORITHM ON MOBILE DEVICE

Patricia Citranegara Kusuma[1]*, Benfano Soewito[2]
Computer Science Department, BINUS Graduate Program - Master of Computer Science, Bina Nusantara University, Jakarta,11480, Indonesia[1,2]
patricia.kusuma@binus.ac.id

Received : 04 September 2023, Revised: 10 November 2023, Accepted : 12 November 2023
**Corresponding Author*

ABSTRACT
This research discusses the importance of enhancing real-time object detection on mobile devices by introducing a new multi-object detection system that uses the quantified YOLOv7 model. Focusing on the complexities of food item detection, particularly in diverse and intricate contexts, our study uses a dataset that includes five food classes. By investigating the influence of data quantity on the detection model, we demonstrate the superiority of larger datasets in both YOLOv5 and YOLOv7. In addition, our comparison shows that YOLOv7 has better precision, recall, and F1-score values compared to YOLOv5. The crucial methodological contribution lies in the successful quantification of the YOLOv7 model, reducing the model size from 28.6 KB to 14.3 KB and enabling seamless mobile application development. This high-performance mobile application displays a real-time interface response time of 235ms, with precision, recall, and F1-score values of 0.923, 0.9, and 0.911, respectively. Beyond the practical implications for informed dietary choices and improved health outcomes, our study develops object detection techniques theoretically, offering valuable insights that can be applied across various domains and emphasizing the potential impact of our approach on both theory and practice.
Keywords: *Multi-Object Detection, YOLOv7, Quantization, Mobile Application.*

1. Introduction

Object detection, recognition, and identification signify remarkable strides in technological advancement, particularly in the domain of computer vision. This intricate field of technology has seen great improvement because of breakthroughs in deep learning, which has significantly increased processing capacity and computation power. Notable examples of Multi-label Object detection and recognition models that leverage deep learning include YOLO, SSD(Kumar et al., 2020), F-RCNN(Y. Wang et al., 2019), and CNN(Bhatt et al., 2021). These models demonstrate their ability to handle this difficult task with impressive accuracy and efficiency by employing bounding boxes to detect objects in images.

The application of object detection and recognition spans various media types, including pictures, photos, and videos(Sharma & Thakur, 2017), and can even be performed on real-time camera-captured objects (Lu et al., 2019). Despite the promise challenges do arise in machine analysis, especially when objects have complex or similar-colored backdrops. For example, determining the maturity level of oil palms with similar forms and colors can be difficult(Mansour et al., 2022). Similar to that, recognizing food objects becomes intricate due to variations in size, shape, and color. Placing food on plates or bowls complicates detection even more. Additionally, occlusion, where objects are partially or completely obscured by other elements in the image, such as plates and silverware, poses another obstacle in food object recognition. Despite these challenges, the continuous progress in object detection and recognition technologies offers promising solutions and brings us closer to more accurate and reliable systems in various practical applications.

The advent of real-time multi-object detection system, equipped with the capability to provide specific information, holds immense potential in various fields and applications, including traffic monitoring (Akhtar et al., 2022), robotics (Zhang et al., 2022), etc. Imagine a system that can perform real-time object detection while providing valuable species-specific information, such as identifying whether a plant or animal is poisonous and offering guidance on how to handle toxic species, as well as details about their habitats. By offering such information, this object detection system empowers users to make informed judgments and take appropriate actions based on the objects they encounter.

The significance of food detection and recognition extends across various domains, including culture, health (Batal et al., 2021) (utilized for assessing the nutritional content of food), tourism(Almansouri et al., 2021), and the economy(Fukase & Martin, 2020). Furthermore, the food recognition system could be further developed and implemented as an automatic billing machine for restaurants and eateries, streamlining their operations and enhancing customer experience. The integration of real-time object detection with pertinent information provision opens exciting opportunities for practical applications in diverse industries and societal contexts.

Our primary goal at the beginning of this research was to develop a smartphone application that can provide real-time detection and nutritional information. This application holds the potential to significantly impact users' daily lives, promote healthy eating and enable individuals to make informed decisions tailored to their specific dietary needs and goals. The success of our research is based on recent advances in deep learning and object recognition, especially by using the YOLOv7 algorithm which has been used for detecting objects in many domains. It should be noted that the implementation of YOLOv7 on smartphones is relatively uncommon, pointing to notable limitations and gaps in the current research that our work attempts to address.

In the landscape of deep learning and object detection, recent advancements and trends play a pivotal role in bolstering the feasibility and success of our research. Specifically, the emergence of object-detection algorithms such as YOLOv7 has been a game-changer. YOLOv7 is a PyTorch-based algorithm that takes advantage of the Python library's deep learning capabilities, particularly in the development of deep neural networks(Redmon et al., 2016). Additionally, the image dataset utilized in this study diverges from prior research endeavors. Researchers will meticulously curate a comprehensive dataset from Google Images, encompassing five types of food: Fried Chicken, Rice, Sambal, Fried Tofu, and Fried Tempe. The selection of this dataset stems from the abundance of food images available on the internet, facilitating comprehensive training. This dataset will undergo rigorous preprocessing, augmentation, and annotation to ensure optimal performance. The YOLOv7 model will be trained and followed comparative analysis with the YOLOv5 model to evaluate training results. After that the YOLOv7 trained model will be quantized to reduce the model size. The application will be built using Android studio and the quantized YOLOv7 model.

In summary, the primary objective of our research is to develop a smartphone application capable of real-time food detection and providing accurate nutritional information, including calories, protein, carbohydrates, and fat content, for recognized food items. By promoting healthier eating habits and empowering individuals to make informed decisions aligned with their specific dietary needs and goals, food detection technology has the potential to revolutionize the way people interact with food and enhance overall well-being. The innovative approach of real-time multi-object detection holds significant promise for practical applications across various domains, making it an exciting and potentially transformative avenue for further exploration.

2. Literature Review

History of Object Detection

In the early period object detection, the paradigm revolved around handcrafted features such as Haar-like features and Histogram of Oriented Gradients (HOG) as the means to encapsulate the essence of objects within images(Arunmozhi & Park, 2018). This methodology entailed the systematic traversal of a sliding window across the image canvas, extracting intricate features from each window. Subsequently, these meticulously discerned features played a pivotal role in training a classifier, honed to distinguish with precision between windows containing objects and those merely portraying background elements. This foundational approach marked the rudimentary stage in the evolution of object detection algorithms, laying the groundwork for the subsequent strides in the realm of computer vision.

The introduction of deep learning has changed the landscape. Deep learning models, such as convolutional neural networks (CNNs), are capable of learning complex object models from data without the need for any knowledge(Dhillon & Verma, 2020). This greatly improves the accuracy and speed of object recognition algorithms. CNN architecture has many variations, including YOLO algorithm, LeNet (Wei et al., 2019), AlexNet (H.-C. Chen et al., 2022), ResNet (Gao et al., 2021), VGG (Sitaula & Hossain, 2021). and others.

Previous research in the domain of object detection has yielded significant findings using various algorithms and datasets. One study utilized the Faster R-CNN, focusing on detecting small objects. Resulting in an accuracy of 87% and 90% of recall (Cao et al., 2019). Another study focused on wheat spike detection under occlusions using UAV images and YOLOv5 model, achieving a mean average precision (mAP) value of 94.1%(Zhao et al., 2021). Using a massive dataset containing over 20,000 images, Kagaya, Aizawa, and Ogawa (Kagaya et al., 2014) extended their research to detect ten types of Japanese food, achieving an impressive accuracy of 89.7% using the CNN model. Additionally, object detection research on the "Xiaomi Blackshark" smartphone utilized the Tiny-YOLOv3 algorithm in conjunction with the COCO dataset, where Tiny-YOLOv3 demonstrated a mAP accuracy of 33.1 with a model size of 33.8 MB, achieving a detection speed of 17.7 frames per second(Martinez-Alpiste et al., 2022).

History of YOLO

YOLO (You Only Look Once) is a CNN-based object detection algorithm that is currently in the development process. The YOLO algorithm was first discovered and introduced by Redmon et al. in 2015, known as YOLOv1 (Redmon et al., 2016). Over time, the YOLO algorithm has undergone continuous development and improvement. In 2017, YOLOv2 and YOLO9000 were introduced (Redmon & Farhadi, 2017),followed by YOLOv3 in 2018 (Redmon & Farhadi, 2018). In 2020, YOLOv4 was introduced by Alexey Bochkovskiy et al. (Bochkovskiy et al., 2020). Subsequently, YOLOv5 was discovered by a company named Ultralytics in 2020. YOLOv5 was found to be a modification of YOLOv4, integrating the anchor box selection process into the model. It is the first YOLO model to use PyTorch as a framework (Zaidi et al., 2022).The YOLO algorithm has advantages, including high accuracy in real-time object detection, simplicity, and efficiency due to using a single neural network for detection and classification, and adaptability for object detection in custom datasets. YOLOv7 is a relatively new object detection model introduced in 2022 by Chien-Yao Wang et al.(C.-Y. Wang et al., 2022), exhibits superior accuracy and speed compared to other object detection algorithms, including YOLOv4, YOLOX, YOLOR, Scaled-YOLOv4, YOLOv5. While sharing a similar architecture with YOLOv5, YOLOv7 incorporates several improvements, resulting in shorter interface and training times in comparison.

The YOLOv5 algorithm, introduced by Ultralytics in 2021, marks the fifth generation of the YOLO series(Zaidi et al., 2022). While YOLOv5 doesn't extensively compare with its predecessor YOLOv4, the latter stands out for its significant contributions, boasting a 10% increase in Average Precision (AP) and a 12% enhancement in Frames Per Second (FPS) over YOLOv3 (Deng, et al., 2020). YOLOv5 is notably the first iteration of YOLO to integrate the PyTorch framework, leveraging the larger PyTorch community compared to the Darknet framework employed by YOLOv4. This utilization of PyTorch grants YOLOv5 an edge in achieving superior results (Thuan, 2021). Architecturally, YOLOv5 possesses a lighter model weight than YOLOv4, which provides advantages encompassing increased accuracy, reduced computational load, and enhanced detection speed (Yan, Fan, Lei, Liu, & Yang, 2021). The architecture of YOLOv5 comprises three main components: a head, neck, and backbone, outlined in Figure 1.

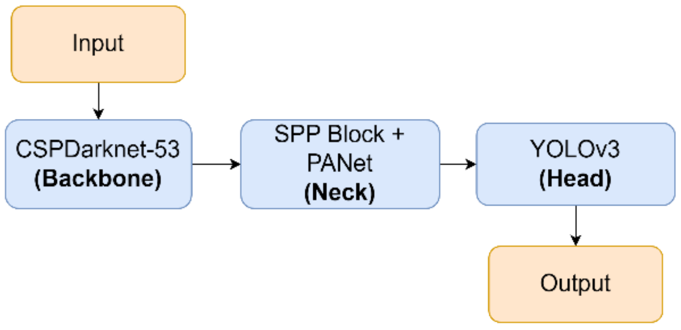


Fig. 1. YOLOv5 Architecture

Its backbone relies on the CSPDarknet53 architecture, a modified version of ResNet, encompassing 53 convolutional layers arranged in residual blocks. This arrangement empowers the YOLOv5 backbone to generate three distinct outputs from feature maps, facilitating multi-scale prediction(Nepal & Eslamiat, 2022). Subsequently, within the YOLOv5 architecture, SPP and PANet are integrated as the neck components. This neck architecture serves the crucial purpose of amalgamating features, primarily focused on enhancing the model's efficacy in detecting objects across various scales (Yao et al., 2021). Notably, the head employed in YOLOv5 remains consistent with the head utilized in both YOLOv4 and YOLOv3.

To effectively detect smaller objects, augmenting the backbone with extra layers and a larger input size becomes imperative. This necessity is underscored by the utilization of CSPDarknet53, a choice attributed to its requirement of a 512 by 512-pixel input image. As we transition to the architecture's neck, a strategic fusion of PAN (Path Aggregation Network) and SPP (Spatial Pyramid Pooling) components comes into play. Through a maximal pooling mechanism, SPP adeptly merges feature maps sized at 19x19x512, utilizing diverse kernel sizes and uniform padding (maintaining spatial consistency). PAN also introduces a transformative change, replacing the traditional summation process with a concatenation approach. The culmination of these collaborative efforts results in the synthesis of four interconnected feature maps, resulting in a comprehensive volume measuring 19x19x2048 (Bochkovskiy et al., 2020). This intricately orchestrated orchestration within the model's architecture contributes significantly to its improved ability to detect objects of varying scales.

YOLOv7 is a novel object detection model introduced in 2022 by Chien-Yao Wang et al.(C.-Y. Wang et al., 2022), boasting exceptional accuracy and speed performance that outperforms established counterparts such as YOLOv4, YOLOX, YOLOR, Scaled-YOLOv4, YOLOv5, DETR, Deformable DETR, DINO-5scale-R50, and ViT-Adapter-B. While sharing architectural similarities with YOLOv5, YOLOv7 integrates several enhancements and modifications. This upgrade yields a streamlined interface and reduced training time when compared to its YOLOv5 counterpart. Both YOLOv5 and YOLOv7 harness the PyTorch framework for their operations. However, a pivotal deviation of the YOLOv7 algorithm from previous iterations lies in its architectural innovation, utilizing the E-ELAN (Extended Efficient Layer Aggregation Network) computation block. This unique component introduces randomization, expansion, and cardinality combinations to enhance the network's learning capabilities without compromising the integrity of the original gradient. E-ELAN also manages the shortest and longest gradient paths in each layer to expedite network convergence and effective learning. A visual representation of the E-ELAN architecture is presented in Figure 2, encapsulating the cutting-edge design principles that underpin YOLOv7's advanced performance.

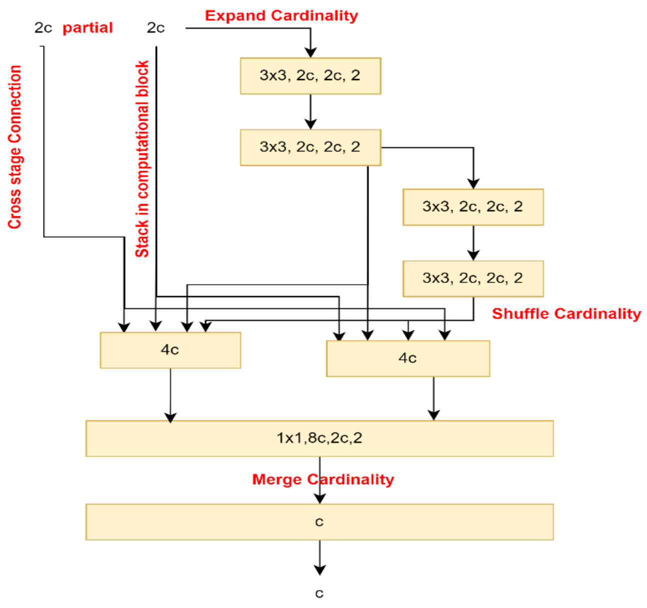


Fig. 2. Extended Efficient Layer Aggregation Networks

In YOLOv7, pivotal features of the model undergo strategic adjustments through a comprehensive scaling strategy, aligning the model to cater to diverse application requisites. This entails augmenting the model's depth (characterized by the number of stages), breadth (indicating the number of channels), and resolution (reflecting the input image's dimensions). When dealing with models founded on amalgamated architectures, YOLOv7 implements a compound model scaling approach, adeptly circumventing challenges posed by volatile ratios that materialize during model expansion. This compound model scaling technique ensures that the model retains its original design characteristics, thus safeguarding its foundational structure.

To improve the model's resistance to the prevalent patterns it attempts to detect, YOLOv7 adeptly employs a re-parameterization strategy that involves the averaging of a variety of model weights. This strategic combination improves the model's ability to capture robust representations of larger trends in the data. YOLOv7 also employs RepConv without an identity connection (RepConvN) to avoid identity connections that may occur when convolutional layers undergo replacements such as residuals or concatenation. This tactical integration effectively reduces any complications caused by identity connections, ensuring the model's convolutional processes' integrity and efficacy (C.-Y. Wang et al., 2022).

At the core of YOLO's architecture are its backbone, neck, and head components, with the head responsible for generating the model's anticipated output. YOLOv7 draws inspiration from Deep Supervision, a technique employed in training deep neural networks, deviating from a rigid single-head structure (Shen et al., 2020). While YOLO heads bear the primary responsibility of producing network predictions, their positioning at the network's terminal end prompts the integration of additional heads at intermediary junctures to facilitate more effective training. Recognizing that auxiliary heads are less efficient in training compared to the final head, YOLOv7 introduces a nuanced coarse-to-fine framework. In this approach, supervision is propagated from the leading head to diverse granularities, enhancing the model's training dynamics. Augmenting network training, a label assigner mechanism enriches YOLOv7's capabilities. This mechanism incorporates network predictions and ground truth, subsequently assigning labels, thereby refining the model's predictive accuracy and overall performance.

The YOLOv7 algorithm stands out as an impressive real-time object detection approach, utilizing neural networks to achieve its capabilities. Previous research on citrus plantations revealed that YOLOv7 offers an advantage over the SSD model in terms of speed and obtaining a more stable mean average precision (mAP) value (J. Chen et al., 2022). The YOLOv7 model achieved an outstanding mAP value of 94.64%, while SSD achieved 92.81%. Moreover, YOLOv7 excels in convergence, reaching it faster compared to SSD, with interface times of 78.27 ms and 91.02 ms, respectively. Furthermore, Wu, et al. conducted research to detect *Candamellia oleifera*, wherein YOLOv7 outperformed other models like YOLOv5s, YOLOv3-spp, and Faster R-CNN in terms of mAP performance, precision, recall, and F1 Score. In addition, YOLOv7 demonstrated the smallest Average Detection Speed when compared to YOLOv5s, YOLOv3-spp, and Faster R-CNN, with values of 0.025, 0.054, 0.072, and 5.167, respectively(Wu et al., 2022). Another research study utilized YOLOv7 to detect small objects, specifically traffic signs, utilizing the public dataset TT100K. The study, conducted with the Improved YOLOv7, achieved an mAP@0.5 of 88.7% (S. Li et al., 2023).

These diverse studies underscore the remarkable progress and potential inherent in object detection technologies. Despite the advancements made, persistent gaps and limitations in existing approaches necessitate further enhancements. Addressing these gaps, our proposed real-time food detection system for smartphones capitalizes on the robust YOLOv7 algorithm. Recent studies, as illustrated above, consistently demonstrate YOLOv7's superiority in terms of speed, stability, and accuracy across various applications. This underscores its potential not only to overcome limitations observed in previous models but also to be effectively implemented in smartphones. The identified gaps in standardized evaluation metrics, dataset diversity, hardware variability, and the underexplored application of YOLOv7 on smartphones emphasize areas where future research can contribute to a more comprehensive and applicable understanding of object detection techniques.

Pytorch

PyTorch, a Python package, emerges as an asset for exploring deep learning, particularly in the construction of complex deep neural networks. PyTorch's essence lies in two goals: first, providing an automated differentiation engine that deftly computes gradients for neural network models, thereby streamlining the training endeavor; and second, providing a conducive training environment. Furthermore, PyTorch includes a NumPy-like library that allows practitioners to perform tensor operations on the GPU, accelerating computational throughput. The benefits of Pytorch include that it is simple to use, quick since it uses the GPU, versatile because it can be used for a variety of machine learning applications, and the Pytorch community is large, active, and has resources to assist large users (Imambi et al., 2021).

Integration of Object Detection a Mobile Technology

Previous research on object detection in mobile technology has employed various methods. J.W. Chen et al. developed a smartphone application for pest detection using the YOLOv4 algorithm, achieving 100% accuracy in mealybugs, 89% in Coccidae, and 97% in Diaspididae (J.-W. Chen et al., 2021). J.W. Chen et al. demonstrated that the main challenge lies in collecting a large dataset. Another study utilized YOLO-5x for Road Damage Detection on smartphone images, achieving an F1-score of 0.58. The authors suggested that the model is suitable for real-time detection (Jeong, 2020)In 2015, real-time food recognition on smartphones was accomplished using a traditional object detection algorithm, linear SVM, and HOG, achieving an accuracy of 79.2% (Kawano & Yanai, 2015). Another study focused on food detection on mobile devices, employing a transfer learning to CNN model and customizing 29 varieties of food, resulting in an impressive accuracy of 95.55% (Fakhrou et al., 2021).

To ensure real-time performance on mobile devices, optimizing the model architecture by favoring lightweight designs and implementing quantization techniques is essential. Utilizing hardware acceleration, such as GPU, DSP, or NPU, along with converting models to formats compatible with mobile inference engines like TensorFlow Lite or Core ML, accelerates inference speed. Strategies like depth-wise separable convolutions and pruning help reduce the number of parameters and computations. Adjusting input resolution, adopting anchor-free detection approaches, and optimizing post-processing steps contribute to algorithmic efficiencies. Efficient memory usage and caching mechanisms, along with exploring edge computing possibilities, enhance overall performance. Adaptive frame skipping strategies and regular updates further contribute to maintaining real-time constraints while ensuring optimal computational efficiency and accuracy.

Quantization

Quantization simplifies machine learning models, reducing size and computational complexity by encoding weights (Garifulla et al., 2021). In this study, we use quantization to shrink the YOLOv7 object detection model, allowing effective performance on modest smartphones. Various methods, including uniform quantization, linear quantization, and bfloat16 quantization, can be applied in this process. Precision is reduced from 32-bit floating-point to lower bit-width representations (Yang et al., 2019). The primary goal is to maintain model performance while significantly decreasing memory and computational demands, enabling the operation of complex models on resource-constrained mobile devices. Essential aspects of model quantization involve precision reduction through bit quantization, exemplified by converting from 32-bit floating-point to 8-bit integers. Quantization schemes like uniform and non-uniform quantization offer diverse strategies for assigning quantization levels. Quantization-aware training initiates at the beginning of the training process to minimize performance impact. Dynamic quantization provides adaptive precision during inference, accommodating each layer's specific needs. The benefits encompass optimizing model size, reducing memory footprint, lowering computational requirements, enabling faster inference, and enhancing energy efficiency. Quantization techniques are pivotal for deploying deep learning models effectively on mobile platforms, ensuring adherence to device constraints while maintaining acceptable performance levels (Yang et al., 2019).

3. Research Methods

The goal of this study is to create a multi-object detection information system for Android mobile devices. To enable its integration into smartphone applications, this study aims to build an information system that makes use of a multi-object detection model while improving both speed and accuracy. Figure 3 illustrates the research framework used to develop this system, outlining the structure and methodology guiding this work.

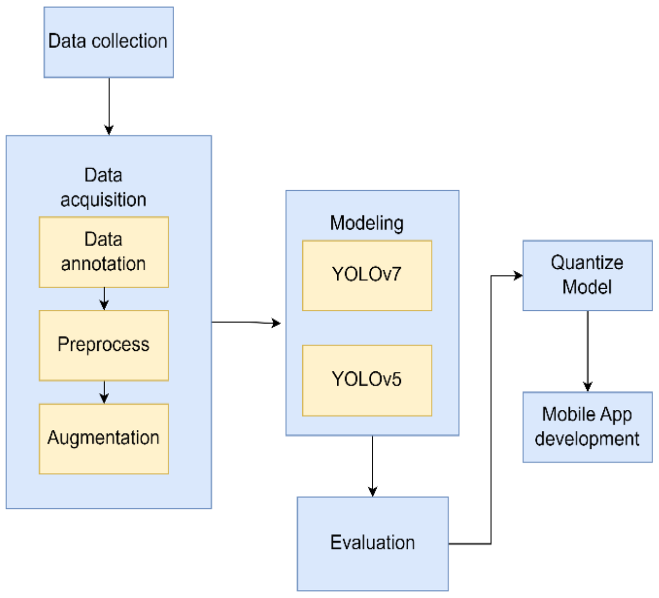


Fig. 3. Research Framework

The initial step in building a multi-object detection system is to choose an appropriate dataset. While there are a variety of datasets available for Multi-Object Detection research, this study focuses on a food dataset with five classes: Fried Chicken, Rice, Sambal, Fried Tofu, and Fried Tempe. This dataset was sourced from Google images and subsequently labeled and annotated using Roboflow tools. The collected data is pre-processed and then divided into training, validation, and testing subsets to ensure optimal training. To assess the impact of data quantity and augmentation on model performance, Roboflow tools are used to expand the dataset by factors 2, 3, and 4.

The modeling phase employs YOLO (YOLOv5 and YOLOv7) models on Google Colab. A comparative analysis is conducted between the two YOLO models, using datasets comprising a total of 552, 1136, 1581, and 2228 instances. Following model selection, hyperparameter tuning is executed, paving the way for the training process. Subsequently, the chosen model undergoes testing on both training and validation data, extending to detection on test data. Performance evaluation encompasses precision, recall, f1-score, and mAP, ultimately identifying the optimal model.

Upon selecting the best model, the researcher proceeds to quantize the best model for implementation on devices. Additionally, an assessment of mAP, precision, recall, and F1-score are conducted based on the volume of training data used. The best model is then quantized and deployed on mobile devices, establishing a system capable of providing information on detected objects via smartphone platforms. This comprehensive approach underscores the intricate process of developing an effective multi-object detection system for practical application.

a. Data Collection and Data Acquisition

The dataset employed in our research comprises five distinct classes: Fried Chicken, Rice, Sambal, Fried Tofu, and Fried Tempe. Figure 4 (a)-(e) provides an illustrative example of the data collection process. The initial data collection yields a total of 552 instances, with each class containing approximately 100-110 images. Following collection, the data undergoes annotation and labeling facilitated by Roboflow tools. Subsequent preprocessing involves resizing all data to dimensions of 512 x 512 pixels. To augment the dataset, a series of techniques are applied, including horizontal and vertical flips, 90° rotations (Clockwise, Counter-Clockwise, Upside

Down), and adjustments in brightness (ranging between -30% and +30%) and saturation (ranging between -20% and +20%). These augmentation techniques significantly expand the dataset, resulting in a total of 1136, 1581, and 2228 data points. Each augmented dataset is then divided into distinct subsets: 70% for training, 20% for validation, and 10% for testing. This approach ensures that the impact of data quantity, image preprocessing, and data augmentation on the object detection model can be thoroughly assessed. By systematically exploring datasets with varying sizes and incorporating diverse preprocessing techniques, this research aims to glean insights into the intricate dynamics of object detection performance.

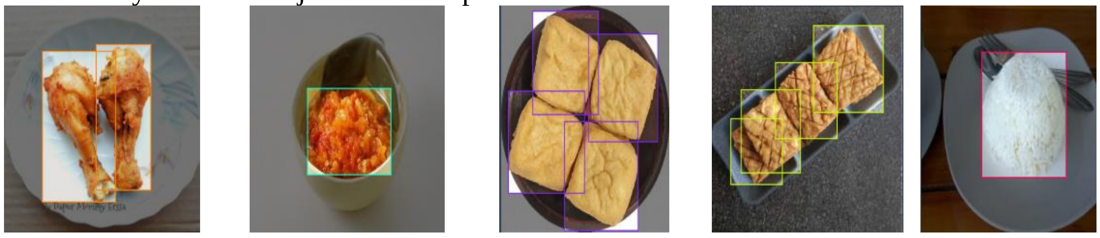

(a) Fried Chicken (b) Sambal (c) Fried Tofu (d) Fried Tempe (e) Rice

Fig. 4. Data Collection

The data collection and annotation process implemented rigorous quality control measures to ensure precision and minimize biases. This involved specific keyword selection for Google Images searches such as "*Ayam goreng","Sambal","Tahu goreng","Tempe goreng","nasi putih"*, followed by meticulous manual labeling conducted by annotators (author). Consistency was maintained through predefined guidelines, with regular reviews and iterations to address errors. Efforts were made to mitigate biases by promoting diversity in the dataset, and annotators underwent training sessions to align labeling standards. Overall, these measures were aimed at producing a reliable and unbiased dataset crucial for the subsequent success of model training and the effectiveness of the object detection system.

b. Model

The research reason for the selection of YOLOv5 and YOLOv7 as object detection models is driven by their specific strengths, geared towards efficient real-time multi-object detection on mobile devices. YOLOv5, recognized for its speed and streamlined architecture with anchor boxes, ensures swift inference crucial for real-time applications. YOLOv7, a more recent version, enhances its predecessors with the E-ELAN computation block for effective learning and anchor-free detection, eliminating predefined anchor boxes for faster and more accurate results. Compatibility with the PyTorch framework and the CSPDarknet53 backbone further solidifies their appeal for seamless integration into the study's mobile application development using Android Studio. Together, these models align with the study's objectives, combining speed, accuracy, and adaptability for effective real-time multi-object detection on mobile platforms. Additionally, the study aims to assess whether the YOLOv7 model surpasses its predecessor, the YOLOv5 model, in terms of performance.

In the training process for both YOLOv5 and YOLOv7 models, achieving optimal performance involves a crucial step of hyperparameter tuning. Hyperparameters, predetermined before training, include the number of epochs, batch size, learning rate, and the optimization algorithm. The models undergo training for multiple epochs (20, 40, 60, 80, and 100), with performance evaluation at each epoch, and the model exhibiting the best performance after 100 epochs is chosen. The batch size is set to 16, representing the number of training samples used in one iteration, impacting model updates and convergence stability. The learning rate, set at 0.01, determines the step size during iterations, with a higher rate enabling faster convergence but risking overshooting, and a lower rate ensuring slower but more precise convergence. Employing the Adam optimizer, a popular algorithm combining features of AdaGrad and RMSProp, adds adaptability to learning rates. By fine-tuning these hyperparameters, the training process aims to balance model convergence, accuracy, and efficiency, crucial for the overall performance of YOLOv5 and YOLOv7 in real-time object detection on mobile devices.

c. Evaluation

The evaluation and analysis process will involve assessing key metrics such as precision, recall, f1-score, and mean Average Precision (mAP) using the testing data. The model exhibiting the highest precision, recall, f1-score, and mAP will then be selected for implementation on mobile devices. To derive the percentages for these metrics, a crucial tool is the confusion matrix. The confusion matrix, designed for classification models, allows for a comparison of predicted and actual class labels across all classes, providing insights into accuracy and error evaluation (Moritz, et al., 2022). The matrix provides values for True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), essential for performance calculations. Notably, true negatives (TN) are not typically employed in object detection, given the indefinite number of undetected bounding boxes within an image (Padilla, Netto, & da Silva, 2020). Refer to Figure 5 depicts a visual representation of the Confusion Matrix. This comprehensive approach will shed light on the model's efficacy and guide its practical implementation in real-world applications.

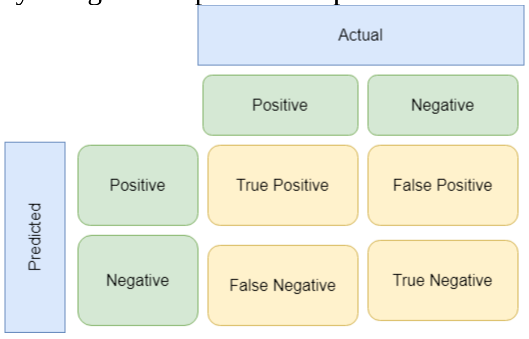


Fig. 1. Confusion matrix

Precision, an important metric, indicates the model's ability to accurately identify desired objects within images. Precision in predictions is calculated as the percentage of successful positive predictions compared to all temporary object detection outcomes. (Padilla, Netto, & da Silva, 2020). Recall, also known as sensitivity, evaluates the model's effectiveness in detecting relevant objects among all existing relevant items. Recall provides insight into the model's ability to detect genuine objects by quantifying the percentage of correctly identified positive predictions against all ground truth data. (Padilla, Netto, & da Silva, 2020). A higher precision value indicates more accurate detection results, whereas a higher recall value indicates successful identification of a large number of ground truth objects.

Through the harmonic average of these values, the F1-score, a composite metric, combines precision and recall into a single measure. This allows for a more balanced evaluation of model performance in classifying positive and negative samples, resulting in a more comprehensive evaluation. (Zhao & Li, 2020). Meanwhile, the mean Average Precision (mAP) emerges as a crucial evaluation tool for object detection models, assessing their proficiency in categorizing and localizing diverse objects. mAP leverages precision and recall values from detection results, offering a comprehensive measure of accuracy across multiple categories. The mAP value itself represents the average precision across all classes, providing an insightful overview of the model's performance (Padilla, Netto, & da Silva, 2020). The concept of Average Precision (AP) embodies a precision-recall curve, elucidating the algorithm's accuracy during iterative processes. The precise formulation of recall, F1-Score and mAP calculation can be seen on the equation 1- 4 below. These metrics collectively enable a comprehensive assessment of the model's proficiency in object detection and classification.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{ground\ truth}$$

$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

d. Quantize Model and Mobile App Development

Quantization, a technique aimed at streamlining machine learning models, serves to diminish the size and computational complexity by encoding model weights and activations using lower-precision data types. Remarkably, this can be achieved without significantly compromising model accuracy (Garifulla, et al., 2021). In the context of this study, the post-training quantization approach will be employed to optimize the YOLOv7 object detection model. Post-training quantization entails transforming a learned model into a lower-precision representation, achieved through diverse methods such as uniform quantization, linear quantization, and bfloat16 quantization. Since the YOLOv7 model is built on the PyTorch framework, the quantization process necessitates converting it to a TFLite model to ensure compatibility with mobile devices. A comprehensive depiction of the conversion process from PyTorch to Quantize TFLite fp-16 can be observed in Figure 6. This critical phase contributes significantly to the model's practical implementation on mobile platforms.

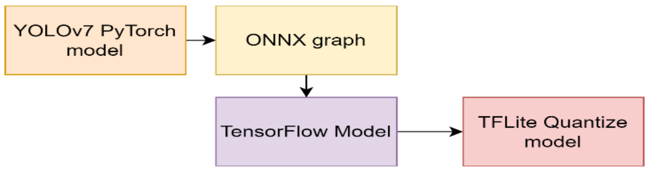
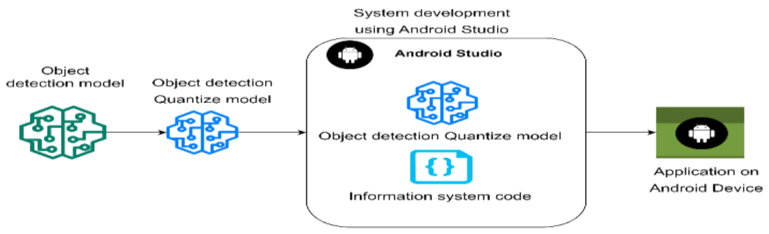


Fig. 2. PyToch to TFLite Process

A series of steps are involved in the process of quantizing the YOLOv7 model using the PyTorch framework. The YOLOv7 model is first transformed into an ONNX graph, a representation that adheres to the Open Neural Network Exchange (ONNX) standard. ONNX is an open standard for machine learning models that defines a unified set of operators as well as a standardized file format. This compatibility allows for seamless model sharing across multiple machine-learning frameworks and simplifies deployment on a variety of hardware platforms. Furthermore, the ONNX graph includes critical model metadata such as the model's name, creator, and description, which aids in the documentation and understanding of the model's functionality (Lin, et al., 2019). Following the conversion to the ONNX model, the model is then translated into the TensorFlow framework. This conversion makes use of the TensorFlow model library, which includes tools for converting to the TensorFlow Lite model. TensorFlow models use full floating-point precision (fp-32) by default. However, this research approach employs Half-Precision Floating Point Quantization (fp-16). This is significant because TFLite fp-16 uses only 16 bits of memory, reducing storage requirements, memory bandwidth, power consumption, and inference latency. These advantages translate into improved operational efficiency, reduced resource demands, and increased computational speed, all of which contribute to lower service latency and lower deployment infrastructure requirements (Garifulla, et al., 2021). This cascading transformation and optimization process ensures that the YOLOv7 model is used effectively in resource-constrained mobile environments.

The integration of the quantized model into the mobile app allows for its deployment on the device. The coding process for application creation becomes critical to ensure the model's compatibility with the device. This includes developing an information system specific for detected objects, which is a critical aspect of mobile application development within the Android Studio environment. Figure 7 shows a comprehensive delineation of the application development process, providing a structured overview of the stepwise progression. The seamless use of real-time object detection on Android devices is highlighted by this cohesive amalgamation of the quantized model and intuitive mobile application construction.


System development using Android Studio
Android Studio
Object detection model
Object detection Quantize model
Object detection Quantize model
Information system code
Application on Android Device
Fig. 3. Application Development Process

The developed application has a user-friendly interface, resembling the experience of opening the smartphone camera. Upon directing the camera towards the target food, the application employs real-time object detection, presenting the results through a bounding box on the detected item. Additionally, crucial nutritional information, including calories, fat, carbohydrates, and protein, is displayed prominently at the top of the detected bounding box. This intuitive design enhances user interaction, providing instant and comprehensive insights into the nutritional composition of the identified food items, thereby promoting informed dietary choices.

4. Results and Discussions

The research was undertaken to assess the impact of data volume and augmentation on the model, while also facilitating a comparative analysis of detection outcomes between the YOLOv5 and YOLOv7 models. Once the model generation phase was completed, the investigation progressed to the practical implementation of the model on Android mobile devices, achieved through model quantization. To arrive at the evaluation results, it was essential to conduct the modeling of the YOLOv5 and YOLOv7 models. This phase encompassed modeling executed across 4 distinct datasets, each with varying data quantities (552, 1136, 1581, and 2228), utilizing an Epoch of 100 and Batch size of 16. The ensuing evaluation, as detailed in Table 1-2, notably indicates that the optimal evaluation outcomes for both models emerged within the dataset containing 2228 data points. This underscores the profound influence of data volume and data augmentation on the performance of the detection model.

Table 1 - YoloV5 Evaluation Result

YOLOv5				
Data	Precision	Recall	mAP@0.5	F1-score
552	0.635	0.649	0.678	0.64
1136	0.781	0.737	0.789	0.76
1581	0.844	0.918	0.895	0.84
2228	0.875	0.918	0.952	0.89

Table 2 - YoloV7 Evaluation Result

YOLOv7				
Data	Precision	Recall	mAP@0.5	F1-score
552	0.702	0.627	0.698	0.66
1136	0.774	0.743	0.812	0.76
1581	0.851	0.838	0.9	0.84
2228	0.974	0.99	0.93	0.98

The evaluation outcomes of both the YOLOv7 and YOLOv5 models, each based on a dataset of 2228 instances, exhibit notable levels of satisfaction. Specifically, for YOLOv5, the precision, recall, mAP@0.5, and F1-score stand at 0.875, 0.918, 0.952, and 0.89 respectively. In contrast, YOLOv7 yields precision, recall, mAP@0.5, and F1-score values of 0.974, 0.99, 0.93, and 0.98. Notably, YOLOv5 demonstrates superior performance in terms of mAP@0.5, suggesting its heightened ability in object detection. Meanwhile, YOLOv7 excels in precision, recall, and F1-score, reflecting better accuracy, sensitivity, and overall model efficacy in object detection. For precision and mAP@0.5 visuals, see Figures 8-11.

The differences in performance between YOLOv5 and YOLOv7 have implications for practical applications. YOLOv5 excels in rapid and accurate object detection, suitable for autonomous vehicles, emphasizing speed. Meanwhile, YOLOv7's superior precision, recall, and F1-score make it ideal for detailed object detection in applications like medical imaging, prioritizing accuracy. Choosing between them involves trade-offs in speed and precision, requiring consideration of factors such as computational resources and real-time processing needs. The decision should align with the specific requirements and constraints of the intended application.

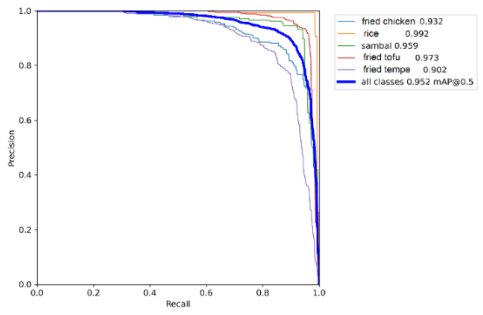


Fig. 4. mAP@0.5 YOLOv5

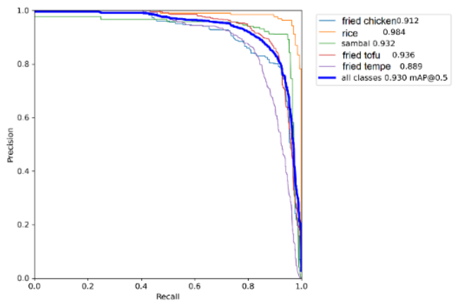


Fig. 5. mAP@0.5 YOLOv7

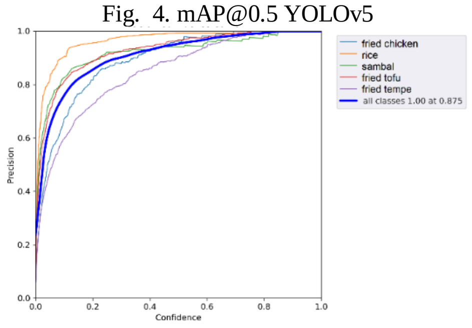


Fig. 6. Precision of YOLOv5

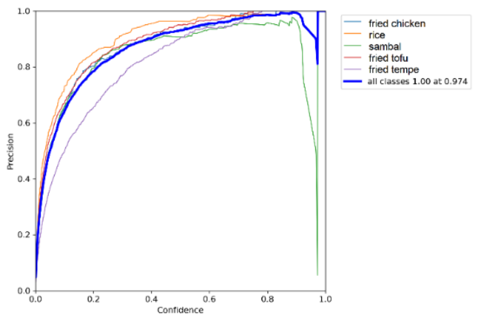
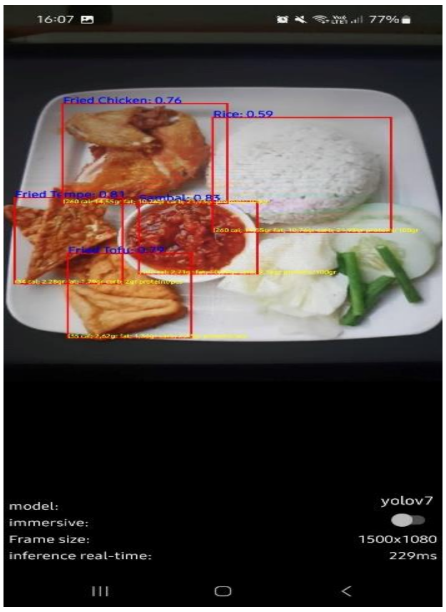


Fig. 7. Precision of YOLOv7

After completing the model training, the YOLOv7 model was chosen for integration into mobile devices. The reason behind selecting the YOLOv7 model for integration into mobile devices, specifically Android devices, stems from the authors' aim to demonstrate a new application creation approach using quantization methods in YOLOv7, as outlined in the proposed method. In addition, the decision was influenced by the limited availability of research addressing the quantization of YOLOv7 models specifically for smartphone utilization. This selection is in line with the goal of contributing to existing knowledge by providing insights and practical guidance regarding the quantization process for YOLOv7, which is specifically designed for effective implementation on mobile platforms. The YOLOv7 PyTorch model was transformed and quantized for successful implementation, resulting in a TFLite fp-16 (floating point 16) quantized model using the proposed method. The quantized model was 14.3 KB in size, compared to the initial model's size of 28.6 KB. The quantized model was then integrated into a real-time object detection application written in the Java programming language and developed in Android Studio. This application offers real-time object detection capabilities coupled with the provision of nutritional information for the detected objects. Application evaluation encompassed assessing real-time interface speed and conducting tests on various objects to determine recall precision and F1-score levels. The real-time object detection application demonstrated the range of interface time of 225 - 232ms. The test outcomes, derived from 43 multi-object images, revealed precision results of 0.923, recall of 0.9, and an F1-score of 0.911. Figure 12 shows the real-time multi-object detection application developed as part of this research.

Figure 12 provides a compelling visual representation of our multi-object detection system's capabilities. Notably, it demonstrates the system's proficiency in detecting and classifying objects across five distinct categories while efficiently delivering pertinent food-related information. Impressively, this process transpires with remarkable efficiency, as evidenced by an interface response time of 229 milliseconds. Furthermore, it is noteworthy that the system exhibits discerning acumen by refraining from detecting objects that do not align with the model's training dataset, thus underscoring its precision and specificity in object recognition.



model: yolov7
immersive:
Frame size: 1500x1080
inference real-time: 229ms

Fig. 8. Real-Time Multi-Object Detection application

Discussions

The outcomes of our study highlight the superior performance of the YOLOv7 model trained for food detection, surpassing previous research efforts. Specifically, our model achieved an impressive mAP@0.5 score of 93%, outperforming the YOLOv7 model tailored for kiwi detection with a lower score of 62.1%(Gallo et al., 2023). Moreover, our study excelled in comparison to others utilizing YOLOv7 for small object detection, achieving a remarkable mAP of 81.1%(K. Li et al., 2023). Additionally, a comparative analysis involving the YOLOv7x model, with 100 and 200 epochs, revealed a 2% improvement in mAP scores, culminating in a final score of 86.3% (Dewi et al., 2023).

Furthermore, our proposed smartphone application for food detection demonstrated clear superiority over existing applications. It boasted an outstanding F1-score of 0.911, surpassing an application designed for road damage detection using deep learning, which achieved a lower F1 score of 0.62(Alfarrarjeh et al., 2018). Additionally, our application outperformed a study creating an application for acne detection using the Faster R-CNN model, where the precision reached 0.54(Huynh et al., 2022) Notably, our proposed application showcased significantly higher precision, reaching an impressive value of 0.923. Furthermore, in another study that implemented YOLOv4 for eye tracking on smartphones, our application achieved a higher F1 score of 0.85(Kumari et al., 2021). These results collectively emphasize the superior performance and efficacy of our study in comparison to existing research endeavor

5. Conclusion

The experimental analysis revealed that datasets with 2228 samples significantly outperformed those with fewer samples across both the YOLOv5 and YOLOv7 models, emphasizing the substantial impact of data volume and augmentation on detection outcomes. The modeling phase involved employing the YOLOv5 and YOLOv7 object detection algorithms with fixed batch parameters 16, learning rate of 0.01, adam optimizer and epoch 100. The analysis of the 2228 data set highlighted YOLOv7's superior precision, recall, and F1-score compared to YOLOv5. This finding highlights the heightened accuracy of YOLOv7 in the task of object detection, particularly its ability to effectively identify positive objects within images. Our proposed methodology successfully facilitated the conversion and quantization of the YOLOv7 model into the TFLite fp-16 format. This achievement marked a significant milestone, resulting in the development of a real-time multi-object detection mobile application. This quantization process led to a notable reduction in the model's size, from 28.6 KB to a compact 14.3 KB. Following this, the application development process was seamlessly executed using the YOLOv7-fp16 quantized model and Android Studio tools. The developed application showcased an average real-time interface response time of 235ms. During application testing, the precision, recall, and F1-score achieved values of 0.923, 0.9, and 0.911, respectively. Beyond these outcomes, the study's implication lies in the successful integration of advanced object detection algorithms into real-time mobile applications, paving the way for enhanced user experiences and practical implementations across diverse scenarios.

References

Akhtar, M. J., Mahum, R., Butt, F. S., Amin, R., El-Sherbeeny, A. M., Lee, S. M., & Shaikh, S. (2022). A Robust Framework for Object Detection in a Traffic Surveillance System. *Electronics*, *11*(21), 3425. https://doi.org/10.3390/electronics11213425

Alfarrarjeh, A., Trivedi, D., Kim, S. H., & Shahabi, C. (2018). A Deep Learning Approach for Road Damage Detection from Smartphone Images. *2018 IEEE International Conference on Big Data (Big Data)*, 5201–5204. https://doi.org/10.1109/BigData.2018.8621899

Almansouri, M., Verkerk, R., Fogliano, V., & Luning, P. A. (2021). Exploration of heritage food concept. *Trends in Food Science & Technology*, *111*, 790–797. https://doi.org/10.1016/j.tifs.2021.01.013

Arunmozhi, A., & Park, J. (2018). Comparison of HOG, LBP and Haar-Like Features for On-Road Vehicle Detection. *2018 IEEE International Conference on Electro/Information Technology (EIT)*, 0362–0367. https://doi.org/10.1109/EIT.2018.8500159

Batal, M., Chan, H. M., Fediuk, K., Ing, A., Berti, P., Sadik, T., & Johnson-Down, L. (2021). Importance of the traditional food systems for First Nations adults living on reserves in Canada. *Canadian Journal of Public Health*, *112*(S1), 20–28. https://doi.org/10.17269/s41997-020-00353-y

Bhatt, D., Patel, C., Talsania, H., Patel, J., Vaghela, R., Pandya, S., Modi, K., & Ghayvat, H. (2021). CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope. *Electronics*, *10*(20), 2470. https://doi.org/10.3390/electronics10202470

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal Speed and Accuracy of Object Detection*.

Cao, C., Wang, B., Zhang, W., Zeng, X., Yan, X., Feng, Z., Liu, Y., & Wu, Z. (2019). An Improved Faster R-CNN for Small Object Detection. *IEEE Access*, *7*, 106838–106846. https://doi.org/10.1109/ACCESS.2019.2932731

Chen, H.-C., Widodo, A. M., Wisnujati, A., Rahaman, M., Lin, J. C.-W., Chen, L., & Weng, C.-E. (2022). AlexNet Convolutional Neural Network for Disease Detection and Classification of Tomato Leaf. *Electronics*, *11*(6), 951. https://doi.org/10.3390/electronics11060951

Chen, J., Liu, H., Zhang, Y., Zhang, D., Ouyang, H., & Chen, X. (2022). A Multiscale Lightweight and Efficient Model Based on YOLOv7: Applied to Citrus Orchard. *Plants*, *11*(23), 3260. https://doi.org/10.3390/plants11233260

Chen, J.-W., Lin, W.-J., Cheng, H.-J., Hung, C.-L., Lin, C.-Y., & Chen, S.-P. (2021). A Smartphone-Based Application for Scale Pest Detection Using Multiple-Object Detection Methods. *Electronics*, *10*(4), 372. https://doi.org/10.3390/electronics10040372

Dhillon, A., & Verma, G. K. (2020). Convolutional neural network: a review of models, methodologies and applications to object detection. *Progress in Artificial Intelligence*, *9*(2), 85–112. https://doi.org/10.1007/s13748-019-00203-0

Fakhrou, A., Kunhoth, J., & Al Maadeed, S. (2021). Smartphone-based food recognition system using multiple deep CNN models. *Multimedia Tools and Applications*, *80*(21–23), 33011–33032. https://doi.org/10.1007/s11042-021-11329-6

Fukase, E., & Martin, W. (2020). Economic growth, convergence, and world food demand and supply. *World Development*, *132*, 104954. https://doi.org/10.1016/j.worlddev.2020.104954

Gallo, I., Rehman, A. U., Dehkordi, R. H., Landro, N., La Grassa, R., & Boschetti, M. (2023). Deep Object Detection of Crop Weeds: Performance of YOLOv7 on a Real Case Dataset from UAV Images. *Remote Sensing*, *15*(2), 539. https://doi.org/10.3390/rs15020539

Gao, M., Qi, D., Mu, H., & Chen, J. (2021). A Transfer Residual Neural Network Based on ResNet-34 for Detection of Wood Knot Defects. *Forests*, *12*(2), 212. https://doi.org/10.3390/f12020212

Huynh, Q. T., Nguyen, P. H., Le, H. X., Ngo, L. T., Trinh, N.-T., Tran, M. T.-T., Nguyen, H. T., Vu, N. T., Nguyen, A. T., Suda, K., Tsuji, K., Ishii, T., Ngo, T. X., & Ngo, H. T. (2022). Automatic Acne Object Detection and Acne Severity Grading Using Smartphone Images and Artificial Intelligence. *Diagnostics*, *12*(8), 1879. https://doi.org/10.3390/diagnostics12081879

Imambi, S., Prakash, K. B., & Kanagachidambaresan, G. R. (2021). *PyTorch* (pp. 87–104). https://doi.org/10.1007/978-3-030-57077-4_10

Jeong, D. (2020). Road Damage Detection Using YOLO with Smartphone Images. *2020 IEEE International Conference on Big Data (Big Data)*, 5559–5562. https://doi.org/10.1109/BigData50022.2020.9377847

Kagaya, H., Aizawa, K., & Ogawa, M. (2014). Food Detection and Recognition Using Convolutional Neural Network. *Proceedings of the 22nd ACM International Conference on Multimedia*, 1085–1088. https://doi.org/10.1145/2647868.2654970

Kawano, Y., & Yanai, K. (2015). FoodCam: A real-time food recognition system on a smartphone. *Multimedia Tools and Applications*, *74*(14), 5263–5287. https://doi.org/10.1007/s11042-014-2000-8

Kumar, A., Zhang, Z. J., & Lyu, H. (2020). Object detection in real time based on improved single shot multi-box detector algorithm. *EURASIP Journal on Wireless Communications and Networking*, *2020*(1), 204. https://doi.org/10.1186/s13638-020-01826-x

Kumari, N., Ruf, V., Mukhametov, S., Schmidt, A., Kuhn, J., & Küchemann, S. (2021). Mobile Eye-Tracking Data Analysis Using Object Detection via YOLO v4. *Sensors*, *21*(22), 7668. https://doi.org/10.3390/s21227668

Li, K., Wang, Y., & Hu, Z. (2023). Improved YOLOv7 for Small Object Detection Algorithm Based on Attention and Dynamic Convolution. *Applied Sciences*, *13*(16), 9316. https://doi.org/10.3390/app13169316

Li, S., Wang, S., & Wang, P. (2023). A Small Object Detection Algorithm for Traffic Signs Based on Improved YOLOv7. *Sensors*, *23*(16), 7145. https://doi.org/10.3390/s23167145

Lu, S., Wang, B., Wang, H., Chen, L., Linjian, M., & Zhang, X. (2019). A real-time object detection algorithm for video. *Computers & Electrical Engineering*, *77*, 398–408. https://doi.org/10.1016/j.compeleceng.2019.05.009

Mansour, M. Y. M. A., D. Dambul, K., & Choo, K. Y. (2022). Object Detection Algorithms for Ripeness Classification of Oil Palm Fresh Fruit Bunch. *International Journal of Technology*, *13*(6), 1326. https://doi.org/10.14716/ijtech.v13i6.5932

Martinez-Alpiste, I., Golcarenarenji, G., Wang, Q., & Alcaraz-Calero, J. M. (2022). Smartphone-based real-time object recognition architecture for portable and constrained systems. *Journal of Real-Time Image Processing*, *19*(1), 103–115. https://doi.org/10.1007/s11554-021-01164-1

Nepal, U., & Eslamiat, H. (2022). Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs. *Sensors*, *22*(2), 464. https://doi.org/10.3390/s22020464

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)*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6517–6525. https://doi.org/10.1109/CVPR.2017.690

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*.

Sharma, K. U., & Thakur, N. V. (2017). A review and an approach for object detection in images. *International Journal of Computational Vision and Robotics*, *7*(1/2), 196. https://doi.org/10.1504/IJCVR.2017.10001813

Shen, Z., Liu, Z., Li, J., Jiang, Y.-G., Chen, Y., & Xue, X. (2020). Object Detection from Scratch with Deep Supervision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(2), 398–412. https://doi.org/10.1109/TPAMI.2019.2922181

Sitaula, C., & Hossain, M. B. (2021). Attention-based VGG-16 model for COVID-19 chest X-ray image classification. *Applied Intelligence*, *51*(5), 2850–2863. https://doi.org/10.1007/s10489-020-02055-x

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*.

Wang, Y., Jia, X., Zhou, M., Xie, L., & Tian, Z. (2019). A novel F-RCNN based hand gesture detection approach for FMCW systems. *Wireless Networks*. https://doi.org/10.1007/s11276-019-02096-2

Wei, G., Li, G., Zhao, J., & He, A. (2019). Development of a LeNet-5 Gas Identification CNN Structure for Electronic Noses. *Sensors*, *19*(1), 217. https://doi.org/10.3390/s19010217

Wu, D., Jiang, S., Zhao, E., Liu, Y., Zhu, H., Wang, W., & Wang, R. (2022). Detection of Camellia oleifera Fruit in Complex Scenes by Using YOLOv7 and Data Augmentation. *Applied Sciences*, *12*(22), 11318. https://doi.org/10.3390/app122211318

Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J., & Hua, X. (2019). Quantization Networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7300–7308. https://doi.org/10.1109/CVPR.2019.00748

Yao, J., Qi, J., Zhang, J., Shao, H., Yang, J., & Li, X. (2021). A Real-Time Detection Algorithm for Kiwifruit Defects Based on YOLOv5. *Electronics*, *10*(14), 1711. https://doi.org/10.3390/electronics10141711

Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M., & Lee, B. (2022). A survey of modern deep learning based object detection models. *Digital Signal Processing*, *126*, 103514. https://doi.org/10.1016/j.dsp.2022.103514

Zhang, H., Liu, L. Z., Xie, H., Jiang, Y., Zhou, J., & Wang, Y. (2022). Deep Learning-Based Robot Vision: High-End Tools for Smart Manufacturing. *IEEE Instrumentation & Measurement Magazine*, *25*(2), 27–35. https://doi.org/10.1109/MIM.2022.9756392

Zhao, J., Zhang, X., Yan, J., Qiu, X., Yao, X., Tian, Y., Zhu, Y., & Cao, W. (2021). A Wheat Spike Detection Method in UAV Images Based on Improved YOLOv5. *Remote Sensing*, *13*(16), 3095. https://doi.org/10.3390/rs13163095