

Tomato ripeness classification with weight-based and texture-adaptive pooling in CNNs

Rut Yatigul¹⁾, Wisanu Jitviriy^{a*1)}, Eiji Hayashi²⁾, Aran Blattler¹⁾, Watcharin Tangsuksant¹⁾ and Teppakorn Sittiwanchai¹⁾

¹⁾Department of Production and Robotics Engineering, King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand

²⁾Department of Creative Informatics, Kyushu Institute of Technology, Fukuoka, 820-8502, Japan

Received 17 February 2026

Revised 30 June 2026

Accepted 4 August 2026

Abstract

This research introduces a weight-based mixed pooling and adaptive fuzzy pooling mechanism based on feature map texture analysis for attention mechanism Squeeze-and-Excitation Networks (SE) and Convolutional Neural Networks (CNNs) in tomato ripeness classification. Traditional CNNs and SE networks rely on deterministic pooling strategies that can be suboptimal for capturing diverse visual features such as texture and color gradients in fruit. A novel fuzzy pooling system is proposed that dynamically adjusts the fusion ratio of min and max pooling based on real-time texture analysis of feature maps via a Laplacian filter. This mechanism was integrated into Darknet53, ResNet34, and ResNet101 architectures, both directly and within the SE attention block. The method was evaluated on a five-class tomato ripeness dataset using 10-fold cross-validation against traditional and fixed-weight pooling methods. The fuzzy and mixed pooling approach demonstrated superior performance on certain models which include ESEnet: Darknet53 + SE (Fuzzy pool) achieved an average F1 score across folds of 0.8429, outperforming Darknet53 + baseline SE (Average pool) at 0.8350; Extended Net: ResNet34 + Fuzzy pool achieved an average F1 score across folds of 0.8305, outperforming baseline ResNet34 at 0.8130 and ResNet101 + Fuzzy pool achieved an average F1 score across folds of 0.8284, outperforming baseline ResNet101 at 0.8267. This improvement was realized with minimal increase in computational overhead. Specifically, measurements of model parameters (M) and FLOPs (GMACs) showed only a slight increase across all architectures after integrating SEnet into the system, and zero increase after integrating the Laplacian filter and fuzzy logic. The study concludes that logic-based adaptive pooling is a robust strategy that improves feature representation in CNNs, achieving a balance between classification accuracy and computational cost. These findings offer practical value for developing more reliable automated harvesting systems and provide a generalizable enhancement for a wide range of computer vision classification tasks.

Keywords: Image classification, Attention mechanism, Fuzzy Pooling, Mixed Pooling, Laplacian filter

1. Introduction

As one of the most important and popular crops worldwide, tomatoes play a crucial role in agriculture. Due to their rich nutritional value and appealing taste, tomatoes are consumed by millions of people worldwide. As a result, they serve as a significant resource in both the agricultural and food industries [1]. Consequently, they have been intensively researched in many aspects to improve industrial production; for instance, Jeeravipoolvarn et al. [2] investigated the effect of different plant watering processes and ratios on tomato seed production capability and quality.

In commercial production, tomatoes are usually harvested during the mature ripening stage. This is because they have sufficient firmness, an extended shelf life, and the potential to turn red after being removed from the plant [3]. Consumer preferences, especially for color and texture [4] along with the need to reduce potential damage during transportation and other supply chain processes, are the main factors influencing the decision to harvest tomatoes at this stage. Although there are many studies focusing on classifying the maturity level of tomatoes, [5] calculate the area of the red HSV color range on the detected tomato and compare it with the areas of green and yellow to classify fully ripe tomatoes from unripe ones. They later implemented this algorithm in a tomato harvesting robot. [6] used infrared imaging to recognize tomato fruits and estimated their maturity levels using probability distributions, which were quantified by combining the color values of detected fruits with the expertise of experienced farmers. However, there are still few studies that focus on using CNNs and attention mechanisms with adaptive pooling strategies to classify the maturity level of tomatoes.

Nowadays, there is intensive study on merging technology with agriculture [7, 8], Convolutional Neural Networks (CNNs) have proven to be an accurate and reliable method of technology for computer vision tasks, such as detection and classification. CNNs technology has been applied on various field of research, one of the fields where this technology has been widely applied is agriculture. Chaowalittawin et al. [9] has proposed banana quality classification system using CST-MobileNetV2 a lightweight deep learning model integrated with a microservice architecture and LINE application for user interaction with the system. To overcome the accuracy limitations of shallower 4-layer CNNs, Thipsrirach et al. [10] proposed DuNet-12, an improved architecture that outperformed state-of-the-art models like LeNet-5 and AlexNet in durian maturity classification in certain setting. Like many other industries, where system parameters are optimized to maximize sustainability and reduce physical waste [11] agricultural sector also experiences rapid

*Corresponding author.

Email address: wisanu.j@eng.kmutnb.ac.th

<https://doi.org/10.64960/easr.2026.266223>

growth in digitalization of operation to ensure optimal resource management. In the past ten years, there are many research papers for example [12-14] that deeply investigate the possibility to integrate the concept of attention mechanisms into their own work in order to improve the performance of computer vision tasks like the work of [15] which focus on using attention mechanism to improve accuracy of image classification task while other research use attention mechanism to apply on different application for example object detection [16], image segmentation [17]. Attention mechanisms can be separate into three different types: channel attention [18], spatial attention [19], and finally mixed attention [20, 21]. Even though there is a significant amount of research focusing on CNN architecture and attention mechanisms, there is still a limited number of studies addressing adaptive pooling strategies that can automatically adjust its reliance weight on two opposite pooling strategies and evaluate its result on both CNNs and attention mechanisms. As mentioned by [18] in their Squeeze-and-Excitation network (SE) paper, their approach to using average pooling as the main pooling method for feature extraction was due to simplicity, and more sophisticated strategies could be employed.

The primary challenge addressed in this research is the variability of image feature map texture levels. Tomato ripeness classification datasets contain diverse texture characteristics. For example, even within the same "100% ripe" class, certain images may exhibit highly dense textures where the fully red tomatoes are occluded by branches and leaves, as seen in Figure 1(a) and 1(b). Conversely, other images may feature low-density textures with fully visible red tomatoes; without any occlusion, these appear simply as red circular surfaces, as shown in Figure 1(c) and 1(d).

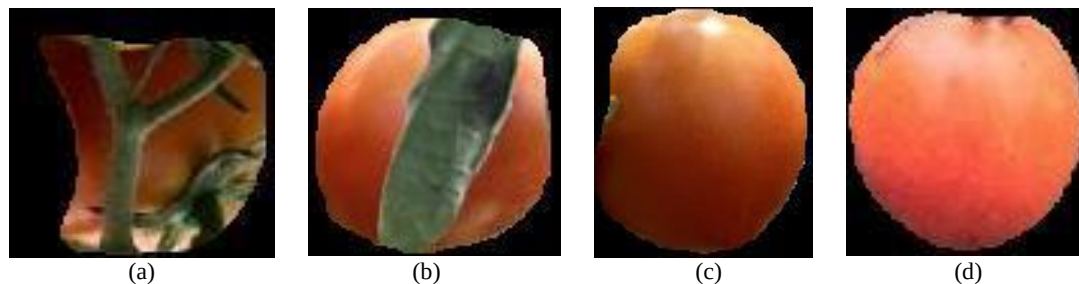


Figure 1 Images from the 100% ripe class of the benchmark dataset used in this study. (a, b) Highly dense texture images showing fully ripe tomatoes occluded by stems and leaves. (c, d) Low-density texture images showing completely visible, fully ripe tomatoes.

Several studies have noted that feature extraction methods—specifically the pooling mechanisms used in Convolutional Neural Networks (CNNs)—can heavily impact model performance when applied to different types of image texture characteristics. For example, Peng et al. [22] state that max pooling discards weaker signals within the pooling region, potentially ignoring important local details. Meanwhile, average pooling calculates the mean of the pooling region, which can result in the down-weighting of strong activations. Similarly, Jie and Wanda [23] note that max pooling loses distinguishing details in high-activation regions by discarding non-maximum values, while average pooling dilutes feature maps when processing zero-heavy inputs. Furthermore, Saeedan et al. [24] explain that max pooling disrupts the flow of gradients in some branches during the backward pass since only one node is selected in every input neighborhood.

Collectively, these studies highlight the limitations of relying on a single pooling mechanism in traditional CNNs. This is particularly problematic for datasets that contain a wide variety of feature map characteristics, such as the mix of highly dense and low-density textures found in the tomato ripeness classification dataset used in this research. Choosing only one type of feature extraction method—as is standard in traditional CNNs that rely exclusively on either max or average pooling—can degrade model accuracy and prevent the architecture from reaching its maximum potential.

The main concept and contribution of proposed method involves creating a pooling method for CNNs and attention mechanisms by combining max pooling and min pooling (an inverse of max pooling) together using two different methods. The first is mixed pooling [25], where a weight function with a fixed weight ratio was used to balance the distribution of max pooling and min pooling; for this research, the ratio is set at 0.5, which equally distributes between max and min pooling. Another method is fuzzy pooling, which includes a logic-based adaptive weight adjustment system to the existing mixed pooling. The logic behind fuzzy pooling is a texture variance analysis using a Laplacian filter, which is a filter used for edge detection, image smoothing, texture enhancement, and low-light image enhancement [26]. In this work, it analyzes the texture density level of each passing feature map. This information is later fed to fuzzy logic, which determines the texture density of the processing feature map and then assigns a proper weight ratio to the mixed pooling for performance improvement in feature extraction.

2. Methods

2.1 Extended Squeeze and Excitation Network (ESEnet)

Living creature whether human or animal that can perceive their surrounding with vision generally do not process an entire scene all at once when visualizing something; instead, they tend to focus on specific parts. This concept is similar to how attention mechanisms—such as spatial or channel-based attention—operate, focusing only on important pixels and disregarding irrelevant ones without processing all pixels. In the original research paper, the main purpose of the Squeeze-and-Excitation Network was to improve the classification accuracy of CNNs in large-scale image classification problems due to its ability to model contextual information [18].

For multi-class classification of tomato ripening stages, the Darknet53, ResNet34 and ResNet101 model were investigated, with the Squeeze-and-Excitation Network integrated into it to improve classification accuracy. The Squeeze-and-Excitation Network was introduced by [18], which focuses on combining spatial and channel-based information while giving less attention weight to an irrelevant feature. The operation of the Squeeze-and-Excitation Network can be separated into two stages. Squeeze: This stage uses global average pooling to compress the spatial dimensions of each feature map into a single value per channel. It captures global context and summarizes how important each channel is. Excitation: The squeezed information is passed through a small neural network (with two fully connected layers) to generate channel-wise weights. These weights are then used to scale (recalibrate) the original feature maps which help amplify an important channels and suppressing less useful ones. ResNet34 and ResNet101 were selected because the

original Squeeze-and-Excitation (SE) Network research paper [18] primarily utilized the ResNet architecture as a baseline. The authors of that study integrated their proposed SE block into the architecture to introduce several new models, such as SE-ResNet-50, SE-ResNeXt-50, and SE-ResNet-101. Therefore, this paper aims to address an area for improvement identified in the original study, where the authors explicitly stated that feature extraction methods warranted further investigation. Furthermore, Darknet53 is the backbone architecture of YOLO—a highly popular model for object detection, classification, and segmentation. Integrating the proposed system into Darknet53 could potentially lead to an improved sub-branch of the YOLO model in future research.

Besides the standard Squeeze-and-Excitation Network, this work also introduces an improved Extended Squeeze-and-Excitation Network (ESEnet) by changing its squeeze operation from average pooling to mixed pooling and fuzzy pooling, based on image texture variance level analysis. This is under the assumption that in high texture intensity areas of the feature map, max pooling is more suitable as a texture extractor due to its ability to extract sharp edges and high-contrast texture areas, while min pooling is used to balance out the max pooling in areas with smooth transitions in texture intensity. This will be controlled by a weighting function, which uses a fixed ratio for mixed pooling and an adaptive ratio for fuzzy pooling.

2.2 Mixed Pooling

In general CNN applications, max pooling and average pooling are two of the most used pooling operations because of their computational efficiency. However, the authors of the original paper of Squeeze-and-Excitation Network (SE) [18] clearly mention that, based on their experimental result on the ImageNet dataset [27], average pooling slightly outperforms max pooling in terms of error rate—22.28% compared to 22.57% for top-1 error, and 6.03% compared to 6.09% for top-5 error. In which they conclude that according to this result, average pooling is selected as the standard pooling operation for the Squeeze-and-Excitation Network.

$$u = \lambda f_{\max}(x) + (1 - \lambda) f_{\min}(x), \quad (1)$$

For mixed pooling, this work introduces a weighting function as expressed in Eq. (1) to combine max and min pooling, using λ as a weighting parameter to control the contribution of each. Notably, min pooling was chosen instead of average pooling to pair with max pooling. This decision is made to increase the contrast between the feature extraction behaviors of max and min pooling, which would be less pronounced if average pooling were used instead. For the weight in mixed pooling, 0.5 was selected as the standard weighting factor, combining max and min pooling in equal proportions. This choice is made to investigate the effect of an equal combination of min and max pooling on the performance of the Extended Squeeze and Excitation Network.

2.3 Fuzzy Pooling

Fuzzy Pooling for the Squeeze-and-Excitation Network is a novel method introduced for the first time in this paper. It shares the same formula structure as Mixed Pooling which can be observed in Eq. (1). However, Fuzzy Pooling introduces logic-based weight selection based on feature map texture analysis using a Laplacian filter.

2.3.1 Laplacian filter

Laplacian filter is one of the classical simple filters. It is a high pass filter from signal point of view [26]. Laplacian filters will highlight regions of rapid intensity change in an image and are partially good at finding fine details in an image [28]. The Laplacian filter $L(x, y)$ of an image with pixel intensity value $I(x, y)$ is given by Eq. (2), and its 3×3 convolution kernel is shown in (3):

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}, \quad (2)$$

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -8 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad (3)$$

This work used a Laplacian filter to measure the texture density of each feature map during the forward pass. The extracted information is then passed to a fuzzy logic system, which assigns an appropriate pooling weight ratio.

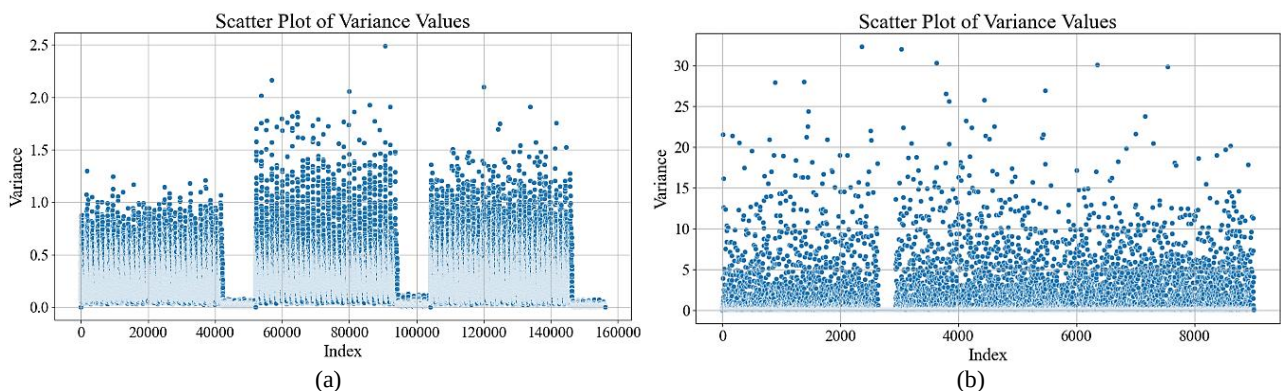


Figure 2 Pixel intensity variance of each feature map from (a) ResNet34 with ESEnet, (b) Extended ResNet34, (c) Darknet53 with ESEnet, (d) Extended Darknet53, (e) ResNet101 with ESEnet, (f) Extended ResNet101

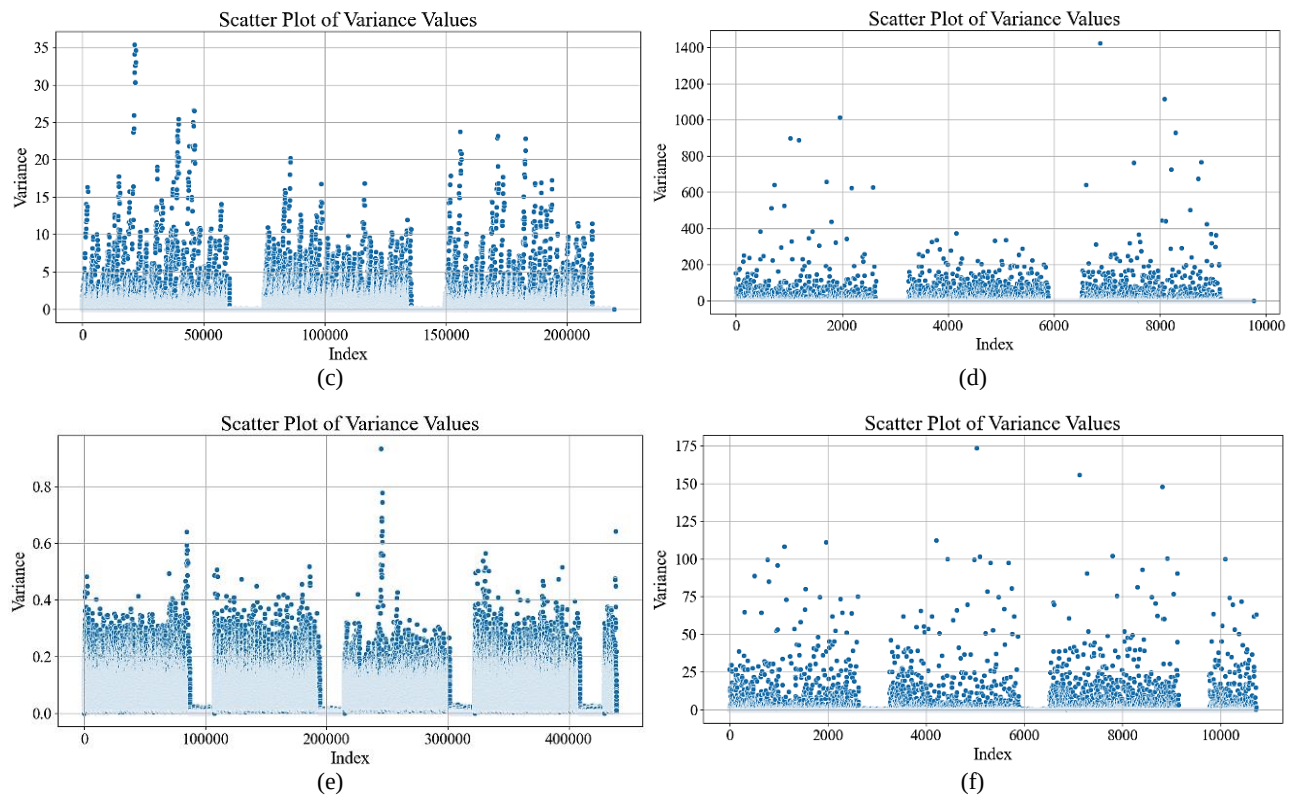


Figure 2 (continued) Pixel intensity variance of each feature map from (a) ResNet34 with ESEnet, (b) Extended ResNet34, (c) Darknet53 with ESEnet, (d) Extended Darknet53, (e) ResNet101 with ESEnet, (f) Extended ResNet101

From Figure 2, the operation of the Laplacian filter can be observed as it analyzes the pixel intensity of each feature map in the tomato ripeness dataset. This information is crucial for the operation of the fuzzy logic system. Figure 2 (a) presents the pixel intensity variance of ResNet34 with ESEnet. The data shown in Figure 2 (a) originates from the fuzzy pooling block operating within the ESEnet. One interesting observation is that the pixel intensity variance captured from the fuzzy pooling block directly integrated into the architecture shown in Figure 2 (b) appears more widely distributed and exhibits a broader range of values compared to the fuzzy pooling block embedded within the ESEnet. This may occur because feature maps that pass through many convolutional layers tend to develop more diverse characteristics, unlike the fuzzy pooling blocks that are directly connected to each convolutional layer in the ESEnet. The Laplacian filter is also applied to Darknet53, both in the Extended Darknet53 (Darknet53 with fuzzy pooling) and in Darknet53 with ESEnet, to measure pixel intensity in the feature maps and provide input to the fuzzy logic system as a weight function ratio λ . Figure 2 (c) represents the pixel intensity variance of feature maps that pass through Darknet53 with ESEnet. When compared to its ResNet34 counterpart, the range of pixel intensity variance appears to be wider. This indicates that, even when using the same dataset, different CNN architectures can produce feature maps with distinct characteristics. As seen in Figure 2 (d), the pixel intensity variance range of the Extended Darknet53 is wider compared to Darknet53 ESEnet. This behavior is consistent with its ResNet34 counterpart, reinforcing the theory that the placement of the adaptive pooling block influences the characteristics of the feature maps it observes. Figure 2 also shows the pixel intensity variance range of (e) ESE ResNet101 and (f) Extended ResNet101, which exhibit the same effect as the previous architectures—a wider variance range when fuzzy pooling replaces the original average pooling in the main ResNet101 architecture. The use of this variance information within the fuzzy logic system will be discussed in the next section.

The Laplacian filter plays a crucial role in this research, as it serves as the indicator for the fuzzy logic system to determine whether to give more weight to max pooling or min pooling. This assumes that in regions of the feature map with low texture intensity, priority should be given to min pooling, while in areas with rich texture intensity, max pooling should be prioritized as the main pooling operation. This decision-making process is made possible through texture density analysis using the Laplacian filter.

2.3.2 Fuzzy logic

Fuzzy logic is an advanced control strategy designed to handle highly nonlinear systems [29]. The term "fuzzy" means to the inherent ambiguity or uncertainty present in many real-world situations, where conditions cannot be strictly classified as true or false. Instead, fuzzy logic offers a range of values between true and false, allowing for greater flexibility in identifying the most appropriate solution to a given problem. In this work, fuzzy logic is introduced to adaptively calibrate the pooling weight function. The input to the fuzzy logic system is the texture level of the feature map, represented as texture variance. A higher variance indicates that the feature map contains sharp edges or low-level features. In such cases, fuzzy logic prioritizes max pooling, based on the assumption that max pooling is more effective at capturing sharp edges and areas with strong texture on plain backgrounds [30]. Conversely, in low-variance cases, fuzzy logic prioritizes min pooling, which is an inverse version of max pooling. Formulas that used to describe system dynamics of fuzzy logic is described as follows

$$\mu_F(x; a, b, c) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{b-a}, & a \leq x \leq b \\ \frac{c-x}{c-b}, & b \leq x \leq c \\ 0, & c \leq x \end{cases}, \quad (4)$$

Eq. (4) represents the general formula for the triangular fuzzy membership function. The symbol represents the membership function, which indicates how much a crisp input (in this case, variance or lambda) belongs to a fuzzy membership function such as low, middle or high. The parameter a and c set the left and right feet or base point of triangle while parameter b set the location of the triangle peak.

$$\mu_{low}(v) = \mu_{tri}(v; a_{low}, b_{low}, c_{low}), \quad (5)$$

$$\mu_{middle}(v) = \mu_{tri}(v; a_{middle}, b_{middle}, c_{middle}), \quad (6)$$

$$\mu_{high}(v) = \mu_{tri}(v; a_{high}, b_{high}, c_{high}), \quad (7)$$

Eqs. (5), (6) and (7) represent the variance membership functions. In this work, the variance level of the feature map was defined as low, middle, and high. This work measures the variance by collecting data from the Laplacian filter attached to the ESE block and main pooling layer, which is connected to each residual block of Darknet53, ResNet34, ResNet101 and Extended Darknet53, Extended ResNet34, Extended ResNet101 during the training process then analyze the data distribution, remove outliers, and define the upper and lower boundaries of the clustered variance values. These boundaries are used to assign the fuzzy membership functions for variance. Note that if the dataset or CNN architecture changes, these membership functions must be readjusted to suit the new environment.

$$\mu_{\lambda, low}(\lambda) = \mu_{tri}(\lambda; a_{low}, b_{low}, c_{low}), \quad (8)$$

$$\mu_{\lambda, middle}(\lambda) = \mu_{tri}(\lambda; a_{middle}, b_{middle}, c_{middle}), \quad (9)$$

$$\mu_{\lambda, high}(\lambda) = \mu_{tri}(\lambda; a_{high}, b_{high}, c_{high}) \quad (10)$$

Eqs. (8), (9) and (10) represent the membership functions of the lambda value, which will later be used to tune the weight function of the real-time adaptive pooling operation. In this case, since lambda serves as a weight function descriptor, its range is limited to between 0 and 1.

$$\lambda_{low}(\lambda) = \min(\mu_{v, low}(v), \mu_{\lambda, low}(\lambda)), \quad (11)$$

$$\lambda_{middle}(\lambda) = \min(\mu_{v, middle}(v), \mu_{\lambda, middle}(\lambda)), \quad (12)$$

$$\lambda_{high}(\lambda) = \min(\mu_{v, high}(v), \mu_{\lambda, high}(\lambda)), \quad (13)$$

Eqs. (11), (12) and (13) present the activation functions. These formulas follow the structure of the Scikit-Fuzzy library, which is the fuzzy logic framework used in this work. Specifically, the three activation functions $\lambda_{low}(\lambda)$, $\lambda_{middle}(\lambda)$, $\lambda_{high}(\lambda)$ are used to determine the degree to which each fuzzy rule contributes to the final output. These functions play a key role in modulating the weight ratio of the adaptive pooling mechanism based on the inferred importance of the feature map regions. The defuzzification formula is shown in Eq. (14) which is calculated based on the center of gravity method.

$$\lambda^* = \frac{\int \mu_a(\lambda) \cdot \lambda \, d\lambda}{\int \mu_a(\lambda) \, d\lambda}, \quad (14)$$

Figure 3 (a) shows the fuzzy membership function of ResNet34 with ESEnet. The Mamdani fuzzy logic system with triangular membership functions is the type of fuzzy logic used throughout this work. The input membership function in Figure 3 (a) is defined based on pixel intensity variance in the feature map, which, for ResNet34 with ESEnet, can be observed in Figure 2 (a). As observed in Figures 2(a) through 2(f), several outlier data points lie far outside the main distribution; these points were subsequently discarded. To define an appropriate cutoff value for the fixed fuzzy membership function, extensive empirical testing was conducted to determine the value that yields the highest classification accuracy. The fuzzy membership function for the Extended ResNet34 is shown in Figure 3 (b), as previously explained in section Laplacian Filter, due to the placement of the fuzzy pooling at the back of ResNet34, the range of recorded pixel intensity variance is significantly greater than that of ResNet34 with ESEnet. Interestingly, the experimental results show that the Extended ResNet34 demonstrates greater improvements over the original ResNet34 compared to ResNet34 with ESEnet. Figure 3 (c) shows the output lambda weight of both ResNet34, which ranges from zero to one, this membership function is the same across all architectures. The same logic also applies to Darknet53 in Figure 3(f), where the output represents both (d) Darknet53 with ESEnet and (e) Extended Darknet53.

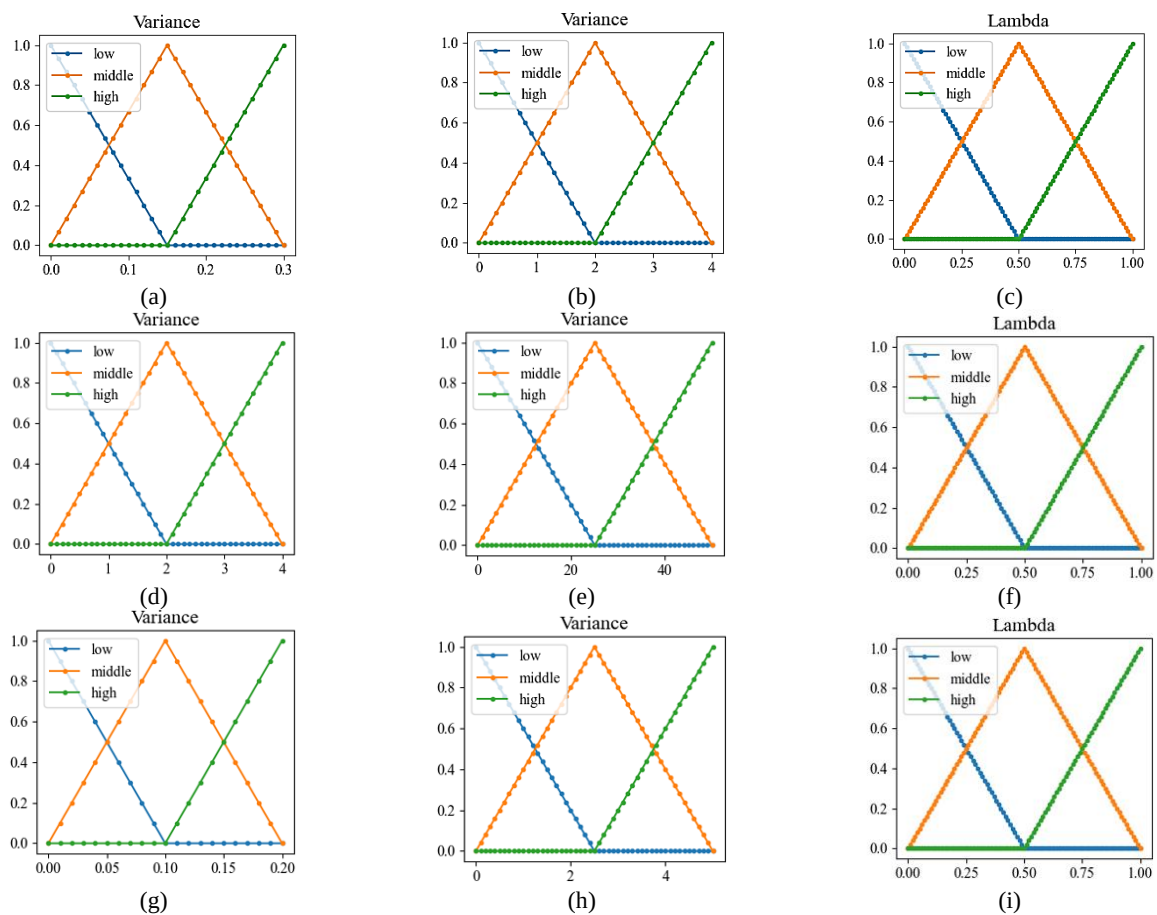


Figure 3 Fuzzy membership function of (a) ResNet34 with ESEnet, (b) Extended ResNet34, (c) both ResNet34 output, (d) Darknet53 with ESEnet, (e) Extended Darknet53, (f) both Darknet53 output, (g) ResNet101 with ESEnet, (h) Extended ResNet101, (i) both ResNet101 output

As can be seen from Figure 3 (d) and (e), and as mentioned in the previous section, both Darknet53 with ESEnet and Extended Darknet53 share a similar characteristic—an increase in the range of feature map intensity variance between the ESEnet and Extended architectures. A significant increase in feature map intensity variance is observed, similar to what was seen with ResNet34. Additionally, Extended Darknet53 appears to show greater improvement over the original architecture compared to Darknet53 with ESEnet. This may be linked to the influence of adaptive pooling, which can express itself more effectively when directly integrated into the main pooling layer rather than operating solely within the ESEnet. Lastly, the ResNet101 fuzzy membership function can be observed in Figure 3 (g) ResNet101 with ESEnet, (h) Extended ResNet101, and (i) the lambda weight output, which is the same for both (g) and (h). ResNet101 also shows similar characteristics to the two previous architectures, including an increased variance in the Extended ResNet101 and higher model accuracy compared to its ESE counterpart.

2.4 Extended Darknet53, ResNet34 and ResNet101 (Extended Net)

Darknet53 is the main convolutional neural network architecture used in YOLOv3 [31]. It is a continuous improvement over YOLOv2 and its predecessor, Darknet19. The architecture employs successive 3×3 and 1×1 convolutional layers and introduces shortcut connections. Darknet53 is larger than Darknet19, consisting of 53 convolutional layers—hence the name "Darknet53". In this work a new Extended Darknet53 was introduced by replacing its original average pooling layer with the proposed mixed and fuzzy pooling layer. The fuzzy pooling layer has also been re-tuned based on the input pixel intensity variance range, as shown in Figure 2 (d), while the mixed pooling layer has a fixed weight ratio set at 0.5. The only difference from the original architecture is the pooling layer, which has been replaced with fuzzy and mixed pooling, while the rest of the architecture remains unchanged.

ResNet34 and ResNet101 (Residual Network) is a state-of-the-art architecture proposed by [32] from Microsoft Research Asia. The core design concept involves introducing a shortcut mechanism that directly connects the inputs to the outputs. This approach effectively addresses the issue of performance degradation that typically arises as network depth increases. In this research the average pooling layer which originally located between the classification sequence and the fully connected layer has been replaced with mixed and fuzzy pooling. The fuzzy pooling layer has been tuned based on the input pixel intensity variance range, as shown in Figure 2 (b) for ResNet34 and (f) for ResNet101, while the mixed pooling layer uses a fixed weight ratio of 0.5.

The purpose of introducing Extended Darknet53, ResNet34 and ResNet101 into the experiment is to further demonstrate the robustness of adaptive pooling (fuzzy and mixed pooling). In this case, the adaptive pooling operations are applied directly to the main pooling layers of the CNN, rather than being confined within the Squeeze-and-Excitation Network. This allows for a potentially greater impact on the system, as the pooling operations now influence the core structure of the network. As a result, not only the effects of adaptive pooling on attention mechanisms like the Squeeze-and-Excitation Network can be observed, but also its effect on the main CNN architecture.

After this, the results will be examined by comparing Extended Darknet53, ResNet34 and ResNet101 against its original architecture. A deep analysis will be conducted on the results obtained from all architectures to explain the behavior of adaptive pooling when applied to different CNN architectures. Additionally, its behavior when applied to attention mechanisms such as ESEnet will be analyzed and compare it with the original Squeeze-and-Excitation Network.

2.5 Tomato ripeness dataset

In this work, tomato ripeness was selected as the benchmark dataset which was obtained from Roboflow public dataset [33], as this research long-term goal is to develop an improved object detection algorithm for tomato harvesting robots. The ability to accurately identify and harvest fully ripe tomatoes is crucial for performance. (Details regarding the tomato harvesting robots themselves are beyond the scope of this research.)

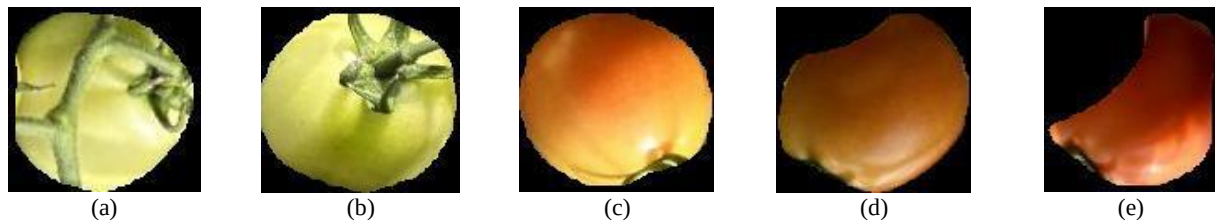


Figure 4 Sample images of each class from the tomato ripeness dataset: (a) 0% Ripe (unripe), (b) 30% Ripe, (c) 50% Ripe, (d) 70% Ripe, (e) 100% Ripe.

The tomato ripeness dataset contains a total of five classes (0%, 30%, 50%, 70%, and 100%), each class shows a different level of ripeness in tomatoes as can be seen from Figure 4. The dataset contains a total of 3,263 images, with 2,935 images used for the training set and 328 images for the test set. The validation set is automatically split from the training set using the K-fold cross-validation method, which will be explained in later sections. The tomato-harvesting competition environment is highly dynamic, requiring the robot's vision system to be fine-tuned to adapt to these conditions. Consequently, the dataset used to train the tomato ripeness classification model must address dynamic scenarios, such as tomatoes occluded by leaves, branches, or other tomatoes, as well as those covered in shadows. This dataset successfully captures all of these variations, with example images shown in Figure 5.

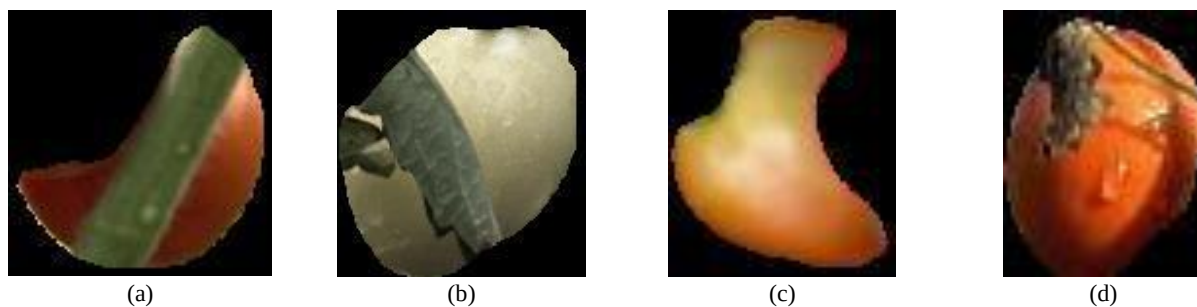


Figure 5 (a) Tomatoes occluded by other tomatoes and branches, (b) Tomatoes occluded by other tomatoes and leaves, (c) Tomatoes severely occluded by other tomatoes, (d) Tomatoes occluded by shadows and leaf.

This work also implemented a class weighting system in the dataset due to class imbalance. This ensures that all the classes have equal importance on gradient updates when model is training regardless of how many samples the dataset have from each class in the training data. This prevents models from predicting the more frequent class more often just because it's more common. The amount of image samples that exist in each class of training set are structure as follows: 0% Ripe (unripe): 732 samples, 30% Ripe: 547 samples, 50% Ripe: 488 samples, 70% Ripe: 668 samples, 100% Ripe: 495 samples.

The training environment used to conduct this research is as follows: CPU – 12th Gen Intel® Core™ i9-12900H 14-Core Processor; GPU – Nvidia GeForce RTX 3090 (24GB VRAM); RAM – 16 GB. The operating system used is Ubuntu 22.04.

3. Results and discussion

Measuring model performance is a challenging task, as it is essential to ensure that the results presented and any observed performance improvements are genuinely due to the model architecture itself, rather than noise, variance in the training process, or mere chance. To address this, this research incorporates K-fold cross-validation, dividing the dataset into 10 equal folds. This allows all models to be trained and validated on all parts of the dataset [34]. This work reports the fold with the highest F1 score and average F1 score across all folds to represent the model's performance. The F1 score is a widely used metric for evaluating CNN architectures across various fields [35]. Additionally, random seed has been fixed at 42 to ensure reproducibility, enabling future researchers to replicate the results.

3.1 Stage1: ESEnet

This work divided the experiment into two stages. Stage 1 presented the results of ESEnet and compared them with the original Squeeze-and-Excitation Network. The average pooling used in the original architecture was replaced with max pooling to observe the differences introduced by another deterministic pooling method, followed by mixed and fuzzy pooling to test and evaluate the proposed

architecture. Three state-of-the-art CNN architectures were used as the base for ESEnet: Darknet53, ResNet34, and ResNet101. The diagram of the proposed system can be observed from Figure 6.

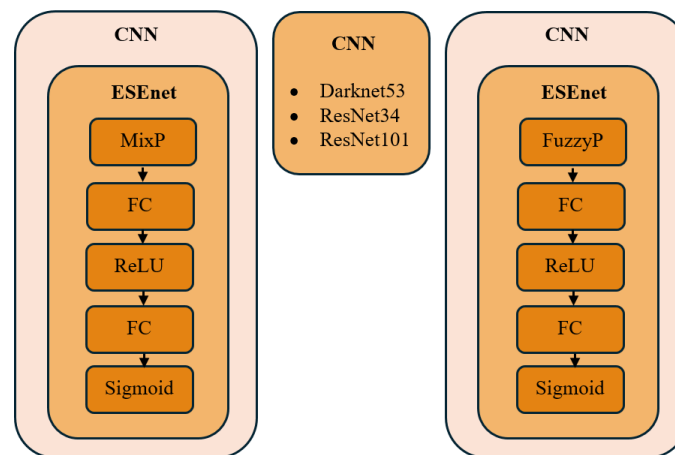


Figure 6 Diagram represents the proposed ESEnet architecture (Remark: MixP = Mixed pooling, FuzzyP = Fuzzy pooling, FC = Fully connected layer, ReLU= Rectified Linear Unit, Sigmoid = Sigmoid activation function)

All of the architectures adopted the same training hyperparameters, starting with a batch size of 96, a learning rate of 0.0001 except ResNet34 which used a learning rate of 0.001, the SGD optimizer, and 1,000 training epochs with early stopping based on validation loss (patience = 100). The SE squeeze ratio was set to 16 for Darknet53 and 8 for ResNet34 and ResNet101. The outcomes of the work were presented in Table 1 and Table 2.

The analysis of model performance on the tomato ripeness dataset revealed several insightful findings. First, looking at Darknet53, Table 1 presented both the average and peak F1 scores measured on the test set across all folds for each model variant, including the original Darknet53, Darknet53+SE (Average Pool), Darknet53+SE (Max Pool), and the proposed Darknet53+ESE (Mixed Pool) and Darknet53+ESE (Fuzzy Pool). The proposed Darknet53+ESE (Fuzzy Pool) achieved the highest peak F1 score of 0.8685 and the highest average F1 score of 0.8429, slightly outperforming the Darknet53+SE (Average Pool) model, which recorded a peak F1 score of 0.8530 and an average F1 score of 0.8350, and the original Darknet53 model with a peak F1 score of 0.8497 and an average F1 score of 0.8381. This suggested that integrating fuzzy pooling with the Squeeze-and-Excitation (SE) block could lead to a modest improvement in Darknet53 model performance. Closely following was Darknet53+ESE (Mixed Pool), which yielded a peak F1 score of 0.8566 and an average F1 score of 0.8385, still slightly outperforming both Darknet53+SE (Average Pool) and the original Darknet53. Interestingly, while Darknet53+SE (Max Pool)'s average F1 score across all folds was the worst among all models at 0.8343, its peak F1 score measured across all folds was in third place right under Darknet53+ESE (Mixed Pool) at 0.8558, outperforming both Darknet53+SE (Average Pool) and baseline Darknet53.

From Table 1, the second section presented experimental results performed on the ResNet34 architecture. ResNet34 + SE (Mixed Pool) achieved the highest average F1 score across all folds at 0.8353, followed closely by ResNet34 + SE (Fuzzy Pool) at 0.8313. Although ResNet34 + SE (Fuzzy Pool) came in second in terms of average F1 score across folds, it achieved first place in terms of peak F1 score across all folds at 0.8662, while ResNet34 + SE (Mixed Pool) also came in second place in this category at 0.8636. Both of these architectures outperformed all other ResNet34 models that were equipped with traditional pooling in all testing categories with minimal increase in computational consumption, as could be observed from Model FLOPs (GMac) and Model Parameters (M). ResNet34 + SE (Average Pool) came in third place for both average F1 score at 0.8284 and peak F1 score at 0.8630, outperforming ResNet34 + SE (Max Pool), which achieved an average F1 score of 0.8209 and a peak F1 score of 0.8472. Baseline ResNet34 came in last place with an average F1 score of 0.8130 and a peak F1 score of 0.8439.

The last section in Table 1 presented experimental results of ResNet101. In this section, surprisingly, ResNet101 + SE (Max Pool) achieved the highest average F1 score at 0.8293, while ResNet101 + SE (Mixed Pool) came closely in second place, achieving an average F1 score of 0.8269, which still outperformed ResNet101 with baseline SE (Average Pool), which achieved an average F1 score of 0.8246. Baseline ResNet101 came in third place at 0.8267, and finally ResNet101 + SE (Fuzzy Pool) in this setting came in last place, achieving an average F1 score of 0.8120. In terms of peak F1 score, baseline ResNet101 came in first place at 0.8536, followed closely by ResNet101 + SE (Average Pool) at 0.8476. ResNet101 + SE (Max Pool) came in third place with a peak F1 score of 0.8472, then ResNet101 + SE (Fuzzy Pool) at 0.8408, and in last place was ResNet101 + SE (Mixed Pool) at 0.8380. This highlighted the fact that even though weight-based and texture-adaptive pooling could enhance model classification accuracy across many CNN architectures, it still posed some limitations in generalization in certain models. More sophisticated methods could be further investigated in the future to further improve the generalization ability of adaptive pooling systems.

Figure 7 represented validation loss, accuracy, and F1 score across all models and architectures that had been implemented with traditional, weight-based, and texture-adaptive pooling SE compared with baseline architectures. When CNN architectures were equipped with SE, it usually caused the training process to take longer before being terminated after reaching early stop. For example, in Figure 7 (a), a baseline Darknet53 usually terminated its training process around epoch 150. Compared to this, in Figure 7 (b) Darknet53+SE (Average Pool), there were many folds that took longer than 150 epochs to reach early stop, with some folds even taking up to 200 epochs before reaching early stop. The same effect could also be seen from Figure 7 (l), (m), (n), (o), in which ResNet101 + SE (Average, Max, Mixed, and Fuzzy Pool) took slightly longer to finish their training processes compared to their baseline counterpart in Figure 7 (k). This effect was consistent with every model that had been equipped with SE, regardless of their pooling operation inside SE or their CNN architecture.



Figure 7 Validation loss, accuracy and F1 score measure across all folds of every models of ESEnet (a) Darknet53, (b) Darknet53+SE (Average Pool), (c) Darknet53+SE (Max Pool), (d) Darknet53+SE (Mixed Pool), (e) Darknet53+SE (Fuzzy Pool), (f) ResNet34, (g) ResNet34 + SE (Average Pool), (h) ResNet34 + SE (Max Pool), (i) ResNet34 + SE (Mixed Pool), (j) ResNet34 + SE (Fuzzy Pool), (k) ResNet101, (l) ResNet101 + SE (Average pool), (m) ResNet101 + SE (Max pool), (n) ResNet101 + SE (Mixed pool), (o) ResNet101 + SE (Fuzzy pool)

Table 1 Experimental results of ESEnet implemented across multiple state-of-the-art CNN architectures

Architecture	Model	Average F1 score (across all folds)	Peak F1 score (across all folds)	Model FLOPs (GMac)	Model Parameters (M)
Darknet53	Darknet53 (Original)	0.8381	0.8497	9.32	40.59
	Darknet53 + SE (Average pool)	0.8350	0.8530	9.33	41.45
	Darknet53 + SE (Max pool)	0.8343	0.8558	9.33	41.45
	Darknet53 + ESE (Mixed pool)	<u>0.8385</u>	<u>0.8566</u>	9.33	41.45
	Darknet53 + ESE (Fuzzy pool)	0.8429	0.8685	9.33	41.45

Table 1 (continued) Experimental results of ESEnet implemented across multiple state-of-the-art CNN architectures

Architecture	Model	Average F1 score (across all folds)	Peak F1 score (across all folds)	Model FLOPs (GMac)	Model Parameters (M)
ResNet34	ResNet34 (Original)	0.8130	0.8439	3.68	21.3
	ResNet34 + SE (Average pool)	0.8284	0.8630	3.69	21.61
	ResNet34 + SE (Max pool)	0.8209	0.8472	3.69	21.61
	ResNet34 + ESE (Mixed pool)	0.8353	<u>0.8636</u>	3.69	21.61
	ResNet34 + ESE (Fuzzy pool)	<u>0.8313</u>	0.8662	3.69	21.61
ResNet101	ResNet101 (Original)	0.8267	0.8536	9.99	42.56
	ResNet101 + SE (Average pool)	0.8246	<u>0.8476</u>	10.02	52.05
	ResNet101 + SE (Max pool)	0.8293	0.8472	10.02	52.05
	ResNet101 + ESE (Mixed pool)	<u>0.8269</u>	0.8380	10.02	52.05
	ResNet101 + ESE (Fuzzy pool)	0.8120	0.8408	10.02	52.05

Remark: Bold values represent the best performance; underline values represent the second-best performance.

3.2 Stage2: Extended Net

This section was the second stage where an experiment was performed on Extended Darknet53, Extended ResNet34, and Extended ResNet101. The word "extended" meant that all these architectures were attached with the proposed weight-based and texture-adaptive pooling, directly replacing their traditional pooling, and then average and peak F1 scores measured across all folds were compared with their baseline counterparts. This work also replaced the average pooling that came with the original architecture, which served as the main feature pooling layer before the fully connected layer, with max pooling to observe the differences when applying another type of deterministic pooling method. The diagram of the proposed system can be observed from Figure 8.

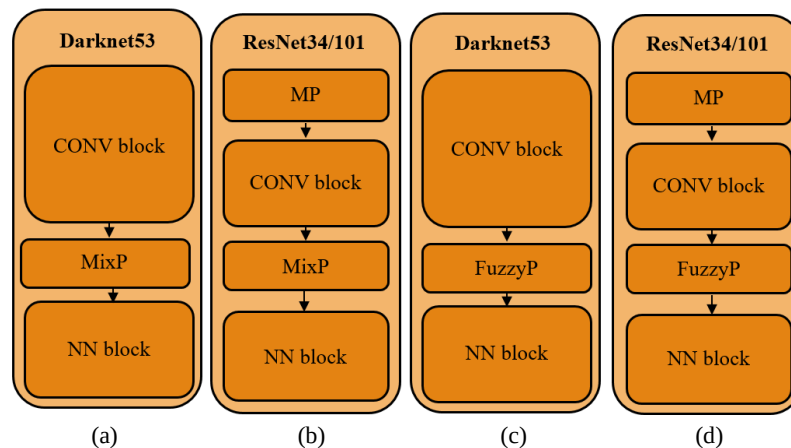


Figure 8 Diagram of the proposed Extended Net architecture: (a, b) Extended Darknet53, ResNet34, and ResNet101 with mixed pooling; (c, d) Extended Darknet53, ResNet34, and ResNet101 with fuzzy pooling. (Remark: MixP = Mixed pooling, FuzzyP = Fuzzy pooling, MP = Max pooling, CONV block= Convolutional block, NN block = Neural network block)

All the CNN architectures in the second stage of the experiment adopted the same training hyperparameters, starting with a batch size of 96, a learning rate of 0.0001 except ResNet34 which used a learning rate of 0.001, the SGD optimizer, and 1,000 training epochs with early stopping based on validation loss. Patience was set at 100 for all architectures. The Squeeze-and-Excitation Network was removed from this stage of the experiment to observe the pure effect of weight-based and texture-adaptive pooling on CNN architecture.

As could be observed from Table 2, first section, Darknet53 + Fuzzy Pool and Darknet53 + Mixed Pool achieved the best peak F1 scores of 0.8722, outperforming baseline Darknet53, which came in second place at 0.8497, and lastly Darknet53 + Max Pool at 0.8441. In terms of average F1 score, the model that performed the best was Darknet53 + Mixed Pool, achieving an average F1 score of 0.8497, followed closely in second place by Darknet53 + Fuzzy Pool, achieving an average F1 score of 0.8478, outperforming both its baseline Darknet53 with an average F1 score of 0.8381 and a modified deterministic pool Darknet53 + Max Pool with an average F1 score of 0.8302 all these improvements were obtained at zero computational increase.

The second section of Table 2 presented ResNet34 and its extended models. ResNet34 + Fuzzy Pool achieved the highest average and peak F1 scores at 0.8305 and 0.8447 respectively, outperforming original ResNet34, which achieved an average F1 score of 0.8130 and a peak F1 score of 0.8439. ResNet34 + Mixed Pool achieved the second-best average F1 score at 0.8223, slightly outperforming baseline ResNet34; however, it still could not outperform baseline ResNet34 in terms of peak F1 score, which it achieved at 0.8357.

ResNet34 + Max Pool slightly outperformed baseline ResNet34 in terms of average F1 score at 0.8178, while still being outperformed by baseline ResNet34 in peak F1 score, which it achieved at 0.8415.

The last section of Table 2 presented ResNet101 and its extended models. ResNet101 + Fuzzy Pool achieved the highest average and peak F1 scores of 0.8284 and 0.8592 respectively, outperforming baseline ResNet101, which came in second place and achieved average and peak F1 scores of 0.8267 and 0.8536 respectively. ResNet101 + Mixed Pool achieved average and peak F1 scores of 0.8097 and 0.8260, slightly outperforming its modified deterministic pooling counterpart ResNet101 + Max Pool, which came in last place and achieved average and peak F1 scores of 0.8036 and 0.8196 respectively.

By integrating weight-based and texture-adaptive pooling mechanisms directly inside the CNN architecture, the results seemed more pronounced, which could be observed from the consistent improvement in results across all architectures. In this setting, all architectures that used fuzzy pool achieved higher average F1 score when compared to their base architectures, which were ResNet34 and ResNet101. Even though Darknet53 + Fuzzy Pool was slightly outperformed by its mixed pool counterpart, its average F1 score across all folds was still very high, outperforming all of its counterparts that were equipped with traditional pooling. In terms of peak F1 score, all the architectures in Table 2 that were equipped with fuzzy pool outperformed every one of their counterparts that were equipped with deterministic pool, highlighting its accuracy-boosting ability when the dataset setup was right. Furthermore, it addresses the challenge—introduced in the introduction section—regarding the limitations of traditional CNNs that rely on a single feature extraction method.

Figure 9 represented validation loss, accuracy, and F1 score across all models and architectures that had been implemented with weight-based and texture-adaptive pooling mechanisms that were directly integrated inside the CNN main pooling layer and then compared with baseline architectures which used traditional pooling. Overall, every model which used deterministic pooling had a relatively similar training process. However, when looking at CNN architectures which used fuzzy pool, a longer training process compared to deterministic pool counterparts could be observed. For instance, Darknet53 with deterministic pool, Figure 9 (a), (b), (c) usually took around 100–150 epochs to reach early stop; however, Darknet53 + Fuzzy Pool, Figure 9 (d) took around 200–300 epochs until reaching early stop on many folds of its training operation. This similar effect could be observed on ResNet101, with its deterministic pooling models, Figure 9 (i), (j), (k) usually taking from 100–200 epochs until reaching early stop; however, ResNet101 + Fuzzy Pool, Figure 9 (l) on average took from 200–300 epochs to reach early stop. However, ResNet34 did not experience this longer training effect on its fuzzy pool as severely as other CNN architectures, which could be observed from Figure 9 (e), (f), (g) that used deterministic pooling and Figure 9 (h) which used fuzzy adaptive pooling. All of these models had relatively the same training time, which took around 100–150 epochs.

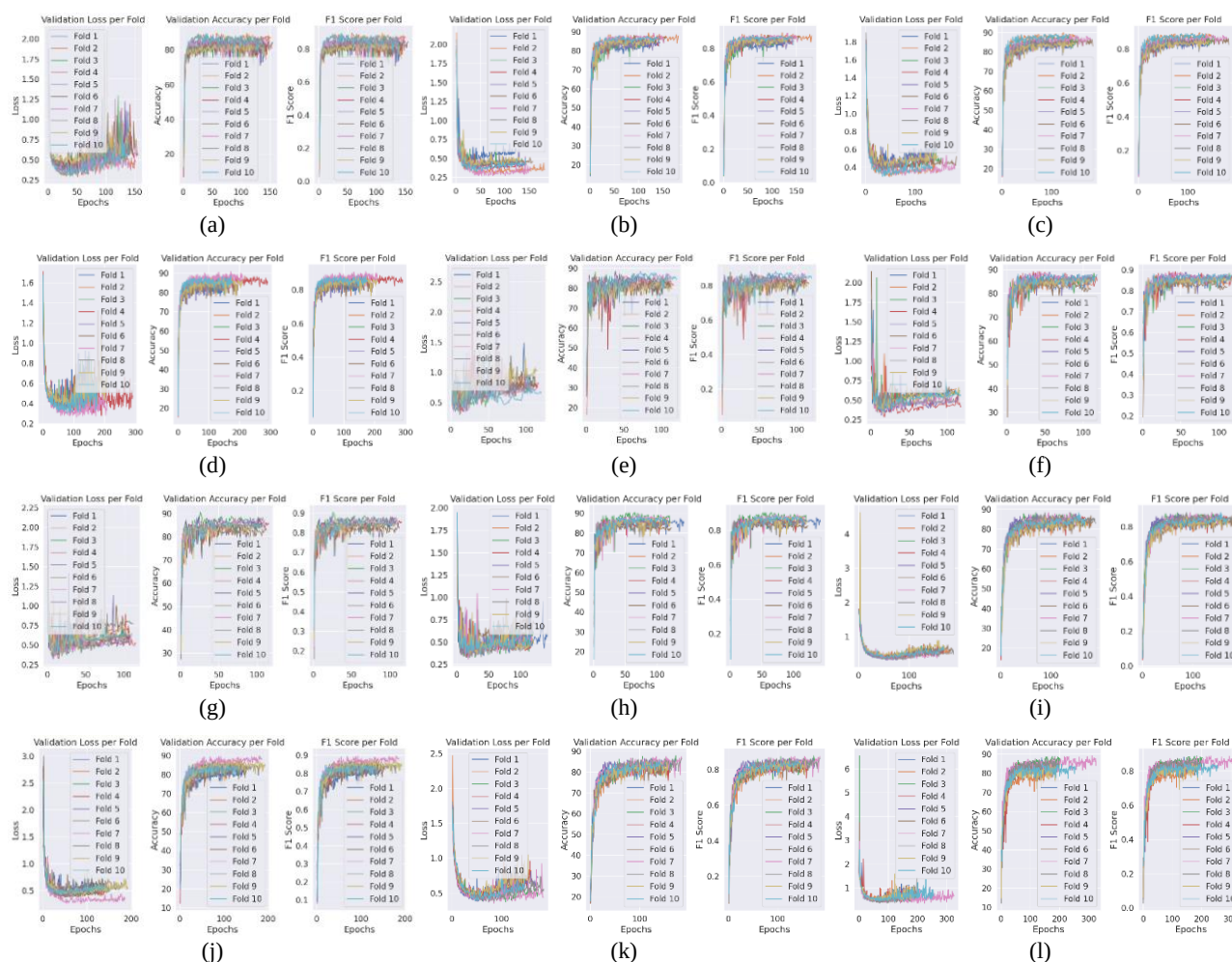


Figure 9 Validation loss, accuracy and F1 score measure across all folds of every models of Extended Net (a) Darknet53, (b) Darknet53 + Max Pool, (c) Darknet53 + Mixed Pool, (d) Darknet53 + Fuzzy Pool, (e) ResNet34, (f) ResNet34 + Max Pool, (g) ResNet34 + Mixed Pool, (h) ResNet34 + Fuzzy Pool, (i) ResNet101, (j) ResNet101 + Max pool, (k) ResNet101 + Mixed pool, (l) ResNet101 + Fuzzy pool

Table 2 Experimental results of Extended Net implemented on multiple CNN architectures

Architecture	Model	Average F1 score (across all folds)	Peak F1 score (across all folds)	Model FLOPs (GMac)	Model Parameters (M)
Darknet53	Darknet53 (Original)	0.8381	0.8497	9.32	40.59
	Darknet53 + Max pool	0.8302	0.8441	9.32	40.59
	Darknet53 + Mixed pool	0.8497	0.8722	9.32	40.59
	Darknet53 + Fuzzy pool	<u>0.8478</u>	0.8722	9.32	40.59
ResNet34	ResNet34 (Original)	0.8130	<u>0.8439</u>	3.68	21.3
	ResNet34 + Max pool	0.8178	0.8415	3.68	21.3
	ResNet34 + Mixed pool	<u>0.8223</u>	0.8357	3.68	21.3
	ResNet34 + Fuzzy pool	0.8305	0.8447	3.68	21.3
ResNet101	ResNet101 (Original)	<u>0.8267</u>	<u>0.8536</u>	9.99	42.56
	ResNet101 + Max pool	0.8036	0.8196	9.99	42.56
	ResNet101 + Mixed pool	0.8097	0.8260	9.99	42.56
	ResNet101 + Fuzzy pool	0.8284	0.8592	9.99	42.56

Remark: Bold values represent the best performance; underline values represent the second-best performance.

3.3 Implementation

The Tomato-Harvesting competition has been organized since 2014 [36] in Kitakyushu, Japan, with the aim of developing agricultural robots, arousing the interest of young researchers in robotics in agriculture, and giving back the developed technology to the agricultural field [37]. Hayashi laboratory at Kyushu Institute of Technology has developed a tomato harvesting robot and has been constantly participating in the Tomato-Harvesting competition since 2014.

This work has been implemented and tested on the tomato harvesting robot developed by Hayashi laboratory by utilizing its hand camera, which is placed at the end effector, to perform close-up classification of tomato ripeness levels.



Figure 10 Hayashi laboratory tomato harvesting robot (AgileX Robotics BUNKER [AGX-BKR-00])



Figure 11 Intel RealSense D435 depth camera of the tomato harvesting robot attached to a UFACTORY xArm 7 robot manipulator

As shown in Figure 10, the tomato fruit will be attached to a rack covered with fake tomato leaf background to simulate the real background the robot will see during competition. One tomato will be classified at a time because the hand camera will only be used for precise measurement of tomato fruit at close distance, since the tomato harvesting robot will use a Kinect DK v2 depth camera at the robot base to perform rough overall tomato detection. Darknet53 + Fuzzy Pool was chosen as the model to perform classification implementation since, among all models with texture adaptive pooling integrated that have been trained and tested in this research, it achieved the highest average F1 score across all folds at 0.8478.

The camera that was used for this experiment was the Intel RealSense D435, which is an RGBD camera manufactured by Intel. Its manufacturing facility for the camera is in Thailand, attached to the end effector of UFACTORY xArm 7 robot manipulator which is a robot manipulator designed and created by UFACTORY company based in Shenzhen, Guangdong, China (Figure 11). This setup was also used by the tomato harvesting robot to detect and measure the depth from the end effector to the tomato fruit and use the manipulator to reach and grasp, but for this work only RGB information will be used. The model and classification program have been moved from the training PC, whose specifications were already declared in section 2.5 Tomato ripeness dataset, to the author's own laptop for demonstration, which has the following specifications: Lenovo IdeaPad Gaming3 15ACH6, CPU – AMD Ryzen 5 5600H with Radeon Graphics; GPU – Nvidia GeForce RTX 3060 Laptop GPU (6GB VRAM); RAM – 16 GB. The operating system used is Ubuntu 22.04.

Darknet53 + Fuzzy Pool was chosen as the model to perform classification implementation since, among all models with texture-adaptive pooling integrated that have been trained and tested in this research, it achieved the highest average F1 score across all folds at 0.8478.

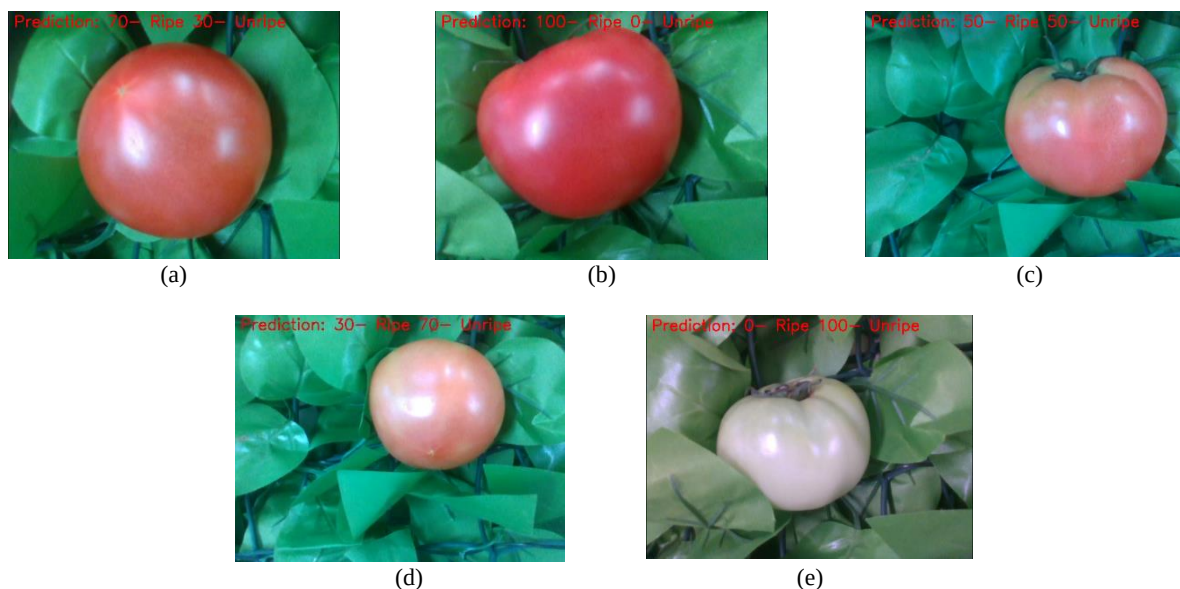


Figure 12 (a), (b), (c), (d), (e) Output result of tomato ripeness classification

Figure 12 shows a classification result of tomato ripeness using a real tomato fruit as a sample for the classification program. In terms of qualitative analysis from observation in real experiments, this research found that a deep red tomato 100% Ripe has a very robust classification process with only minor misclassification observed. The same goes for fully green tomatoes 0% Ripe (Unripe), which only have minor misclassification. This can be explained by the fact that these two classes of tomatoes have very distinct characteristics, such as high color contrast from other classes. While there are still quite a few misclassifications observable in the three other classes 30% Ripe, 50% Ripe, 70% Ripe, this makes sense due to the fact that these three classes share quite similar characteristics in color, where all three classes only have slight differences in shade of red, which could cause misclassification to occur. To test the applicability of the proposed approach, a quantitative investigation was conducted by running the proposed model (Darknet53 + Fuzzy Pool) live using the Hayashi laboratory tomato-harvesting robot's vision system, which captured an image sequence during testing on the experimental tomato fence shown in Figure 13.

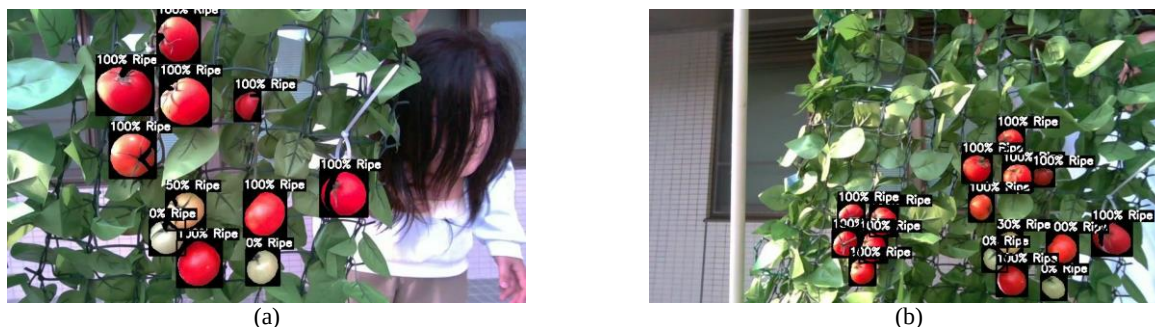


Figure 13 (a), (b) Example of output result captured from robot

The quantitative analysis was conducted by constructing a dataset from images captured at the experimental tomato fence. These images were separated into five classes of tomato ripeness (0%, 30%, 50%, 70%, and 100%), exactly matching the main dataset [33] used in the primary research. The results are shown in Table 3.

Table 3 Experimental results of Darknet53 + Fuzzy Pool implemented on a dataset collected from the experimental tomato fence.

Class name	Original Input Resolution	Model Input Resolution	F1 score	Support
0% Ripe	81x93	256x256	0.92	102
100% Ripe	81x93	256x256	0.98	779
30% Ripe	81x93	256x256	0.59	46
50% Ripe	81x93	256x256	0.74	53
70% Ripe	81x93	256x256	0.20	20
Accuracy			0.92	1000
Macro Average			0.69	1000
Weight Average			0.93	1000

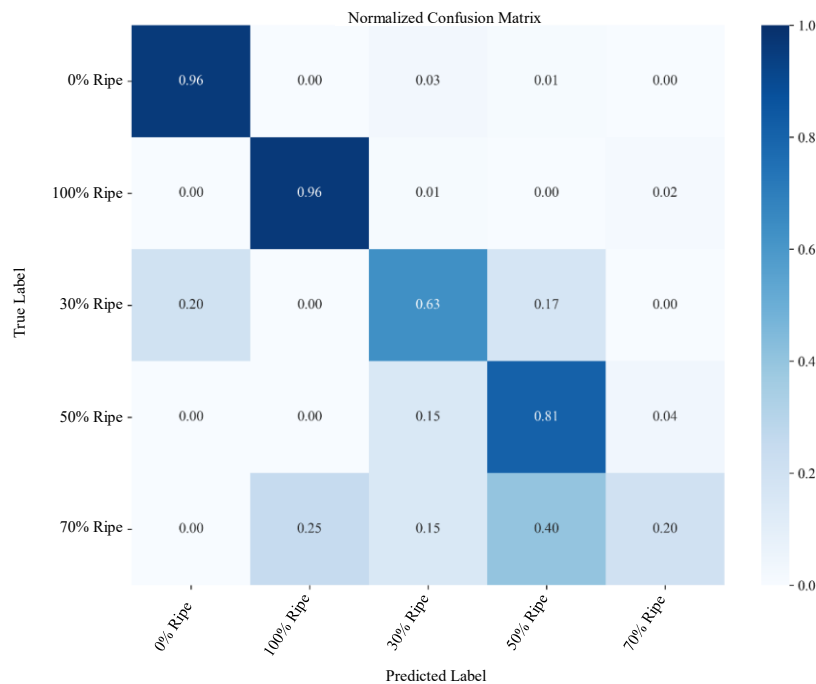


Figure 14 Normalized Confusion Matrix of Darknet53 + Fuzzy Pool

As can be observed from Table 3, classes with unique and clear characteristics, such as 0% Ripe (entirely green tomatoes) and 100% Ripe (entirely red tomatoes), achieved very high F1 scores of 0.92 and 0.98, respectively. These classes had a very low percentage of misclassifications, as can be seen in the confusion matrix in Figure 14. Although there are quite a characteristic similarity between class 30% Ripe and 70% Ripe classes, the 50% Ripe class still achieved a high F1 score of 0.74. It had a low rate of misclassifications which by quantitative result, incorrectly predicted as 70% Ripe and 30% Ripe in 4% and 15% of cases, respectively.

The 30% Ripe class has shown an average performance, obtaining an F1 score of 0.59. This can be explained by its similar visual characteristics, 20% of these instances were incorrectly classified as 0% Ripe, and 17% as 50% Ripe. Lastly, the 70% Ripe class performed the worst, obtaining an F1 score of only 0.20. It had a high misclassification rate, being incorrectly predicted as 50% Ripe in 40% of cases, along with 25% and 15% misclassifications into the 100% Ripe and 30% Ripe classes, respectively. This highlights room for improvement in optimizing future models to distinguish between tomato classes with highly similar appearances.

4. Conclusion

This research proposed a weight-based mixed pooling and novel texture-adaptive fuzzy pooling method for analyzing the ripeness of tomato fruit based on feature map texture intensity analysis. The system was implemented on CNNs by replacing their pooling layers. To prove its robustness, this research implemented the proposed systems on three state-of-the-art CNN architectures: ResNet34, ResNet101, and Darknet53, all of which yielded positive results with improved classification accuracy. In addition to directly implementing the proposed pooling methods on CNNs, this research also implemented mixed and fuzzy pooling on the attention mechanism "Squeeze-and-Excitation Networks." This aimed to fill the research gap clearly identified in previous studies, which stated that the average pooling used in Squeeze-and-Excitation Networks was implemented due to simplicity and could be replaced and further investigated in future research. This implementation also yielded positive results, showing improved classification accuracy on Darknet53, ResNet34 and ResNet101 models where Squeeze-and-Excitation Networks were implemented, compared to traditional methods which used average pooling. This research also utilized K-fold cross-validation method for result collection process, dividing the dataset into ten different folds and training on all parts of the dataset. This approach ensured that any improvement recorded in this research stemmed from actual model performance rather than random noise occurring during the training process. The algorithm was

also implemented on a tomato harvesting robot to analyze qualitative results for real-world application in automatic tomato harvesting and tomato ripeness classification, which is a very important aspect for autonomous tomato harvesting robots.

Although improvement in model classification accuracy by using weight-based mixed pooling and texture-adaptive fuzzy pooling was present in this research, the generalization ability was still limited, as presented in this research that not all CNN models could have their classification accuracy improved by implementing the proposed pooling algorithm. Further investigation could be conducted on more CNN architectures, different types of attention mechanisms, including further investigation into different types of datasets. The mechanism that was used to automatically adjust the weight function, whether to rely more on max or min pooling, could also be changed from fuzzy logic to a deep learning-based or another type of adaptive optimization technique through mathematical framework [38] as well.

5. Author contributions

Conceptualization: Rut Yatigul, Wisanu Jitviriya, Teppakorn Sittiwanchai, Aran Blattler; Methodology: Rut Yatigul, Wisanu Jitviriya, Teppakorn Sittiwanchai, Watcharin Tangsuksant; Formal analysis: Rut Yatigul, Wisanu Jitviriya, Teppakorn Sittiwanchai; Software: Rut Yatigul, Wisanu Jitviriya, Teppakorn Sittiwanchai; Writing—Original Draft: Rut Yatigul; Writing—Review & Editing: Wisanu Jitviriya, Teppakorn Sittiwanchai, Aran Blattler, Watcharin Tangsuksant; Hardware support: Eiji Hayashi; Supervision: Wisanu Jitviriya, Teppakorn Sittiwanchai, Aran Blattler, Watcharin Tangsuksant.

6. Acknowledgements

The '16th run uploaded images e50' dataset, publicly available on the Roboflow Universe platform, was utilized for this research. The authors gratefully acknowledge King Mongkut's University of Technology North Bangkok for its financial support of this research.

7. Reference

- [1] Wiset L, Poomsa-ad N, Onsaard W. Drying characteristics and quality evaluation in microwave-assisted hot air drying of cherry tomato. *Eng Appl Sci Res.* 2021;48(6):724-31.
- [2] Jeeravipoolvarn B, Juangpanich M, Masnate V, Kovithyakorn L. Study on the effects of various irrigation methods on tomato seed production. *KKU Eng J.* 2013;21(1-2):21-7. (In Thai)
- [3] Bapat VA, Trivedi PK, Ghosh A, Sane VA, Ganapathi TR, Nath P. Ripening of fleshy fruit: molecular insight and the role of ethylene. *Biotechnol Adv.* 2010;28(1):94-107.
- [4] Oltman AE, Jervis SM, Drake MA. Consumer attitudes and preferences for fresh market tomatoes. *J Food Sci.* 2014;79(10):S2091-7.
- [5] Miao Z, Yu X, Li N, Zhang Z, He C, Li Z, et al. Efficient tomato harvesting robot based on image processing and deep learning. *Precision Agric.* 2023;24(1):254-87.
- [6] Fujinaga T, Yasukawa S, Ishii K. Tomato growth state map for the automation of monitoring and harvesting. *J Robot Mechatron.* 2020;32(6):1279-91.
- [7] Chaibung N, Sethanan K. The selecting sugar cane plantation development system for harvesting by considering the accessing to the sugar cane plantation. *KKU Eng J.* 2012;39(4):375-82. (In Thai)
- [8] Srisorn W, Doungpueng K, Muenkaew P, Chuan-Udom S. Behavior and model of grain separation for a small axial flow maize shelling unit. *Eng Appl Sci Res.* 2023;50(6):538-47.
- [9] Chaowalittawin V, Krungseanmuang W, Sathaporn P, Morita F, Archevapanich T, Purahong B. Banana quality classification using lightweight CNN model with microservice integration system. *Eng Appl Sci Res.* 2025;52(4):430-8.
- [10] Thipsrirach Y, Kosacka-Olejnik M, Poonikom K. Classification of durian maturity using a convolutional neural network. *Eng Appl Sci Res.* 2023;50(3):233-43.
- [11] Wongkvanklom A, Sata V, Sanjayan JG, Chindaprasirt P. Setting time, compressive strength and sulfuric acid resistance of a high calcium fly ash geopolymer containing borax. *Eng Appl Sci Res.* 2018;45(2):89-94.
- [12] Tian Y, Li X, Ma S. Lightweight encoder with attention mechanism for pipe recognition network. *J Robot Mechatron.* 2024;36(2):343-52.
- [13] He H, Peng Z, Shang P, Wang W, Pei X. An intent recognition method for aerial swarm based on attention pooling mechanism. *J Adv Comput Intell Inform.* 2025;29(1):5-11.
- [14] Yang T, Chan AB. Visual tracking via dynamic memory networks. *IEEE Trans Pattern Anal Mach Intell.* 2021;43(1):360-74.
- [15] Mungklachaiya S, Salaiwarakul A. Attention-X: Enhancing the classification of natural attraction scenes with advanced attention mechanisms. *Eng Appl Sci Res.* 2025;52(3):337-51.
- [16] Khan FS, van de Weijer J, Vanrell M. Modulating shape features by color attention for object recognition. *Int J Comput Vis.* 2012;98:49-64.
- [17] Fu J, Liu J, Tian H, Li Y, Bao Y, Fang Z, et al. Dual attention network for scene segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15-20; Long Beach, USA. USA: IEEE; 2019. p. 3141-49.*
- [18] Hu J, Shen L, Sun G. Squeeze-and-excitation networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18-23; Salt Lake City, USA. USA: IEEE; 2018. p. 7132-41.*
- [19] Jaderberg M, Simonyan K, Zisserman A, Kavukcuoglu K. Spatial transformer networks [Internet]. *arXiv [Preprint].* 2015 [cited 2026 Jan 13]. Available from: <https://arxiv.org/abs/1506.02025>.
- [20] Woo S, Park J, Lee JY, Kweon IS. CBAM: Convolutional block attention module. *Proceedings of the European Conference on Computer Vision (ECCV); 2018 Sep 8-14; Munich, Germany. Cham: Springer; 2018. p. 3-19.*
- [21] Li X, Wu J, Chang D, Huang W, Ma Z, Cao J. Mixed attention mechanism for small-sample fine-grained image classification. *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC); 2019 Nov 18-21; Lanzhou, China. USA: IEEE; 2019. p. 80-5.*
- [22] Peng Y, Chen DZ, Sonka M. Spectral U-Net: Enhancing medical image segmentation via spectral decomposition. *2025 IEEE 22nd International Symposium on Biomedical Imaging (ISBI); 2025 Apr 14-17; Houston, USA. USA: IEEE; 2025. p. 1-5.*

- [23] Jie HJ, Wanda P. RunPool: A dynamic pooling layer for convolution neural network. *Int J Comput Intell Syst.* 2020;13(1):66-76.
- [24] Saeedan F, Weber N, Goesele M, Roth S. Detail-preserving pooling in deep networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2018 Jun 18-23; Salt Lake City, USA. USA: IEEE; 2018. p. 9108-16.
- [25] Zafar A, Aamir M, Mohd Nawi N, Arshad A, Riaz S, Alruban A, et al. A comparison of pooling methods for convolutional neural networks. *Appl Sci.* 2022;12(17):8643.
- [26] Gong Y, Tang W, Zhou L, Yu L, Qiu G. Quarter Laplacian filter for edge aware image processing. *2021 IEEE International Conference on Image Processing (ICIP)*; 2021 Sep 19-22; Anchorage, USA. USA: IEEE; 2021. p. 1959-63.
- [27] Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*; 2009 Jun 20-25; Miami, USA. USA: IEEE; 2009. p. 248-55.
- [28] Bhairannawar S, Patil A, Janmane A, Huilgol M. Color image enhancement using Laplacian filter and contrast limited adaptive histogram equalization. *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*; 2017 Apr 21-22; Vellore, India. USA: IEEE; 2017. p. 1-5.
- [29] Prasotphon T, Selamassakul K, Sittiwanchai T, Jitviriyaw W. Fuzzy-PID vs. PID controllers: A comparative study on DC motor speed control. *2024 1st International Conference on Robotics, Engineering, Science, and Technology (RESTCON)*; 2024 Feb 16-18; Pattaya, Thailand. USA: IEEE; 2024. p. 64-8.
- [30] Nirthika R, Manivannan S, Ramanan A, Wang R. Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study. *Neural Comput Appl.* 2022;34(7):5321-47.
- [31] Redmon J, Farhadi A. YOLOv3: An incremental improvement [Internet]. *arXiv [Preprint]* 2018 [cited 2026 Jan 13]. Available from: <https://arxiv.org/abs/1804.02767>.
- [32] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; 2016 Jun 27-30; Las Vegas, USA. USA: IEEE; 2016. p. 770-8.
- [33] Thesis tomato. 16th run uploaded images e50 Dataset [Internet]. *Roboflow Universe, Roboflow*; 2023. [cited 2025 May 12]. Available from: <https://universe.roboflow.com/thesis-tomato/16th-run-uploaded-images-e50>.
- [34] Sittiwanchai T, Khawnuan U, Yodpijit N. Enhancing human activity recognition with lightweight CNN models and integrated blocks. *Sci Eng Health Stud.* 2024;18:24040001.
- [35] Khawnuan U, Sittiwanchai T, Yodpijit N. Convolutional neural network for wearable fall detection systems. *Sci Eng Health Stud.* 2023;17:23040002.
- [36] Ishii K, Matsuo T, Takemura Y, Sonoda T, Sanada A, Tominaga M, et al. Report on the 8th tomato-harvesting competition toward smart agriculture. *Proceedings of International Conference on Artificial Life and Robotics*; 2023 Feb 9-12; Oita, Japan. p. 484-8.
- [37] Matsuo T, Takemura Y, Sonoda T, Nishida Y, Yasukawa S, Ishii K. Toward smart tomato greenhouse: the 6th tomato-harvesting-robot competition and regulation changes aiming at practical application. *J Robot Netw Artif Life.* 2022;9(1):13-9.
- [38] Senasutham R, Bureerat S, Laohavanich J, Chiawchanwattana C, Yangyuen S. Multi-objective optimization of a threading machine for tobacco leaves. *Eng Appl Sci Res.* 2022;49(2):218-27.