

CLASSIFICATION OF RICE VARIETIES FROM MILLED RICE
GRAIN IMAGES BY OBJECT DETECTION METHOD



A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Applied Mathematics
Suranaree University of Technology
Academic Year 2022

การจำแนกพันธุ์ข้าวจากภาพของเมล็ดข้าวสารด้วยวิธีการตรวจหาวัตถุ



นายวุฒิชัย วัชรรัตน์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาคณิตศาสตร์ประยุกต์


มหาวิทยาลัยเทคโนโลยีสุรนารี

ปีการศึกษา 2565

CLASSIFICATION OF RICE VARIETIES FROM MILLED RICE
GRAIN IMAGES BY OBJECT DETECTION METHOD

Suranaree University of Technology has approved this thesis submitted in partial fulfillment of the requirements for a Master's Degree.

Thesis Examining Committee



Amornrat Suriyawichitseranee.

(Dr. Amornrat Suriyawichitseranee)

Chairperson

J. Tanthanuch

(Asst. Prof. Dr. Jessada Tanthanuch)

Member (Thesis Advisor)

Pisamai Kittipoom

(Asst. Prof. Dr. Pisamai Kittipoom)

Member

Chatchai Jothityang

(Assoc. Prof. Dr. Chatchai Jothityangsoon)

Vice Rector for Academic Affairs

and Quality Assurance

Santi Maensiri

(Prof. Dr. Santi Maensiri)

Dean of Institute of Science

WUTTICHAJ WATCHARARAT : CLASSIFICATION OF RICE VARIETIES FROM MILLED RICE GRAIN IMAGES BY OBJECT DETECTION METHOD. THESIS ADVISOR : ASST. PROF. JESSADA TANTHANUCH, Ph.D. 47 PP.

Keyword : RICE VARIETIES CLASSIFICATION/MILLED RICE GRAIN/OBJECT DETECTION/ARTIFICIAL INTELLIGENCE

This thesis has applied the YOLOv5 object detection model to help classify rice varieties from images of rice grains from the following varieties: Karacadag, Jasmine, Ipsala, Basmati, and Arborio. The research was divided into three main parts: data engineering, which involved developing a Python program to prepare data for artificial intelligence learning; data science operations using Python programming in conjunction with Google Colaboratory for rice grain detection; and the development of model accuracy evaluation method. In the data preparation phase, single-grain JPEG images were obtained from <https://www.muratkoklu.com/datasets/>, and noise reduction, background removal, and conversion to PNG format were performed. These images were then placed into 800×800 pixels images, each containing 20-60 rice grains, with varying degrees of overlapping: no overlap, and 5%, 10%, 15%, 20%, and 25% overlap. The non-overlapping images were used to train the YOLOv5 model, which was then used to classify rice varieties and identify the locations of various rice grains in the images. The research results showed that the YOLOv5 model could effectively classify all five rice varieties. Evaluating the model's accuracy at a threshold of 0.6, it was found that the model could correctly classify rice varieties in images with 0%, 5%, 10%, 15%, 20%, and 25% grain overlap, with accuracy rates of 99.13%, 99.00%, 98.62%, 98.19%, 97.56%, and 96.89%, respectively.

School of Mathematics
Academic Year 2022

Student's Signature _____
Advisor's Signature _____

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Asst. Prof. Dr. Jessada Tanthanuch, for his unwavering support and guidance throughout my research. I am deeply grateful to my family and friends for their emotional support and encouragement throughout my studies. Finally, I would like to mention the financial support provided by the Kitti Bundit scholarship, which made this research possible.

Wuttichai Watchararat



CONTENTS

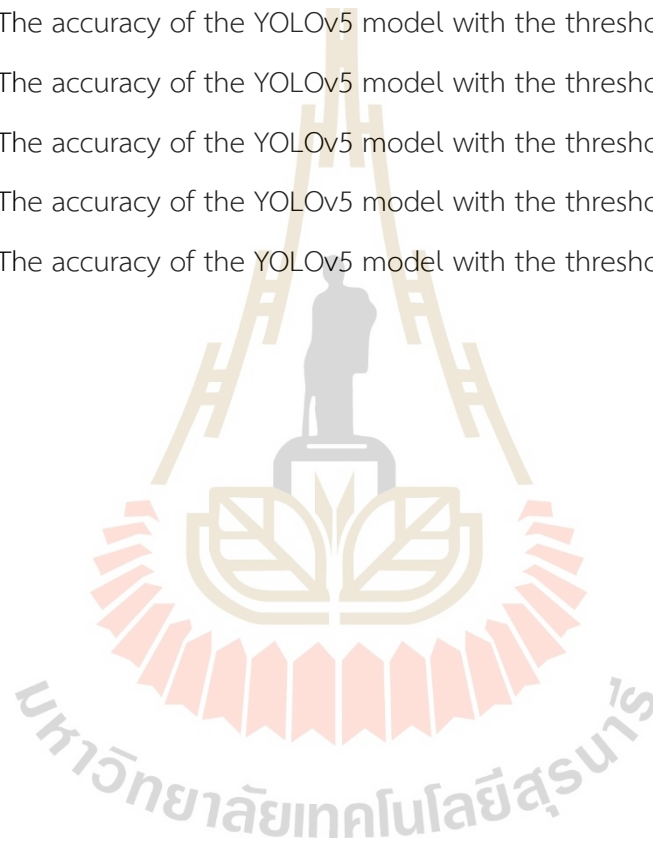
	Page
ABSTRACT IN THAI	I
ABSTRACT IN ENGLISH	II
ACKNOWLEDGEMENTS	III
CONTENTS	IV
LIST OF TABLES	VI
LIST OF FIGURES	VII
CHAPTER	
I INTRODUCTION	1
1.1 Research objective	2
1.2 Scope and limitations	2
1.3 Research procedure	3
1.4 Expected results	3
II LITERATURE REVIEW	4
2.1 Digital Image Processing	4
2.1.1 Grayscale Image	4
2.1.2 Binary Image	5
2.1.3 Morphology	5
2.2 Data Labeling	9
2.3 Intersection over Union (IoU)	10
2.4 Non-Maximum Suppression (NMS)	10
2.5 YOLOv5 model	11
2.6 Researches Related with the rice grain classification and the object detection	13
III RESEARCH METHODOLOGY	16
3.1 Data Import	16

CONTENTS (Continued)

	Page
3.2 Data Engineering for Train and Test data	17
3.3 Modeling by the YOLOv5 model	19
3.4 Accuracy measurement for the YOLOv5 model	20
IV RESULTS AND DISCUSSION	23
4.1 Data Engineering	23
4.2 Accuracy measurement for the YOLOv5 model	25
4.3 Accuracy consideration of the YOLOv5 model	28
V CONCLUSION AND RECOMMENDATION	29
REFERENCES	31
APPENDICES	
APPENDIX A APPLICATION OF PYTHON CODE IN DATA ENGINEERING AND MODELING	35
A.1 The transparency image by Python code in Jupyter Notebook	36
A.2 A mixed rice grain image and labeling data by Python code in Jupyter Notebook	37
A.3 Modeling by Python code in Google Colaboratory	40
A.4 Object Detection by Python code in Google Co- laboratory	42
A.5 Accuracy measurement for the YOLOv5 model by Python code in Jupyter Notebook	42
CURRICULUM VITAE	47

LIST OF TABLES

Table		Page
4.1	The total number of rice grains in mixed rice grain images.	25
4.2	The total number of rice grains detected.	25
4.3	The accuracy of the YOLOv5 model with the threshold value = 0.75.	26
4.4	The accuracy of the YOLOv5 model with the threshold value = 0.70.	26
4.5	The accuracy of the YOLOv5 model with the threshold value = 0.65.	26
4.6	The accuracy of the YOLOv5 model with the threshold value = 0.60.	27
4.7	The accuracy of the YOLOv5 model with the threshold value = 0.55.	27



LIST OF FIGURES

Figure		Page
2.1	Set translation and set reflection.	7
2.2	Example of dilation.	8
2.3	Example of erosion.	8
2.4	Example of opening.	8
2.5	Example of closing.	9
2.6	Example of data labeling.	9
2.7	Example data of data labeling.	9
2.8	Structure of One-Stage Object Detector.	12
2.9	YOLOv5 model system.	12
3.1	A five-variety rice grain.	16
3.2	A rice grain image.	17
3.3	The noise elimination.	17
3.4	The noise elimination with contour.	18
3.5	Example of overlapping rice grains in a mixed rice grain image.	19
3.6	Result of training the model to detect the location and identify the varieties of rice grains.	19
3.7	The detection of a mixed rice grains image.	20
3.8	The creation of a filled mixed image.	21
3.9	The detectable area of the YOLOv5 model.	21
3.10	Flowchart of the workflow in this thesis.	22
4.1	The transparency of a rice grains image.	23
4.2	A mixed rice grains image to train model.	24
4.3	The overlapping of a rice grains image.	24
4.4	The graph of the accuracy value of the model.	28

CHAPTER I

INTRODUCTION

In the first half-year of 2022, Thailand was considered the second-largest rice exporter in the world after India. According to the Isra News Agency (2022), the top three rice-exporting countries during that period were as follows: India, accounting for 9.27 million tons of rice; followed by Thailand, exporting 3.51 million tons of rice; and Vietnam, exporting 3.31 million tons of rice. Thailand's rice exports from January to July 2022 amounted to 4,085,198 tons, valued at 71,105.4 million Thai Baht, according to the Thai Rice Exporters Association (2022). Thailand's rice exports are mainly classified into six types: white rice, jasmine rice, parboiled rice, broken rice, glutinous rice, and brown rice (Chaiyanukulkitti, 2022). The form of rice can be classified in accordance with the Notification of the Ministry of Commerce Re: Rice Product Standard B.E. 2559 (Rice Product Standard, 2016) and Re: Prescribing Thai Hom Mali Rice as a Standard Product and Thai Hom Mali Rice Product Standard (No. 3) B.E. 2559 (Thai Hom Mali Rice Product Standard, 2016). This classification takes into account various factors, such as rice variety, product acquisition process (including husking, milling, and steaming), moisture content, grain shape and color, and amylose content (which affects the texture of cooked rice). However, each type of rice has a different price, and the reliability of rice exports is affected by the amount of adulteration of cheaper rice varieties. Various methods for detecting rice adulteration are currently being developed, including DNA extraction.

Currently, artificial intelligence (AI) plays a significant role in research related to object recognition and detection, such as facial recognition, personal identification, patient classification, disease diagnosis, and the classification of types and models of vehicles. In the agriculture-related industry, there have been developments, such as the creation of an expert egg grading system based on machine vision and artificial intelligence techniques (Omid et al., 2013), the use of deep learning for fruit classification (Mimma et al., 2022), mushroom classification (Tarawneh et al., 2022), and the detection and species

classification of orchid flowers (Steven & Toon, 2015). However, there is also ongoing research related to the use of AI or machine learning techniques in working with rice, such as the work of Aki, Güllü, and Uçar (2015), Zareiforush et al. (2016), Cinar and Köklü (2019, 2021, 2022), Köklü, Cinar, and Taspinar (2021), among others. They have applied many techniques to classify various varieties of rice grain. The results of these works have helped in the development of classification performance in terms of both accuracy and time efficiency.

The aim of this work is to study the use of image data of milled rice grains, namely Karacadag, Jasmine, Ipsala, Basmati, and Arborio, which are publicly available from Dr.Murat KÖKLÜ, an instructor at Selçuk University's Faculty of Technology, Department of Computer Engineering and Software. The dataset consists of 15,000 JPEG format images for each grain of rice, with a resolution of 250x250 pixels (75,000 images in total). The first stage of our research is the data engineering process which organizes data to be able the computer system to recognize different varieties of rice grains in images. The second stage is the data science stage which provides a model for future image classification of other grain varieties. On the last stage, all obtained models will be validated and verified.

1.1 Research objective

To apply mathematical knowledge in digital image processing field and object detection techniques for the classification of rice varieties from milled rice grain images.

1.2 Scope and limitations

1. The dataset was selected from muratkoklu, public data of Dr.Murat KÖKLÜ, the instructor of Selçuk University, Faculty of Technology, Department of Computer Engineering, Department of Software, retrieved from <https://www.muratkoklu.com/datasets>.
2. Using Python language version 3.7, the programming language to generate images

and detect an object, working on ASUS-TUF Gaming FX505DU-AL052T, CPU AMD Ryzen 7 3750H with Graphic Processing Radeon Vega Mobile Gfx 2.30 GHz, 16GB RAM, and Microsoft Windows 11 OS.

1.3 Research procedure

The research work proceeded as follows:

1. Study mathematical knowledge in digital image processing for preprocessing data to train a model.
2. Study the YOLOv5 model and technique for apply a model to detect a rice grain in an image.
3. Apply a model to detect, classify and locate a rice grain in an image with Python language.
4. Accuracy measurement for the YOLOv5 model.

1.4 Expected results

It is expected to use the YOLOv5 model as a prototype model to detect, classify and locate each rice grain in the test images.

CHAPTER II

LITERATURE REVIEW

This chapter presents the algorithm of machine learning to detect an object and technique for data engineering.

2.1 Digital Image Processing

Digital image processing refers to the process of manipulating and analyzing images using a computer to convert them into digital data. This allows us to obtain both qualitative and quantitative information.

2.1.1 Grayscale Image

A grayscale image is typically represented as a matrix of pixel values, where each pixel represents a point in the image and the value of the pixel represents the intensity of the grayscale color at that point.

A grayscale image is a 2-dimensional array where the values in the array represent the intensity of the gray color of each pixel. Each pixel in a grayscale image is represented by a single scalar value ranging from 0 (black) to 255 (white), where 0 represents the absence of any light and 255 represents the maximum amount of light, a grayscale image can be represented as a function $f(x, y)$, where x and y are the spatial coordinates of the pixel and $f(x, y)$ is the intensity value at that point. This function can be thought of as a continuous function that describes the intensity variation of the grayscale color across the image.

The matrix representation of a grayscale image can be used to perform various mathematical operations on the image, such as filtering, transformation, and analysis.

2.1.2 Binary Image

Converting a color image to binary allows it to be displayed on a device with only two levels of intensity, or 1 bit, where 0 represents black and 1 represents white. This conversion reduces storage space and is also useful in preprocessing, specifically through the use of the thresholding binary technique. This technique separates objects from the background by changing pixel values to either 0 or 1 based on whether they are less than or greater than/equal to the threshold value, as defined by equation (2.1),

$$b(x, y) = \begin{cases} 0, & p(x, y) < \text{Thr} \\ 1, & \text{otherwise} \end{cases} \quad (2.1)$$

when $b(x, y)$ is a pixel value of binary image, $p(x, y)$ is a pixel value of the input image and Thr is the threshold constant value, it is between 0 and the maximum intensity of the point in the image.

2.1.3 Morphology

In mathematics, morphology is set theory, the objects' shapes in an image are a set in morphology. In binary images, the sets are members of integer space Z^2 , an element of a set is a tuple (vector in Z^2) whose coordinates are the (x, y) coordinates of a black pixel in the image. Morphological operations bring input images manipulated by a shape matrix called structuring elements, and the output image is obtained. Most often this concept is used to separate objects from the background or to modify the shape of an object with the limitation that the image used to separate the object from the background must be a binary image.

Let A and B be sets in Z^2 , with components $a = (a_1, a_2)$ and $b = (b_1, b_2)$, respectively. The translation of A by $x = (x_1, x_2)$, denoted $(A)_x$, is defined as

$$(A)_x = \{c | c = a + x, \text{ for all } a \in A\}. \quad (2.2)$$

The reflection of B , denoted \hat{B} , is defined as

$$\hat{B} = \{x | x = -b, \text{ for all } b \in B\}. \quad (2.3)$$

The complement of set A is

$$A^c = \{x | x \notin A\}. \quad (2.4)$$

Finally, the difference of two sets A and B , denoted $A - B$, is defined as

$$A - B = \{x | x \in A, x \notin B\} = A \cap B^c. \quad (2.5)$$

Set B is commonly as the structuring element in dilation, as well as in other morphological operations. Let \emptyset denoting the empty set, the dilation of A by B , denoted $A \oplus B$, is defined as

$$A \oplus B = \{x | (\hat{B})_x \cap A \neq \emptyset\}. \quad (2.6)$$

The erosion of A by B , denoted $A \ominus B$, is defined as

$$A \ominus B = \{x | (B)_x \subseteq A\}. \quad (2.7)$$

Dilation expands an image and erosion shrinks it. The opening of set A by structuring element B , denoted $A \circ B$, is defined as

$$A \circ B = (A \ominus B) \oplus B. \quad (2.8)$$

The closing of set A by structuring element B , denoted $A \bullet B$, is defined as

$$A \bullet B = (A \oplus B) \ominus B. \quad (2.9)$$

Opening, it's an operation that brings erosion and dilation together. Image erosion may help reduce noise in the lost image followed by dilation of the image to help bring the image back close to the original. Closing, also tends to reduce noise in the lost image but, with a different sequence of operations, closing starts with a dilation first followed by an erosion.

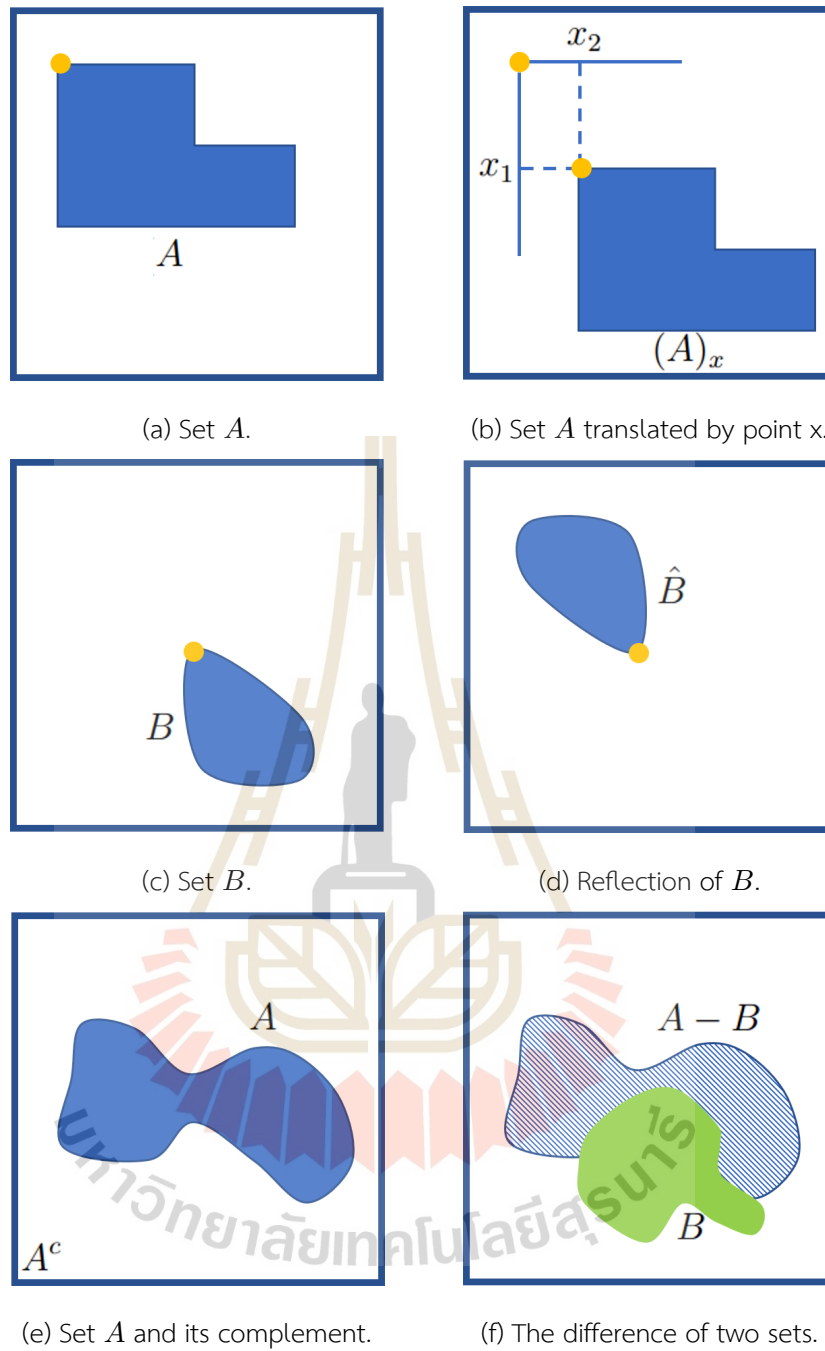


Figure 2.1 Set translation and set reflection.

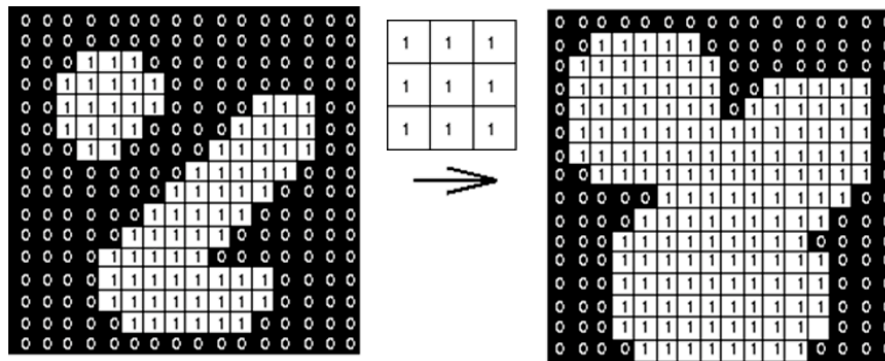


Figure 2.2 Example of dilation.

source: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm>

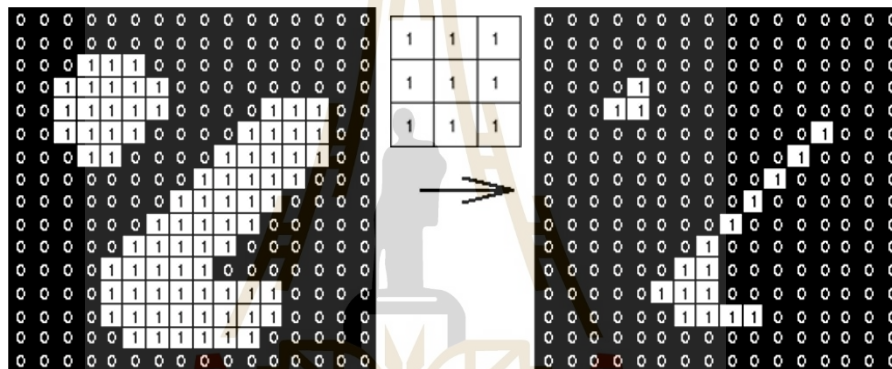


Figure 2.3 Example of erosion.

source: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm>

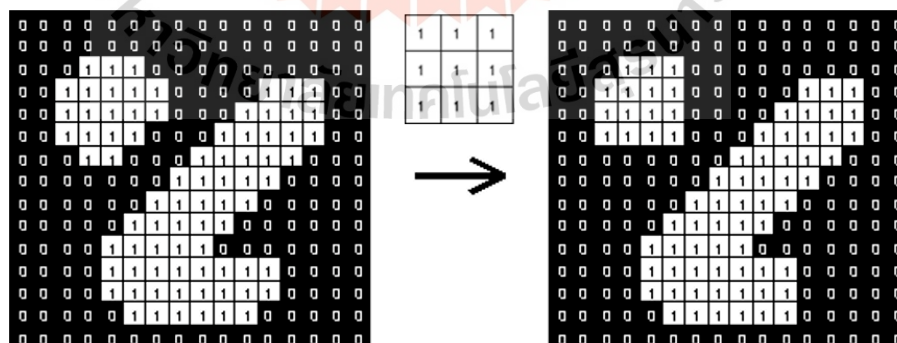


Figure 2.4 Example of opening.

source: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm#guidelines>

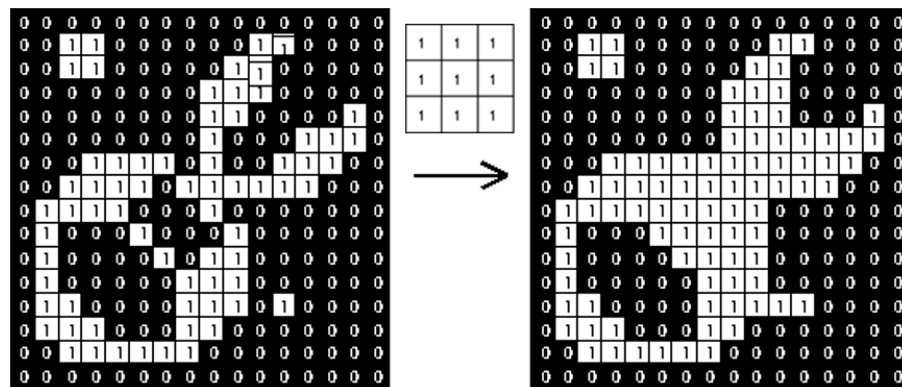


Figure 2.5 Example of closing.

source: <https://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm#variants>

2.2 Data Labeling

Data labeling means that we define the objective of the work we want to do so that the model can understand it. In image detection, labeling involves determining the location of an object and classifying it by creating a bounding box that covers the object.

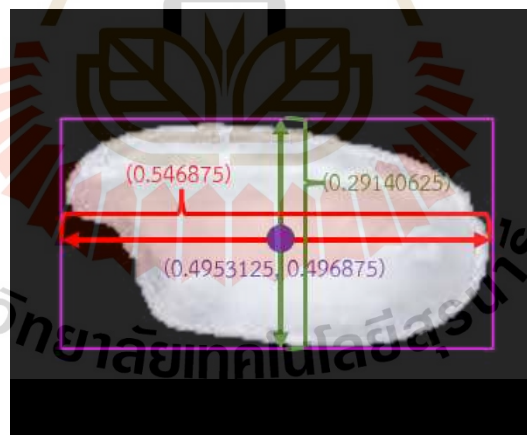


Figure 2.6 Example of data labeling.

0 0.4953125 0.496875 0.546875 0.29140625

Figure 2.7 Example data of data labeling.

In Figure 2.7, the first value represents the object type, while the second and third values indicate the center position of the object in (x, y) form. The fourth and fifth values represent the width and height of the object, respectively.

2.3 Intersection over Union (IoU)

Intersection over Union (IoU) is a common metric used in object detection to evaluate the performance of an algorithm that detects objects in an image. IoU measures the similarity between the predicted bounding box (i.e., the region of an image where the algorithm thinks the object is located) and the ground truth bounding box (i.e., the true location of the object in the image).

IoU is defined as the ratio of the area of the intersection between the predicted and ground truth bounding boxes to the area of their union. The formula for calculating IoU is:

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}. \quad (2.10)$$

To calculate the area of intersection, we first determine the overlapping region between the predicted and ground truth bounding boxes. This is the region where the predicted box and the ground truth box both have non-zero area. We then calculate the area of this region.

To calculate the area of union, we add the areas of the predicted and ground truth bounding boxes and subtract the area of intersection. This is because the intersection is counted twice (once in each bounding box) when we add the areas of the two boxes.

Once we have calculated IoU for a set of predictions and ground truth boxes, we can use it as a performance metric to evaluate the accuracy of the object detection algorithm. A common threshold for determining whether a detection is correct is an IoU of 0.5 or greater. This means that if the IoU between a predicted bounding box and a ground truth bounding box is greater than or equal to 0.5, we consider the detection to be correct.

2.4 Non-Maximum Suppression (NMS)

Non-Maximum Suppression (NMS) is a post-processing step often used in object detection algorithms to eliminate duplicate detections of the same object.

NMS involves three inputs:

1. A set of bounding boxes $B = \{b_1, b_2, \dots, b_n\}$.
2. A set of confidence scores $C = \{c_1, c_2, \dots, c_n\}$, where c_i is the confidence score for the i -th bounding box b_i .
3. A threshold value θ .

The NMS algorithm works as follows:

1. Sort the bounding boxes in B in descending order of their confidence scores in C .
2. Select the bounding box with the highest confidence score, and remove all other bounding boxes that have a high overlap with it, i.e. the boxes having an intersection over union (IoU) greater than a threshold θ . The remaining set of boxes after this step is denoted as B' .
3. Repeat step 2 until there are no more bounding boxes left in B .

2.5 YOLOv5 model

The YOLOv5 model is a One-Stage Object Detection algorithm that can simultaneously detect the object and the area in which it appears in the image. Other object detection algorithms, such as R-CNN, Fast R-CNN, and Faster R-CNN, are two-stage or multi-stage algorithms that detect a region of an object before identifying what it is. While Two-Stage or Multi-Stage Object Detection may be more accurate in detecting objects than One-Stage Object Detection, the YOLOv5 model offers faster performance. The YOLOv5 model's operations are divided into four sub-stages: Input, Backbone, Neck, and Head, as shown in Figure 2.8. In the Input stage, the model imports data, which can be an image, patches, or an image pyramid. In the Backbone stage, the model processes features using neural networks such as VGG16, ResNet-50, ResNeXt-101, and Darknet53. The Neck stage aggregates features using neural networks such as FPN and PANet. Finally, in the Head stage, the model prepares for prediction using neural networks such as RPN, YOLO, SSD, RetinaNet, and FCOS.

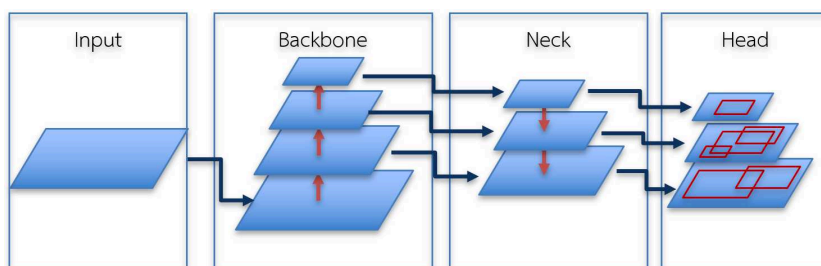


Figure 2.8 Structure of One-Stage Object Detector.

The object analysis process begins with the model looking at grid cells of size $S \times S$. Each grid cell predicts the bounding box surrounding the object in the image and the probability that it belongs to the object. The model only predicts objects with a probability greater than 50%, allowing it to identify which objects are of interest within the box boundaries. This process is illustrated in Figure 2.9.

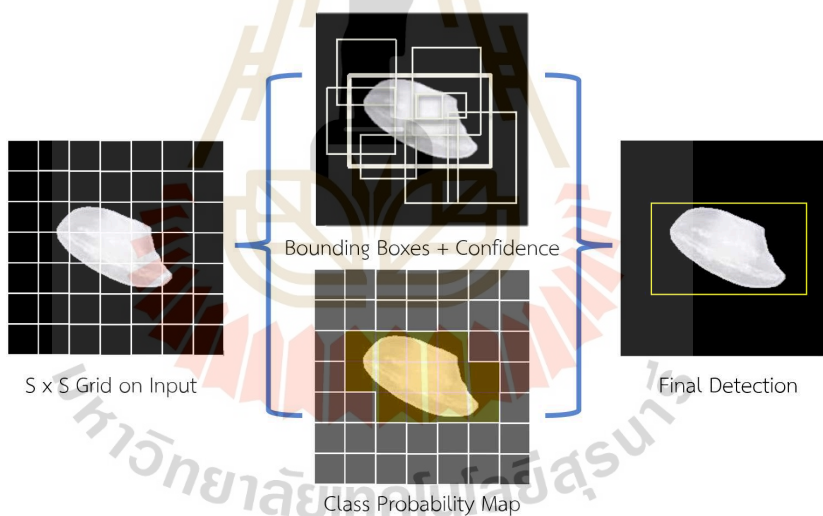


Figure 2.9 YOLOv5 model system.

2.6 Researches Related with the rice grain classification and the object detection

Ren, He, Girshick, and Sun (2016) present a new object detection approach called Region Proposal Network (RPN), which efficiently generate high-quality region proposals by sharing full-image convolutional features with the detection network. The RPN is trained end-to-end and provides region proposals that are used by the Fast R-CNN detection network. The authors also propose to merge the RPN and Fast R-CNN into a single network, using "attention" mechanisms to guide the network's focus. This integration achieves state-of-the-art detection performance while maintaining efficiency.

Cinar and Köklü (2019) design a computerized vision system to differentiate between two different types of rice. The researchers employ several machine learning approaches, including logistic regression (LR), multilayer perceptron (MLP), support vector machine (SVM), decision tree (DT), random forest (RF), Naive Bayes (NB), and k-nearest neighbors (k-NN), to assess the system's performance. The achieved classification accuracy rates are 93.02% for LR, 92.86% for MLP, 92.83% for SVM, 92.49% for DT, 92.39% for RF, 91.71% for NB, and 88.58% for k-NN.

Zhao, Zheng, Xu, and Wu (2019) provide a comprehensive overview of recent advances in deep learning-based object detection. The review covers various deep learning architectures, including Faster R-CNN, YOLO, SSD, and RetinaNet, and examines the strengths and weaknesses of each method. Overall, the authors highlight the remarkable progress made in deep learning-based object detection and the potential for further improvements in the future. They also discuss recent developments in object detection with attention mechanisms and multi-scale feature fusion.

Hamzah and Mohamed (2020) conduct a study to explore innovative approaches in utilizing technology for rice quality classification to avoid contamination. They employ an artificial neural network (ANN) to detect impurities in rice grains. The results show that using a multilayer perceptron neural network (MLPNN) in the ANN produces the highest precision, achieving a rate of 96%.

Truong Hoang , Van Hoai, Surinwarankoon, Duong, and Meethongjan (2020) aim

to automate rice recognition using a combination of hand-crafted descriptors and CNN methods. The researchers conduct their experiment on the VNRICE dataset and achieve a high level of accuracy using the DenNet21 framework. Their results demonstrate the effectiveness of their proposed method in rice recognition with an accuracy rate of 99.04%.

Cinar and Köklü (2021) organize a study in which they utilize feature extraction techniques to analyze five distinct rice varieties from a single brand. The researchers incorporate morphological, shape, and color features in their analysis, resulting in the extraction of a total of 106 features after conducting pre-processing operations. Morphological feature analysis provides 12 characteristics obtained through shape analysis, while shape features is comprised of four features. Moreover, the researchers extract 90 color features using five different color spaces, including RGB, HSV, Lab*, YCbCr, and XYZ. The study aim to identify the top five features that are most effective and specific in distinguishing between the rice varieties through various tests. These features are roundness, compactness, shape factor, aspect ratio, and eccentricity, out of a total of 106 features. The findings of this research suggest the potential of feature extraction techniques in classifying rice varieties based on unique characteristics.

Köklü, Cinar, and Taspinar (2021) focus on the classification of the five most frequently grown rice varieties in Turkey, namely Arborio, Basmati, Ipsala, Jasmine, and Karacadag. The dataset consists of 75,000 grain images, with 15,000 images collected for each rice variety. The second dataset is comprised of 106 features, which includes 12 morphological, four shape, and 90 color features, extracted from these images. The researchers apply Artificial Neural Network (ANN) and Deep Neural Network (DNN) algorithms to process the feature dataset, and the Convolutional Neural Network (CNN) algorithm for the image dataset to perform the classification processes. These models produce a significantly high level of classification accuracy, with ANN achieving a 99.87% accuracy rate, DNN achieving 99.95%, and CNN achieving 100% accuracy. These findings suggest that feature extraction and machine learning techniques can play an instrumental role in accurately classifying rice varieties based on morphological, shape, and color characteristics.

Bhupendra, Moses, Miglani, and Kankar (2022) develop a machine vision system that constructed a dataset of 8048 high-resolution images of damaged rice grains, span-

ning seven damage classes. They use five different state-of-the-art memory-efficient Deep Convolutional Neural Network (CNN) models, including EfficientNet-B0, ResNet-50, InceptionV3, MobileNetV2, and MobileNetV3, which are fine-tuned for damage classification of milled rice grains. The study shows that the EfficientNet-B0 model outperformed the other models with an overall classification accuracy of 98.37%, achieving high individual class accuracy rates ranging from 95.45% to 100%. Moreover, the model has a significantly reduced size (47 MB) and a short prediction time of 0.122 seconds, making it an efficient system for the classification of damaged rice grains.

Cinar and Köklü (2022) explore the classification of five distinct varieties of rice by utilizing features such as morphology, shape, and color. The researchers collect a total of 75,000 images of rice grains, with 15,000 images per variety, which are pre-processed using MATLAB software and prepare for feature extraction. A total of 106 features are extracted from the images, including 12 morphological, 4 shape, and 90 color features from various color spaces. To develop classification models, the researchers utilize machine learning techniques and employed algorithms such as decision tree, k-nearest neighbor, logistic regression, multilayer perceptron, random forest, and support vector machines. Results indicate that the random forest algorithm achieves the highest average classification accuracy, with an accuracy of 97.99% for morphological features and 98.04% for morphological and shape features. The logistic regression algorithm achieves a classification accuracy of 99.25% for color features, while the multilayer perceptron algorithm has the highest accuracy of 99.91% when using all three types of features (morphological, shape, and color). This research sheds light on the potential of utilizing machine learning algorithms in the classification of rice varieties based on various features.

CHAPTER III

RESEARCH METHODOLOGY

This chapter presents the process used in this research. The process comprises of 4 parts as follows: 1) data import, 2) data engineering for train and test data, 3) modeling by the YOLOv5 model, 4) accuracy measurement for the YOLOv5 model.

3.1 Data Import

The rice grain image data used in this thesis was obtained from "muratkoklu", public data of Dr.Murat KÖKLÜ, available on <https://www.muratkoklu.com/datasets>, the dataset consisting of images of Arborio rice grains, Basmati rice grains, Ipsala rice grains, Jasmine rice grains, and Karacadag rice grains, with 15,000 images of each variety.



(a) A Arborio rice grain. (b) A Basmati rice grain. (c) A Ipsala rice grain.



(d) A Jasmine rice grain. (e) A Karacadag rice grain.

Figure 3.1 A five-variety rice grain.

3.2 Data Engineering for Train and Test data

From an image of a rice grain, the background is not completely black (some pixel values are not equal to zero), as shown in Figure 3.2.



Figure 3.2 A rice grain image.

We solved this by removing the background of each grain of rice (making it transparent), we use closing and opening in morphological operations to eliminate noise, as shown in Figure 3.3.



(a) A noisy image. (b) A closing image. (c) A opening image.

Figure 3.3 The noise elimination.

In this particular image, there is a large of noise that cannot be eliminated by using closing and opening techniques. Instead, we use contour techniques to isolate the main object in the image, as shown in Figure 3.4.

We created a blank image with a size of 800×800 pixels to generate a mixed rice grain image using Python and the Jupyter Notebook program. This was done to create images with multiple objects in one image. The image was created by randomly sampling images of rice grains from all 5 varieties, with 6,000 images of each variety (30,000 total



Figure 3.4 The noise elimination with contour.

images), and placing them in the overall image of the rice grains, resulting in 7,000 images without overlapping. Each mixed rice grain image contains 20 to 35 different rice grains, randomly placed in different positions, and we created a file to write data on the variety, position, and size of each rice grain (labeling data).

For the test data, we created mixed rice grain images for the model to detect rice grains. However, these images are different from the ones used for training data because it may occur the overlapping of rice grain in the images for the real situation. Here, we used a transparent rice grain image to create a rice grain collection to measure the accuracy of the YOLOv5 model. The mixed rice grain images with the overlap of the rice grains were done as the followings. The images allow overlap rice grain images in size with an

3.4 Accuracy measurement for the YOLOv5 model

To measure the accuracy of the YOLOv5 model, we detect a rice grain in a overlapping rice grain image with $\text{conf} = 0.25$ (conf-thres is the smallest value to be shown or precision). If it is less than 0.25, it will not be shown. The results of detection by the YOLOv5 model are a detected image and data of a variety, position, and size of a rice grain.



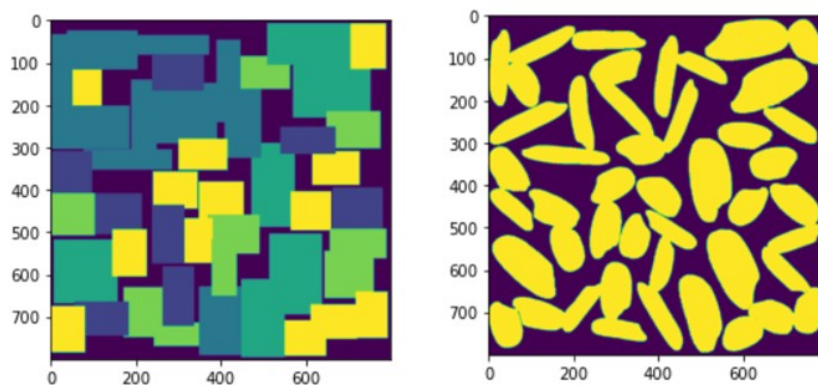
(a) A detected image.

(b) A detected data.

Figure 3.7 The detection of a mixed rice grains image.

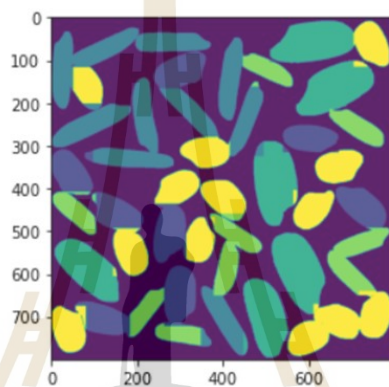
We use labeling data to create a filled image with different pixel values (a pixel value depends on variety), as shown in Figure 3.8 (a).

We bring a mixed rice grain image and convert it to a grayscale image, as shown in Figure 3.8 (b). Then we reconstruct a filled mixed rice grain image with the shape of the grain according to the mixed rice grain image and the pixel value according to the filled image. This gives us an overview of the rice grains with pixel values according to the rice varieties, as shown in Figure 3.8 (c).



(a) A filled image.

(b) A grayscale imag.



(c) A filled mixed image.

Figure 3.8 The creation of a filled mixed image.

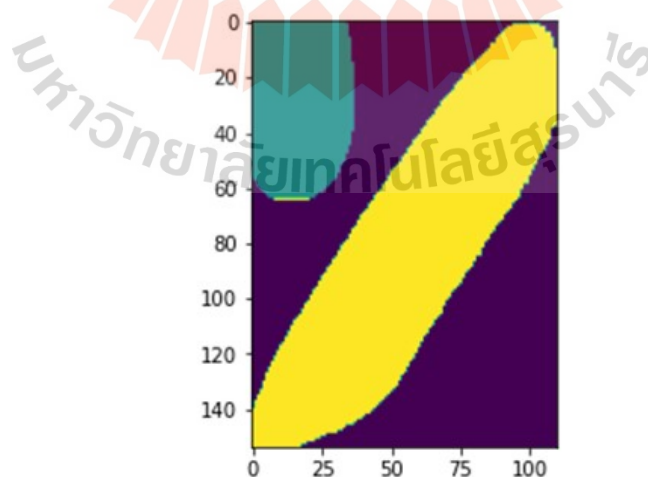


Figure 3.9 The detectable area of the YOLOv5 model.

The accuracy is defined as the ratio of the number of rice grain variety pixels detected in the detectable area of the YOLOv5 model to the total number of pixels detected, as shown in Figure 3.10.

We assume that the YOLOv5 model correctly detected the rice grain if the number of rice grain variety pixels was greater than the threshold value.

Overall, the workflow of this thesis is presented in the following flowchart.

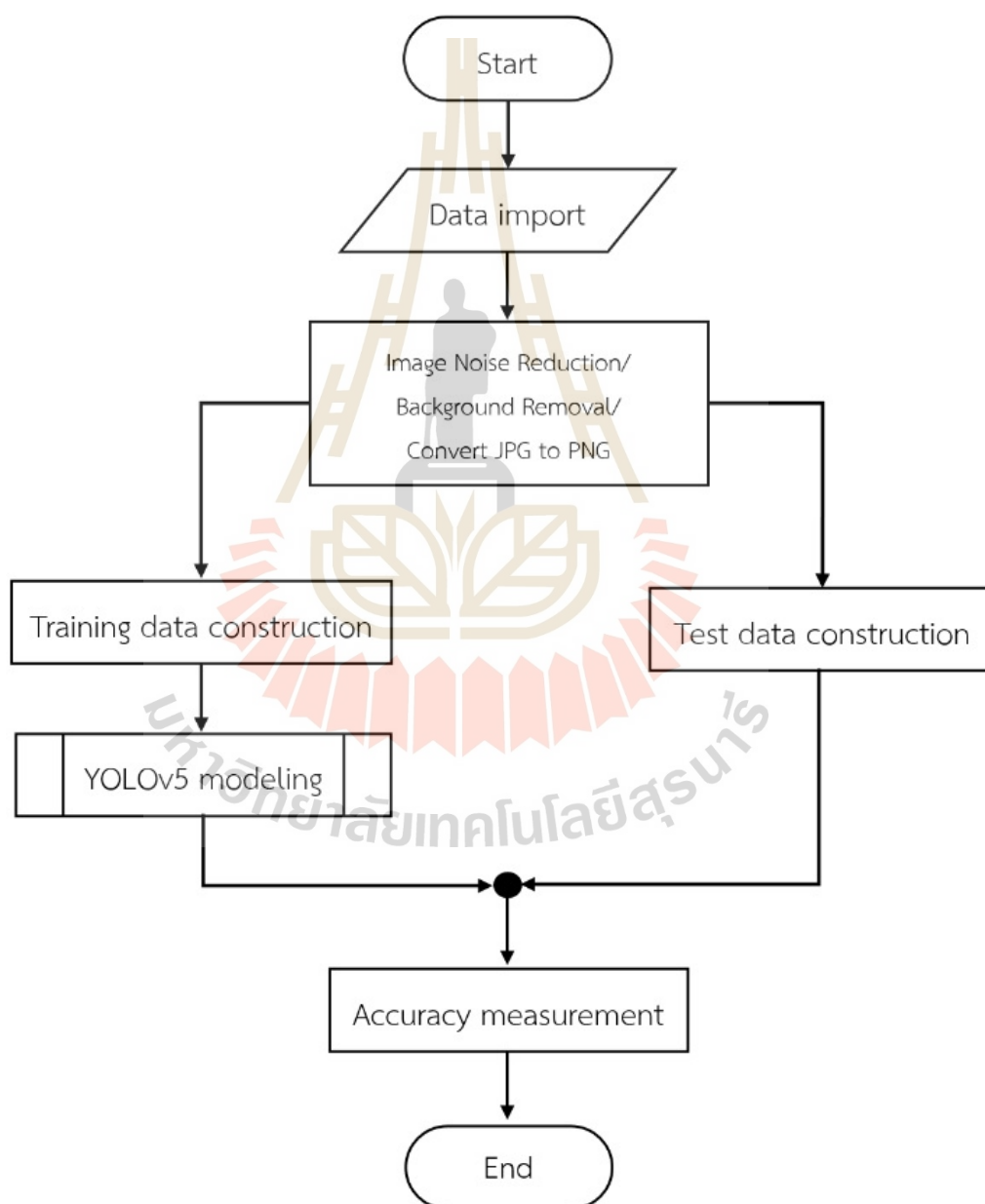


Figure 3.10 Flowchart of the workflow in this thesis.

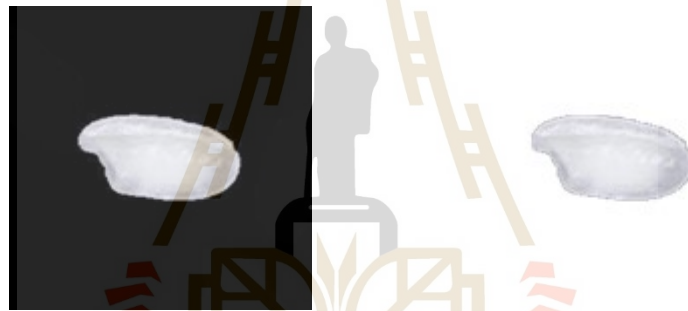
CHAPTER IV

RESULTS AND DISCUSSION

This chapter presents the results from the data engineering process and the accuracy measurement for the YOLOv5 model.

4.1 Data Engineering

We use Python code and morphological operations to eliminate noise and remove the background from a rice grain image, The results are shown in Figure 4.1.

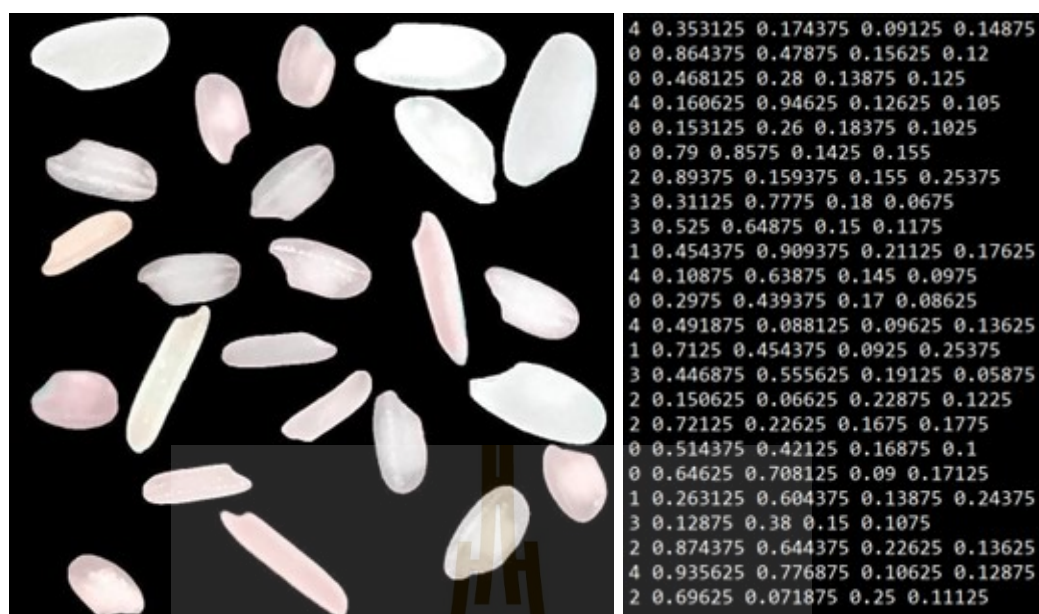


(a) A normal rice grain image. (b) A transparent rice grain image.

Figure 4.1 The transparency of a rice grains image.

We use Python code to create a composite image of mixed rice grains, and a text file to record data on the variety, position, and scale of each grain. The results are shown in Figure 4.2.

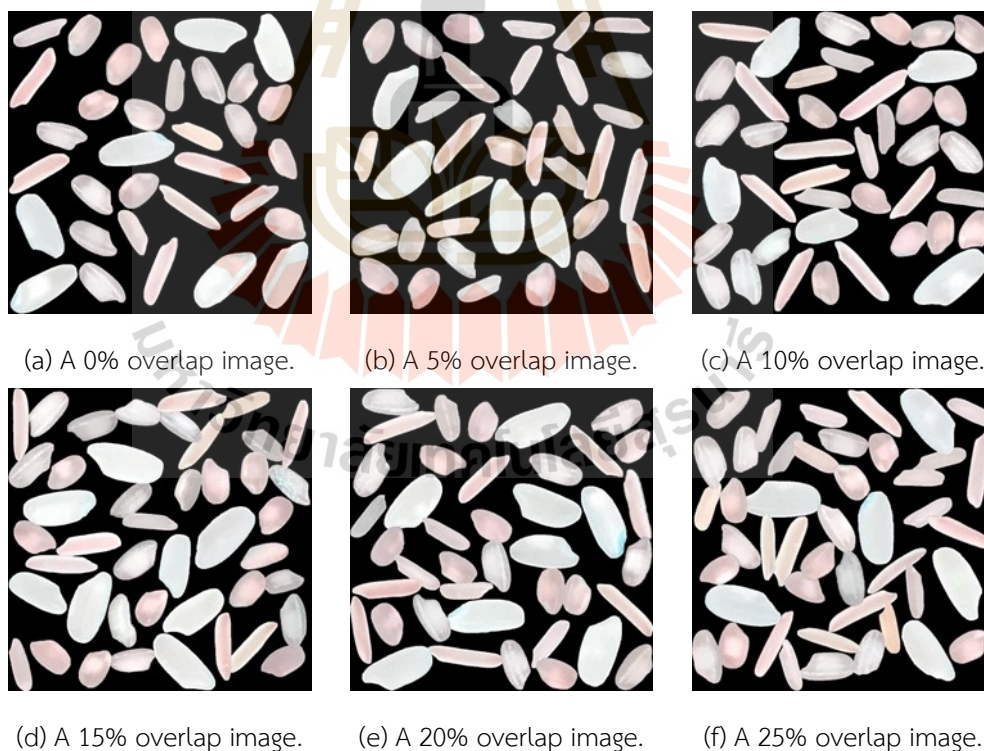
To create an overlapping image of rice grains, the results are shown in Figure 4.3.



(a) A mixed rice grain image.

(b) Data of rice grains labeling.

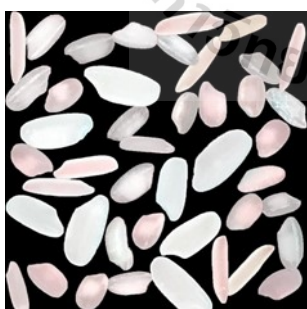
Figure 4.2 A mixed rice grains image to train model.



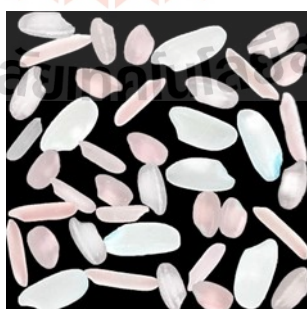
(a) A 0% overlap image.

(b) A 5% overlap image.

(c) A 10% overlap image.



(d) A 15% overlap image.



(e) A 20% overlap image.



(f) A 25% overlap image.

Figure 4.3 The overlapping of a rice grains image.

4.2 Accuracy measurement for the YOLOv5 model

For the measurement of the accuracy of the YOLOv5 model used in the detection of all 5 types of rice grains, we set the threshold value at 0.75, 0.7, 0.65, 0.6, and 0.55.

Table 4.1 The total number of rice grains in mixed rice grain images.

	overlap					
	0%	5%	10%	15%	20%	25%
Arborio	3799	4145	4295	4348	4637	4882
Basmati	3009	3448	3571	3759	3875	3930
Ipsala	2474	2584	2771	2893	3027	3140
Jasmine	4010	4580	4516	4725	5105	5135
Karacadag	4221	4630	4868	4827	5145	5396

Table 4.2 The total number of rice grains detected.

	overlap					
	0%	5%	10%	15%	20%	25%
Arborio	3889	4259	4447	4562	4877	5194
Basmati	3011	3443	3588	3774	3914	3968
Ipsala	2473	2585	2772	2894	3029	3142
Jasmine	4047	4630	4568	4792	5179	5214
Karacadag	4208	4602	4830	4778	5084	5302

Table 4.3 The accuracy of the YOLOv5 model with the threshold value = 0.75.

	overlap					
	0%	5%	10%	15%	20%	25%
Total of True	17418	19117	19579	19940	20805	21176
Total of False	210	402	626	860	1278	1644
Accuracy	0.9881	0.9794	0.9690	0.9586	0.9421	0.9280

Table 4.4 The accuracy of the YOLOv5 model with the threshold value = 0.70.

	overlap					
	0%	5%	10%	15%	20%	25%
Total of True	17459	19229	19757	20170	21145	21612
Total of False	169	290	448	630	938	1208
Accuracy	0.9904	0.9851	0.9778	0.9697	0.9575	0.9471

Table 4.5 The accuracy of the YOLOv5 model with the threshold value = 0.65.

	overlap					
	0%	5%	10%	15%	20%	25%
Total of True	17471	19294	19859	20326	21390	21933
Total of False	157	225	346	474	693	887
Accuracy	0.9911	0.9885	0.9829	0.9772	0.9686	0.9611

Table 4.6 The accuracy of the YOLOv5 model with the threshold value = 0.60.

	overlap					
	0%	5%	10%	15%	20%	25%
Total of True	17475	19324	19926	20423	21545	22111
Total of False	153	195	279	377	538	709
Accuracy	0.9913	0.9900	0.9862	0.9819	0.9756	0.9689

Table 4.7 The accuracy of the YOLOv5 model with the threshold value = 0.55.

	overlap					
	0%	5%	10%	15%	20%	25%
Total of True	17475	19332	19952	20460	21630	22228
Total of False	153	187	253	340	453	592
Accuracy	0.9913	0.9904	0.9875	0.9836	0.9795	0.9740

4.3 Accuracy consideration of the YOLOv5 model

We plotted the accuracy values of the YOLOv5 model on a graph, with the horizontal axis representing overlapping percentages. Specifically, 1 denotes 25% overlapping, 2 represents 20% overlapping, 3 is 15% overlapping, 4 is 10% overlapping, 5 represents 5% overlapping, and 6 represents non-overlapping.

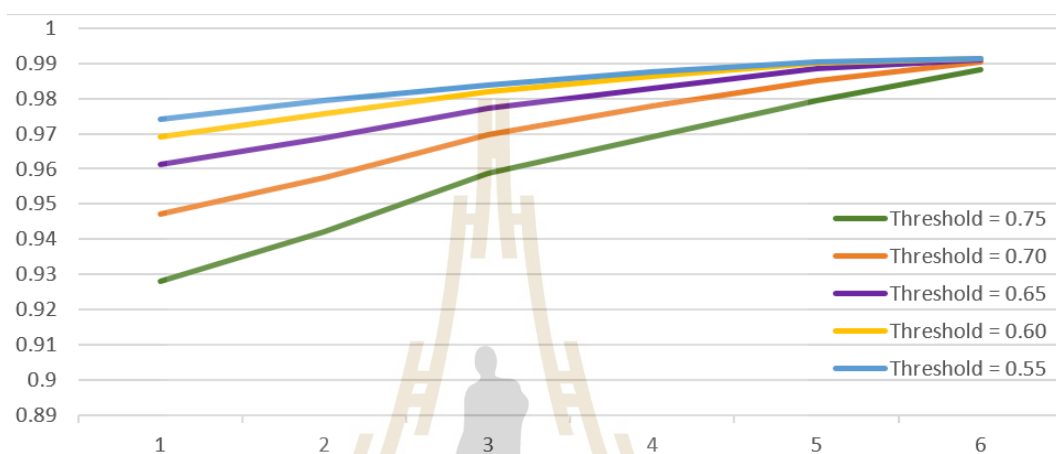


Figure 4.4 The graph of the accuracy value of the model.

Upon examining the accuracy graph, we observed that the accuracy value for non-overlapping rice grain images at a threshold of 0.60 was identical to that at a threshold of 0.55. As a result, we determined that a threshold of 0.60 would be the optimal choice for assessing the accuracy of the YOLOv5 model.

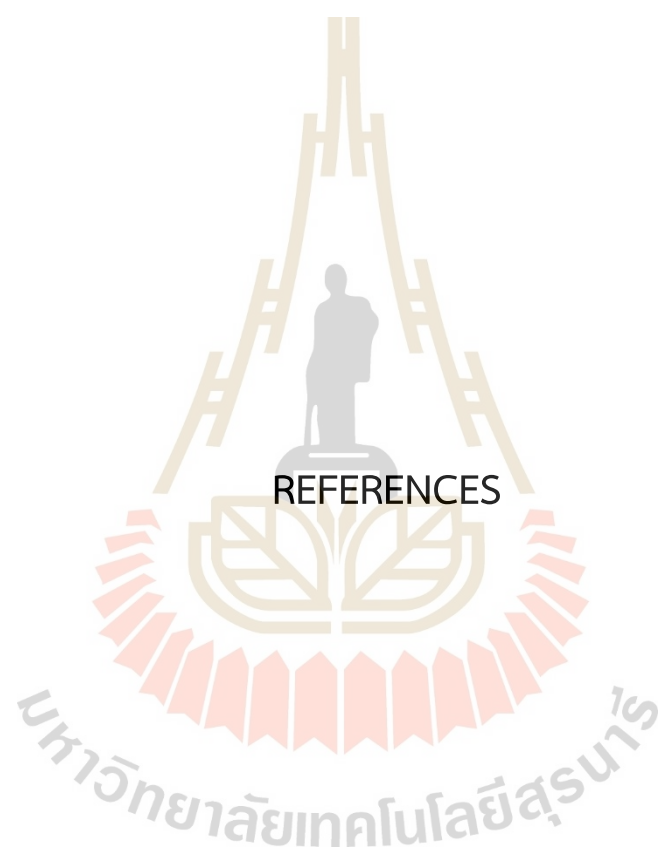
CHAPTER V

CONCLUSION AND RECOMMENDATION

In this research, our aim is to classify different varieties of rice grains using the YOLOv5 model as a detection method. We began by preparing rice grain image data and labeling it to train the YOLOv5 model. To measure the accuracy of the YOLOv5 model, we used overlapping rice grain images and applied the threshold technique at values of 0.75, 0.70, 0.65, 0.60, and 0.55. The accuracy value of the YOLOv5 model with non-overlapping rice grain images at a threshold of 0.60 was the same as the accuracy value with non-overlapping rice grain images at a threshold of 0.55. Based on this, we concluded that a threshold value of 0.60 is optimal for measuring the accuracy of the YOLOv5 model.

As part of our recommendations, we suggest using data on various rice grain varieties in Thailand to reduce rice adulteration and increase the reliability of exports.





REFERENCES

REFERENCES

- Aki, O., Güllü, A., and Uçar, E. (2015). Classification of Rice Grains Using Image Processing and Machine Learning Techniques. *International Scientific Conference "UNITECH 2015"*, 20 – 21 November, 2015, Gabrovo, 352-354.
- Bhupendra, Moses, K., Miğlani, A., and Kankar, P. K. (2022). Deep CNN-based damage classification of milled rice grains using a high-magnification image dataset. *Computers and Electronics in Agriculture*, 195(2022), 106811. doi.org/10.1016/j.compag.2022.106811.
- Charniak, E. (2018). *Introduction to Deep Learning*. The MIT Press.
- Cinar, I., and Köklü, M. (2019). Classification of Rice Varieties Using Artificial Intelligence Methods. *International Journal of Intelligent Systems and Applications in Engineering (IJISAE)*, 7(3), 188–194.
- Cinar, I., and Köklü, M. (2021). Determination of Effective and Specific Physical Features of Rice Varieties by Computer Vision in Exterior Quality Inspection. *Selcuk Journal of Agriculture and Food Sciences (SJAFS)*, 35(3), 229-243.
- Cinar, I., and Köklü, M. (2022). Identification of Rice Varieties Using Machine Learning Algorithms. *Journal of Agricultural Sciences (Tarim Bilimleri Dergisi)*, 28(2), 307-325.
- Gonzalez, R. C., and Woods, R. E. (1992). *Digital Image Processing*. United States of America, Addison-Wesley Publishing Company.
- Hamzah, S. A., and Mohamed, A. (2020). Classification of white rice grain quality using ANN: a review. *IAES International Journal of Artificial Intelligence (IJ-AI)*, 9(4), 600-608.
- Köklü, M., Cinar, I., and Taspınar, Y. S. (2021). Classification of Rice Varieties with Deep Learning Methods. *Computers and Electronics in Agriculture*, 187(2021), 1-8.
- Köklü, M. (2022). Datasets. *Online Education*. Available Source: <https://www.muratkoklu.com/datasets/>, 13 September, 2022

- Aepanich, N., and Urairong, H. (2019). *Improving Verification Method for Thai Hom Mali Rice and Pathumthani Rice Variety*. Agricultural Biotechnology Research and Development Group, Biotechnology Research and Development Bureau, Department of Agriculture.
- Mimma, N.-E.-A., Ahmed, S., Rahman, T., and Khan, R. (2022). Fruits Classification and Detection Application Using Deep Learning, *Scientific Programming, 2022*, Article ID 4194874, 16 pages. <https://doi.org/10.1155/2022/4194874>
- National Bureau of Agricultural Commodity and Food Standards. (2017). *Thai jasmine rice*. Agricultural Standards TAS 4000-2017. Ministry of Agriculture and Cooperatives, published in the Government Gazette. Announcement and general works, volume 134, special section 221 d., 8 September, 2017.
- Neapolitan, R. E., and Jiang, X. (2018). *Artificial Intelligence: With an Introduction to Machine Learning*. (2 ed.) CRC Press.
- Chayanukulkiti, P. (2022). *Rice*. Bureau of Agricultural and Industrial Trade Promotion, March, 2022, Retrieved from <https://www.ditp.go.th>
- Omid, M., Soltani, M., Dehrouyeh, M. H., Mohtasebi, S. S., and Ahmadi, H. (2013) An expert egg grading system based on machine vision and artificial intelligence techniques, *Journal of Food Engineering, 118*(1), 70-77.
- Ren, S., He, K., Grishick, R., and Sun, J. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
- Steven, P., and Toon, G. (2015). Visual detection and species classification of orchid flowers. *IAPR International Conference on Machine Vision Applications (MVA2015)At: Tokyo, Japan*, 505-509. 10.1109/MVA.2015.7153241.
- Tarawneh, O., Tarawneh, M., Sharrab, Y., and Altarawneh, M. (2022). Mushroom classification using machine-learning techniques. *International Computer Sciences*

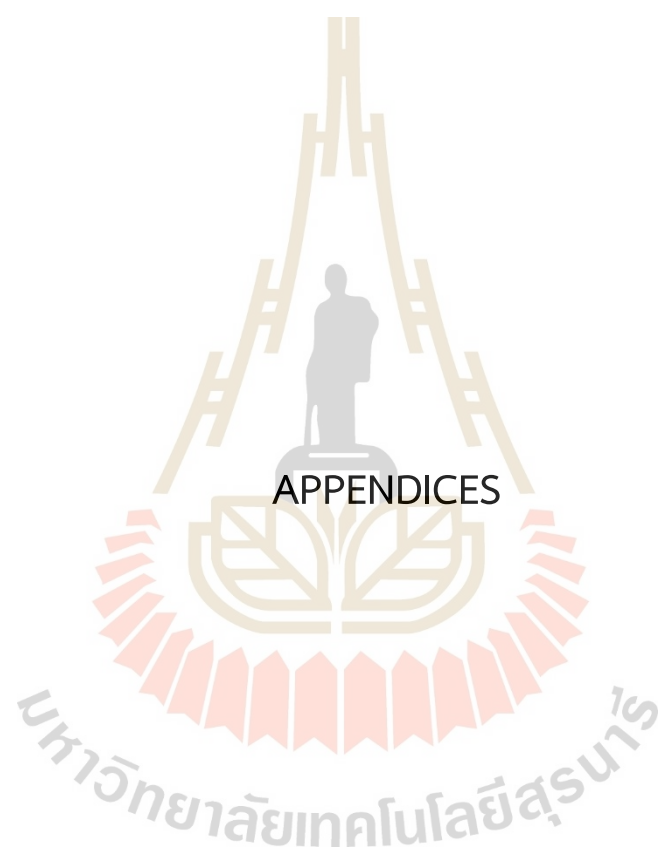
and Informatics Conference (ICSIC-2022).At: AMMAN ARAB UNIVERSITY, Jordan, https://www.researchgate.net/publication/362374571_Mushroom_classification_using_machine-learning_techniques

Thai Rice Exporters Association. (2022). *Statistics of rice exports*. Retrieved from <http://www.thairiceexporters.or.th>

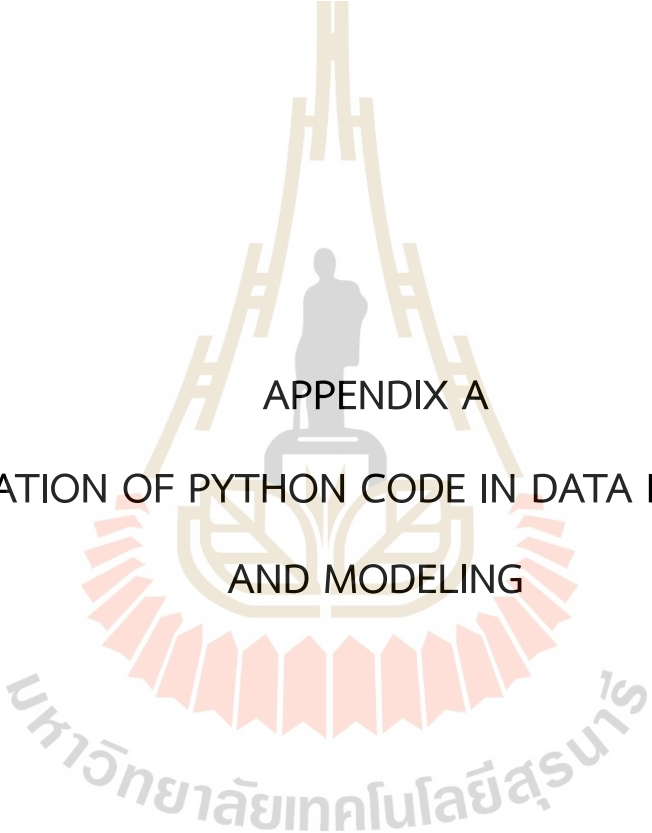
Truong Hoang, V., Van Hoai, D. P., Surinwarangkoon, T., Duong, H.-T., and Meethongjan., K. (2020). A Comparative Study of Rice Variety Classification based on Deep Learning and Hand-crafted Features. *ECTI-CIT Transactions*, 14(1), 1–10. doi.org/10.37936/ecti-cit.2020141.204170.

Zareiforoush, H., Minaei, S., Alizadeh, M. R., and Banaka, A. (2016). Qualitative Classification of Milled Rice Grains Using Computer Vision and Metaheuristic Techniques. *Journal of Food Science Technology*, 53(1), 118-131.

Zhao, Z.-Q., Zheng, P., Xu, S.-T., and Wu, X. (2019). Object Detection with Deep Learning: A Review. *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*.



APPENDICES



APPENDIX A
APPLICATION OF PYTHON CODE IN DATA ENGINEERING
AND MODELING

This chapter presents some Python code using in this thesis.

A.1 The transparency image by Python code in Jupyter Notebook

The transparency image process is the following:

```
import cv2
import matplotlib.pyplot as plt
from skimage import filters
import os
import numpy as np
path = 'C:/Users/Asus/Desktop/JPG/'
path2 = 'C:/Users/Asus/Desktop/PNG/'
os.chdir(path)
for file in os.listdir():
    if file.endswith(".jpg"):
        f_name = f"{path}/{file}"
        wf_name = path2+f"{file}".replace('.jpg', '.png')
        print(f_name)
        print(wf_name)
        fimg = cv2.imread(f_name, 1)
        height = fimg.shape[0]
        width = fimg.shape[1]
        blank_img = np.zeros((height,width,3), np.uint8)
        gray = cv2.cvtColor(fimg, cv2.COLOR_BGR2GRAY)
        _, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
        kernel = np.ones((9,9), np.uint8)
        mask = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, kernel)
        mask = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel)
        # put mask into alpha channel of result
        result = fimg.copy()
        result = cv2.cvtColor(result, cv2.COLOR_BGR2BGRA)
        result[:, :, 3] = mask
        gray = cv2.cvtColor(result, cv2.COLOR_BGR2GRAY)
        # using contours to find the biggest object
        _, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
```

```

contours, hierarchy = cv2.findContours(thresh,cv2.RETR_LIST, \
                                       cv2.CHAIN_APPROX_SIMPLE)

mx = (0,0,0,0)    # biggest bounding box so far
mx_area = 0

for cont in contours:
    x,y,w,h = cv2.boundingRect(cont)
    area = w*h

    if area > mx_area:
        mx = x,y,w,h
        mx_area = area

x,y,w,h = mx
crop_img = result[y:y+h,x:x+w]
plt.imshow(crop_img)
plt.show()

blank_img2 = np.zeros((height,width,4), np.uint8)
blank_img2[y:y+h, x:x+w] = crop_img

new_width = width
new_height = height
result2 = cv2.resize(blank_img2,(int(new_width),int(new_height)))
cv2.imwrite(wf_name, result2)

```

We start by reading an image of a single rice grain in JPEG format. Next, we apply a grayscale transformation to the image. This process allows us to highlight the distinct features and characteristics of the rice grain, making it easier to analyze and manipulate. Finally, we save the grayscale image in PNG format. This format is ideal for storing grayscale images because it can maintain high-quality detail while minimizing file size, making it easier to share and store the image.

A.2 A mixed rice grain image and labeling data by Python code in Jupyter Notebook

The mixed rice grain image and labeling data process is the following:

```

import cv2
import matplotlib.pyplot as plt

```

```

import os

import numpy as np

import random

path = 'C:/Users/Asus/Desktop/Train'
path2 = path+'/Label/'
path3 = path+'/IM/'
meta_name = path3+'data.csv'
metaf = open(meta_name, 'w')
metaf.write('File name,Arborio ,Basmati ,Ipsala ,Jasmine ,Karacadag,SUM\n')
os.chdir(path)

imgwidth = 800
imgheight = 800
blank_img = np.zeros((imgwidth,imgheight,3), np.uint8)
for rnn in range(1,7001):
    blank_img = np.zeros((imgwidth,imgheight,3), np.uint8)
    background = blank_img
    wf_name = path2+str(rnn).zfill(4)+' .txt'
    ftxt = open(wf_name, 'w')
    k_count = j_count = i_count = b_count = a_count = 0
    count_type = np.zeros(5)
    numseed = random.randint(20, 35)
    for seedcount in range(1,numseed+1):
        file = random.choice(os.listdir(path))
        if file.endswith(".png"):
            f_name = f"{path}/{file}"
            fimg = cv2.imread(f_name, cv2.IMREAD_UNCHANGED)
            height = fimg.shape[0]
            width = fimg.shape[1]
            x,y,w,h = cv2.boundingRect(fimg)
            countcheck = 0
            while True:
                countcheck += 1
                xoffset = random.randint(0, imgwidth-w)
                yoffset = random.randint(0, imgheight-h)

```



```

crop_img2 = blank_img[yoffset:yoffset+h,xoffset:xoffset+w]
summ = crop_img2.sum()
if summ==0:
    overlay = fimg[y:y+h,x:x+w]
# IMREAD_UNCHANGED => open image with the alpha channel

alpha_channel = overlay[:, :, 3] / 255 # convert from 0-255 to 0.0-1.0
overlay_colors = overlay[:, :, :3]

alpha_mask = np.dstack((alpha_channel, alpha_channel, alpha_channel))

h, w = overlay.shape[:2]
background_subsection = background[yoffset:yoffset+h, xoffset:xoffset+w]

composite = background_subsection*(1-alpha_mask)+overlay_colors*alpha_mask

background[yoffset:yoffset+h, xoffset:xoffset+w] = composite
f_name2 = f_name.upper()
if f_name2.find('ARBORIO')!=-1:
    rice_tag = '0 '
    count_type[0]+=1
if f_name2.find('BASMATI')!=-1:
    rice_tag = '1 '
    count_type[1]+=1
if f_name2.find('IPSALA')!=-1:
    rice_tag = '2 '
    count_type[2]+=1
if f_name2.find('JASMINE')!=-1:
    rice_tag = '3 '
    count_type[3]+=1
if f_name2.find('KARACADAG')!=-1:
    rice_tag = '4 '
    count_type[4]+=1
txt = rice_tag+str((xoffset+w/2)/imgwidth)+' '
    +str((yoffset+h/2)/imgheight)+' '
    +str(w/imgwidth)+' '+str(h/imgheight)

```

```

        ftxt.write(txt+'\n')
    break
    if countcheck>10000:
        break

ftxt.close()
pix_name = str(rnn).zfill(4)+'.jpg'
plt.imshow(path3+pix_name,blank_img)
metaf.write(pix_name+', '+str(count_type[0])+', '+str(count_type[1])+', '
            +str(count_type[2])+', '+str(count_type[3])+', '
            +str(count_type[4])+', '+str(count_type.sum()))+'\n')

metaf.close()
print('DONE!')

```

To begin, we create a blank image using Python code. Next, we generate a text file to write data of the variety, position, and scale of each rice grain we want to include in the image.

Once the text file is complete, we randomly select a rice grain image to input into the blank image. In this particular code, we are creating a mixed rice grain image that contains between 20 and 35 rice grains, with each grain placed in a non-overlapping position on the blank image. However, it's important to note that if we want to create a mixed rice grain image with overlapping grains, we can adjust the number of rice grains and the value of the sum pixels of input area. The average sum value of pixels for each rice grain image is around 2,000,000 pixels, and the average sum number of pixels for each rice grain image is around 13,700 pixels (which corresponds to approximately 46 rice grains in an image of size 800 × 800 pixels), then a 5% overlap image is values for **summ=<100,000** and **numseed = random.randint(44, 48)**.

A.3 Modeling by Python code in Google Colaboratory

The modelling by the YOLOv5 model process is the following:

```

from google.colab import drive
drive.mount('/content/drive')

```

To begin our analysis, we import both the training and testing datasets from Google Drive. This allows us to leverage flexibility of Google's cloud-based storage system to access and work with our data.

```
!git clone https://github.com/ultralytics/yolov5 # clone
%cd yolov5
%pip install -qr requirements.txt # install
```

```
import torch
import utils
display = utils.notebook_init() # checks
```

To obtain the necessary code for our YOLOv5 model, we clone the folder from the following GitHub repository: <https://github.com/ultralytics/yolov5>. This folder contains all of the Python code necessary to implement and train the model, allowing us to fine-tune the model for our particular use case.

```
%pip install -q wandb
import wandb; wandb.login()
```

WandB offers a variety of functions to assist with your machine learning workflow, including the ability to record training results, collect datasets, generate reports, optimize hyperparameters, and visualize data.

Before training model, it is important to modify the `yolov5s.yaml` file by changing the number of classes from `nc:80` to `nc:5` (the number of classes in model). Additionally, we need to reconstruct the `coco128.yaml` file to ensure it is aligned with these modifications.

```
train: /content/train # path of train dataset
val: /content/train # path of train dataset
# Classes
names: # rice varieties
  0: Arborio
  1: Basmati
  2: Ipsala
  3: Jasmine
```

4: Karacadag

Finally, we can train the YOLOv5 model with an image size of 800 and for a total of 5 epochs. This will enable us to fine-tune the model and optimize its performance for our specific use case.

```
!python train.py --img 800 --epochs 5 --data coco128.yaml --weights yolov5s.pt
```

A.4 Object Detection by Python code in Google Colaboratory

The object detection by the YOLOv5 model process is the following:

```
!python detect.py --weights /content/bestRM.pt --img 800 --conf 0.25 --source /content/test --save-txt
```

To detect a mixed rice grain image, once the detection process is complete, we can then save a text file that contains information about each rice grain's variety, position, and scale. This data can then be used for further analysis or processing as needed.

A.5 Accuracy measurement for the YOLOv5 model by Python code in Jupyter Notebook

The accuracy measurement for the YOLOv5 model process is the following:

```
import cv2
import matplotlib.pyplot as plt
import os
import numpy as np
import random
import re

imgwidth = 800
imgheight = 800
Tr = Fl = 0
Thr = 1
cg1 = ca1 = cb1 = ci1 = cj1 = ck1 = 0
```

```

cg2 = ca2 = cb2 = ci2 = cj2 = ck2 = 0
print('THR = ',Thr)
for im in range(1,501):
    path = 'C:/Users/Asus/Desktop/T2/LB0K/'+str(im).zfill(4)+'.txt'
    pathim = 'C:/Users/Asus/Desktop/T2/IM0K/'+str(im).zfill(4)+'.jpg'
    path2 = 'C:/Users/Asus/Desktop/Detect/D0K/'+str(im).zfill(4)+'.txt'
    #Original
    ftxt = open(path, 'r')
    text = ftxt.readline()
    blank_imgo = np.zeros((imgwidth,imgheight), np.uint8)
    count = 0
    count1 = count2 = count3 = count4 = count5 = 0
    while text:
        text2 = text.split(" ")
        ricetype = int(text2[0])+1
        xo = int(float(text2[1])*imgwidth)
        yo = int(float(text2[2])*imgwidth)
        widtho = int(float(text2[3])*400)
        heighto = int(float(text2[4])*400)
        crop_imgo = blank_imgo[yo-heighto:yo+heighto,xo-widtho:xo+widtho]
        crop_imgo.fill(ricetype)
        text = ftxt.readline()
        count +=1
        if ricetype == 1:
            count1 +=1
        if ricetype == 2:
            count2 +=1
        if ricetype == 3:
            count3 +=1
        if ricetype == 4:
            count4 +=1
        if ricetype == 5:
            count5 +=1
    cg1 +=count
    ca1 +=count1
    cb1 +=count2

```

```

ci1 +=count3
cj1 +=count4
ck1 +=count5

fimg = cv2.imread(pathim)
gray = cv2.cvtColor(fimg, cv2.COLOR_BGR2GRAY)
thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY+ cv2.THRESH_OTSU)[1]
for i in range(0,np.shape(thresh)[0]):
    for j in range(0,np.shape(thresh)[1]):
        if thresh[i,j] == 0:
            blank_imgo[i,j] = 0

#IOU
ftxt = open(path2, 'r')
text = ftxt.readline()
count = 0
count1 = count2 = count3 = count4 = count5 = 0
T = F = 0
while text:
    text2 = text.split(" ")
    ricetype = int(text2[0])+1
    xo = int(float(text2[1])*imgwidth)
    yo = int(float(text2[2])*imgwidth)
    widtho = int(float(text2[3])*400)
    heighto = int(float(text2[4])*400)
    crop_img2 = blank_imgo[yo-heighto:yo+heighto,xo-widtho:xo+widtho]
    ct = cn = 0
    for i in range(0,np.shape(crop_img2)[0]):
        for j in range(0,np.shape(crop_img2)[1]):
            if crop_img2[i,j] == ricetype:
                ct +=1
            if crop_img2[i,j] != ricetype:
                if crop_img2[i,j] > 0:
                    cn+=1
    IOU = ct/(ct+cn)
    if IOU >= Thr:

```

```

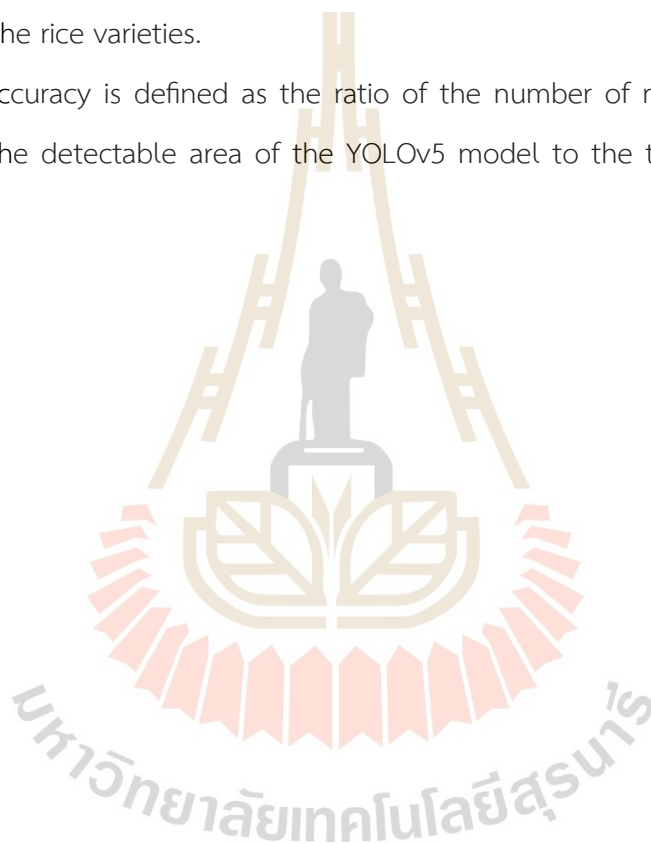
        T +=1
    if IOU < Thr:
        F +=1
    text = ftxt.readline()
    count +=1
    if ricetype == 1:
        count1 +=1
    if ricetype == 2:
        count2 +=1
    if ricetype == 3:
        count3 +=1
    if ricetype == 4:
        count4 +=1
    if ricetype == 5:
        count5 +=1
Tr +=T
Fl +=F
cg2 +=count
ca2 +=count1
cb2 +=count2
ci2 +=count3
cj2 +=count4
ck2 +=count5
print('Image ',str(im).zfill(4),'DONE!!')
print('Grains of Original = ',cg1)
print('Aborio of Original = ',ca1)
print('Basmati of Original = ',cb1)
print('Ipsala of Original = ',ci1)
print('Jasmine of Original = ',cj1)
print('Karacadag of Original = ',ck1)
print('Grains Detection = ',cg2)
print('Aborio Detection = ',ca2)
print('Basmati Detection = ',cb2)
print('Ipsala Detection = ',ci2)
print('Jasmine Detection = ',cj2)
print('Karacadag Detection = ',ck2)

```

```
print('Total of True =',Tr)
print('Total of Flase =',Fl)
print('Accuracy = ',Tr/(Tr+Fl))
```

To measure accuracy for the YOLOv5 model, we start with bring data variety in text file to generate filled image, after that we reconstruct a filled mixed rice grain image with the shape of the grain according to the mixed rice grain image and the pixel value according to the filled image. This gives us an overview of the rice grains with pixel values according to the rice varieties.

The accuracy is defined as the ratio of the number of rice grain variety pixels detected in the detectable area of the YOLOv5 model to the total number of pixels detected.



CURRICULUM VITAE

NAME : Wuttichai Watchararat

GENDER : Male

EDUCATION BACKGROUND:

- Bachelor of Science (Mathematics), Suranaree University of Technology, Thailand, 2020

SCHOLARSHIP:

- His Majesty the King's 7th Cycle Birthday Anniversary Suranaree University of Technology Scholarship
- Kitti Bundit Suranaree University of Technology Scholarship

CONFERENCE:

- The 10th Nonsi Isan National Academic Conference Year 2022 (Best Oral Presentation), Innovation and Technology, Thailand, 26 November, 2022
- The 9th Undergraduate in Applied Mathematics Conference (Bronze medal Oral Presentation), Computational Mathematics, Thailand, 23-24 April, 2021

EXPERIENCE:

- Teaching Assistant in Calculus I, Calculus II, and Calculus III.
- Teaching Assistant in Mathematics in daily life.
- Teaching Assistant in Probability and Statistics.