การตรวจจับอาคารจากภาพรับรู้ระยะไกลโดยใช้โยโล

นายนพดล พุ่มพงษ์

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาคณิตศาสตร์ประยุกต์และวิทยาการคณนา
ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2563

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

BUILDING DETECTION FROM REMOTE SENSING IMAGES USING YOLO

Mr. Noppadon Pumpong

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Applied Mathematics and

Computational Science

Department of Mathematics and Computer Science

Faculty of Science

Chulalongkorn University

Academic Year 2020

Copyright of Chulalongkorn University

Thesis Title	BUILDING DETECTION FROM REMOTE SENSING IMAGES USING YOLO
By	Mr. Noppadon Pumpong
Field of Study	Applied Mathematics and Computational Science
Thesis Advisor	Associate Professor Nagul Cooharojananone, Ph.D.
Thesis Co-advisor	Associate Professor Petarpa Boonserm, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.... Dean of the Faculty of Science
(Professor Polkit Sangvanich, Ph.D.)

THESIS COMMITTEE

.... Chairman
(Assistant Professor Krung Sinapiromsaran, Ph.D.)

.... Thesis Advisor
(Associate Professor Nagul Cooharojananone, Ph.D.)

.... Thesis Co-advisor
(Associate Professor Petarpa Boonserm, Ph.D.)

.... Examiner
(Thap Panitanarak, Ph.D.)

.... External Examiner
(Suriya Natsupakpong, Ph.D.)

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

นพดล พุ่มพงษ์ : การตรวจจับอาคารจากภาพรับรู้ระยะไกลโดยใช้โยโล. (BUILDING DETECTION FROM REMOTE SENSING IMAGES USING YOLO) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : รศ.ดร.นกุล คูหะโรจนานนท์, อ.ที่ปรึกษาวิทยานิพนธ์ร่วม : รศ.ดร.เพชรอาภา บุญเสริม 98 หน้า.

การตรวจจับอาคารจากภาพรับรู้ระยะไกลนั้นได้รับการศึกษาอย่างกว้างขวาง ซึ่งในวิทยานิพนธ์นี้เราจะเสนอแบบจำลองสำหรับการตรวจจับอาคารของสนามบินในภูมิภาคเอเชียผ่านภาพรับรู้ระยะไกลในระดับความสูงหลายระดับแบบจำลองที่ได้นำเสนอนั้นได้รับการปรับปรุงจากการใช้อัลกอริทึมโยโลซึ่งอิงตามแนวคิดของโครงข่ายประสาทแบบคอนโวลูชัน นอกจากนี้เรายังปรับปรุงรูปภาพที่จะใช้ส่งเข้าไปในแบบจำลองของเราโดยใช้แผนภาพเด่นชัดแบบเจท โดยอาคารที่เราต้องการตรวจจับสำหรับการศึกษาครั้งนี้ ได้แก่ อาคารผู้โดยสาร อาคารควบคุม อาคารขนส่งสินค้าและโรงเก็บเครื่องบิน ซึ่งชุดข้อมูลดังกล่าวได้รับการเก็บรวบรวมจากสนามบิน 322 แห่งในภูมิภาคเอเชีย นอกจากนี้แบบจำลองที่ถูกปรับปรุงแล้วยังได้รับการตรวจสอบประสิทธิภาพและความแม่นยำซึ่งผลลัพธ์จากการตรวจสอบแสดงให้เห็นว่าสามารถตรวจจับวัตถุที่ต้องการได้อย่างมีประสิทธิภาพและให้ความแม่นยำสูงกว่าแบบจำลองโยโลดั้งเดิม

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

ภาควิชา	คณิตศาสตร์และ	ลายมือชื่อนิสิต ....
	วิทยาการคอมพิวเตอร์	ลายมือชื่อ อ.ที่ปรึกษาหลัก ....
สาขาวิชา	คณิตศาสตร์ประยุกต์	ลายมือชื่อ อ.ที่ปรึกษาร่วม ....
	และวิทยาการคณนา	
ปีการศึกษา	2563	

6171976023 : MAJOR APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

KEYWORDS : BUILDING DETECTION / SALIENCY MAP / JET COLORMAP / YOLO NETWORK / MORPHOLOGICAL OPERATIONS

NOPPADON PUMPONG : BUILDING DETECTION FROM REMOTE SENSING IMAGES USING YOLO. ADVISOR : ASSOC. PROF. NAGUL COOHAROJANANONE, Ph.D., COADVISOR : ASSOC. PROF. PETARPA BOONSERM, Ph.D., 98 pp.

Building detection system through the remote sensing of images has been widely studied. In this thesis, we propose a model for detecting buildings at airports in Asia through different levels of remote sensing image. The proposed model is improved using the You Only Look Once (YOLO) algorithm based on the convolutional neural network (CNN). We also adjust an inputted image to our model using the Jet Saliency Map. The buildings to be detected in this study are the passenger terminals, the control towers, the cargo buildings, and the hangars. The data set has been collected from 322 different airports in Asia. Furthermore, our improved model is also examined for efficiency and accuracy. The results show that it can detect the intended objects efficiently and provides higher accuracy than the original model.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Department : Mathematics and Computer Science
Field of Study : Applied Mathematics and Computational Science
Academic Year : 2020

Student's Signature ....
Advisor's Signature ....
Co-advisor's Signature ....

ACKNOWLEDGEMENTS

I would like to thank Associate Professor Nagul Cooharojananone, my thesis advisor, and Associate Professor Petarpa Boonserm, my co-advisor, for their innovative ideas that helped me develop this research. This thesis would not have been possible without their assistance and the numerous opportunities from my advisor and my co-advisor.

I am grateful to Project for the Development and Promotion of Science and Technology Talents (DPST) and the Institute for the Promotion of Teaching Science and Technology (IPST) for their financial support. Furthermore, I would like to thank all the professors who taught me to be more knowledgeable and all committees who gave advice and guidelines for solving problems at work. I hope that when I graduate, I will use the knowledge I have gained to the maximum benefit.

In addition, I would like to express my gratitude to the Department of Mathematics and Computer Science and my friends for their constant support.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

CONTENTS

Page

ABSTRACT IN THAI iv

ABSTRACT IN ENGLISH v

ACKNOWLEDGEMENTS vi

CONTENTS vii

LIST OF TABLES x

LIST OF FIGURES xi

CHAPTER

1 Introduction **1**

- 1.1 Objective 5
- 1.2 Scopes and Assumptions 5

2 Background Knowledge **6**

- 2.1 Image Processing 6
  - 2.1.1 Digital Image Definitions 9
  - 2.1.2 Basic Mathematical Tools Used in Image Processing 10
    - 2.1.2.1 Elementwise Operations 10
    - 2.1.2.2 Arithmetic Operations 11
    - 2.1.2.3 Logical Operations 11
  - 2.1.3 Spatial Filtering 13
    - 2.1.3.1 Gaussian Filters 13
  - 2.1.4 Color Image Processing 15
    - 2.1.4.1 Color Models 16
    - 2.1.4.2 Color Mapping 19
  - 2.1.5 Morphological Image Processing 21
    - 2.1.5.1 Erosion and Dilation 22
    - 2.1.5.2 Opening and Closing 25
  - 2.1.6 Binary image 27
    - 2.1.6.1 Thresholding 27
- 2.2 Machine Learning 30

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

CHAPTER Page
2.2.1 Supervised learning . . . 30
2.2.2 Unsupervised learning . . . 30
2.2.3 Semi-supervised learning . . . 31
2.2.4 Reinforcement learning . . . 31
2.2.5 Transfer learning . . . 32
2.3 Deep Learning . . . 32
2.3.1 Linear regression . . . 34
2.3.2 Logistic regression . . . 35
2.3.3 Activation function . . . 36
2.3.4 Weight . . . 37
2.3.5 Bias . . . 38
2.3.6 Backpropagation . . . 38
2.4 Artificial Neural Network . . . 38
2.5 Convolutional Neural Networks . . . 41
2.5.1 Input layer . . . 42
2.5.2 Convolution layer . . . 43
2.5.3 Activation layer . . . 45
2.5.4 Pooling layer . . . 46
2.5.5 Fully connected layer . . . 47
2.6 Object Detection . . . 47
2.7 You Only Look Once Algorithm . . . 48
2.7.1 Overview of the YOLO Pipeline . . . 49
2.7.2 Non-Max Suppression . . . 50
2.7.3 Deficiencies of YOLO . . . 52
2.8 YOLOv2 and YOLO9000 . . . 53
2.8.1 Changes from YOLO . . . 54
2.8.1.1 Anchor Boxes . . . 54
2.8.1.2 Anchor Box Dimensions . . . 55

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

CHAPTER Page

2.8.1.3 Constrained x, y Predictions . . . 55
2.8.1.4 High Resolution Classifier . . . 57
2.8.1.5 Multi Scale Training . . . 57
2.8.1.6 Darknet-19 . . . 57
2.8.1.7 Hierarchical Classification . . . 58
2.9 YOLOv3 . . . 59
2.9.1 Backbone . . . 59
2.9.2 Prediction feature maps . . . 60
2.9.3 Losses . . . 61
2.9.4 Backbone Architecture . . . 63
2.9.5 Residual Block . . . 65
2.10 Performance Evaluation . . . 66
2.10.1 Intersection over Union . . . 66
2.10.2 Confusion Matrix . . . 67
2.10.3 Precision Recall and F1 Score . . . 68
3 Proposed Method . . . 70
3.1 Jet Saliency Map . . . 72
3.2 Applying Jet Saliency Map with YOLOv3 . . . 75
4 Results and Discussions . . . 77
5 Conclusions . . . 92
5.1 Future Work . . . 92
REFERENCES . . . 94
VITA . . . 98

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

LIST OF TABLES

Table Page
2.1 The example of a truth table for AND, OR, and NOT operators 12
4.1 Detection results on the testing set 82

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

LIST OF FIGURES

Figure Page

1.1 Complex shapes of specific buildings [1] 1
1.2 Different sizes of the buildings in the airport [1] 2
1.3 Remote sensing image [1] 2
1.4 Buildings and trees obscured airport buildings [1] 3
2.1 Teleconference via teleconferencing system using image compression techniques [2] 7
2.2 Fingerprint examination using the fingerprint scanning system [3] 7
2.3 Satellite imagery using the principle of image processing [4] 8
2.4 Applying rescue robots as accident prevention and first aid provision system [5] 8
2.5 The example of a grayscale image (left), a grayscale image combined with the intensity arrays (middle), the image intensity arrays (right) [6] 10
2.6 The AND operator gives out 1 only if both (A) and (B) are equal to 1 12
2.7 The OR operator gives out 1 if (A) or (B) or both equal 1 12
2.8 The COMPLEMENT (NOT) operator gives out 0 when (A) = 1 12
2.9 Example of spatial filtering 13
2.10 The graphical representation of the one-dimensional Gaussian distribution . . . 14
2.11 (a) The graphical representation of 2-dimensional Gaussian distribution and (b) The example of a 3×3 gaussian kernel 15
2.12 The RGB color model on a three-dimensional axis [7] 17
2.13 The HSV color model on a three-dimensional axis [8] 18
2.14 The example of a grayscale [9] 19
2.15 The Jet color scales [10] 20
2.16 The evaluation of Jet colormap [11] 21
2.17 The example of erosion using different structural elements 23
2.18 The result of using erosion with a structural element of size 7×7 23
2.19 The example of dilation using different structural elements 24
2.20 The result of using dilation with a structural element of size 7×7 25

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.21 The results of using the morphological opening operation with a square structural element of size 17 × 17 26

2.22 The results of using the morphological closing operation with a square structural element of size 17 × 17 27

2.23 Examples of real-life problems of outcomes from supervised learning and unsupervised learning (a) Patient classification of disease (b) House price prediction by a regression model (c) Clustering of shopping behaviour customer 31

2.24 Applying learning knowledge previously to new models 32

2.25 An example of the layer structure of a deep learning model 33

2.26 Comparison of data feature extraction differences between machine learning and deep learning 34

2.27 An example shows the application of linear regression to estimate the best house price. 35

2.28 An example shows logistic regression determining students who pass and fail from midterm and final scores 36

2.29 An example of the heaviside step function [12] 37

2.30 Examples of the weights in deep learning 38

2.31 The biological neuron [13] 39

2.32 The visualization of an artificial neuron in the hidden layer 40

2.33 Type of artificial neural network (a) Feedforward (b) Feedback 41

2.34 Examples of convolutional neural network structures [14] 42

2.35 The example of a convolutional operation with kernel of size 3 × 3 44

2.36 The movement of the kernel 45

2.37 The rectified linear activation function 46

2.38 The result using average pooling and max pooling with size 2×2 47

2.39 The visualization of an example image with ground truth bounding boxes [15] 48

2.40 Comparison of object detection methods on the Pascal VOC 2007 dataset as of 2015 [16] 49

2.41 The visualization of You Only Look once (YOLO) network structure [17] . . . 50

2.42 The result of using Non-Max Suppression [18] 51

2.43 An example of the confidence score threshold [19] 51
2.44 An example of the final output of the YOLO network [20] 52
2.45 Example of detecting a faulty person (object) of the Yolo network [21] 53
2.46 Comparison of object detection methods on the Pascal VOC 2007 + 2012 dataset as of 2015 [16] 54
2.47 Object box predictions from one anchor box belonging to one particular grid cell [22] 56
2.48 The visualization of You Only Look once v2 (YOLOv2) network structure [22] 58
2.49 Contributions of each of the changes to mAP scores in YOLOv2 [22] 59
2.50 YOLOv3 backbone outputs 60
2.51 The visualization of the You Only Look once v3 (YOLOv3) network structure [23] 65
2.52 The examples of Residual block application 66
2.53 The examples of the Intersection over Union (IoU) 67
3.1 Krabi International Airport, Krabi, Thailand [24] 71
3.2 The visualization of our proposed method architecture [24] 72
3.3 Example of input images [1] 72
3.4 Example of jet color images 73
3.5 Example of jet color images after blurring with Gaussian blur 73
3.6 Example of grayscale images from jet color images after blurring with Gaussian blur 74
3.7 Example of binary images from grayscale 74
3.8 Example of binary images after going through steps 5 and 6 75
3.9 Example of the saliency map image 75
3.10 Flowchart of saliency map creation process [24] 76
4.1 Example images from our dataset (1375 x 770 pixels) [1] 77
4.2 Example images from our dataset (1375 x 770 pixels) [1] 78
4.3 Example images from our dataset (1375 x 770 pixels) [1] 78
4.4 Example images of the result using the original YOLO (1375 x 770 pixels) . . . 79
4.5 Example images of the result using the original YOLO (1375 x 770 pixels) . . . 79

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

4.6 Example images of the result using the original YOLOv3 (1378 x 958 pixels) . 80

4.7 Example images of the result using the original YOLOv3 (1375 x 515 pixels) [24] 81

4.8 Example images of the result using the original YOLOv3 (601 x 277 pixels) . . 81

4.9 Graphs show the results of the measurement values obtained from YOLOv3 . . 83

4.10 Graphs show the results of the measurement values obtained from YOLOv3 with Jet saliency map 83

4.11 Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 793 pixels) 84

4.12 Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 613 pixels) 84

4.13 Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 623 pixels) 85

4.14 Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 643 pixels) 85

4.15 Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 709 pixels) 86

4.16 Example images of the result using the original YOLOv3 with the Jet saliency map (601 x 356 pixels) 86

4.17 Example images of the result using the original YOLOv3 with the Jet saliency map (1375 x 515 pixels) [24] 87

4.18 Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 1239 pixels) 87

4.19 Example images of the result using the original YOLOv3 with the Jet saliency map (1375 x 584 pixels) 88

4.20 Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 958 pixels) 88

4.21 Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 651 pixels) 89

4.22 Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 932 pixels) 89

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

4.23 Example images of the result using the original YOLOv3 with the Jet saliency map (1384 x 1271 pixels) 90
4.24 Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 643 pixels) 91
4.25 Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 747 pixels) 91
5.1 Example of failure detection [24] 93

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

CHAPTER I

INTRODUCTION

The detection of buildings in a given place is a primary step for close observation applications, which allows for the classification of specific buildings. There have been various researches in this branch, which have had many benefits; for instance, an urban planning, a state cadastral inspection, and an infrastructure development of cellular and telecommunication companies. The major problem is that many factors can affect this process, such as the complicated shapes of specific buildings, the building size, image occlusion by other buildings or trees, and remote sensing image clarity. The examples of images are shown in the following figures.

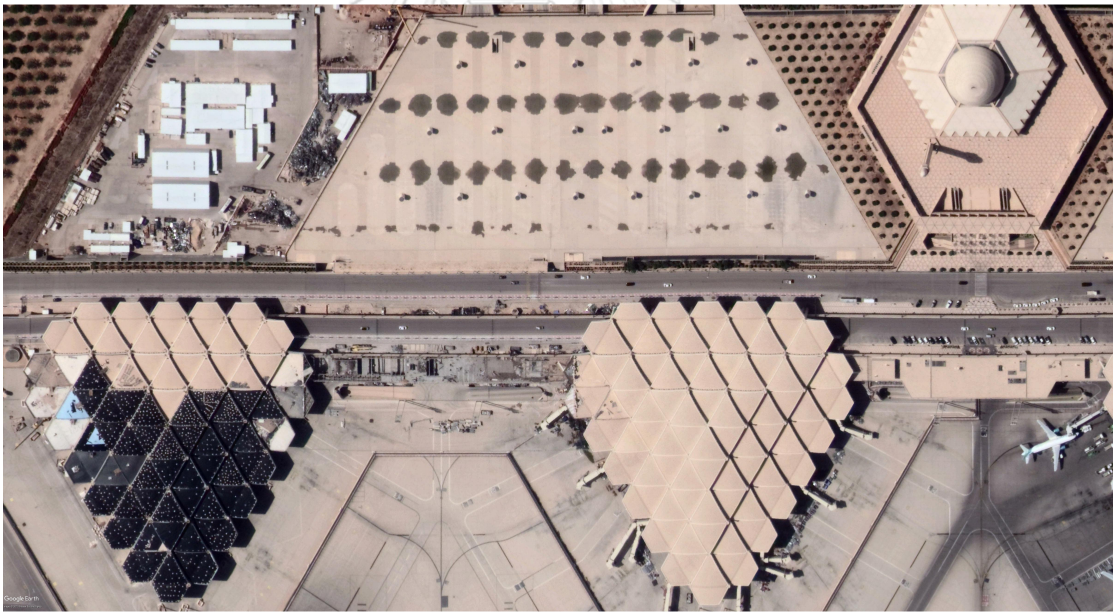


Figure 1.1: Complex shapes of specific buildings [1]

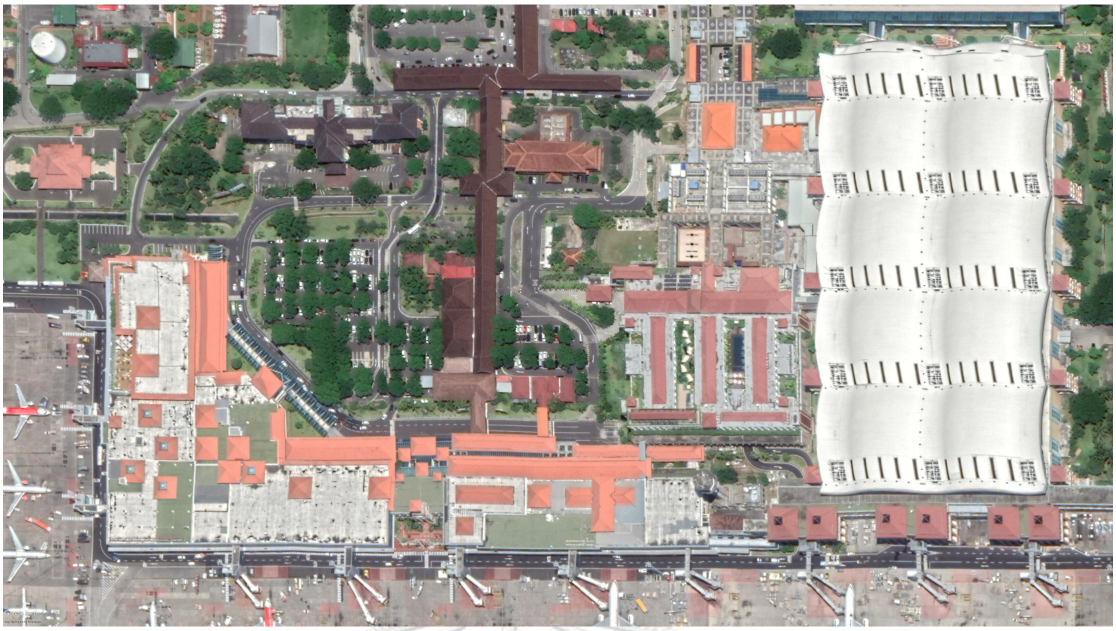


Figure 1.2: Different sizes of the buildings in the airport [1]

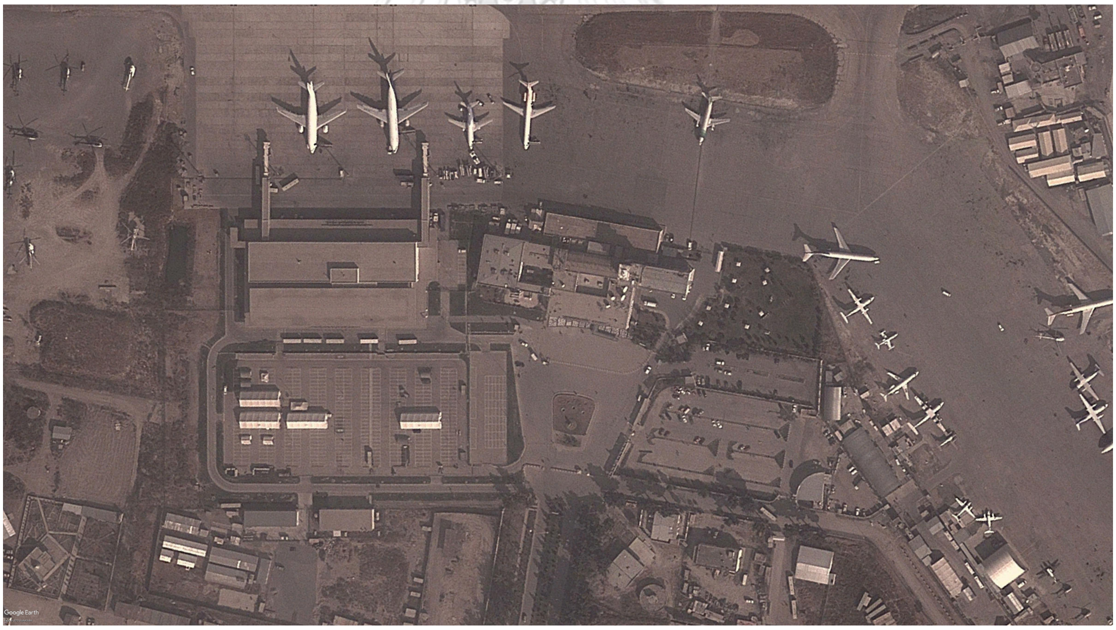


Figure 1.3: Remote sensing image [1]

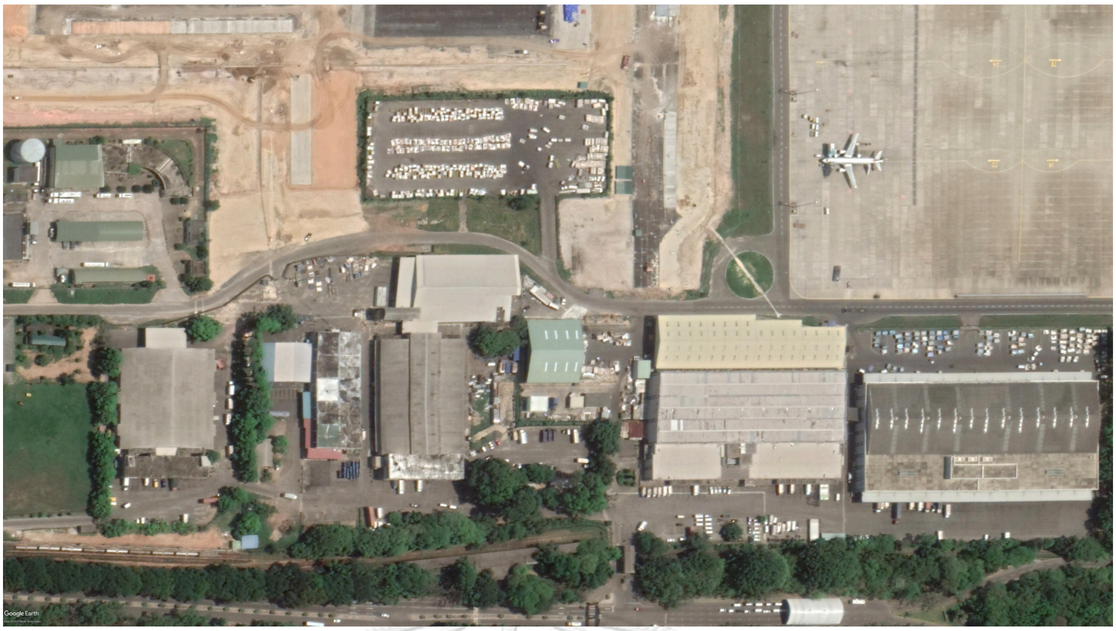


Figure 1.4: Buildings and trees obscured airport buildings [1]

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

To analyze remote sensing images for detecting or classifying buildings, computer scientists may have to do these manually, which requires a lot of processing time and it is hardly impractical when applied to regional or global scales. Thus, it is very important to develop methods that can accurately and automatically detect buildings from high-resolution remote sensing images.

This study addresses the specific building detection within the data set of 322 airports in Asia, collected from a satellite, with some assumptions. One is that each image includes only the airport areas. We are only interested in the main building for the detected buildings, namely, passenger terminal buildings, control towers, cargo buildings, and hangars. Next, each collected image is labeled with the specific building detected using a built-in function of the Visual Object Tagging Tool program.

In technical terms, the concept of the convolutional neural network (CNN) is applied to manipulate the labeled images [25]. CNN has been used in many ways for the detection of interesting objects on images, such as in the region-based convolutional neural network (R-CNN), Faster R-CNN, You Only Look Once (YOLO), and YOLO9000 or YOLOv2

[26, 27, 20, 22]. To achieve our objectives effectively, the concept of Darknet53 in YOLOv3 is utilized to detect the buildings in the labeled images.

Redmon and Farhad have proposed the YOLOv3. It is a method that uses single CNN to predict the boundary of the box and to classify [28]. YOLOv3 is also a regression-based object detection method that transforms the detection problems into regression problems. Because of its high efficiency, which consumes less computational time for an object detection, the regression-based object detection methods are more suitable for our work than those of the region-based methods. In 2019, Zheng et al.[29] improved the structure of YOLOv3 and proposed some applications of the improvement in aircraft recognition through remote sensing images. In addition, the improved method has also been tested by the aircraft industry, with both low and high-quality remote sensing of images. The results have pointed to extremely high accuracy and recall rates of 99.72 % and 98.34 %, respectively. It shows that the improved method is better than the original YOLOv3, even though the images have overexposure and cloud occlusion. Recently, to detect an aircraft in a given place, Sun et al. [30] and also Lilek [31] employed the model based on the CNN method, which has been named as the practical saliency map and makes the background noise decrease significantly.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

As mentioned in this work, we have studied the detection of buildings at the airport area using the remote sensing of images through the concept of YOLOv3, which is a CNN-based object detection method with a saliency map. The study aims to use YOLOv3 to efficiently detect airport buildings through remote sensing images with improved accuracy. Since our data is collected from different remote sensing images, the colors of buildings and background become quite similar. This makes it difficult to separate the buildings from the background visually. To overcome this issue, we need to adjust their colors to be more different before the step of detection by applying the Jet colormap. For instance, for detecting people, Ren et al. [32] has recommended the jet colormap method to encode the raw depth of an image rather than using the gray-scale encoding method directly.

In this thesis, we divide the thesis into five chapters. In Chapter I, we discuss the scope of our study and the details of the materials and methods used. And in Chapter II, the methodology is described in detail, including the basic knowledge used and some evaluation indicators used to investigate our improved method. Chapter III explains the improvement of our work. Chapter IV discusses the accuracy and efficiency of our results through experiments involved with the model. The last Chapter V concludes an overview of the work in this thesis.

1.1 Objective

To detect the building in an airport from remote sensing images using the You Only Look Once deep learning method.

1.2 Scopes and Assumptions

1. The remote sensing images used in this work are acquired 0.6 km, 0.8 km, and 1.0 km above ground. The datasets were collected from 322 Asian airports during the period of May 1, 2019, through January 31, 2020.

2. The whole buildings in Asian airports should be clearly visible from remote sensing images.

3. The resolution of images must be at least 416 × 416 pixels.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

CHAPTER II

BACKGROUND KNOWLEDGE

2.1 Image Processing

Image processing means processing or computerized calculations to get the required information in qualitative and quantitative terms. It is a method to perform some operations on an image, for example, enhancing an image, removing signal noise, extracting the interesting objects from the image in order to analyze for quantitative data such as size, shape, and direction of object movement in the image. Then, we can analyze these quantitative data and create a system to exploit tasks such as fingerprint recognition to determine whose the existing fingerprint image belongs to, the industrial production quality inspection system, the quality sorting system of agricultural crops. The automatic postal code reading system is for sorting out the destination of a large number of daily mail by using an image of the postal code on the envelope. The system stores vehicle information in and out of the building using pictures of license plates for safety purposes. The system monitors road traffic by counting the number of cars on the road in CCTV photos at different intervals. Face recognition systems to monitor terrorists in landmark buildings or immigration and medical imaging analysis. From the various systems mentioned above, it can be seen that image processing requires mass images which are repeated several times. If humans manually analyze these images, it takes a lot of time and labor. The manual analysis may cause fatigue, resulting in a crash in image processing, so computers play an essential role in performing these functions. It is also known that computers can speedily calculate and process large amounts of data, making it extremely useful to optimize image processing and analyze data from images in different systems. Examples of the utilization of image processing as shown in Figures 2.1, 2.2, 2.3, and 2.4.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

Figure 2.1: Teleconference via teleconferencing system using image compression techniques [2]

CHULALONGKORN UNIVERSITY

Figure 2.2: Fingerprint examination using the fingerprint scanning system [3]

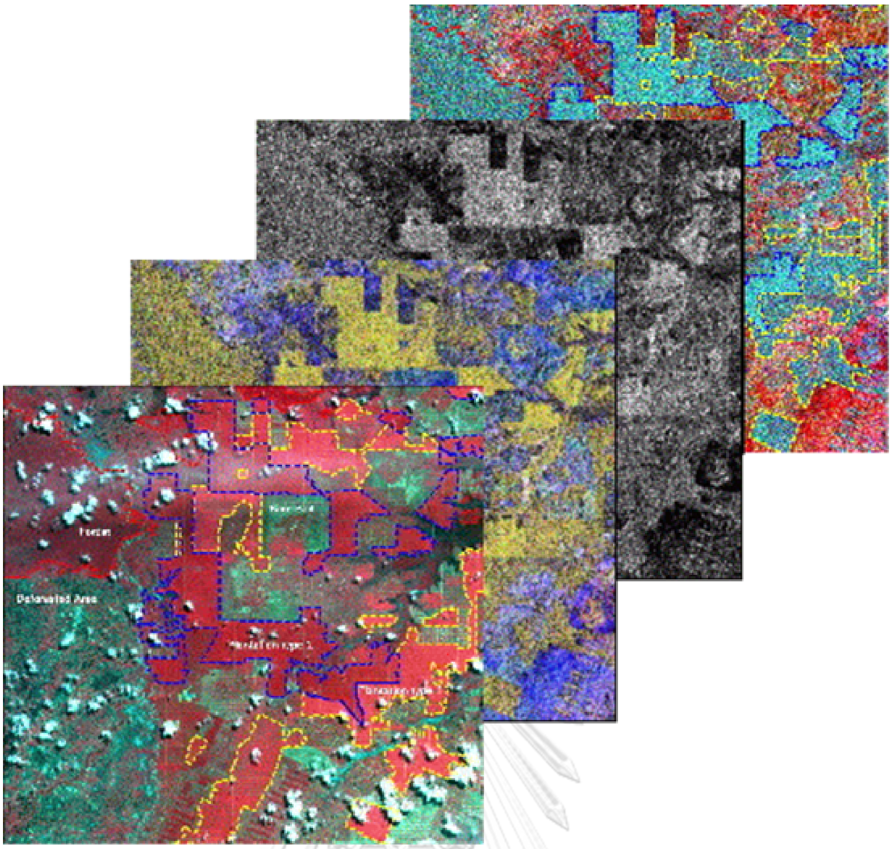


Figure 2.3: Satellite imagery using the principle of image processing [4]

Figure 2.4: Applying rescue robots as accident prevention and first aid provision system [5]

We will explain the basics and details about digital images involved in this thesis in the following sections.

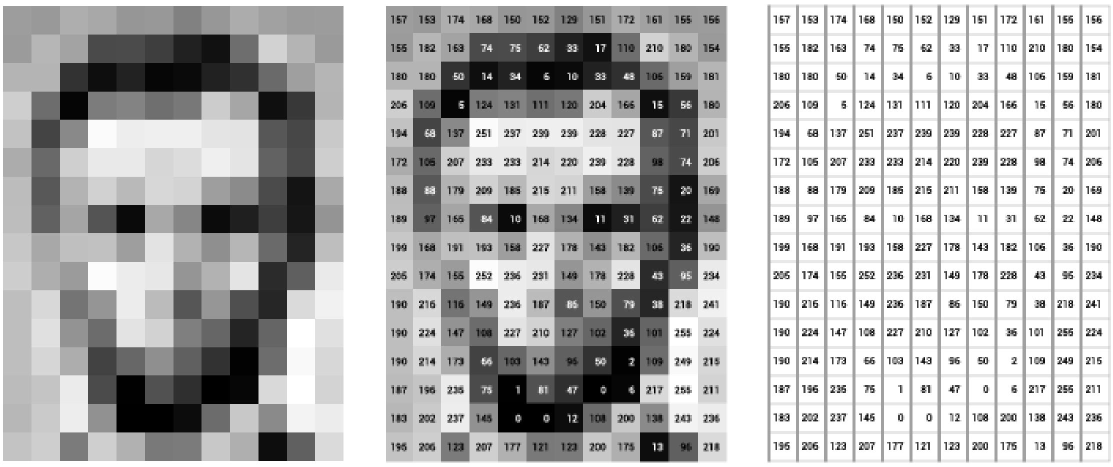
2.1.1 Digital Image Definitions

A digital image is a two-dimensional display of an image in a unit called a pixel. It can be defined as a two-dimensional function $f(x, y)$, where x and y are the image coordinates. The amplitude of f at any (x, y) within the image is the intensity of the image at that position. Provided that x and y are the image coordinates, defined by $x = 1, 2, 3, ..., M - 1$ and $y = 1, 2, 3, ..., N - 1$, and the amplitude of f are finite values, so this image is called a digital image. Let $f(x, y)$ be an $M \times N$ matrix where the origin of the image is at the coordinate $(x, y) = (0, 0)$. Then, $f(x, y)$ can be written in a matrix form as follows

$$f(x,y) = \begin{pmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{pmatrix}.$$

All values in the matrix are referred to as pixels, and pixels begin at $(0, 0)$ in the top left corner of the image. The positions of the pixel are arranged in order from left to right and top to bottom. A bit-mapped image or raster image collects the intensity of a digital image in memory in this way.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Figure 2.5: The example of a grayscale image (left), a grayscale image combined with the intensity arrays (middle), the image intensity arrays (right) [6]

Now that we describe the basics and details about digital images, we'll describe digital images operators related to this thesis in the next section.

2.1.2 Basic Mathematical Tools Used in Image Processing

For image processing tasks, mathematical tools are helpful and essential because they can help in many ways, such as the image enhancement, the noise reduction, and the feature extraction.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.1.2.1 Elementwise Operations

An elementwise operation between one or two images is used to perform actions pixel by pixel. For example, consider the following 3×3 image array:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \text{and} \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}.$$

This formula defines the elementwise $\odot$ product between these two matrices:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \odot \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} & a_{12}b_{12} & a_{13}b_{13} \\ a_{21}b_{21} & a_{22}b_{22} & a_{23}b_{23} \\ a_{31}b_{31} & a_{32}b_{32} & a_{33}b_{33} \end{pmatrix}.$$

2.1.2.2 Arithmetic Operations

Arithmetic operations such as addition, subtraction, multiplication, and division are often applied to tasks involving images. Arithmetic operations between two images, the dimensions $M \times N$, $f(x, y)$, and $g(x, y)$ are expressed as

1. Addition: $A(x, y) = f(x, y) + g(x, y)$. The image enhancement is used to average the image to reduce noise. This type of operation is performed in the image enhancement.

2. Subtraction: $S(x, y) = f(x, y) - g(x, y)$. The image subtraction is commonly used in medical images, especially, removing the background data or increasing the prominence of objects.

3. Multiplication: $M(x, y) = f(x, y) \times g(x, y)$.

4. Division: $D(x, y) = f(x, y) \div g(x, y)$. Both the image multiplication and division are used to correct the grayscale due to varying intensity.

$A(x, y), S(x, y), M(x, y)$ and $D(x, y)$ are images of the same size $M \times N$.

2.1.2.3 Logical Operations

Logical operations like AND, OR, and NOT are frequently used to combine two binary images, each of which has two colors: black (0) and white (1). The logical operation is applied elementwise (bitwise) to integer images. A truth table for AND, OR, and NOT operators are shown in table 2.1. Figures 2.6, 2.7, and 2.8 show the results of using the AND, OR, and NOT operators with a binary image, respectively.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

Table 2.1: The example of a truth table for AND, OR, and NOT operators

A	B	(A) AND (B)	(A) OR (B)	NOT (A)
1	1	1	1	0
1	0	0	1	0
0	1	0	1	1
0	0	0	0	1

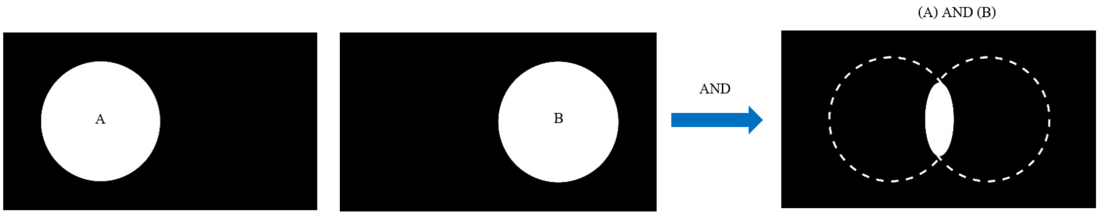


Figure 2.6: The AND operator gives out 1 only if both (A) and (B) are equal to 1

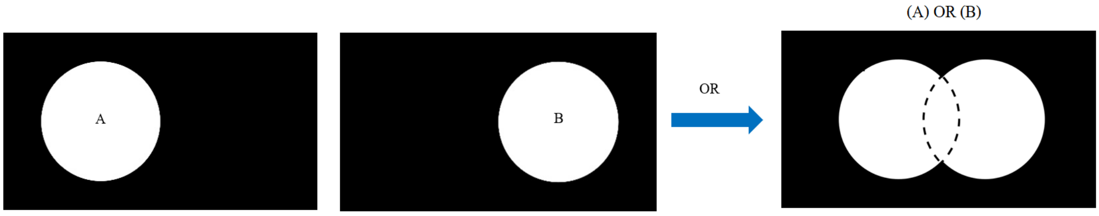


Figure 2.7: The OR operator gives out 1 if (A) or (B) or both equal 1

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

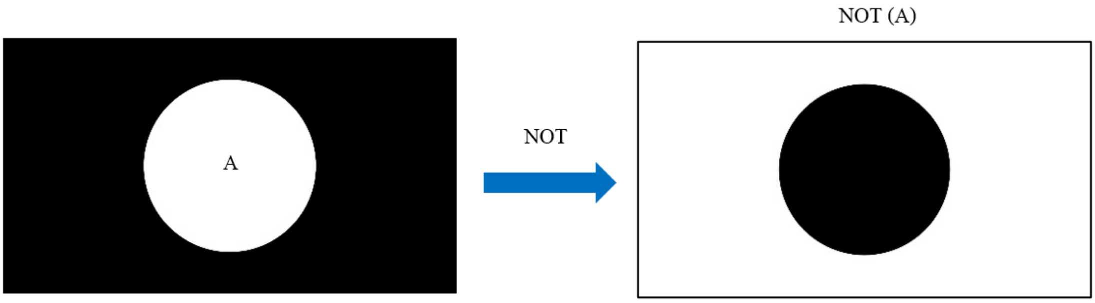


Figure 2.8: The COMPLEMENT (NOT) operator gives out 0 when (A) = 1

After describing the digital images operators, we will describe spatial filtering as necessary in computer vision and essential to this work in the next section.

2.1.3 Spatial Filtering

Spatial filtering is a relatively simple technique and emphasizes low-frequency spatial sharpness and high-frequency spatial sharpness and emphasizes the edge line in numerical image data. Spatial filtering is an average set of points using square grid points, which has dimensions, such as 3×3, 5×5, and 7×7, which always have an odd number of vertical and horizontal points. We will use a square grid to calculate the image pixel mean at the center of the image matrix. The matrix of numbers used to average the values of each pixel image from the neighborhood is called a coefficient to calculate the resulting value of the center image.

First, we multiply the coefficients with the intensity of the image data in the corresponding positions. Then, we combine the results obtained together. Finally, we take all sums divided by the sum of the coefficients. The result is shown in Figure 2.9.

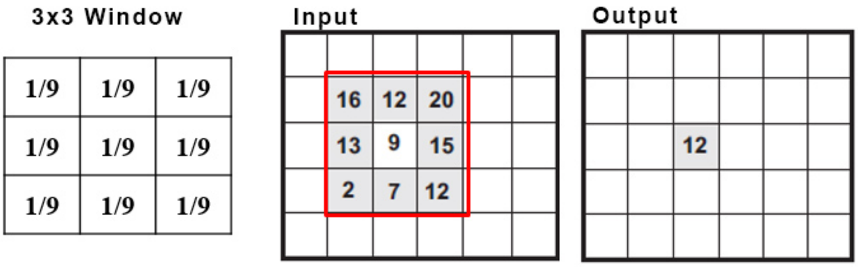


Figure 2.9: Example of spatial filtering

2.1.3.1 Gaussian Filters

The Gaussian filter is a spatial filter with an inverted bell shape extensively used in image processing for smoothing, noise reduction, and computing derivatives.

The one-dimensional gaussian filter equation has the following equation:

$$G(x) = \sqrt{\frac{a}{\pi}} e^{-ax^2}.$$

It can be defined using the deviation parameter with the following equation:

$$G(x) = \frac{e^{-\frac{x^2}{2\sigma^2}}}{\sqrt{2\pi\sigma}},$$

where x is variable value in the x-axis, and σ is a deviation value.

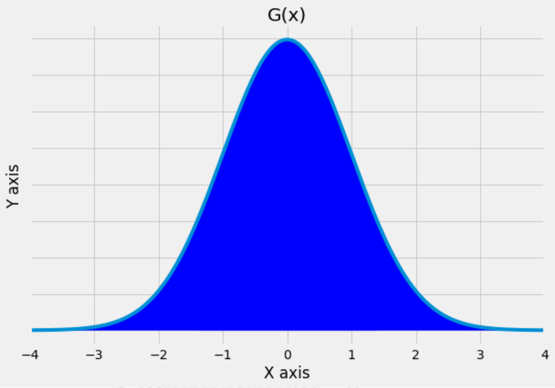


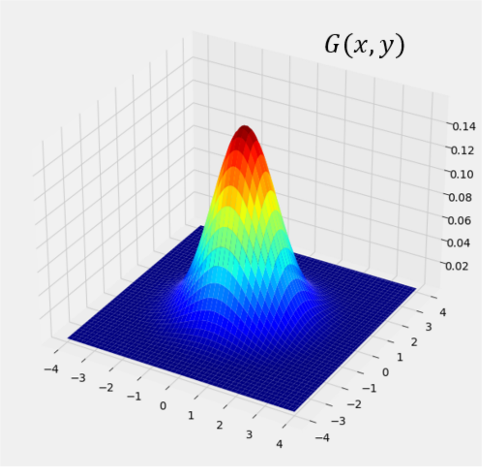
Figure 2.10: The graphical representation of the one-dimensional Gaussian distribution

The two-dimensional Gaussian distribution equation used to create the gaussian kernel has the following equation

$$G(x, y) = Ke^{-\frac{x^2+y^2}{2\sigma^2}},$$

where x is the variable value in the x-axis, y is the variable value in the y-axis, σ is the standard deviation value, and K is the amplitude.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



(a)

$$\frac{1}{4.8976} \times \begin{array}{|c|c|c|} \hline 0.3679 & 0.6065 & 0.3679 \\ \hline 0.6065 & 1 & 0.6065 \\ \hline 0.3679 & 0.6065 & 0.3679 \\ \hline \end{array}$$

(b)

Figure 2.11: (a) The graphical representation of 2-dimensional Gaussian distribution and (b) The example of a 3 × 3 gaussian kernel

Figure 2.11(a) depicts a graphical representation of the gaussian distribution with K = 1 and $\sigma = 1$, and Figure 2.11(b) depicts the example of a 3 × 3 gaussian kernel.

The following section describes the details of color image processing, which is one essential aspect of computer vision, including more information and types of color models.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.1.4 Color Image Processing

Color is one of the essential indicators of image processing. Color is often used to classify objects and to define the class of each pixel in an image. Many researchers use color indicators to achieve their goals. For example, color is used for x-ray imaging or classifying cancer cells and normal cells in medicine. In satellite imagery, color is used to distinguish the class of space taken by satellites, and in engineering, color is used to find fault in circuit boards.

2.1.4.1 Color Models

The color model or the color system describes colors people see in nature by separating them into various elements which can be replaced by numbers that are measured and processed. Normally, the color model can be explained by using the coordinate system in which each color is represented by a single point in the coordinate area. Moreover, this color model is also defined as a standard referred to a particular color, which can be usually seen from the monitor, photos, and prints. This model uses the properties of primary color to generate other different colors. However, there are many mixing criteria of colors. In this research, we use three color models, including, the RGB color model, the HSV color model, and the Grayscale model.

The RGB Color Model

The RGB color model is made up of three primary colors: red, green, and blue. The desired color generation is identified by three-color values mixed in different ratios. Color depends on the intensity of that color. If there is high color intensity when combined, it will produce white, but the low color intensity will make black, so this RGB color model is called positive color mixing. The RGB color model is suitable for display on lighting devices such as monitors, digital cameras, or web images. The RGB color model is shown in Figure 2.12.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

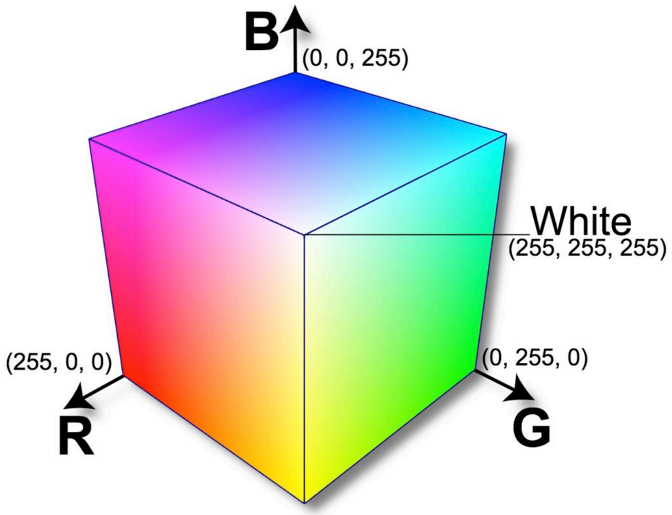


Figure 2.12: The RGB color model on a three-dimensional axis [7]

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

The HSV Color Model

The HSV color model is a pyramid, hexagon, or color cone, converted from an RGB color model that undergoes a non-linear transformation. Therefore, in the HSV color model, hue, saturation, and brightness are used to specify color values, for example, changing a light red to a light green with the same saturation level. All three values must be changed for the RGB color model to produce a light green color, but only the color values are changed in the HSV color model. It shows that the HSV color model is a color model that makes it easier to select colors and differentiate colors.

The HSV color model can be described as an inverted hexagonal pyramid.

The upper surface is a regular hexagon, showing a change in the shade in the Hue (H) direction from 0° to 360°, i.e., the entire spectrum of visible light. The six corners of the hexagon represent the six colors: red, yellow, green, cyan, blue, and magenta.

Saturation (S) is denoted by the direction S from the center to the hexagonal region, and the value varies from 0 to 1. The closer to the hexagonal area, the higher the color saturation. The hexagonal part's color is most saturated, $S = 1$; The color saturation in the hexagon center is 0, i.e., $S = 0$.

The height of the hexagonal pyramid (also known as the core), is denoted by the V value. It demonstrates the gradient of colors from black ($V = 0$) to white ($V = 1$) as bottom to top directions, see Figure 2.13.

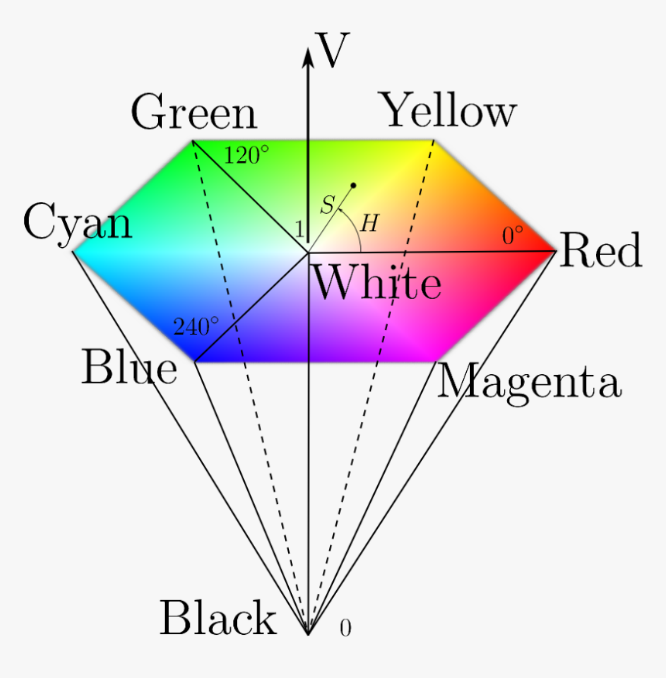


Figure 2.13: The HSV color model on a three-dimensional axis [8]

The Grayscale Model

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

A grayscale image is an image that represents only one shade of gray. The value of each point in the picture is grayscale intensity. They are often stored in 8-bit sizes and offer a range of grayscale from white to black from 0-255. Grayscale images are commonly used in various image processing processes because the grayscale shade is essential and can distinguish its feature. An example of grayscale coloring is shown in Figure 2.14.

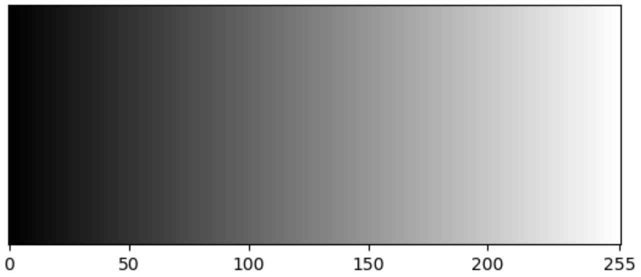


Figure 2.14: The example of a grayscale [9]

There are several approaches to convert an RGB color image into a grayscale image. However, in this research, we use the following equation to convert them.

$$gray(x, y) = 0.299 \cdot r(x, y) + 0.587 \cdot g(x, y) + 0.114 \cdot b(x, y), \tag{2.1}$$

where $r(x, y), g(x, y)$, and $b(x, y)$ is the intensity of red, green, and blue, the model of RGB color images at any pixel (x, y), respectively.

The following section is an essential part of the idea to improve our work efficiency. This section describes the color mapping process, including details and workflows.

2.1.4.2 Color Mapping

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

The color mapping method is intended to change the color of an image or video from one space to another. These approaches have received a lot of attention in recent years, both in the academic literature that provides popularity to study and apply color mapping to various fields. Image processing work applied color mapping with current images or videos such as image data used in medical analysis, weather image data, or landscape image data will transform the image's color into another color space to gain more insights into the images. Color mapping plays a vital role in visualization to increase the effectiveness and efficiency of data and provide more significant insights. However, poor color-mapping selection may provide decreasing insights into the image and also drop the algorithm's performance. Therefore, colormap selection must be suitable for the data or type of the considering image. One of the most popular colormaps uses the Jet

Colormap, which is described as follows.

The color scales in Jet Colormap [33] was designed to be:

1. Many different color tones can be used to render images, as humans can perceive hues more than grayscale, which increases the amount of detail that can be perceived in the image. For example, Pichao Wang and al. published a paper in 2016 that applied jet colormap to simulated a sequence of skeletons to simulate human movement [34], and Lin Li and al. published a paper in 2020 that applied jet colormap to x-ray images of the lungs to detect COVID-19 and pneumonia of patients [35]. Jet colormap shades are shown in Figure 2.15.

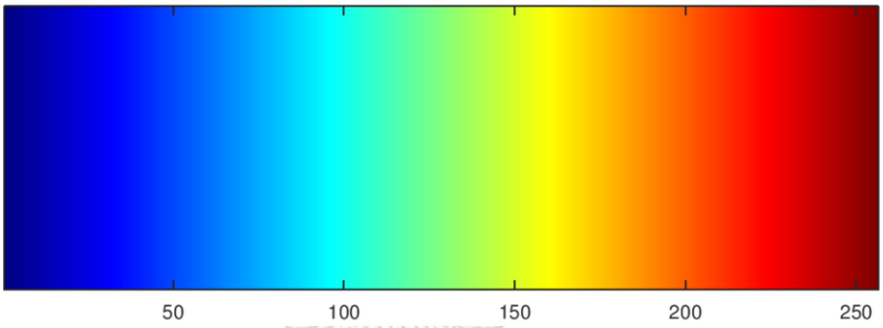


Figure 2.15: The Jet color scales [10]

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2. Obviously, when we use Jet colormap, it is noticed that when the colors of different objects are similar, Jet colormap distinguishes the colors of these two objects clearly, according to the paper of Ellert van der Velden [11], published in 2020.

The statistics and performance of the Jet colormap are depicted in Figure 2.16. The various plots show changes in perceived saturation and lightness in the Jet colormap, including perceptual derivatives and lightness derivatives.

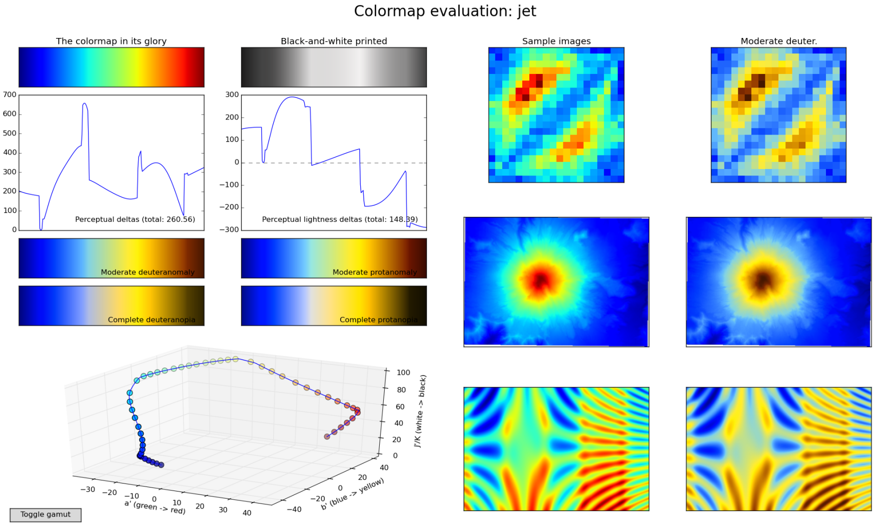


Figure 2.16: The evaluation of Jet colormap [11]

Morphological image processing is another thing that improves our data efficiency and increases its detail. The following section describes the details of each type of morphological image processing quite clearly defined as follows.

2.1.5 Morphological Image Processing

Morphological operations, such as erosion and dilation, are non-linear image processing operations that process images based on shapes. A helpful technique for adjusting each pixel and improving the shape of objects in an image is morphological operation. The structuring element is a small template used to process all possible locations in the image by comparing them to the corresponding neighbor pixels.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Furthermore, in morphological operations, the principle of set reflection and translation is commonly used. $\hat{B}$ denotes the reflection of structuring element B, which is defined as

$$\hat{B} = \{w \mid w = -b, b \text{ for } \in B\}, \tag{2.2}$$

where $\hat{B}$ is a sequence of points in B with their coordinates reversed.

The translation of structuring element B by point z denoted by $(B)z$ defined as

$$B_z = \{c \mid c = b + z, b \text{ for} \in B\}, \tag{2.3}$$

where B_z the set of points in B whose coordinates have been shifted by point z.

2.1.5.1 Erosion and Dilation

Image processing techniques such as erosion and dilation have a widespread presence to improve the image. These techniques are used to reduce background noise and reassemble some of the artifacts that have been separated.

Erosion

Erosion is the corrosion objects in an image, making them smaller by determining their structural element. Then, the input image data is operated by the structural element, which will move to every position to compare with the image data. If they match, the image data is configured to reduce the object's size and eliminate the noise in the image.

It is a binary input image and is a structural element from the Erosion of A and B denoted as $A \ominus B$ defined by

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

$$A \ominus B = \{z \,|B_z \subseteq A\}, \tag{2.4}$$

where B_z is the translation of the structural elements B according to point z.

Figure 2.17 shows examples of Erosion using various structural elements. On the other hand, Figure 2.18 shows the effect of using Erosion to remove image components with a 7 × 7 square structural element.

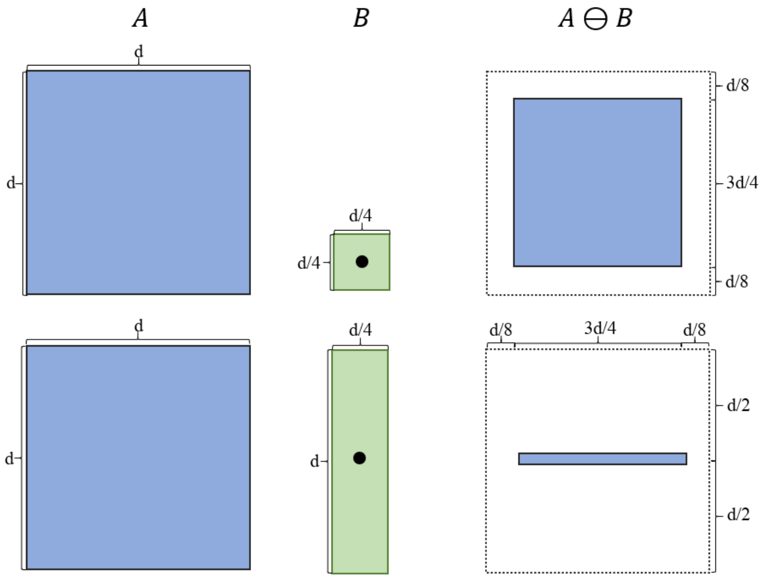


Figure 2.17: The example of erosion using different structural elements

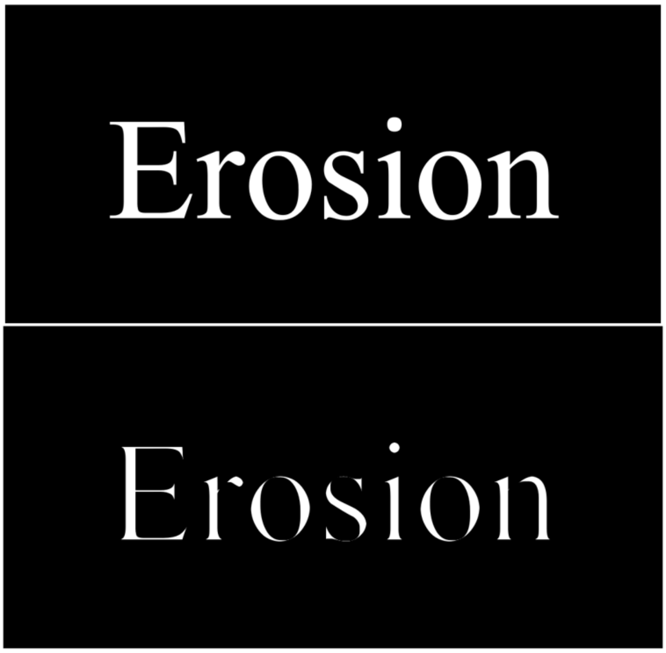


Figure 2.18: The result of using erosion with a structural element of size 7 × 7

Dilation

Dilation aims to expand the image's objects. Structural element determination is similar to erosion. It is taken to indiscriminately equate all image data with image data, scrolling to every position. If they match, the image data is determined by joining two objects close together, joining broken objects, sealing gaps, and removing image noise.

The image gives A is the binary input image and B is the structural element, then dilation A and B denoted as $A \oplus B$ defined by

$$A \oplus B = \left\{ z \,\middle|\, \hat{B}_z \cap A \neq \phi \right\}, \tag{2.5}$$

where $\hat{B}_z$ is the reflection translation of the structural element B according to the z point.

Figure 2.19 shows an example of dilation with different structures, while Figure 2.20 shows the results of using dilation to connect the images component with a 7 × 7 square structural element.

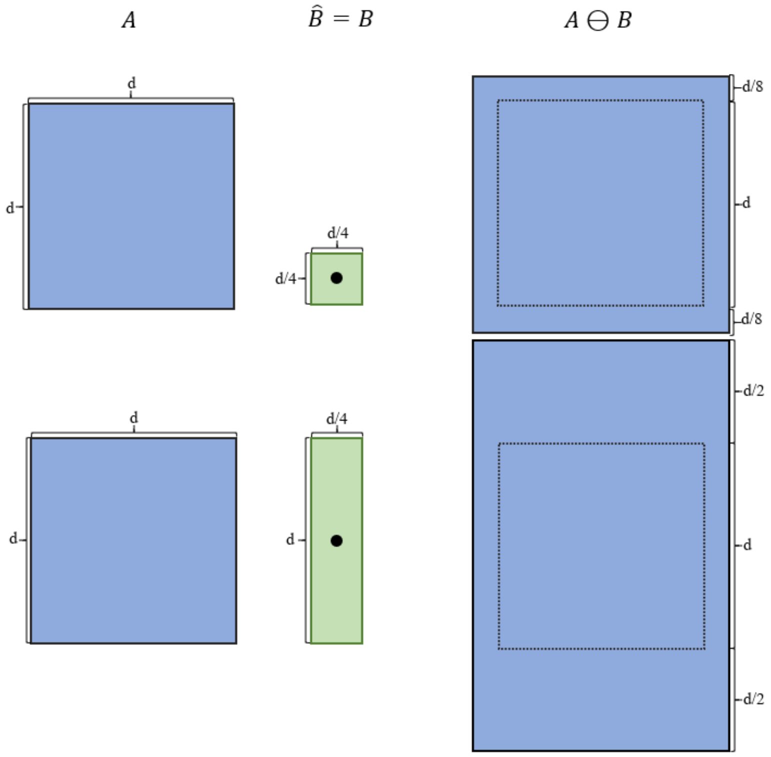


Figure 2.19: The example of dilation using different structural elements

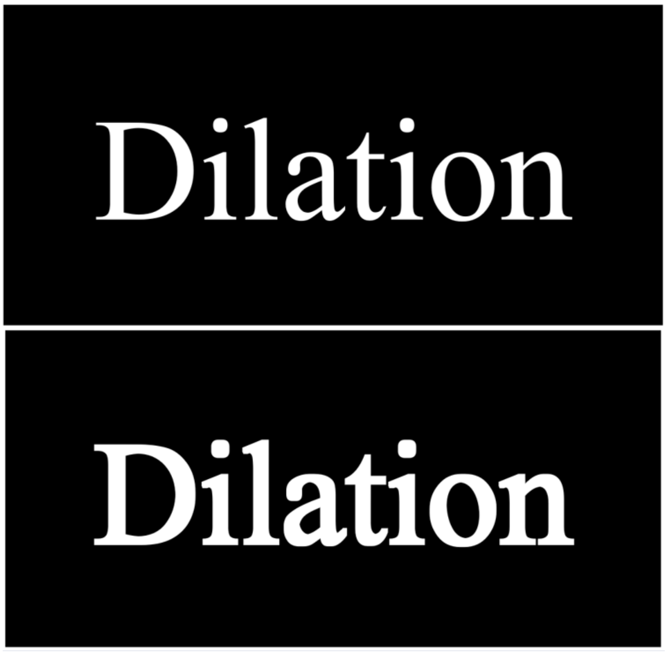


Figure 2.20: The result of using dilation with a structural element of size 7 × 7

2.1.5.2 Opening and Closing

Opening

The opening implements the erosion process followed by the dilation process, generally used to smooth out the object's shape and remove the thin protrusions. It is also used to remove the noise and edge of the object.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

The opening of a binary image A by the structural element B denoted as $A \circ B$ defined by

$$A \circ B = (A \ominus B) \oplus B. \tag{2.6}$$

Therefore, opening A by B is the erosion of A by B followed by dilation by B.

Figure 2.21 shows the results of using a morphological opening operation on a 17×17 rectangular structural element. We can observe that morphological openings can be used to eliminate the noise around the object.

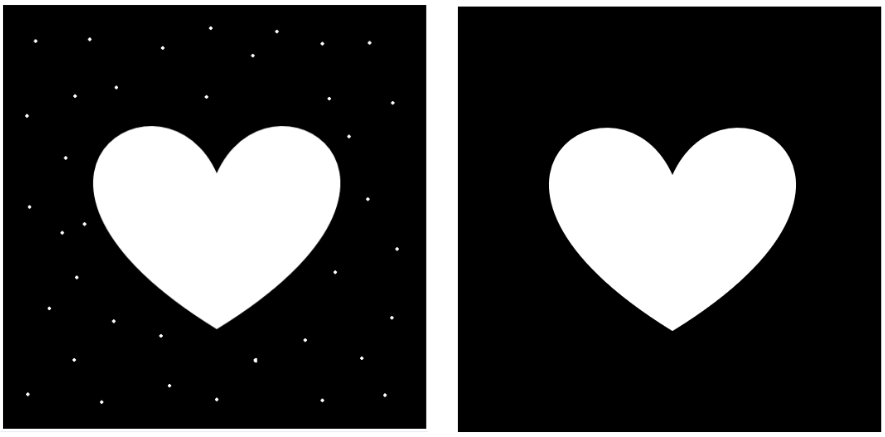


Figure 2.21: The results of using the morphological opening operation with a square structural element of size 17 × 17

Closing

Closing is the implementation of the dilation process, followed by the erosion process, which is used to smooth out the object's contour, eliminate the object's gaps by contouring the small missing objects, and connect separate objects.

Closing of binary image A by structural element B denoted as $A \bullet B$ defined by

$$A \bullet B = (A \oplus B) \ominus B. \tag{2.7}$$

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Therefore, closing of A where B is an dilation of A with B followed by Erosion B.

Figure 2.22 shows the results of using a morphological closing operation on a 17×17 rectangular structural element. We can observe that morphological closing can be used to fill holes within the object of interest.

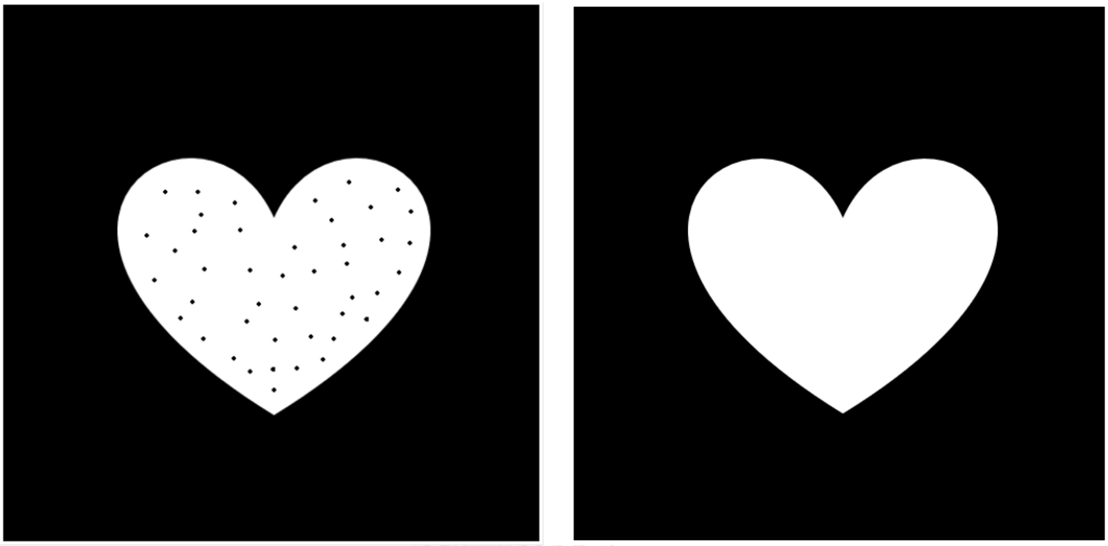


Figure 2.22: The results of using the morphological closing operation with a square structural element of size 17 × 17

A computer vision work that we do in this work, most of the data we use is image data. The processes indispensable for computer vision work are binary images and processing to convert color images to binary images. This process helps us analyze and improve image data. In the next section, we will explain the detail of the binary image and processing to convert color images to binary images.

2.1.6 Binary image

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

A binary image is a monochrome image with an intensity of 0 (black) and 1 (white) at each pixel only. This binary image is a fundamental principle of image processing and is often used for image segmentation, edge detection or thresholding, etc.

2.1.6.1 Thresholding

Thresholding is a technique for splitting an image into two halves, separating the foreground from the background. Choosing a threshold T to divide each pixel into one or two levels for any point (x, y) in the image is the primary method of separating objects and background. $g(x, y)$ represents a thresholding image, which is defined as

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases},$$

where $f(x, y)$ is the image intensity at (x, y) coordinate, and T is the appropriate constant for the entire image.

Otsu's method

Otsu's method, which uses grayscale variance to determine the threshold value, was first introduced in 1979. According to Otsu's method, the object and the background are two distinct groups of data in an image that can be separated. Both data sets can be classified with just one threshold if the object and background are entirely separated, and the variance between the group is low. However, for any data, the grayscale values within the same group have low variance. For example, we want to distinguish the car (object) from the road (background) by assuming that their grayscale values of two groups are different. As a result, the grayscale of the group of the car should be spread over the same range, resulting in a low variance grayscale of the car group, which will be the same in the group of roads. That means that if we combine the road pixels with the car pixels will make the variance higher. As a result, if we can correctly separate the data groups so the variance of the two groups will be the lowest.

CHULALONGKORN UNIVERSITY

The Intra-Class Variance as the weighted sum of the variances of two groups can be calculated as follows

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t),$$

where the probabilities of two classes separated by threshold t are ω_0 and ω_1, and the variances of these two classes are σ_0^2 and σ_1^2.

The class probabilities $\omega_0(t)$ and $\omega_1(t)$ are calculated using the histogram of L bins

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i),$$
$$\omega_1(t) = \sum_{i=t}^{L-1} p(i).$$

Minimizing intra-class variance is equivalent to maximize inter-class variance for two classes:

$$\begin{aligned}\sigma_b^2(t) &= \sigma^2 - \sigma_w^2(t)\\ &= \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2\\ &= \omega_0(t)\omega_1(t)\left[\mu_0(t) - \mu_1(t)\right]^2\end{aligned}$$

which can be expressed in terms of class probabilities ω and class mean μ, where $\mu_0(t)$, $\mu_1(t)$, and μ_T are the class means:

$$\mu_0(t) = \frac{\sum_{i=0}^{T-1} ip(i)}{\omega_0(t)},$$
$$\mu_1(t) = \frac{\sum_{i=t}^{L-1} ip(i)}{\omega_1(t)},$$
$$\mu_T = \sum_{i=0}^{L-1} ip(i).$$

The following relationships are simple to verify:

$$\omega_0\mu_0 + \omega_1\mu_1 = \mu_T,$$
$$\omega_0 + \omega_1 = 1.$$

It is possible to compute the class probabilities and means iteratively. This concept results

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

in a useful algorithm.

Artificial Intelligence (AI) is a program built and developed to be intelligent, think, analyze, plan, and make decisions from the processing of large databases and modify the processing and application according to different situations. The types and details related to AI are described in the following sections.

2.2 Machine Learning

Machine learning is the study of computer algorithms that improve automatically by learning from a large amount of data that computers observe previous data patterns and make decisions to predict outcomes or answer questions that are likely to occur, which can be set without the data analyst. Machine learning is a different approach to artificial intelligence (AI). For AI, developers create step-by-step rules for computers in order to work following a set of instructions. On the other hand, machine learning allows computers to learn from information and improve problem-solving efficiency intelligently. For example, an intelligent autonomous vehicle technology-driven driver, Siri voice recognition technology is that users can use voice commands to request or ask for help promptly, etc. Machine learning is categorized into five main categories as follow

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.2.1 Supervised learning

Supervised learning is the machine learning from labeled sample data in order to construct the algorithms or models for predicting the results and understand the relation between input and output data. This learning model can be applied to new input data, and supervised learning can assess the accuracy or error of the results.

2.2.2 Unsupervised learning

Unsupervised learning is the machine learning that uses for unlabeled sample data in order to search results of the input data, simulate the basic model underlying infrastructure in the input data. Unsupervised learning contrasts with supervised learning because the computer does not know the value or type of the data and tries to identify a group of data.

In the case of supervised learning, labeled data is collected and used for grouping or classifying. However, the case of unsupervised learning is to group or classify the unlabeled data into meaningful groups without any known results of that data. The results of the supervised learning were in classification and regression models, and the outcome of the unsupervised learning was a clustering, as shown in Figure 2.23.

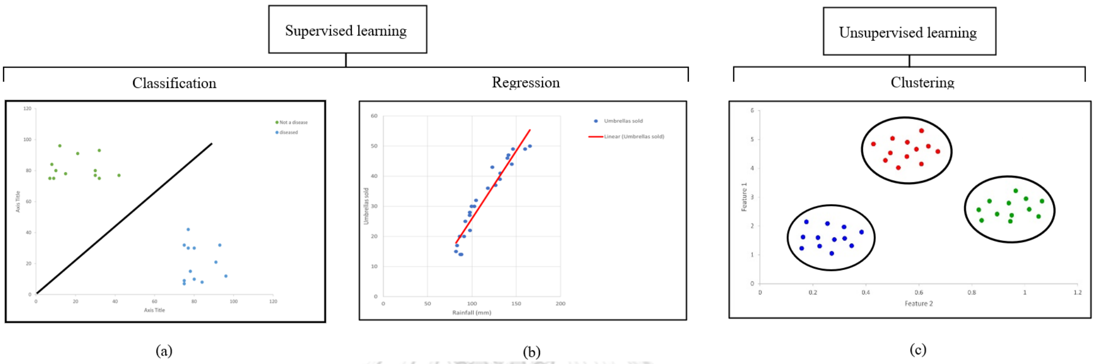


Figure 2.23: Examples of real-life problems of outcomes from supervised learning and unsupervised learning (a) Patient classification of disease (b) House price prediction by a regression model (c) Clustering of shopping behaviour customer

2.2.3 Semi-supervised learning

Semi-supervised learning is blended learning between supervised learning and unsupervised learning using two types of data: labeled and unlabeled. The number of unlabeled data is greater than the number of labeled data. Then, these data are used via the machine learning to create algorithms or models for predicting results.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.2.4 Reinforcement learning

Reinforcement learning is a type of targeted learning in which a machine learns from its surroundings in various ways, such as input data or sensors, etc. In reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training data, it is bound to learn from its experience. For example, in games, players know when they will win or lose, but they have no idea how to play. Each step of this reinforcement learning is learned through experience and the environment. From the steps mentioned, the players can anticipate the game's stages.

2.2.5 Transfer learning

Transfer learning is a technique that reduces training time for deep learning models by incorporating parts of a trained model with similar tasks as part of the new model. In other words, the transfer learning is commonly used in deep understanding, as the pre-trained model is used as a starting point for computer vision and computing natural language processing (NLP). The pre-trained model can help reduce computations and save time in developing neural networks, because it can be initiated from the model learned through the transfer learning process to solve similar problems of previous work as shown in Figure 2.24.

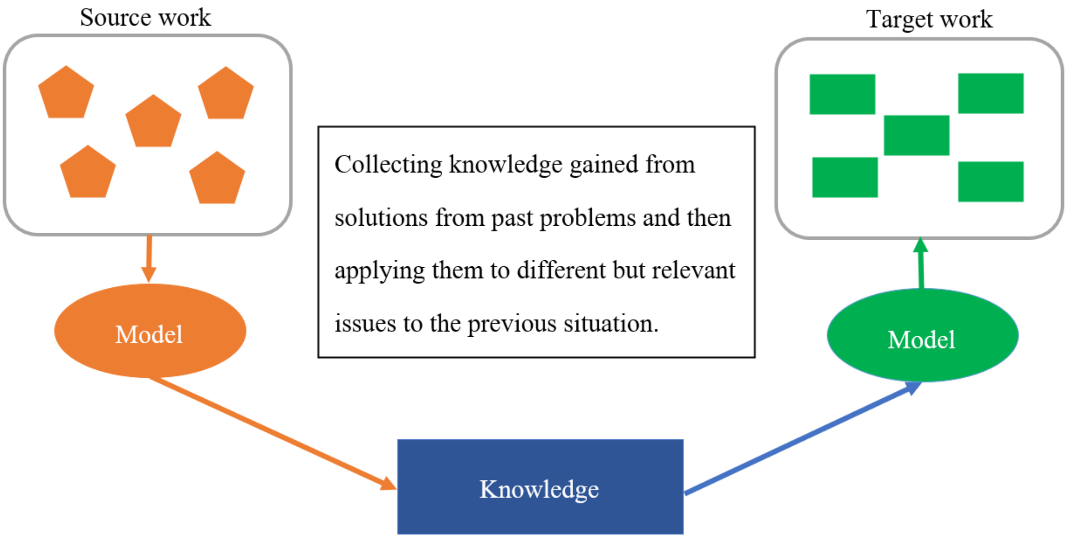

Source work
Target work
Collecting knowledge gained from solutions from past problems and then applying them to different but relevant issues to the previous situation.
Model
Model
Knowledge

Figure 2.24: Applying learning knowledge previously to new models

2.3 Deep Learning

Deep learning is a branch of machine learning where each step is subdivided into individual layers within the deep learning model. Each layer is created from the result of the previous layer. Then, all layers are combined into a neural network. This is similar to how the solution is distributed by neurons in the human brain. Thus, the deep learning algorithm is similar to the work of neural structures, where each neuron is connected to the other and included the transmission of information between nerve cells.

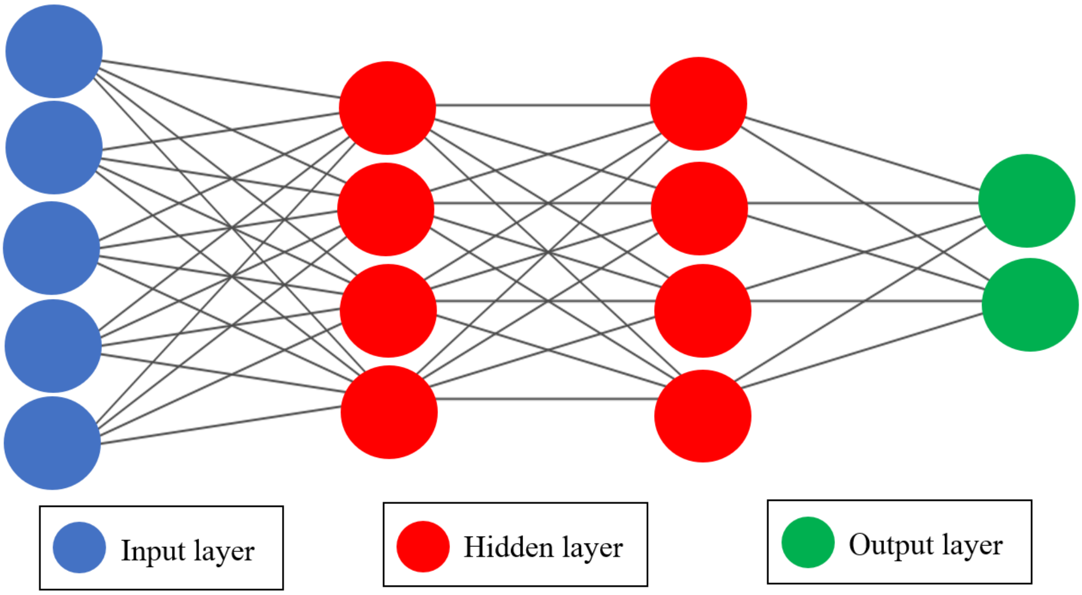


Figure 2.25: An example of the layer structure of a deep learning model

The deep learning model works in consecutive layers as shown in Figure 2.25. Generally, the deep learning model has at least three layers, each of which receives data from the previous layer and forwards it to the next layer. Additionally, the performance of the deep learning model tends still to benefit from large amounts of data, while the performance increase of machine learning is hardly ever changing. The difference between machine learning and deep learning lies in the feature extraction of data, as shown in Figure 2.26. In machine learning, the human being is extracting the features of the data and the features passed into the model for results. However, in deep learning, the model extracts features of data by itself. There are theories involved which form the basis of deep learning as follows

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

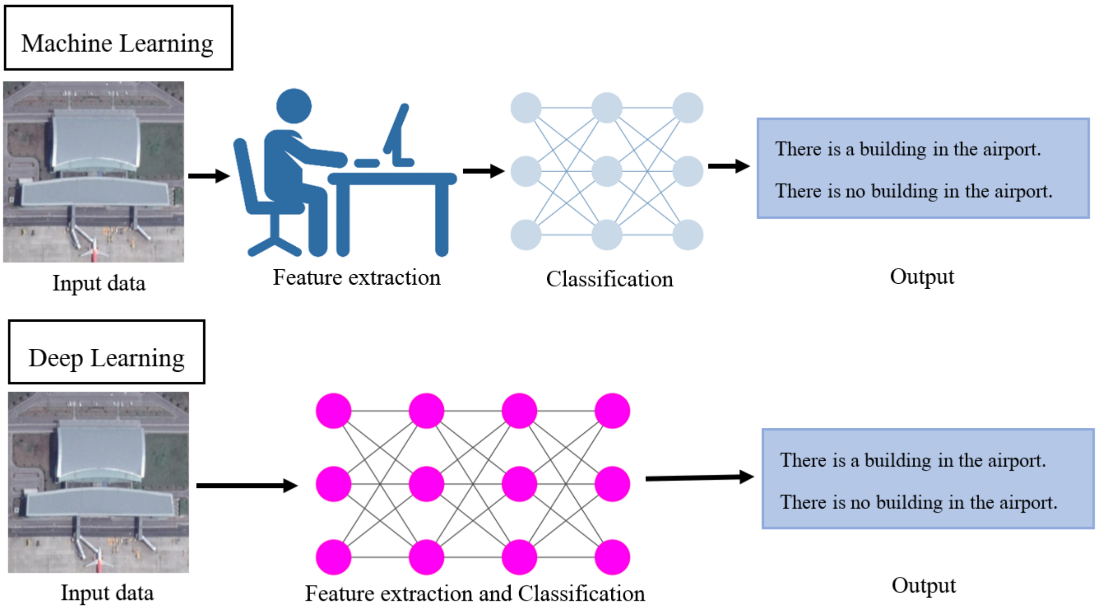

Machine Learning
Input data
Feature extraction
Classification
There is a building in the airport.
There is no building in the airport.
Output
Deep Learning
Input data
Feature extraction and Classification
There is a building in the airport.
There is no building in the airport.
Output

Figure 2.26: Comparison of data feature extraction differences between machine learning and deep learning

2.3.1 Linear regression

Linear regression is a linear relationship between two or more variables: the predictor variable x and the response variable y. For example in one-dimensional data, if we have more enough data, we can use the linear regression to find the relation of these data, i.e., let x and y be correlated multiple times to find the correlation equation, where the simple linear regression equation is expressed in equation (2.8).

$$y = ax + b, \tag{2.8}$$

where x is the predictor variable, y is the response variable, a is the slope, and b is the Y-intercept.

For example, we predict home prices based on house size using the simple linear regression as shown in Figure 2.27.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

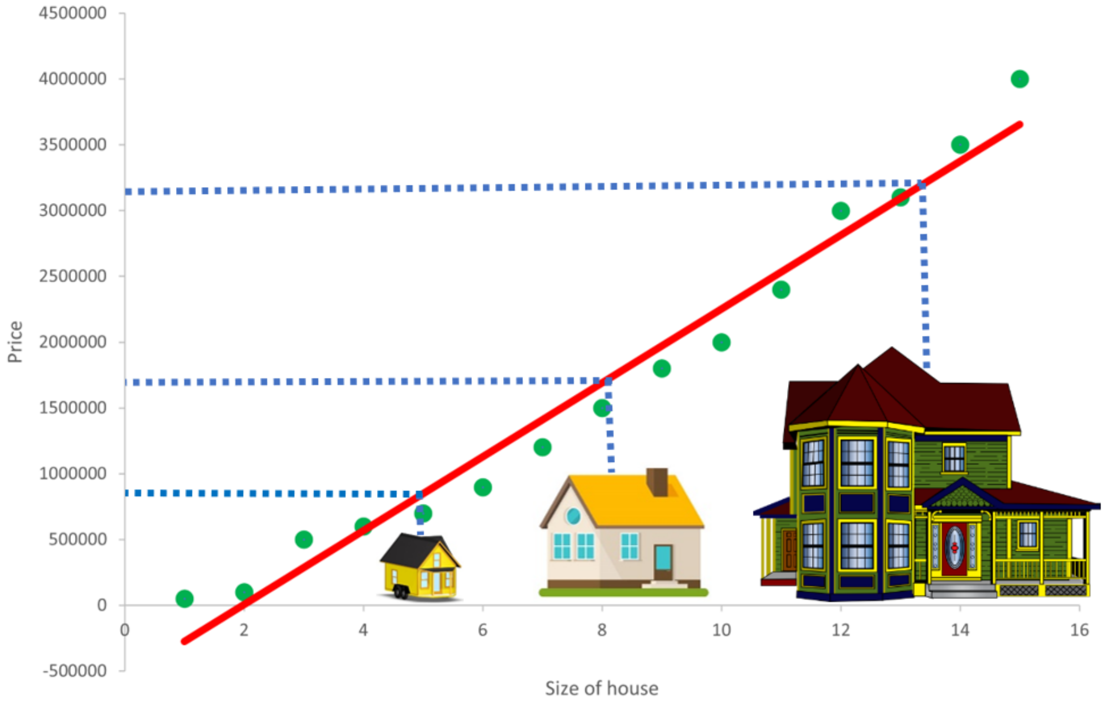


Figure 2.27: An example shows the application of linear regression to estimate the best house price.

We use a simple linear regression to find the line between points marked on a graph with low errors, and when the line has low errors, it is possible to predict house prices by the size of the house.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.3.2 Logistic regression

Logistic regression is a statistical method for analyzing data sets in which at least one independent variable determines the outcome. There are only two possible outcomes: true or false. For example, we have the data in terms of midterm and final scores of students. If we would like to know whether these students will pass and fail, the logistic regression can be applied to find the straight line that separates these two types of data (pass and fail) from each other as shown in Figure 2.28.

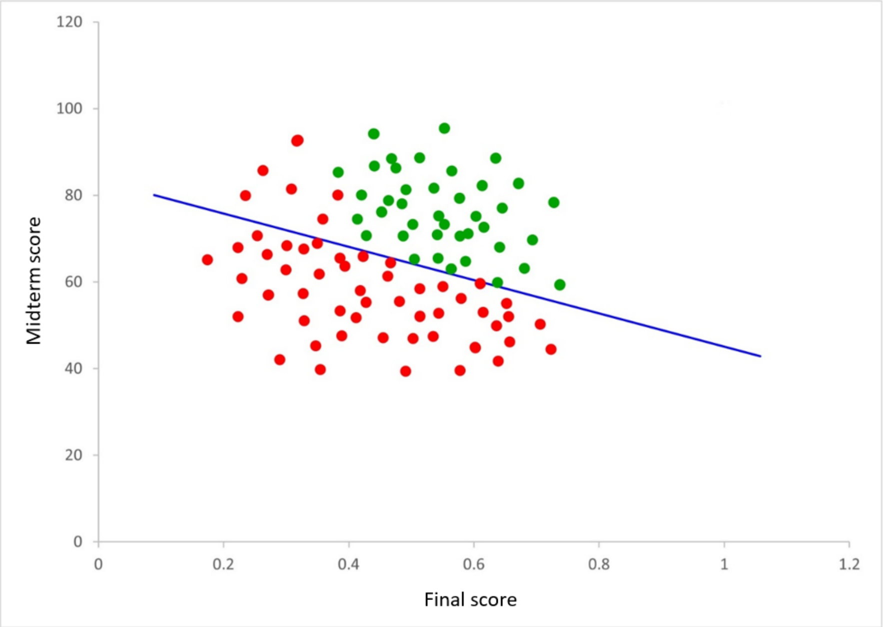


Figure 2.28: An example shows logistic regression determining students who pass and fail from midterm and final scores

2.3.3 Activation function

The activation function is a function that decides and defines a single standardized output for each node of a neuron and helps make the output more efficient, minimizing the possibility of the output variable. For example, the heaviside step function is an example of the most straightforward trigger function, which the function returns 0 when the linear sum is less than 0 and will return the value of 1 when the linear sum is a positive number or equal to zero, which is calculated from Equation (2.9). Figure 2.29 shows an example of the heaviside step function.

$$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases} . \tag{2.9}$$

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

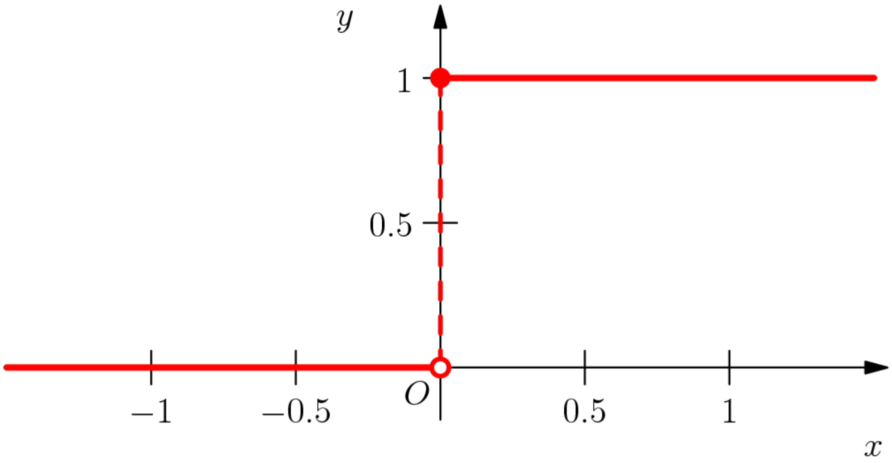


Figure 2.29: An example of the heaviside step function [12]

2.3.4 Weight

Weight is a value associated with each connection between neurons, indicating the importance of the input data. When an input data enters the neuron, it is multiplied by the weight value assigned to the data. Usually, the initial weight value is set randomly, and when the neurons begin to learn, the weight value is adjusted to obtain the result closest to the desired answer. For example, the university has criteria for testing students using test scores and grades, so this neuron has two associated weights and can be adjusted for each weight value. If the obtained weight values produce the accurate result that is very close to the desired result, the weight values will not be adjusted. But if the efficiency is low, the weight value is adjusted through a specific calculation. The examples of weights are shown in Figure 2.30.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

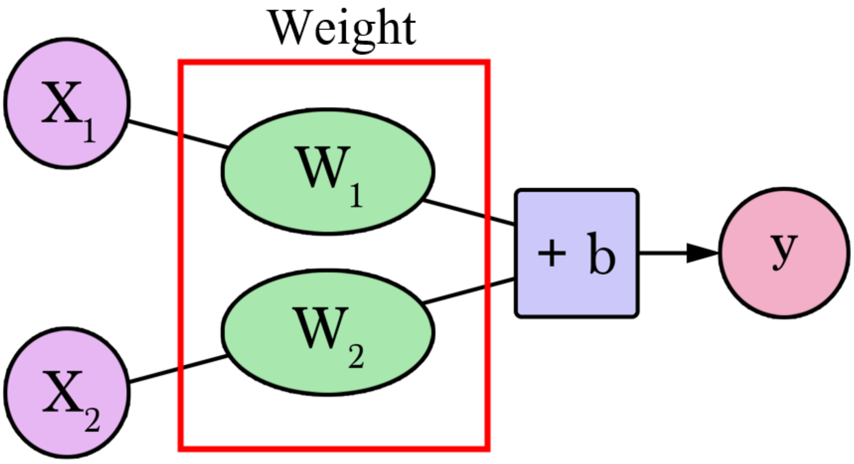


Figure 2.30: Examples of the weights in deep learning

2.3.5 Bias

Bias is a value used to shift the activation function by adding a constant to the input data. Moreover, the bias prevents the neurons from sending inputs with no value (equal to 0) to the output layer. The bias value is always adjusted while the data is trained as well as the weight value.

2.3.6 Backpropagation

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Backpropagation is the process of adjusting the weights of a network in order to train that network to be suitable for the data. It is a basic component of a network. This is accomplished by feeding data into the network, comparing the output to the ground truth, and calculating an error rate using a specialized loss function. This error rate is then cascaded backward through the network to modify the weights so that the prediction is closer to the ground truth the next time the same or similar input is passed through the network.

2.4 Artificial Neural Network

An artificial neural network (ANN) is a mathematical model that simulates the function of biological nerve cells called neurons in the human brain. The dendrite, cell body, and axon are three main components of a neuron, as shown in Figure 2.31. Receiving a signal through the dendrite, transforming a signal through the cell body, and sending the transformed signal via the axon is how a neuron sends a signal to other neurons.

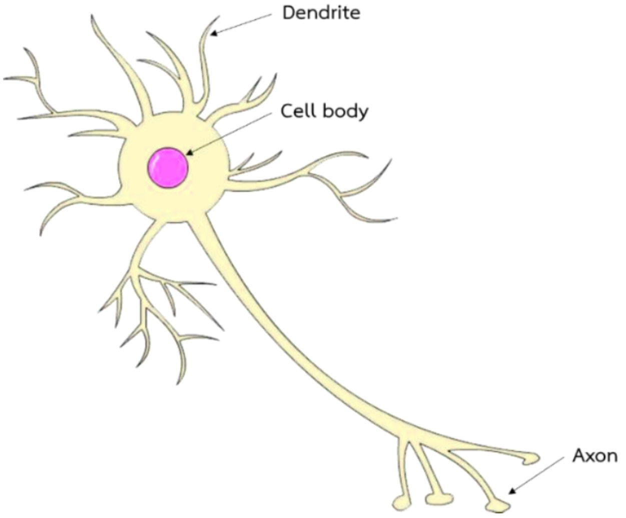


Figure 2.31: The biological neuron [13]

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

The input layer, the hidden layer, and the output layer are three main layers in ANN. Each layer of ANN is made up of multiple artificial neurons or nodes, which are simple processing units. Artificial neurons attempt to mimic the structure and behavior of biological neurons by performing a dot product between input values and weights, then adding a bias and using the activation function to transform these values into results, as shown in Figure 2.32.

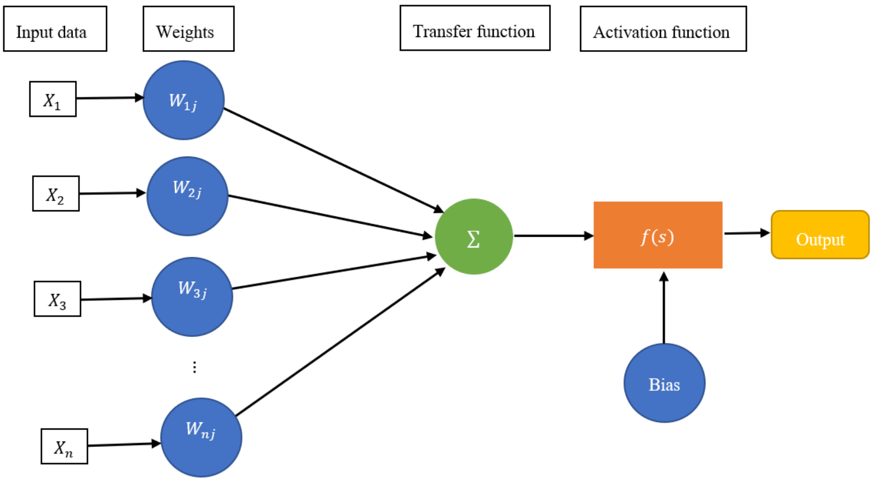


Figure 2.32: The visualization of an artificial neuron in the hidden layer

The operation of the artificial neural network can be seen in Figure 2.32. The artificial neuron receives the input value, weighted the input value, and used the transfer function to combine the weighted input value and the activation function to transform the weighted input value and send it to the next artificial neuron. We use the activation function to avoid 0, which the activation function uses thresholds to transform the sum of the weighted input value, and bias is also added to the sum of it.

The output of an artificial neuron in a hidden layer is defined by equation (2.10) given the number of input is N, The input values are $X = (x_1, x_2, x_3, ..., x_N)$, The weight values are $W = (w_1, w_2, w_3, ..., w_N)$, The bias is b, and The activation function is φ

$$y = \varphi\left(\sum_{i=1}^{N} w_i x_i + b\right). \tag{2.10}$$

In general, the output layers of artificial neurons do not use an activation function because the final output layer is frequently used to represent class scores and predict the class of input data.

The main types of artificial neural networks are divided into two types as follows

1) Feedforward is the simplest type of a neural network. The connections between the layers are non-loopable. The input is directed to the output in one direction through the weight values of each neural.

2) Feedback is a loopback connection. The results obtained from the network are re-circulated to improve the efficiency of the results. This feedback network can be complex, but it is more efficient than the forward feed type, which is commonly used in various applications.

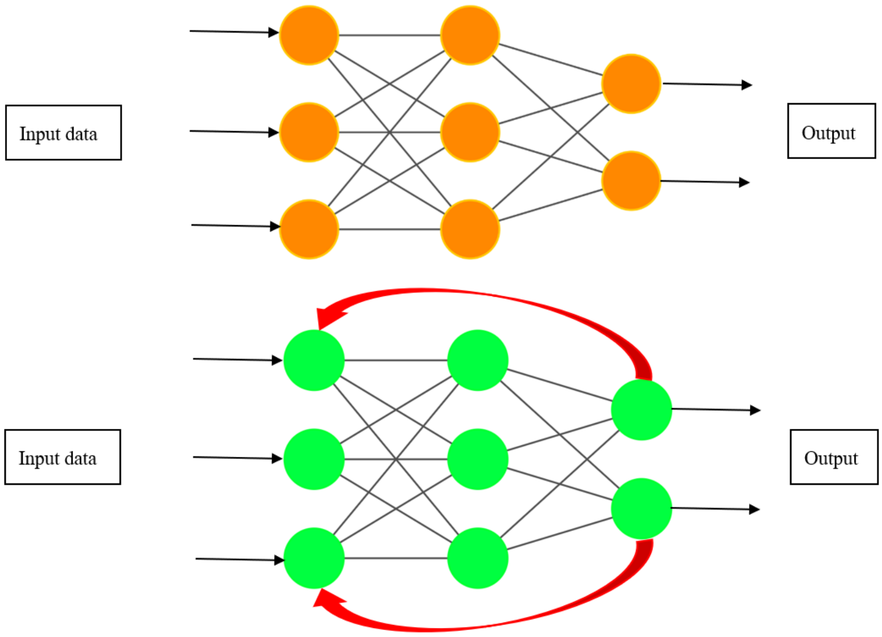


Figure 2.33: Type of artificial neural network (a) Feedforward (b) Feedback

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.5 Convolutional Neural Networks

There are specific types of artificial neural networks that are widely used is the convolutional neural network. The convolutional neural network is similar to artificial neural networks. However, the structure is specifically designed for matrices as input. The layers of conventional neural networks contain neurons arranged in three dimensions: width, height, and depth, which are suitable for spatial data. Usually, the convolutional neural network is applied with object recognition, object detection, object classification, and so on.

The convolutional neural network is used to process input image data through the convolutional layer. Images from an RGB model separated into three primary colors are red, green, and blue. In this separation of this type of the color model, depth is added to the image data, causing the image data from the RGB model to be 3D image data. The image data is then processed with a filter to extract the feature of image data by the convolutional layer, and this step enables the network to detect edges and low-level features in previous layers and detect complex features of deeper layers.

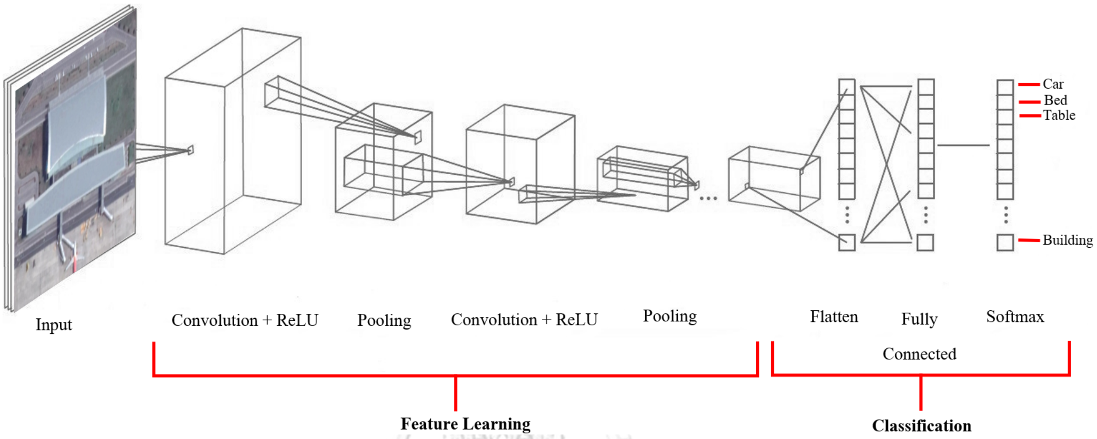


Figure 2.34: Examples of convolutional neural network structures [14]

Basically, the convolutional neural network structure is divided into two parts, including, feature learning and classification as shown in Figure 2.34. Feature learning is the part of adjusting the data by learning the features of input data through many layers which mainly consist of convolution layer, activation layer (ReLU), and pooling layer, respectively. The part of classification consists of fully connected layers which is the layer that converts the results of the convolution and generates the final output of the convolutional neural network. However, the details of those mentioned layers will be explained in the next section.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.5.1 Input layer

An input layer is often used for image data, where every image is a matrix of pixel values, with the range of pixel values that can be encoded at each pixel depending on the bit size of that image. Therefore, the possible range of pixel values can be represented from values $0-255$. The color image in the RGB model is separated into three channels of color, where each channel has added a depth of data. As a result, it becomes the 3D input data, consisting of width × height × depth of each color channel in the image. For example, for a 255×255 (width × height) RGB image, the input layer has three matrices associated with the image, each matrix representing each color channel and consists of a 3D structure called the input volume, which is $255 \times 255 \times 3$

Filters or kernels are represented by weight vectors that are used to join with input data. We can add many filters to increase the number of feature maps extracted from the image. Each of the feature maps that we extract will try to learn different properties of the image, such as edge or color pixel, etc.

2.5.2 Convolution layer

The convolution layer consists of a learnable filter, where filters are made up of matrix width, length, and depth. While the filter slides through the image matrix, they have sought the convoluted values by multiplying the filter matrix with the image matrix at each corresponding point (x, y). The basic concept of convolution comes from the fourier transform. For kernel of W size $m \times n$, assume that $m = 2a + 1$ and $n = 2b + 1$, where a and b are non-negative integers. Given $f(x, y)$ is the value of the array at point (x, y), the convolution of W and f denoted as $h(x, y)$, is defined by the following equation:

$$h(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} W(s,t) f(s+x, t+y). \tag{2.11}$$

An example of finding the convolution $h(x, y)$ is depicted in Figure 2.35.

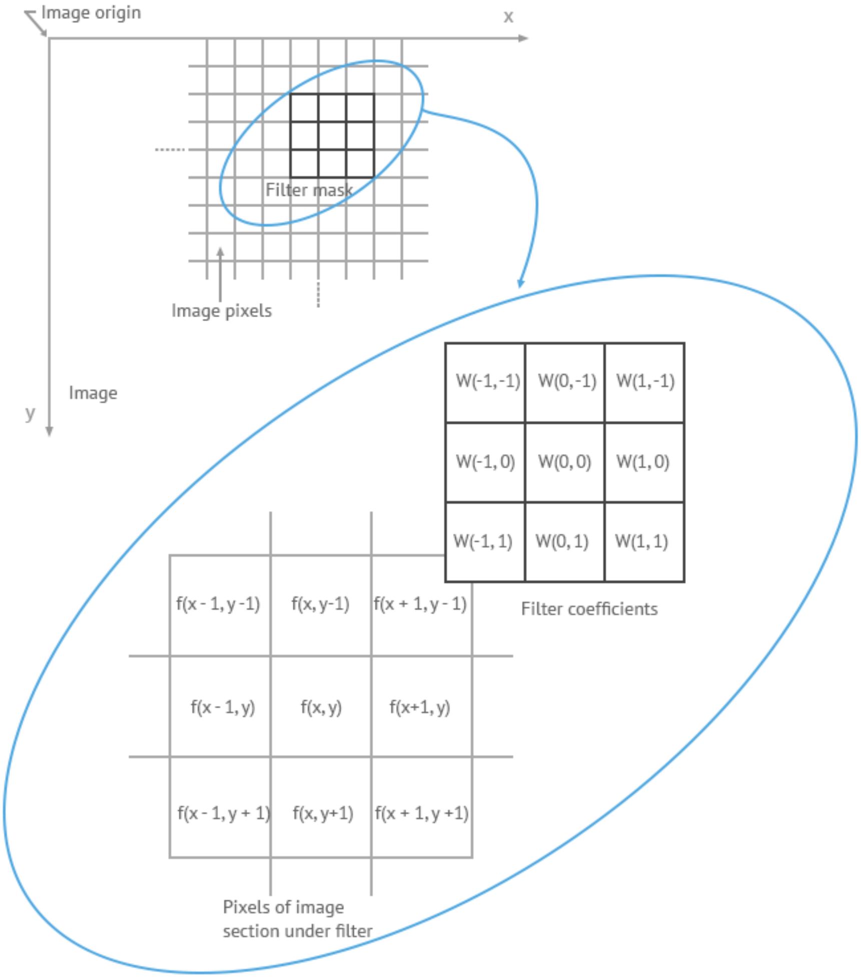


Figure 2.35: The example of a convolutional operation with kernel of size 3×3

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Let us introduce the stride value, it means the pixel number that the filter is slid as that number. The kernel slide with a stride value from top left to top right until the entire width. Then, with the same stride value, it slide down and slide from left to right, and continues the procedure until the whole image matrix. The movement of the kernel is displayed in Figure 2.36.

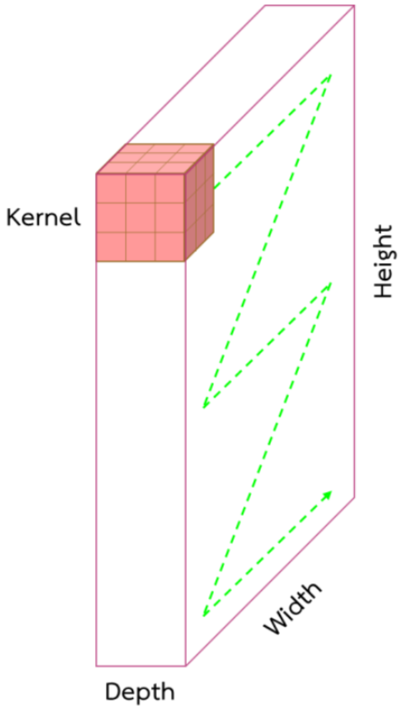


Figure 2.36: The movement of the kernel

2.5.3 Activation layer

The activation layer is a function that transforms input data and determines its output value by a threshold, and sent it to the next node. This process is called the activation function. A popular activation function in a convolutional neural network is the Rectified Linear Unit (ReLU) which reduce the complication of input data, resulting in a faster and more efficient network learning by changing the negative value from the input to 0. The ReLU is shown in Figure 2.37 and it can be expressed by equation (2.12).

$$R(z) = \max(0, z), \tag{2.12}$$

where $R(1) = \max(0, 1) = 1$, and $R(-1) = \max(0, -1) = 0$.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

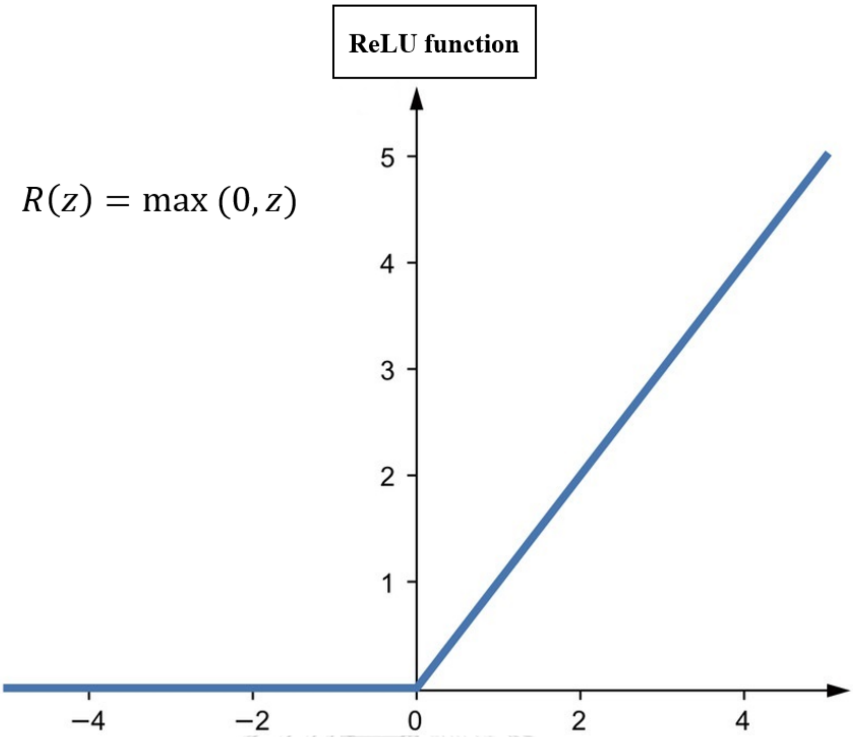


Figure 2.37: The rectified linear activation function

Additionally, the softmax function is a logistical regression function used for a multi-layer classification. The softmax function is used in the last layer of a network which takes in real values of different classes and returns a probability distribution and it can be written as follows (2.13).

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

$$f_j(z) = \frac{e^{z_j}}{\sum_{i=1}^{K} e^{z_i}}. \tag{2.13}$$

2.5.4 Pooling layer

Pooling layers is used to reduce the size of feature map, while its depth remains the same. Most of the popular pooling methods consists of two types: max pooling and average pooling. Pooling layers are performed on each layer of feature map by using the mathematical operations of each pooling type, as demonstrated example of the max and average poolings in Figure 2.38. In addition, the pooling layers help to simplify the feature map and reduce the number of feature that the network learns.

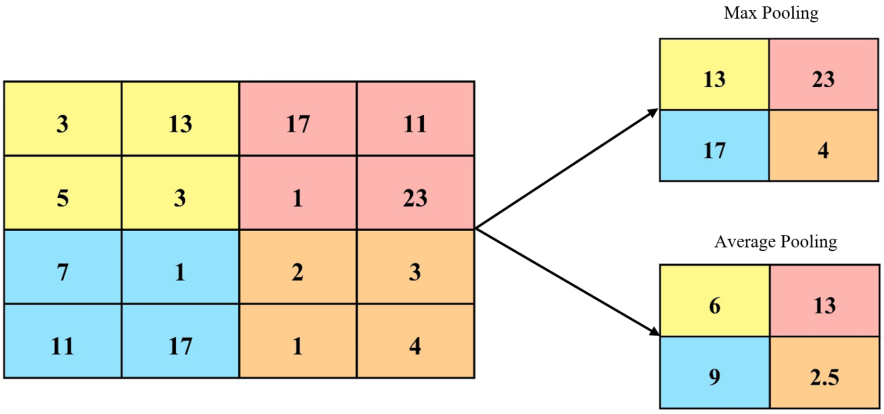


Figure 2.38: The result using average pooling and max pooling with size 2×2

2.5.5 Fully connected layer

The fully connected layer is the last layer to perform by using the feature map that is previous performed in the part of feature learning. It is employed to predict types of the data. This layer is connected to the all neurons in previous layers. Also, the result in this layer is shown in the form of K-dimensional vector, where K is the number of classes that the network needs to predict the result. The vectors mentioned herein contain each category of images that will be categorized as a class.

The convolutional neural network layers are implemented, with each class learning the features to look for from the input. The advantage of using this type of network is variable sharing and association, where variable sharing can reduce the number of weight variables to one layer without affecting the accuracy of the results to be expected. Furthermore, convolution also breaks down the input feature into more minor features. This makes each result value dependent on a small amount of input data, resulting in quick adjustments.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.6 Object Detection

Object detection is a computer vision technique for detecting objects in images and videos to produce meaningful results, and object detection algorithms typically use machine learning or deep learning. We can recognize and locate objects of interest in images or videos in a matter of seconds when we look at them. Hence, the goal of object detection is to use a computer to replicate this intelligence. An example of detecting an object is shown in Figure 2.39.

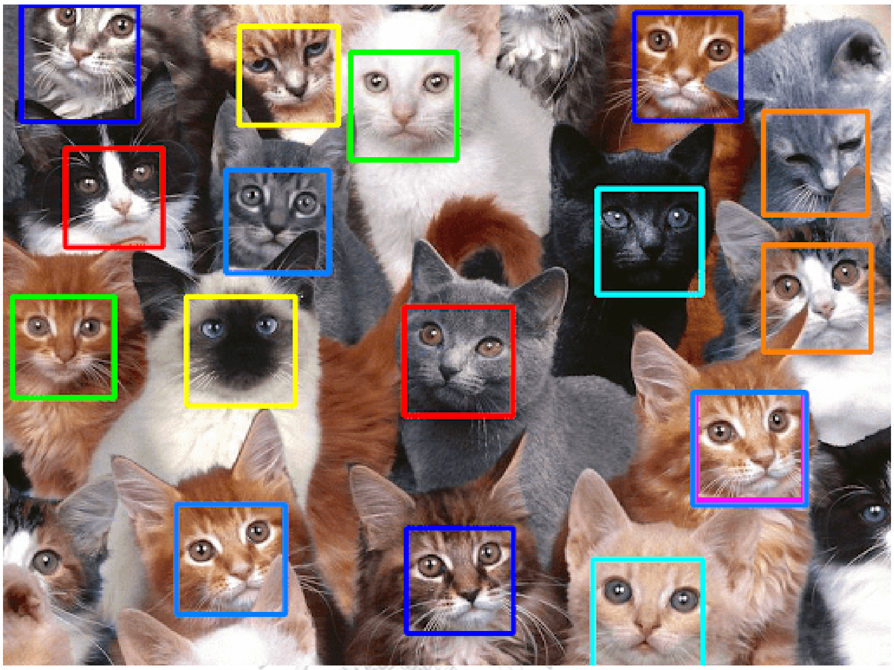


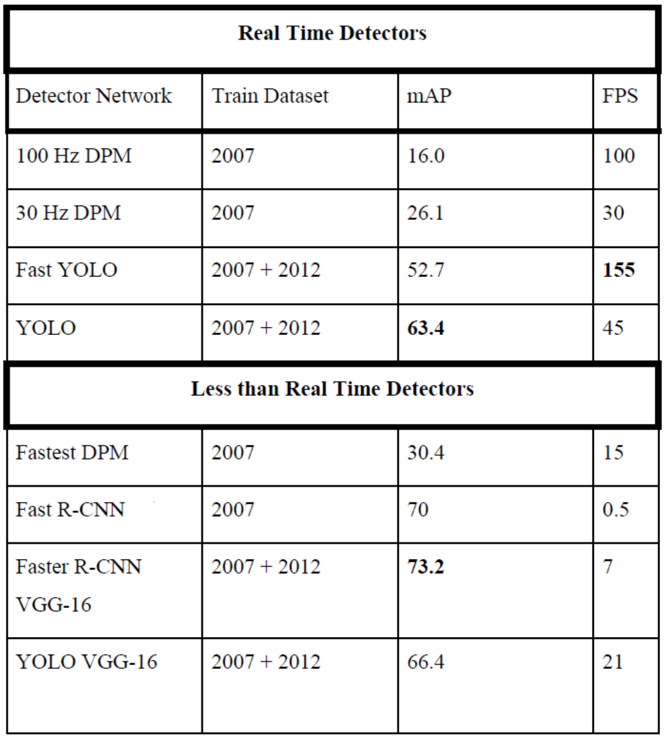
Figure 2.39: The visualization of an example image with ground truth bounding boxes [15]

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

The following sections describe the details and components of the You Only Look Once(YOLO) model that we choose for this work. We explain in detail the three different versions of the YOLO model with the following details.

2.7 You Only Look Once Algorithm

You Only Look Once (YOLO) is a deep neural network method presented in 2016 by Joseph Redmon. YOLO is used for object detection in images or videos. It offers a real-time object detection network that can detect objects. The efficiency of object detection in Pascal VOC 2015 was recorded using the Pascal VOC 2007 dataset with an mAP of 63.4, as shown in Figure 2.40.



Real Time Detectors			
Detector Network	Train Dataset	mAP	FPS
100 Hz DPM	2007	16.0	100
30 Hz DPM	2007	26.1	30
Fast YOLO	2007 + 2012	52.7	**155**
YOLO	2007 + 2012	**63.4**	45
Less than Real Time Detectors			
Fastest DPM	2007	30.4	15
Fast R-CNN	2007	70	0.5
Faster R-CNN VGG-16	2007 + 2012	**73.2**	7
YOLO VGG-16	2007 + 2012	66.4	21

Figure 2.40: Comparison of object detection methods on the Pascal VOC 2007 dataset as of 2015 [16]

Figure 2.40 shows that YOLO could detect objects better than other detectors in real-time object detection because YOLO has reducing learnable parameters and network complexity, YOLO has faster performance than others methods. i.e., Faster R-CNN and Fast R-CNN. Details of Yolo will be explained in the next section.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.7.1 Overview of the YOLO Pipeline

The primary YOLO pipeline discussed here, YOLO extracts feature maps from images using standard convolutional neural networks. The specific network used for this purpose is user-dependent and can be changed according to its requirements. Ideally, the feature extractor must have few learnable parameters without compromising accuracy metrics.

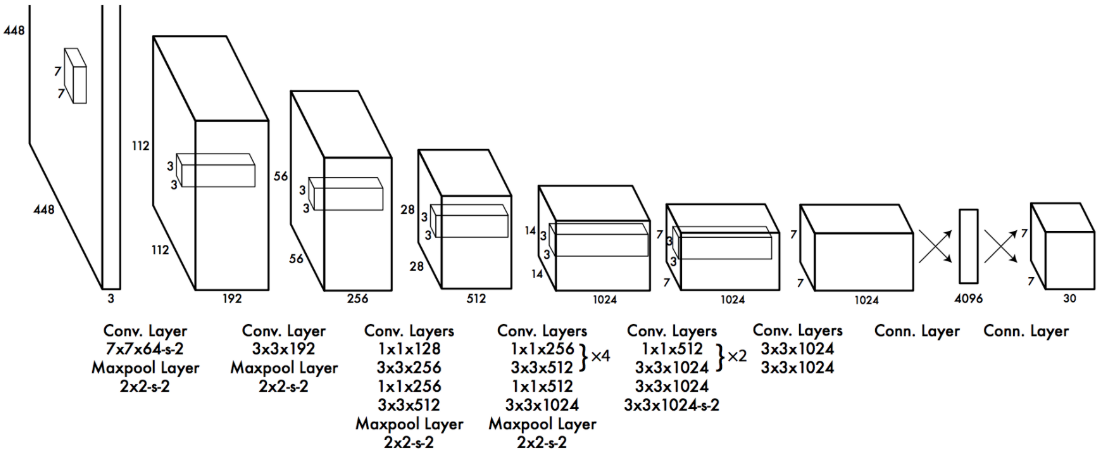


Figure 2.41: The visualization of You Only Look once (YOLO) network structure [17]

From the network in Figure 2.41, the feature map dimensions are dependent upon the size of the input and the close clustering of objects in an image. Output with dimensions of $7 \times 7 \times d$ where the feature map number is d, as shown in the equation (2.14).

$$d = B \times 5 + c, \tag{2.14}$$

where B is the maximum number of objects the network can guess in each 7×7 grid cell, and c is the number of classes.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.7.2 Non-Max Suppression

When an object area is spread over more than one grid cell, the network may not determine the object's exact center point. For this reason, multiple grid cells may contribute to predict the position of the object, resulting in more than one prediction of the same object. An example is shown in Figure 2.42.

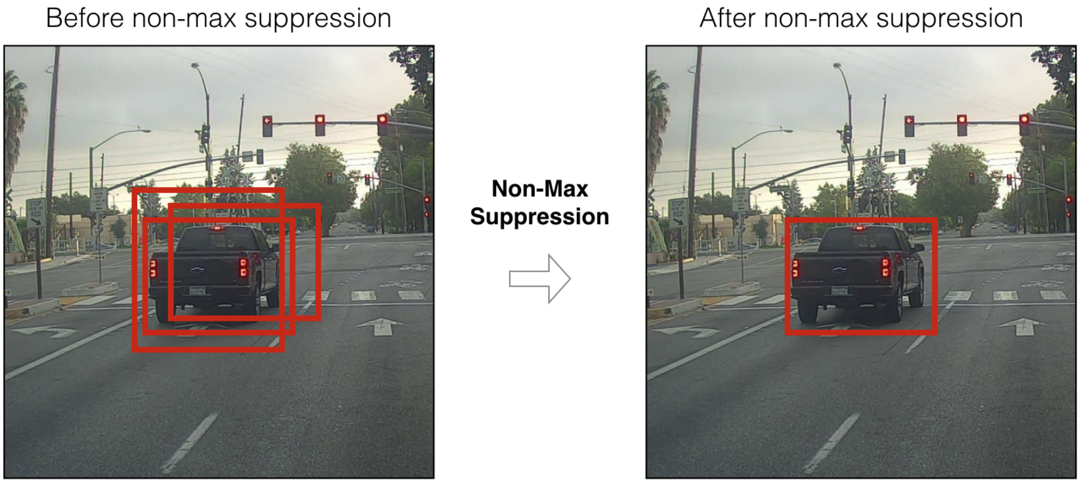


Figure 2.42: The result of using Non-Max Suppression [18]

This is especially prevalent when the threshold confidence levels are set to a lower value, as shown in Fig 2.43.

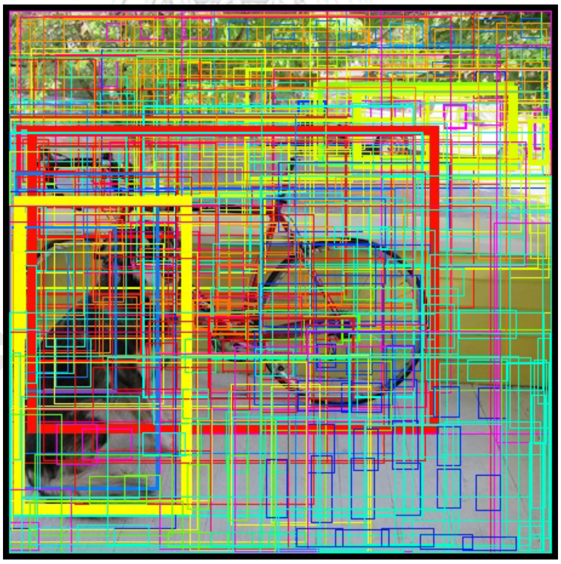


Figure 2.43: An example of the confidence score threshold [19]

Events with more than one prediction of the same object can be resolved by suppressing duplicate outputs or low confidence results. This is achieved through a process known as non-max suppression. Non-max suppression is done greedily in YOLO. First, the bounding box with the highest level of confidence is chosen. With this bounding box selected, all bounding boxes with a high IOU are discarded. Next, the bounding box with the highest confidence is selected from the remaining bounding boxes, and the process continues. This is the last step of the YOLO detection process. Figure 3.5 shows the final result of YOLO on an example image.

This is the final step of the YOLO detection process. The final result from YOLO in the preview is shown in Figure 2.44.

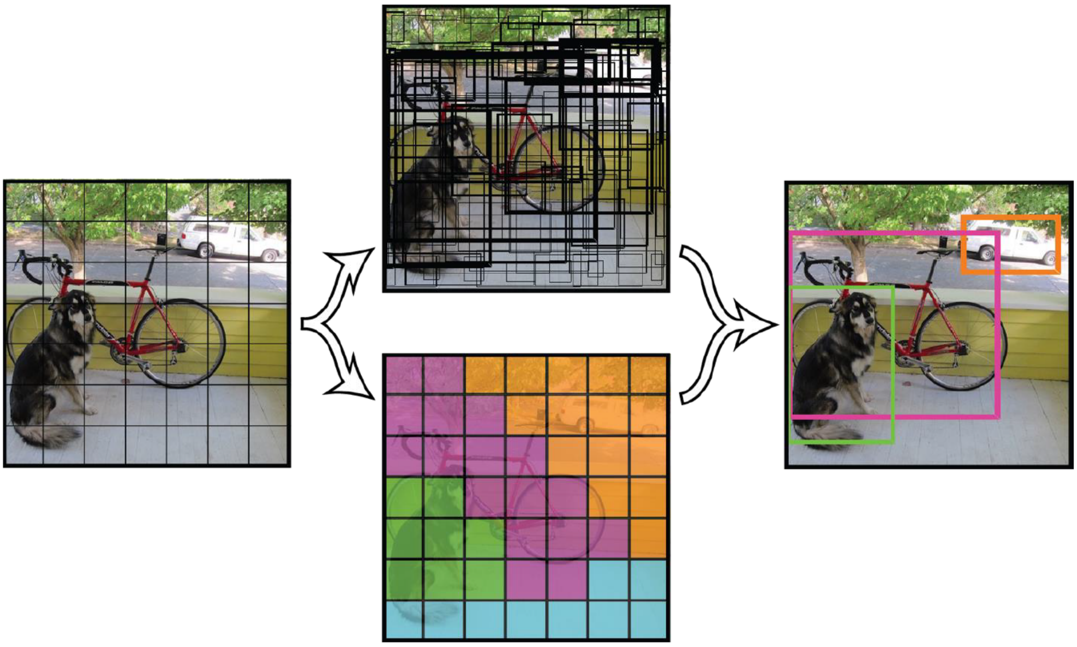


Figure 2.44: An example of the final output of the YOLO network [20]

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.7.3 Deficiencies of YOLO

Although YOLO is a state-of-the-art object detection network with significantly reduced processing times compared to other networks, YOLO has obvious drawbacks compared to other popular networks because YOLO gave mean average precision values. YOLO also made an error when objects were grouped close together, resulting in detection errors due to limitations on the number of objects it can detect. In the grids, each cell and another obvious problem is that when small objects or objects in the image are very different in size, YOLO may have erroneously detected an object, as shown in Figure 2.45.

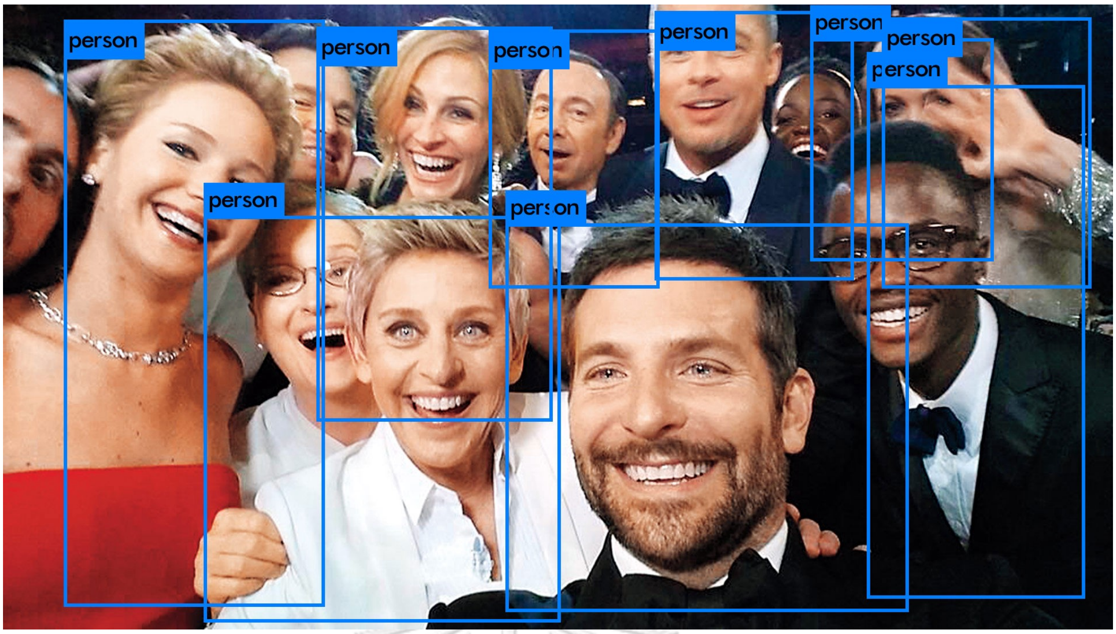


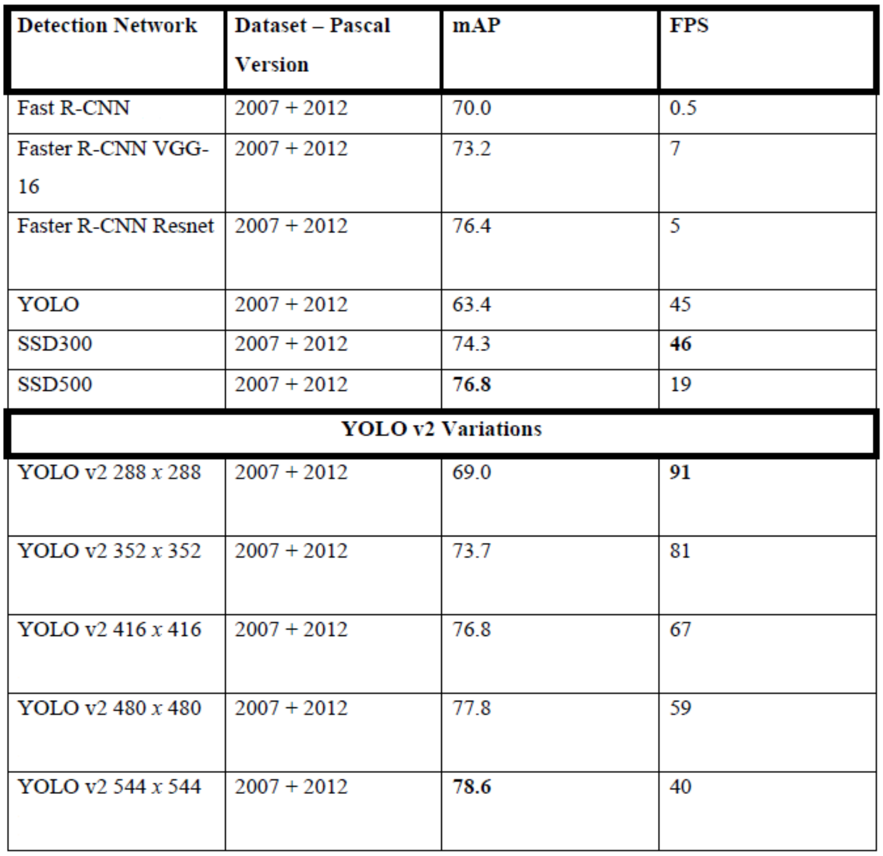
Figure 2.45: Example of detecting a faulty person (object) of the Yolo network [21]

Because YOLO uses convolutions to subsample an image, small objects can often go undetected. For small objects, YOLO is not a good detector. YOLO finds it difficult to generalize when objects of the same class are of different sizes, as shown in Figure 3.6.

2.8 YOLOv2 and YOLO9000

YOLOv2 and YOLO9000 are improvements from YOLO, with several YOLO differences, which will be explained in detail in the next section. YOLOv2 and YOLO9000 are based on YOLO prototype networks, all of which work together to build a faster and more reliable network, introduced in CVPR 2016 by Joseph Redmon and Ali Farhadi, demonstrating that more than 9000 object types can be detected, with the performance of object detection on the work Pascal VOC 2015 using Pascal VOC 2007 + 2012 dataset. YOLOv2 obtained mAP of 76.8 at 67 FPS and mAP of 78.6 at 40 fps, observed in the Table 2.46. This improved performance also shows that YOLOv2 has good performance to object detection. It also provides good results for object class classification, providing more detailed output on WordTree [36] representation of ImageNet [37]. This can be very useful for a wide variety of object detection or classification tasks.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



Detection Network	**Dataset – Pascal Version**	**mAP**	**FPS**
Fast R-CNN	2007 + 2012	70.0	0.5
Faster R-CNN VGG-16	2007 + 2012	73.2	7
Faster R-CNN Resnet	2007 + 2012	76.4	5
YOLO	2007 + 2012	63.4	45
SSD300	2007 + 2012	74.3	**46**
SSD500	2007 + 2012	**76.8**	19
YOLO v2 Variations			
YOLO v2 288 *x* 288	2007 + 2012	69.0	**91**
YOLO v2 352 *x* 352	2007 + 2012	73.7	81
YOLO v2 416 *x* 416	2007 + 2012	76.8	67
YOLO v2 480 *x* 480	2007 + 2012	77.8	59
YOLO v2 544 *x* 544	2007 + 2012	**78.6**	40

Figure 2.46: Comparison of object detection methods on the Pascal VOC 2007 + 2012 dataset as of 2015 [16]

2.8.1 Changes from YOLO

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

YOLOv2 has several enhancements and changes over YOLO, which YOLOv2 increases performance for the detect object in images and videos. The following improvements are available.

2.8.1.1 Anchor Boxes

Improved from YOLO with the addition of anchor boxes, which a set of user-defined anchor boxes is taken at each grid cell, the current grid cell is responsible for predicting the change in height and width anchor boxes instead of the absolute width and the absolute height. This was more efficient than other modern object detection networks such as R-CNN and faster R-CNN. It also made it easier for the network to learn the size of the box.

In the YOLOv2 network in Figure 2.46, 3 anchor boxes are selected for each grid cell. If the sample image contains 7 × 7 cells, there will be 7 × 7 × 3 anchor boxes for that feature map, with each anchor boxes able to predict one object. The equations for predicting the width and the height of an object can be denoted in Equation (2.15).

$$\begin{aligned} b_w &= p_w e^{t_w}, \\ b_h &= p_h e^{t_h}, \end{aligned} \tag{2.15}$$

where b_w is the width and b_h is the height.
t_w and t_h are the predicted width and height values of the network.

From Equation (2.15), it can be noted that the exponential function is used because of its favorable properties during backpropagation, and using exponential functions also prevents negative predictions because width and height cannot be negative for objects.

2.8.1.2 Anchor Box Dimensions

Even if the anchor box is user-defined, there may be enough improper anchor box sizes for the dataset. Therefore, we have figured out a way to determine the anchor box's size. We can choose the appropriate size using K-Means clustering and group the anchor boxes to suit the data set by selecting the anchor boxes with 9 different boxes.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.8.1.3 Constrained x, y Predictions

The corresponding grid cell in YOLO predicts the center of an object. The grid cell in the object's center is in charge of predicting the exact (x, y) location of the object box with itself. The coordinates in region proposal networks are calculated as follows:

$$\begin{aligned} x &= (t_x \times w_a) - x_a, \\ y &= (t_y \times h_a) - y_a, \end{aligned} \tag{2.16}$$

where x, y is the center position of the object;

w_a and h_a is the width and height of the anchor box;

t_x and t_y are the predictions;

x_a and y_a are for the center of the region.

Using equation (2.16), The coordinates predicted by any grid cell could end up in any part of the image. YOLO v2 introduces constraints on this by allowing each grid cell to predict coordinates anywhere within itself only. The center of an object predicted by a particular grid cell cannot be outside of itself. The equation is shown in (2.17).

$$\begin{aligned} b_x &= \sigma(t_x) + c_x, \\ b_y &= \sigma(t_y) + c_y, \end{aligned} \tag{2.17}$$

where b_x and b_y are the center x and y of the object.

The sigmoid function is applied to t_x and t_y, a network predicted where c_x and c_y are the grid cell's positions. With this equation, the object's center cannot extend beyond the boundary of the grid cell.

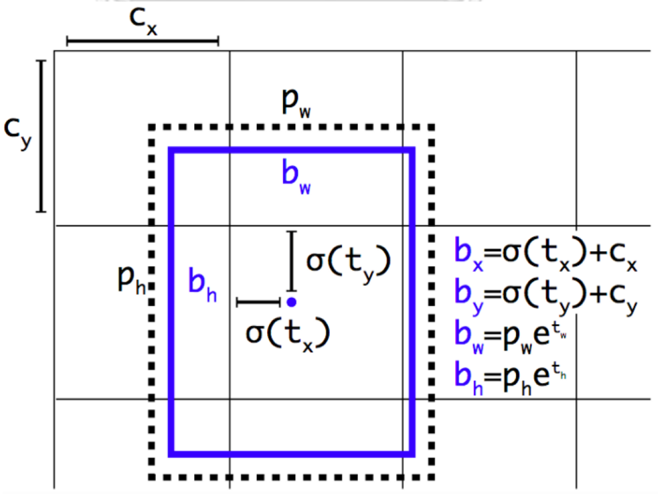


Figure 2.47: Object box predictions from one anchor box belonging to one particular grid cell [22]

2.8.1.4 High Resolution Classifier

YOLO is designed for an image size 244 × 244. With its small image size, the small objects in the image are difficult to detect, especially when we resize from image larger than size 244 × 244 becomes image size 244 × 244 makes the small objects contained in the original image considerably smaller. For this reason, YOLO has difficulty detect small objects in images and videos, so YOLOv2 is trained and tested on a 416 × 416 image which is a larger image. Moreover, the backbone architecture allows the network to accept a higher resolution of the image, although this increases training and prediction time.

2.8.1.5 Multi Scale Training

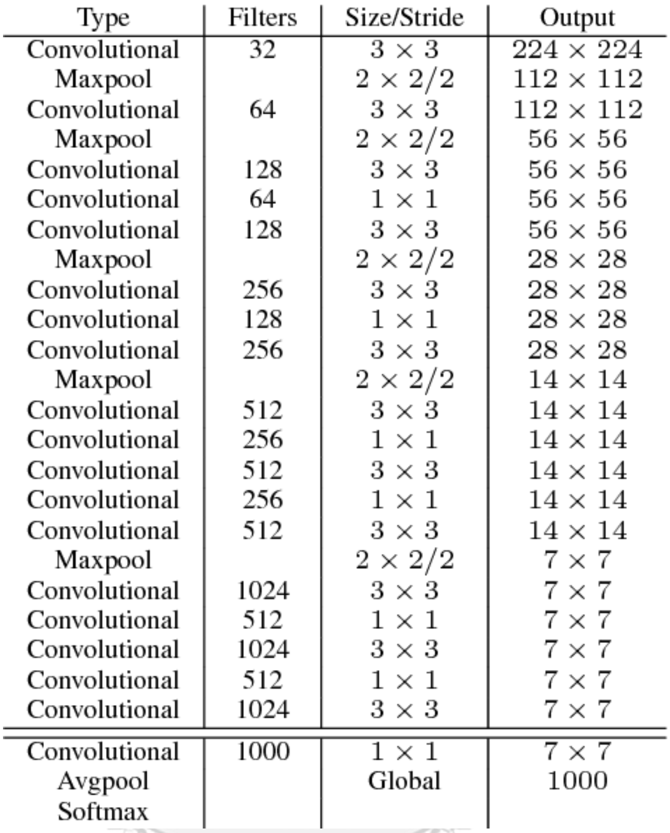
While YOLO does not allow training or testing with multiple scales images, in YOLOv2, the input image can be any dimension as long as the grid number of the generated cell is odd. This allows training to be performed with multiple scales and can also increase the stability and accuracy of the network. Therefore, networks using Multi-Scale Training have better prediction and higher test scores independent of object scale.

2.8.1.6 Darknet-19

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

Most object detection networks use VGG [38] as their backbone. Although this state-of-the-art network offers good feature extraction, millions of parameters need to be learned, which significantly slows down the network, where YOLOv2 uses Darknet-19, a faster backbone than a regular VGG with 19 convolution layers and 5 fully connected layers. Most of the convolutions in Darknet-19 often uses with 3 × 3 matrix, and the number of feature maps doubled after going through all of the pooling steps.

For the training phase, the network was first trained as an object classification network, in which Darknet-19 was combined with fully connected layers at the end to be trained on ImageNet1000. Once this is done, the last fully connected layers are removed and replaced with YOLO prediction maps. This new network will be trained for detection, which process uses the size of the ImageNet training data to train more generalized filters for convolution in the first few layers.



Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Figure 2.48: The visualization of You Only Look once v2 (YOLOv2) network structure [22]

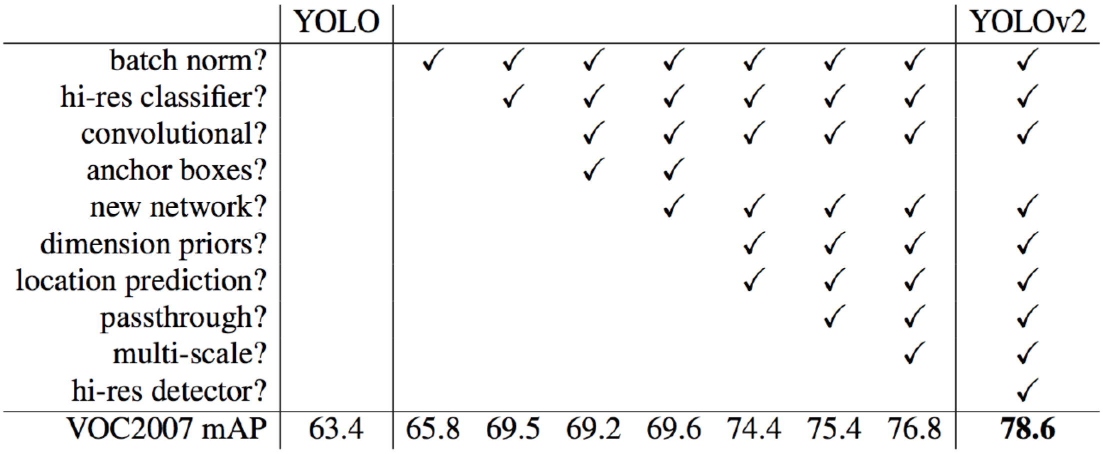
จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.8.1.7 Hierarchical Classification

YOLOv2 is trained to extract ImageNet labels from WordNet. WordNet is a language database associated with words, which helps the network relate images and objects. For example, the norfolk terrier is categorized as a hunting dog, a dog category.

The extracted features need to be processed to generate meaningful predictions corresponding to the object's location and class. The next step of the YOLO pipeline is focused on creating such predictions.

The contributions of each of the changes to mAP scores in YOLOv2 as shown in Figure 2.49.



	YOLO								YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

Figure 2.49: Contributions of each of the changes to mAP scores in YOLOv2 [22]

2.9 YOLOv3

2.9.1 Backbone

The backbone is the feature extractor of the image. The image is transmitted through the convolution layer, pooling layer, batch normalization, and activation layer to distinguish the specific feature of the data, and this backbone rejects other unrelated features. There are many variations between backbone, which the backbone is typically a deep neural network. However, the depth and the number of this network's parameters must be reasonable because excessive depth and the number of network parameters will complicate the operation, increasing the time to process. For example, network training and testing are significantly slower, with Darknet53 being a feature extractor that reduces network complexity without compromising accuracy. The backbone network consists of multiple convolution operations since the image is subsampled over the convolutions, causing small objects to lose their resolution and detail. Outputs are taken at three different convolution stages. The YOLOv3 network uses these three scales of outcomes for prediction.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

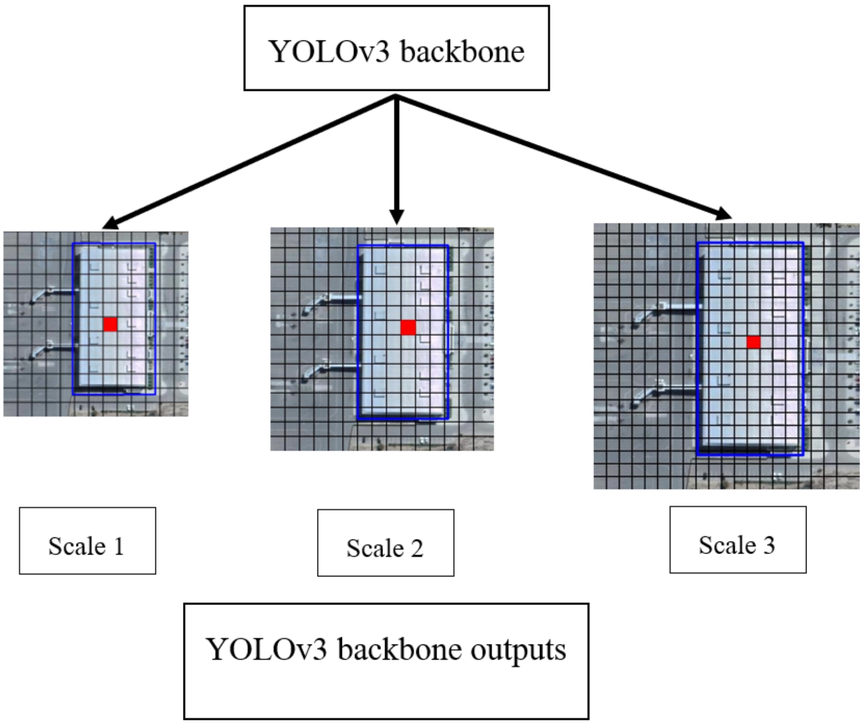


Figure 2.50: YOLOv3 backbone outputs

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.9.2 Prediction feature maps

YOLO predicts the output tensors in three different scales as shown in Figure 2.50. The scales of output tensors correspond to the numbers of grid cells in input images of each scale. The depth of output tensors is $B \times (4+1+c)$, where the feature map predicts x and y. Includes values of change in the height and width of anchor boxes and c values, where c is the number of classes and B is the number of anchor boxes on the same scale, which helps to localize objects of different sizes. YOLOv2 and YOLOv3 make use of anchor boxes for better prediction with tighter bounding boxes.

The backbone network divided each feature map into a cell of each $s \times s$. Each anchor boxes solves five values are the values of x and y where (x, y) are the midpoints of the bounding box, the change of the anchor boxes width (w) and height (h), and objectness or the confidence of an object within that cell if an object has more than one grid cell, only the middle grid cell is responsible for detecting it.

This prediction pattern is repeated for each of three scales, so if there are 3 anchor boxes on each scale, the backbone network will predict $3 \times (4 + 1 + c)$ feature maps for every one of the scales.

However, as described above, each cell can predict only three objects for each scale or one object at any one anchor box aspect ratio for each scale. Many objects form a group, or objects are tiny, making it impossible to detect objects efficiently.

2.9.3 Losses

Since YOLO has $B \times (5 + c)$ prediction in each $s \times s$ cell, it has to have losses for all different types of predictions, with the loss function measuring the error of the position and size of the predicted bounding box. The loss function is responsible for predicting the position of the object in ith cell which has the center (x, y) with width w and height h that is defined by

$$\begin{aligned}&\lambda_{\text{coord}} \sum_{i=1}^{s^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{\text{obj}} \left((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right) \\ &+ \lambda_{\text{coord}} \sum_{i=1}^{s^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(\left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 + \left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 \right),\end{aligned}$$

where the index i refers to the ith cell ordering by upper left cell until to lower right cell for $i = 1, 2, 3, \dots, s^2$, the index j refers to the jth prediction box for $j = 1, 2, 3, \dots, B$, λ_{coord} is a weight of the loss calculated by boundary box coordinates, and $\mathbb{1}_{ij}^{\text{obj}} = 1$ if the jth boundary box in cell i is responsible for detecting an object, otherwise 0.

The factor λ_{coord} assigns an unequal weight to the losses of objects based on their bounding box sizes. The loss factor for smaller objects is higher, while the loss factor for larger objects is lower.

Confidence loss is used to predict whether the network can detect an object's existence, which is a measure of objectiveness. If an object is detected in a particular $s \times s$

cell, the confidence loss for that cell is shown in the following equation

$$\sum_{i=1}^{s^2}\sum_{j=1}^{B}\mathbb{1}_{ij}^{\text{obj}}\left(C_i-\hat{C}_i\right)^2,$$

where $\hat{C}_i$ is the confidence score of the box j in cell i and $\mathbb{1}_{ij}^{\text{obj}}=1$ if the jth boundary box in cell i is responsible for detecting an object, otherwise 0.

If the object is not detected, the confidence loss for that cell is shown in the following equation

$$\lambda_{\text{noobj}}\sum_{i=1}^{s^2}\sum_{j=1}^{B}\mathbb{1}_{ij}^{\text{noobj}}\left(C_i-\hat{C}_i\right)^2,$$

where $\mathbb{1}_{ij}^{\text{noobj}}$ is the complement of $\mathbb{1}_{ij}^{\text{obj}}$, $\hat{C}_i$ is the confidence score of the box j in cell i, and λ_{noobj} is a weight of the loss when detecting background.

Finally, in each cell, we need to define the probability of each class c where the class loss is similar to a cross-entropy loss for a classification network

$$\sum_{i=1}^{s^2}\mathbb{1}_{ij}^{\text{obj}}\sum_{c\in\text{classes}}\left(p_i(c)-\hat{p}_i(c)\right)^2,$$

where $\mathbb{1}_{ij}^{\text{obj}}=1$ if an object appears in cell i, otherwise 0 and $\hat{p}_i(c)$ denotes the conditional class probability for class c in cell i.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

All YOLO losses can be described as follows

$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=1}^{s^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{\text{obj}} \left((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right) \\
&+ \lambda_{\text{coord}} \sum_{i=1}^{s^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(\left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 + \left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 \right) \\
&+ \sum_{i=1}^{s^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=1}^{s^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=1}^{s^2} \mathbb{1}_{ij}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 .
\end{aligned}
$$

2.9.4 Backbone Architecture

Although training a network from scratch takes a long time, there is a way to use a previously trained network for other purposes. For example, the backbone of YOLOv3 is Darknet-53, and Joseph Redmon et al. used a variant of Darknet-53 trained on ImageNet with over 20,000 categories. Transfer learning is a type of training in which a network that has already been trained is used for a different purpose. This is accomplished by removing a few detection and classification layers near the network's end and then training new detection layers with the new data. For example, a network's feature extraction layers have learned to detect essential features from images, convolutional layers and weights associated with this feature extraction have already completed a large portion of the training.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

Darknet-53 is a high-performance backbone that can be used as a structure for various applications, eliminating the need for new models to be trained from scratch. In the context of this thesis, the term training or learning refers to the process of applying transfer learning to the problem at hand, which is the object detection in a custom dataset with buildings in an airport.

Darknet-53 is a convolutional network with 53 layers and residual connections. It is primarily used for image classification. The last three layers of Darknet-53 consist of the average pooling layer, fully connected layer, and softmax layer. We only use Darknet-53 to extract image features in this object detection task. Thus, we ignore the last three layers of Darknet-53. For the detection task, there are two networks of darknet-53 are stacked together, accumulating to a total of 106 layers of fully convolutional architecture. Since the increasing number of layers, this network reduces in speed compared to the second version, which only has 30 layers. In the convolutional layers, kernels of shape 1x1 are applied on feature maps of three different sizes at three different scales in the network. The algorithm makes predictions at three scales, given by downsampling the image's dimensions by a stride of 32, 16, 8, respectively. Downsampling, the reduction in spatial resolution while keeping the same image representation, is done to reduce the size of the image. Every scale uses three anchor bounding boxes per layer. There are three sizes of anchor bounding boxes. There are three large boxes for the first scale, three medium boxes for the second scale, and three small boxes for the last scale. Because there are many sizes of anchor bounding boxes so each layer good for detecting large, medium, or small objects.

The You Only Look once v3 (YOLOv3) network structure is shown in Figure 2.51.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

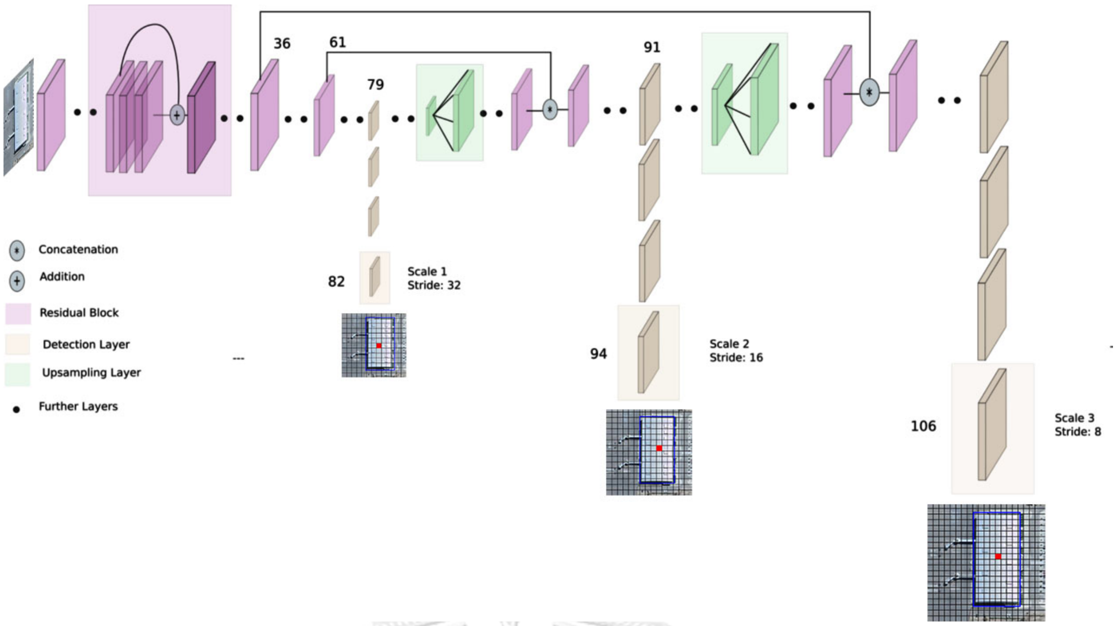


Figure 2.51: The visualization of the You Only Look once v3 (YOLOv3) network structure [23]

2.9.5 Residual Block

Residual block is a new type of block used by Darknet-53. It is hard to train deep neural networks. As the depth of the network increases, the accuracy of the network can become saturated, resulting in higher training errors. The residual block was created to fix this issue. The addition of a skip connection distinguishes the residual block from the standard convolution block in terms of architecture. The input is carried to the deeper layers via the skip connection.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

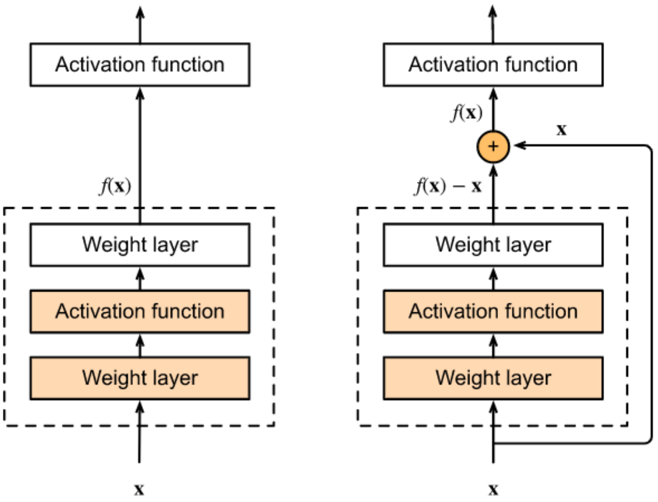


Figure 2.52: The examples of Residual block application

Let x be an input data and f be the desired mapping that is learning through the input data x. On the left in Figure 2.52, the dotted box learns the mapping $f(x)$ directly. The residual block is depicted on the right side in Figure 2.52. The dotted box portion learns $f(x) - x$, a slightly different mapping. The input is added by $f(x) - x$ to get the actual mapping $f(x)$. A residual connection or shortcut connection is a solid line that connects the input x to the mapping. Because the addition of x acts as a residual, the residual block was invented.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.10 Performance Evaluation

The measurement of model performance in this work consists of several measurements using Intersection over Union, Confusion Matrix, Precision, Recall, and F1 Score.

2.10.1 Intersection over Union

The IoU is a popular value for measuring object detection accuracy. Pascal VOC competitions, a well-known object detection competition, used the IoU value to evaluate the competition score. The IoU value was calculated from the area of overlap ratio between the prediction area and the ground truth area divided by the area of the union of the predicted area and the truth area, as shown in the equation (2.18).

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}. \tag{2.18}$$

In this thesis, we use the IoU value in assessing the accuracy of airport building detection. If the IoU value is more than 0.5, the airport buildings are detected. An example of measuring IoU values can be shown in Figure 2.53.

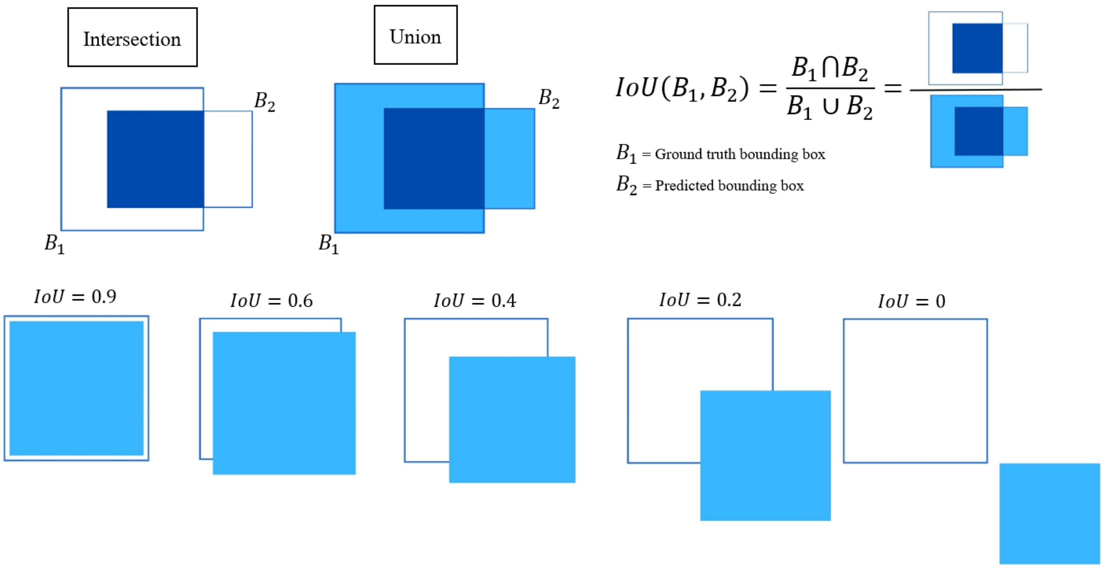

Intersection
Union
B_2
B_1
B_2
B_1
$IoU(B_1, B_2) = \frac{B_1 \cap B_2}{B_1 \cup B_2} =$
B_1 = Ground truth bounding box
B_2 = Predicted bounding box
$IoU = 0.9$
$IoU = 0.6$
$IoU = 0.4$
$IoU = 0.2$
$IoU = 0$

Figure 2.53: The examples of the Intersection over Union (IoU)

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.10.2 Confusion Matrix

A confusion matrix is a table that shows how well a classification model (or a classifier) performs on a set of test data for which the actual values are known.

		Actually class	
		Actually positive (Building)	Actually negative (Not building)
Predicted class	Predicted positive (Building)	True Positive (TP)	False Positive (FP)
	Predicted negative (Not building)	False Negative (FN)	True Negative (TN)

The confusion matrix of two types of data classification, as in this thesis, consists of two kinds: airport building and not airport building. The values in the table are assigned the following values:

1) True Positive (TP) is an area that the system predicts is an airport building, and that area is an actual airport building.
2) False Positive (FP) is an area that the system predicts is an airport building, but that area is not an airport building.
3) True Negative (TN) is the area that the system predicts is not an airport building, and that area is not an airport building.
4) False Negative (FN) is the area that the system predicts is not an airport building, but that area is an airport building.

2.10.3 Precision Recall and F1 Score

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Classification performance measurement can calculate Precision, Recall, and F1 Score as follows

Precision is the value of all airport buildings correctly predicted by the system as a percentage, calculated from the predicted ratio of the correct airport buildings (TP) to the number predicted to be all airport buildings (TP + FP).

A recall is the value of all airport buildings correctly predicted by the system as a percentage, calculated from the ratio that predicted the correct airport buildings (TP) and the total airport building (TP + FN).

F1 score is the weighted average of precision and recall. It can be calculated from twice the precision times the recall value divided by a precision plus recall.

The performance measurement is defined as follows:

$$
\begin{aligned}
&\text{Precision } (P) = \frac{TP}{TP + FP}.\\
&\text{Recall } (R) = \frac{TP}{TP + FN}.\\
&\text{Average Precision} = \int_0^1 P(R)\, dR.\\
&\text{F1-Score} = 2 \cdot \frac{P \times R}{P + R}.
\end{aligned}
$$

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

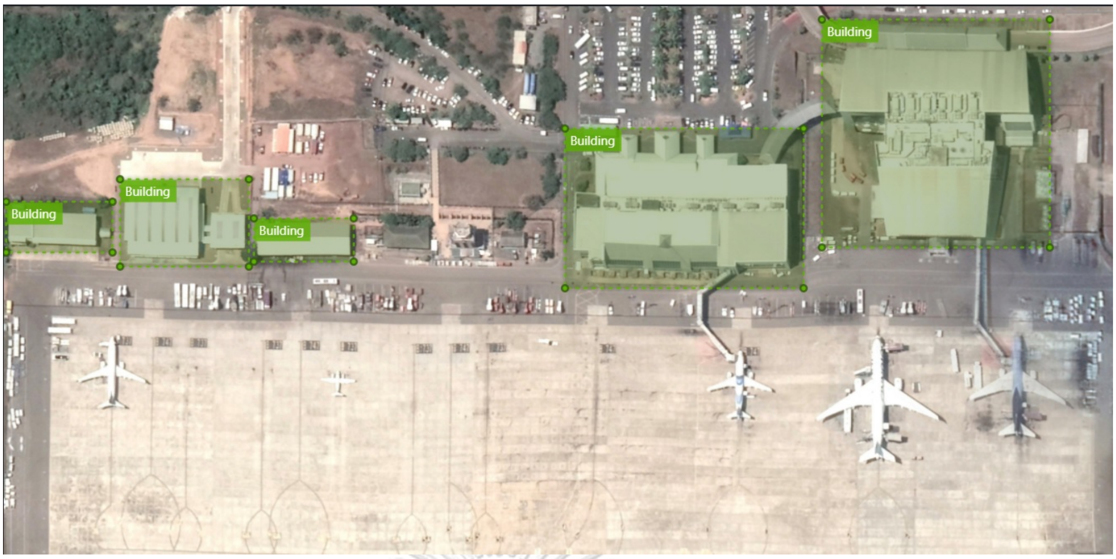
CHAPTER III

PROPOSED METHOD

In this thesis, we provide the data sets that have been used to test our proposed method. These data sets are composed of remote sensing images of more than 322 Asian airports. In each airport, we captured the levels of high ground above the airport at 0.6, 0.8, and 1 kilometers, of which we have collected 465 images with 4800×2682 pixels.

Since the acquired data images cannot directly be applied with the YOLOv3 model, we need to slice the obtained image blocks into the specified size and label them before using them with the YOLOv3 model. Roughly, the collected images are first cut into 4,743 images with 416×416 pixels for each image, which consists of a labeled building. As mentioned in the introduction, for each image block, the Visual Object Tagging Tool is used to label the buildings into the PASCAL VOC format. To make the descriptions above more tangible, the sample and types of labeled buildings are illustrated in Figures 3.1.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



(a) Labeled samples of the buildings

(b) The passenger terminal for international flights

(c) The passenger terminal for domestic flights

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Figure 3.1: Krabi International Airport, Krabi, Thailand [24]

Then, when we finished preparing the RGB image dataset for preliminary model practice, then next, the conversion of the RGB image dataset from 3D to 4D. We are introducing a method that combines a Jet saliency map with You only look once v3 (YOLOv3) network. We create a one-dimensional image called Jet saliency map, which can reduce the complexity of the background and combine it with a 3D RGB input image to be a 416 × 416 four-dimensional input image before feeding it through the YOLOv3 network. The architecture of our approach is shown in Figure 3.2.

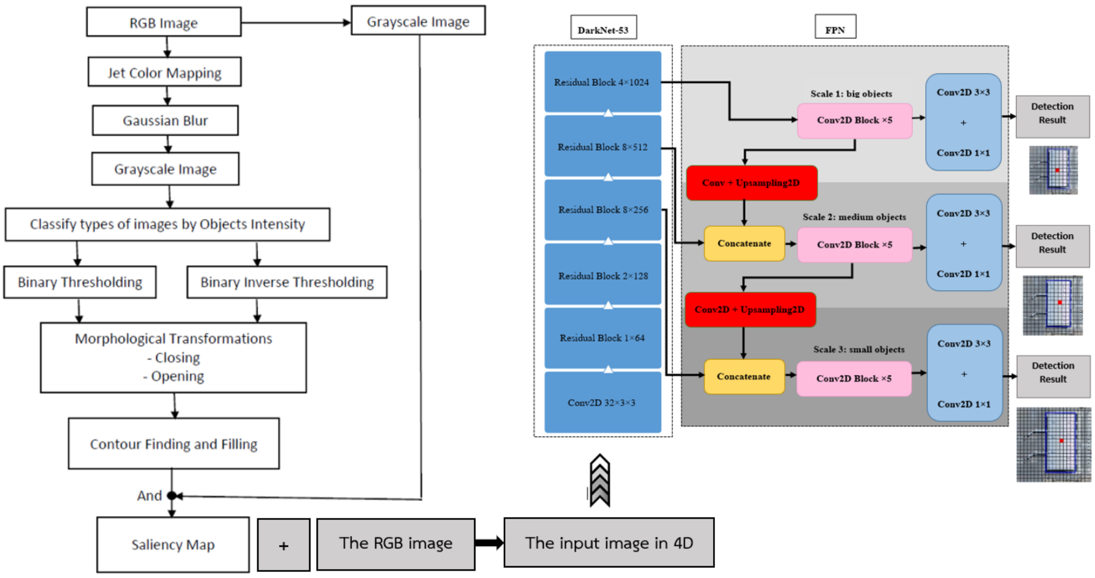


Figure 3.2: The visualization of our proposed method architecture [24]

3.1 Jet Saliency Map

In this section, we describe our improved process for constructing the Jet saliency map as demonstrated via the flowchart in Figure 3.10. The details for each block are as following.

1. We first transform the inputted 3D RGB images, see Figures 3.3, to Jet color images in 3D format by changing the color space as Jet color map. Thus, we have a new Jet color image, as seen in Figures 3.4.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

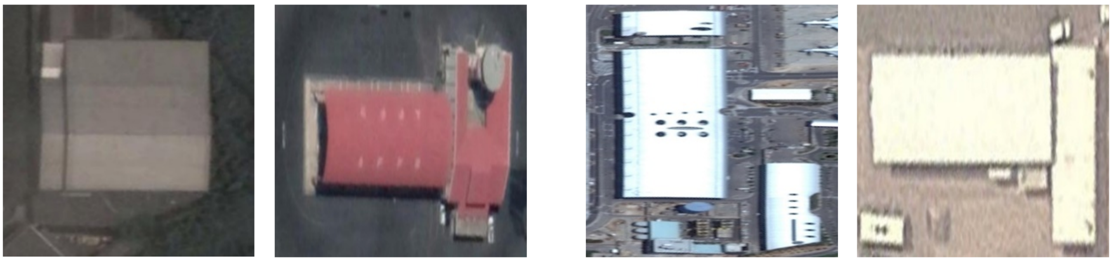


Figure 3.3: Example of input images [1]

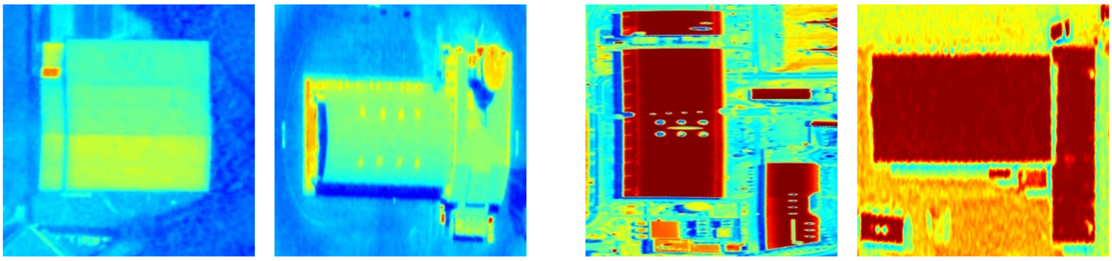


Figure 3.4: Example of jet color images

2. We then blur the obtained Jet color images using the Gaussian blur with 5×5 filters to denoise from the images, e.g., trees, vehicles, and small objects that are unrelated to our main object of interest. The Gaussian blur is a type of image-blurring filter that uses a Gaussian function

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}},$$

where, σ^2 is the variance, for computing the transformation at each pixel in the images.

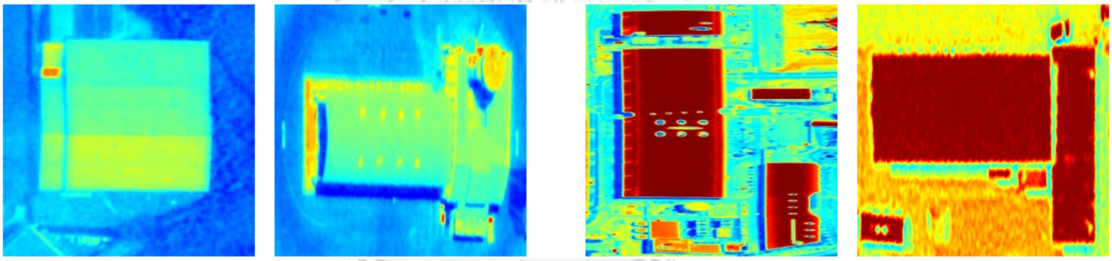


Figure 3.5: Example of jet color images after blurring with Gaussian blur

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

3. However, we found that the previous step provides the Jet color images, which can divide our objects of interest into two shading types, in gray scale. Therefore, in this step, we transform the blurred images obtained through the previous step into two groups of gray-scale images. The first group is the gray-scale image that has the foreground intensity higher than the background intensity. Conversely, the second group is the gray-scale image with the value of the foreground intensity less than that of the background intensity, as seen in Figure 3.6. The formula for transforming an RGB image into a gray-scale image is defined by

$$Y = 0.3R + 0.59G + 0.11B,$$

where, Y represents the intensity at a pixel and R, G, B represent the red, green, blue values at a pixel, respectively.

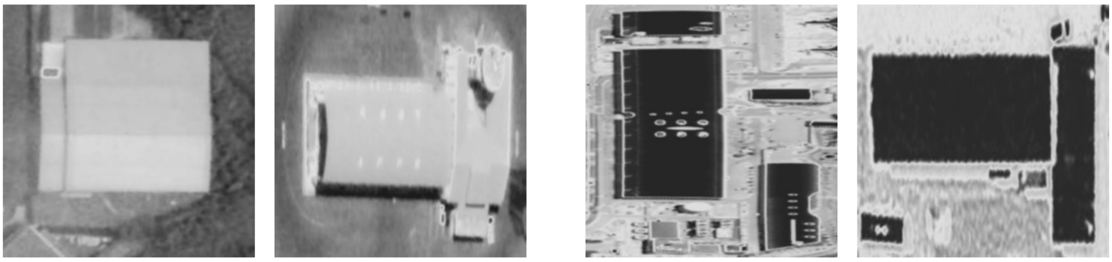
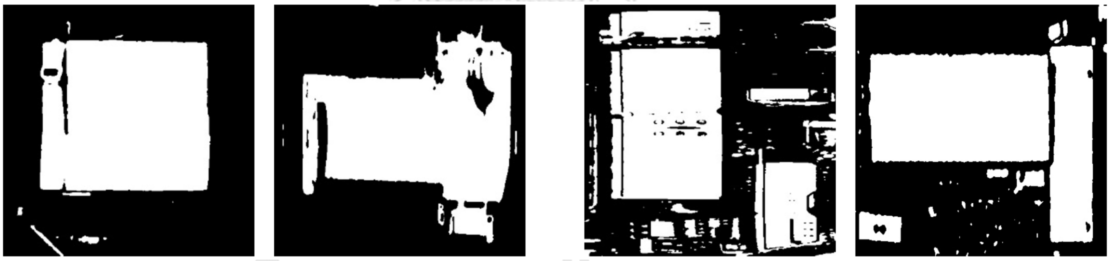


Figure 3.6: Example of grayscale images from jet color images after blurring with Gaussian blur

4. Next, we continue to transform an image of the first group into a binary image as using the binary threshold. Furthermore, in a series of computational experiments, we found that the suitable value of the threshold is 180. For the second group, we used the inverse binary threshold by transforming a gray-scale image into the inverse binary image as in Figure 3.7. For this, the suitable threshold value that we found is 150.

   Note that, the two selected thresholds are obtained by observing the results of the obtained images manually. As a result, these selected thresholds are appropriate when the resulting images provide the necessary detail and coverage that we want.



CHULALONGKORN UNIVERSITY

Figure 3.7: Example of binary images from grayscale

5. We then employed the morphological transformations, which consists of two operators. The first operator is the closing operator, which is used to close small holes inside the foreground objects. The second operator is the opening operator, which is used to remove the remaining noises in the image.
6. We use the command “CHAIN APPROX SIMPLE” in python, a type of contour approximation method, to find the contour of all objects in the image and fill the inside of the images to get a more detailed visual of the objects. The results image obtained from actions in steps 5 and 6 are shown in Figure 3.8.

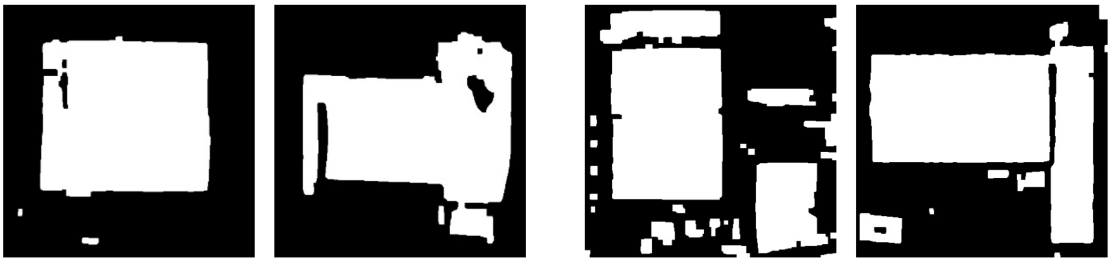


Figure 3.8: Example of binary images after going through steps 5 and 6

7. Finally, we will construct the saliency map using the bitwise AND operators to combine our binary image obtained in the preceding step with the original gray-scale image. Then, we achieve the 1D saliency map.

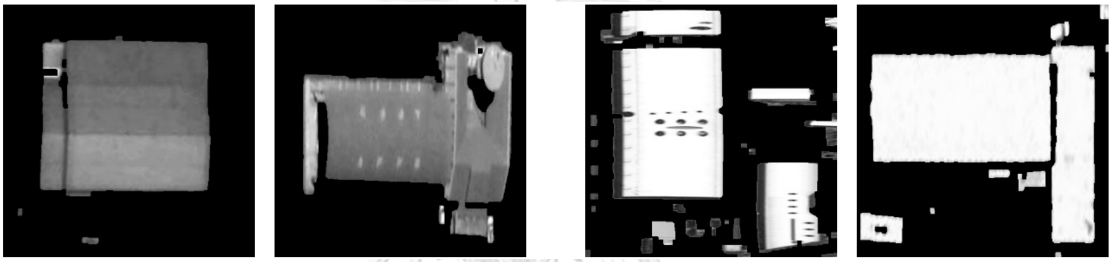


Figure 3.9: Example of the saliency map image

Note that, the fusion of the image obtained from the saliency map with the original RGB image, gives us the input image in 4D.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

3.2 Applying Jet Saliency Map with YOLOv3

To combine the Jet saliency map with the YOLOv3 network, we combine the RGB input image with the Jet saliency map as a 4D input array and then feed it into the YOLOv3 network to detect the airport buildings. The YOLOv3 network parameters we used in this work are the same as in the YOLO paper. In this evaluation of the efficiency of the method we proposed, the results are described in Chapter IV.

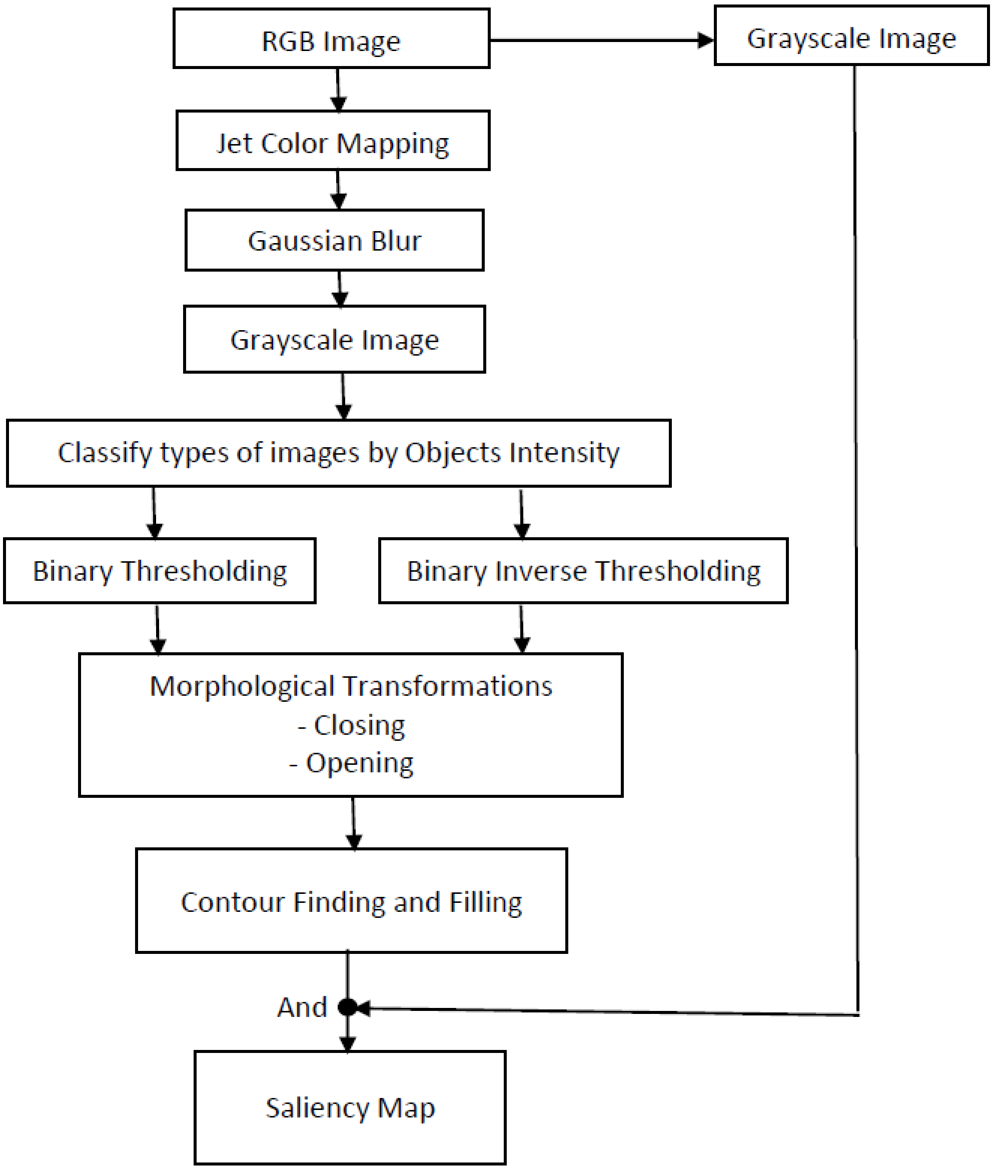
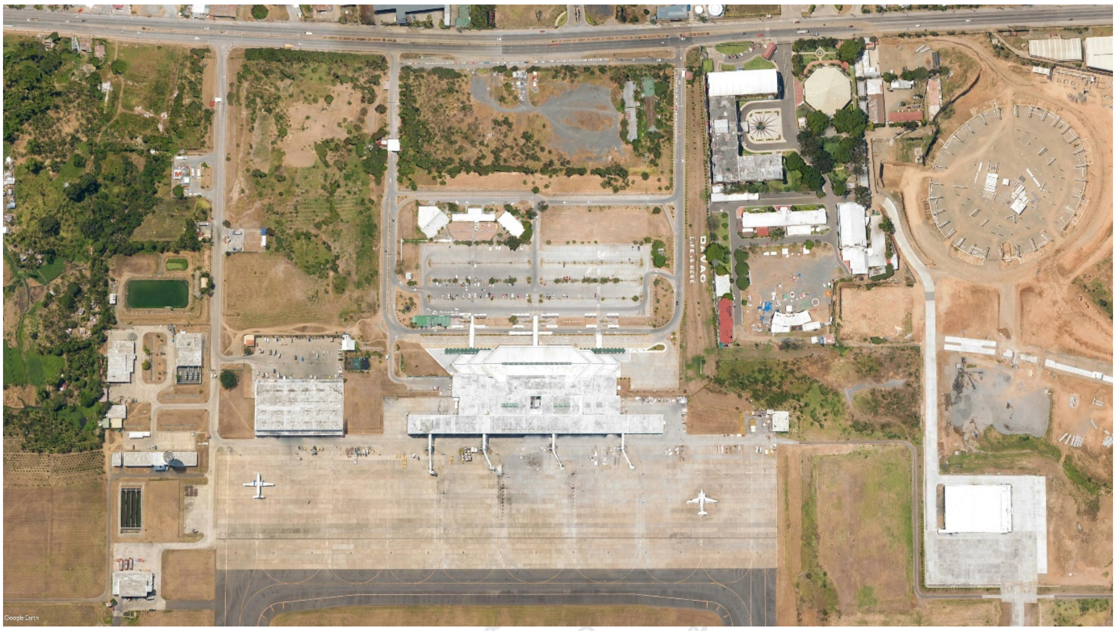

RGB Image
Grayscale Image
Jet Color Mapping
Gaussian Blur
Grayscale Image
Classify types of images by Objects Intensity
Binary Thresholding
Binary Inverse Thresholding
Morphological Transformations
- Closing
- Opening
Contour Finding and Filling
And
Saliency Map

Figure 3.10: Flowchart of saliency map creation process [24]

CHAPTER IV

RESULTS AND DISCUSSIONS

At the beginning of the experiment, we tried using the original YOLO architecture with the parameters from the original paper. The images we used to test the model are 190 images containing 809 target buildings for detection. Example images used to test the model are shown in Figures 4.1, 4.2, and 4.3.



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Figure 4.1: Example images from our dataset (1375 x 770 pixels) [1]

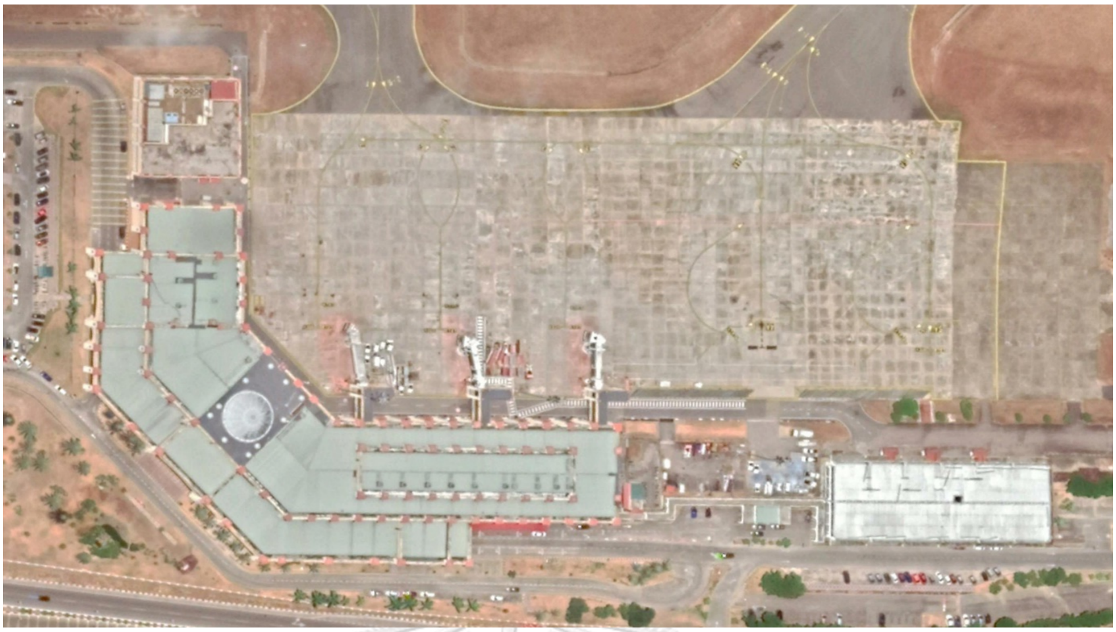


Figure 4.2: Example images from our dataset (1375 x 770 pixels) [1]

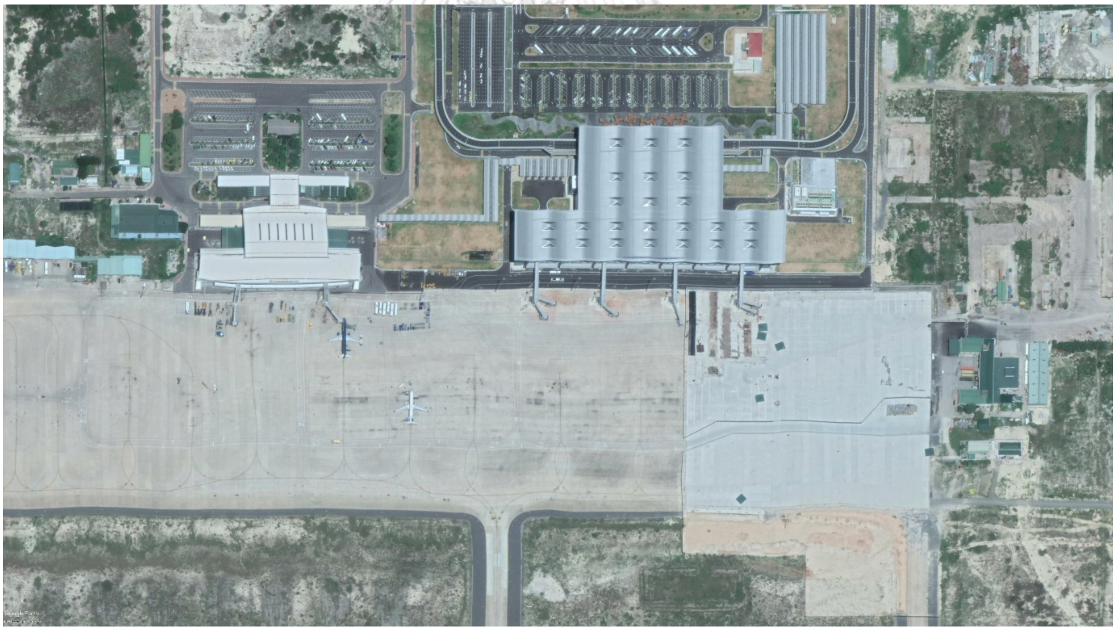


Figure 4.3: Example images from our dataset (1375 x 770 pixels) [1]

The results are unsatisfactory as predicted bounding boxes do not fit in the airport buildings. Many airport buildings are not detected because their shape is very diverse. In addition, it can not detect airport buildings that are small and found additional problems that if other objects obscure the buildings interested, they will not be able to detect. Also, a building having a color similar to the background color is another problem that makes it impossible to detect. Images of the result using the original YOLO architecture are shown in Figures 4.4, and 4.5

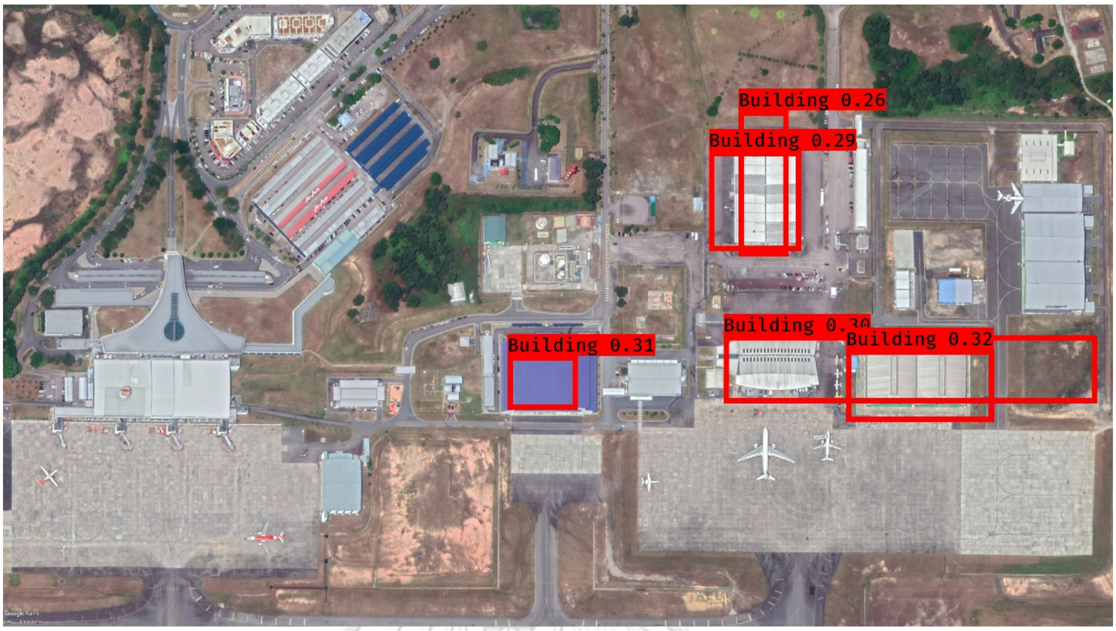


Figure 4.4: Example images of the result using the original YOLO (1375 x 770 pixels)

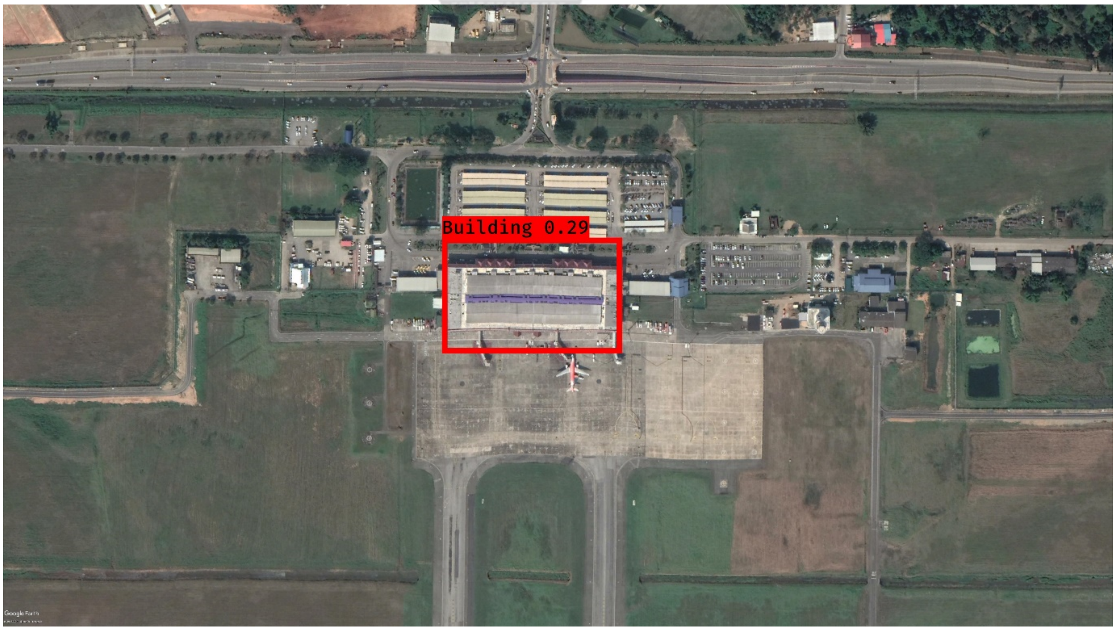


Figure 4.5: Example images of the result using the original YOLO (1375 x 770 pixels)

After that, we will solve the problem mentioned-above. First, the predicted bounding boxes do not fit in airport buildings. Next, there are many buildings in the airport which cannot be detected, because of the complicated shapes and small buildings.

Therefore, we use the YOLOv3 architecture instead of the original YOLO architecture to solve this problem that the model cannot detect the small objects in the image. Because the YOLOv3 has been improved accuracy and efficiency of the backbone structure and the predicted bounding boxes. Then, it provides better results than original architecture as illustrated in Figures 4.7, 4.6, and 4.8.

Note that:

- **Red boxes** represent predicted bounding boxes
- **Black boxes** represent ground truth bounding boxes

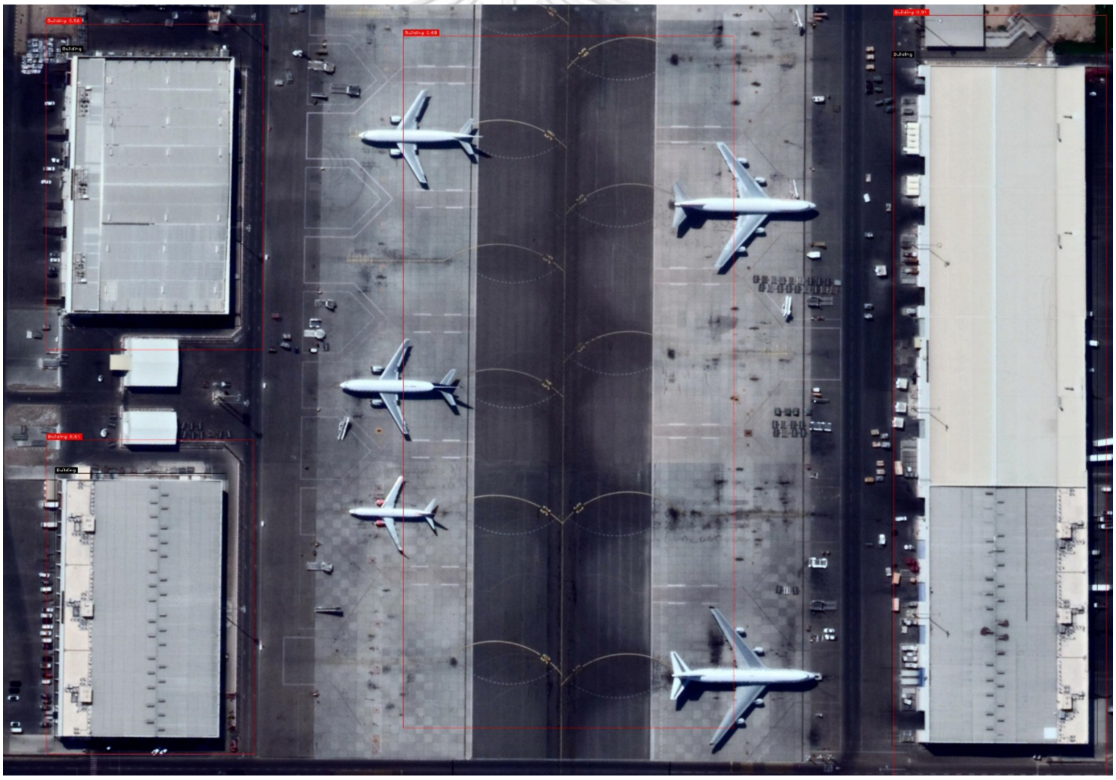


Figure 4.6: Example images of the result using the original YOLOv3 (1378 x 958 pixels)

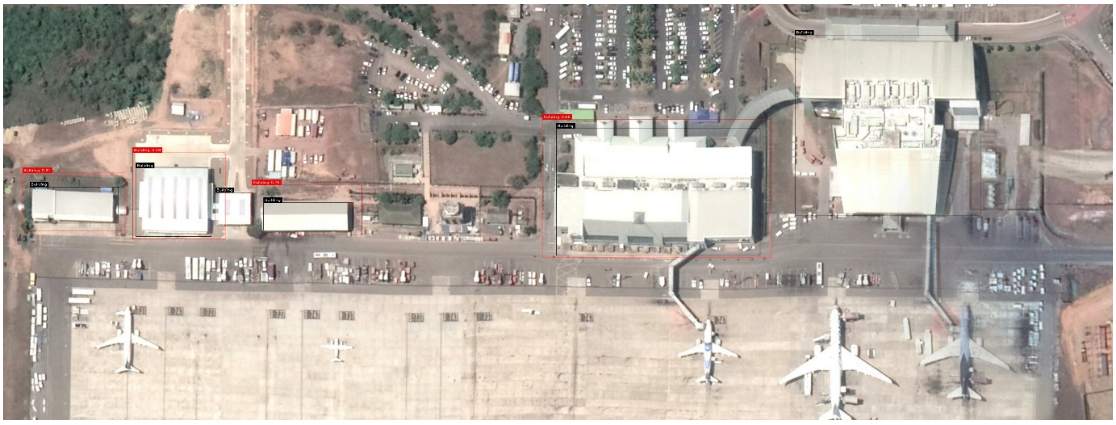


Figure 4.7: Example images of the result using the original YOLOv3 (1375 x 515 pixels) [24]

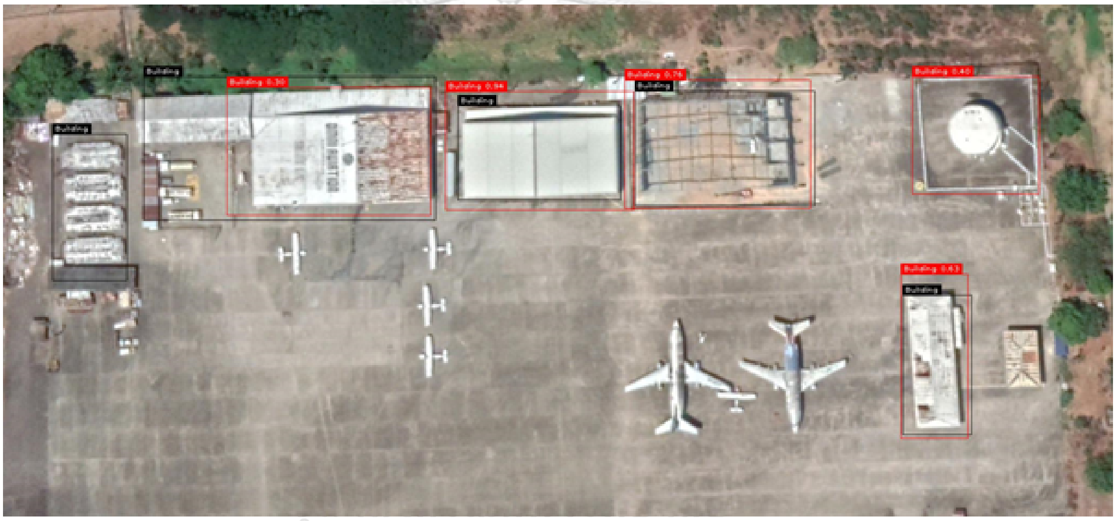


Figure 4.8: Example images of the result using the original YOLOv3 (601 x 277 pixels)

Chulalongkorn University

From the example images of the result using the original YOLOv3, it can be noted that the detection performance has improved when we switch to the YOLOv3 architecture, which allows the model to detect buildings in airports with greater accuracy. However, it has failure detection which means that the model cannot fully detect the building due to the possibility of obstructing the building or a variety of shapes. It also detects non-building that is of interest to us because its color or shape is similar to its background. So we come up with a solution to this problem that we try to get rid of the background or non-building stuff using the Jet saliency map (see in Chapter III) to solve the problem and increase the efficiency of the YOLOv3 architecture. We test our improved model using 190 images, which contain 809 target objects for detection. These images used for testing have been manually labeled to reflect the actual ground truth. In addition, the efficiency of our proposed algorithm has also been determined via several measurements using the value of IoU = 0.5. Experimental results are demonstrated in Table 4.1, which compares the original model of YOLOv3 and our improved model based on YOLOv3 with Jet saliency map.

Table 4.1: Detection results on the testing set

Measurements	Original model (%)	Our improved model (%)
Precision	74.5209	85.3254
Recall	81.7058	89.1223
Average Precision	75.6322	85.4174
F1-Score	77.9481	87.1826

From Table 4.1, we can see that our improved model using the Jet saliency map provides higher accuracy than the original method of up to about 10 percent for all measurements. Hence, the proposed model is more efficient and has a significantly improved level of accuracy.

The measurement values from Table 4.1 can be derived from the calculation, which can be displayed from plotting graphs as shown in Figures 4.9 and 4.10.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

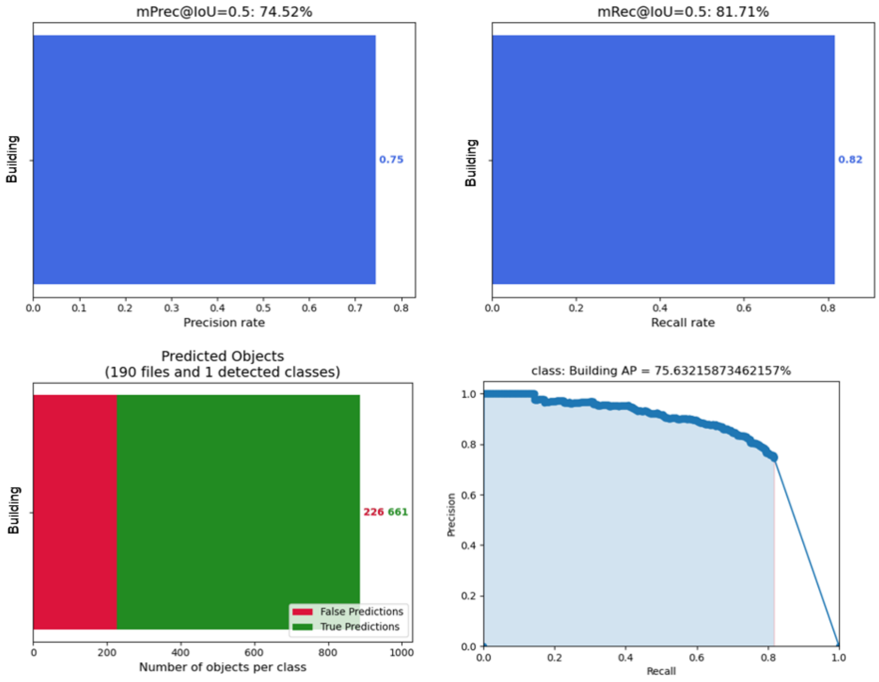


Figure 4.9: Graphs show the results of the measurement values obtained from YOLOv3

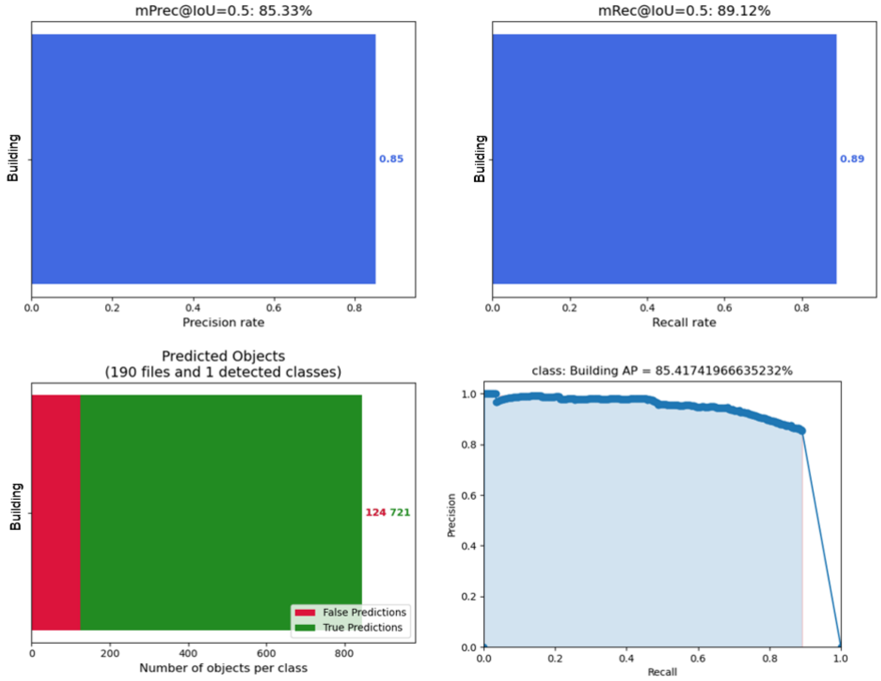


Figure 4.10: Graphs show the results of the measurement values obtained from YOLOv3 with Jet saliency map

The resulting images from our improved model based on YOLOv3 with Jet saliency map are shown in the example images of the result from using the original YOLOv3 with the Jet saliency map as follows

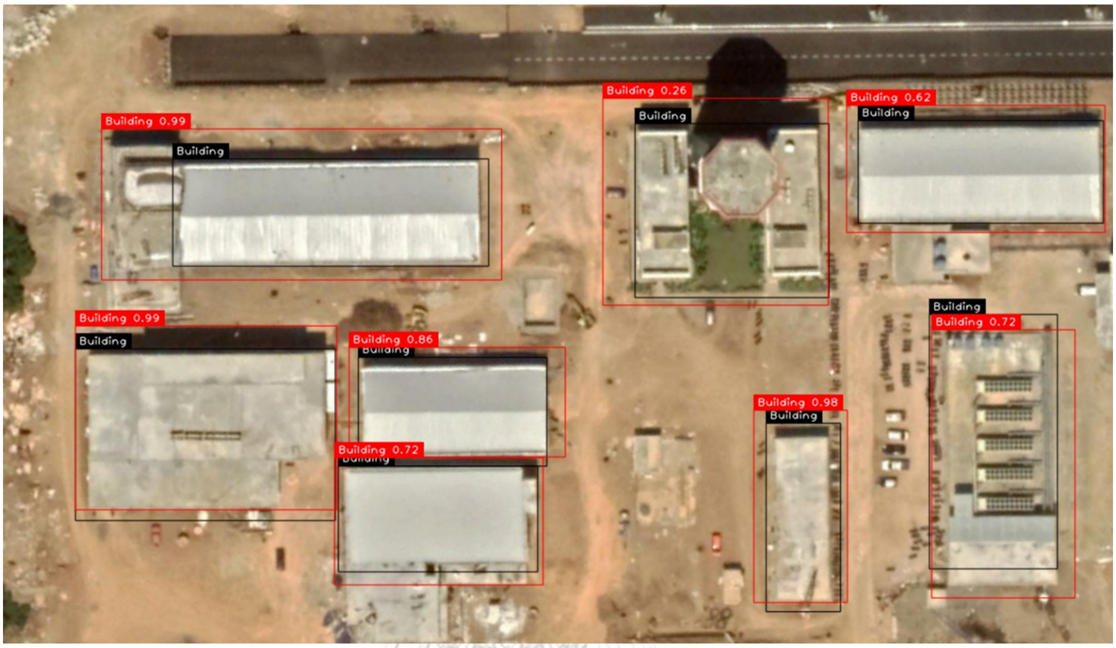


Figure 4.11: Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 793 pixels)

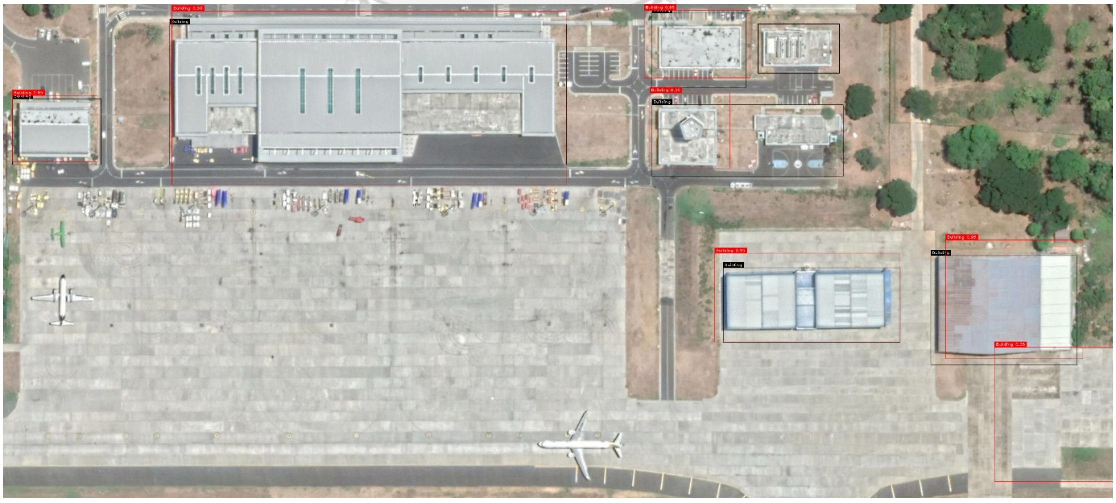


Figure 4.12: Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 613 pixels)

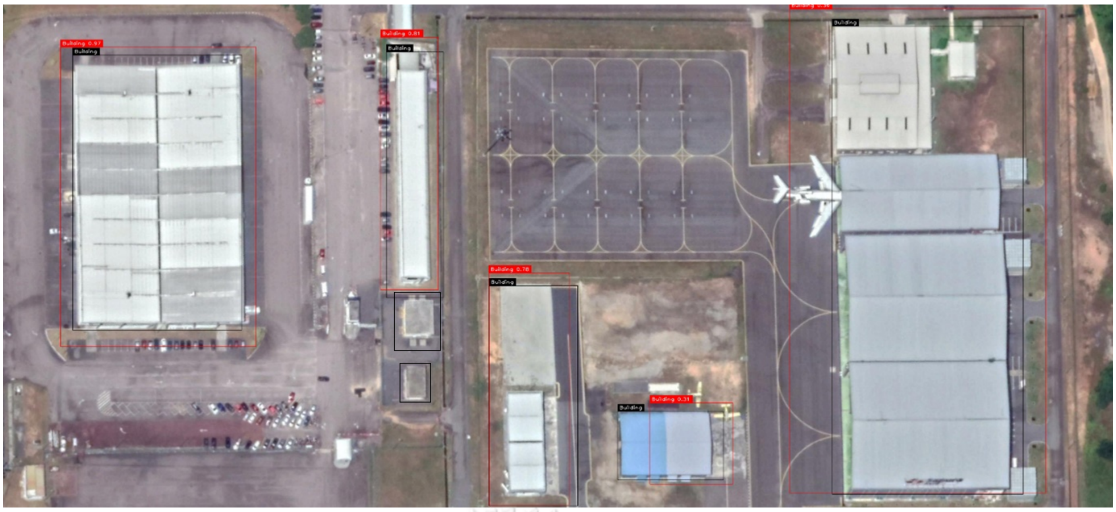


Figure 4.13: Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 623 pixels)

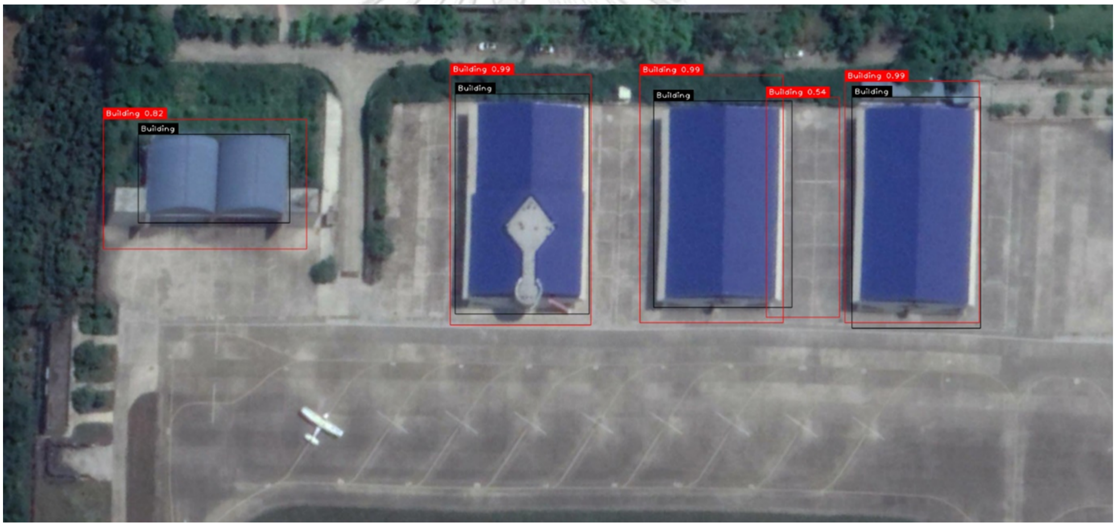


Figure 4.14: Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 643 pixels)

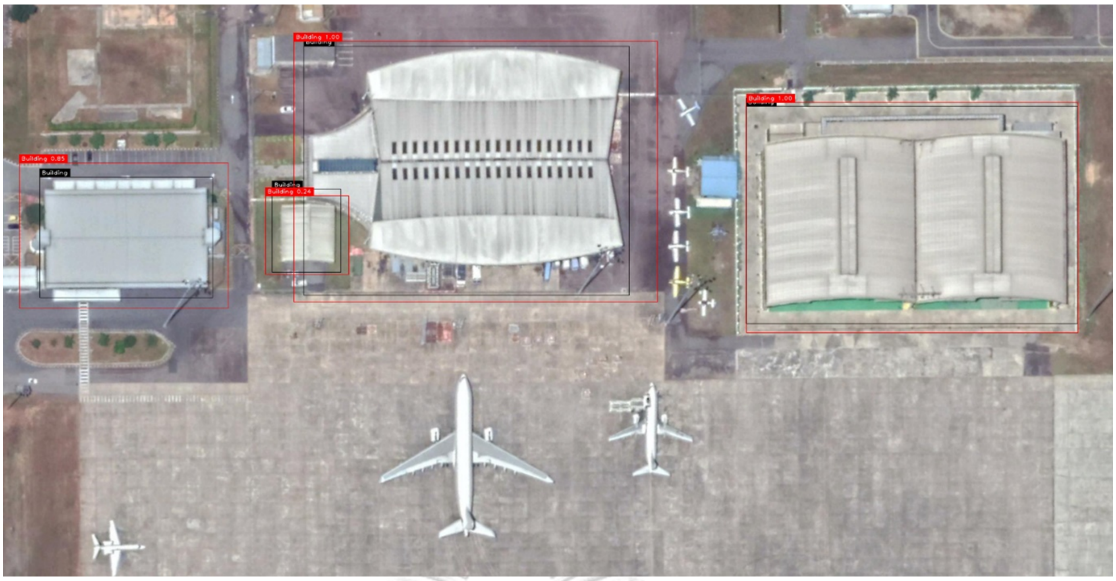


Figure 4.15: Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 709 pixels)

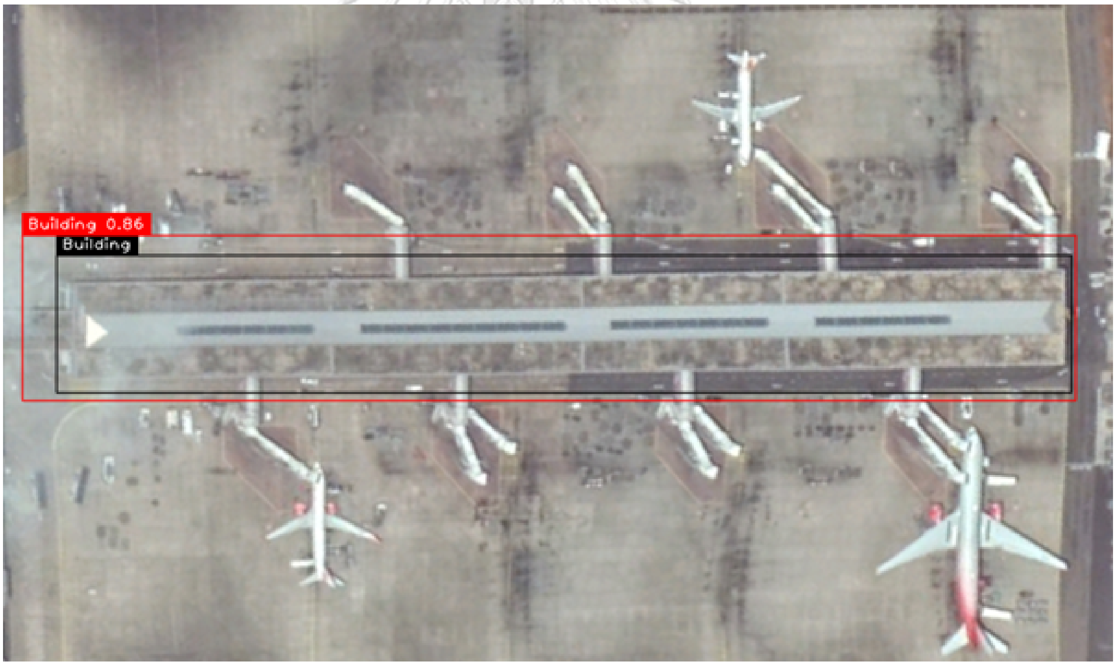


Figure 4.16: Example images of the result using the original YOLOv3 with the Jet saliency map (601 x 356 pixels)

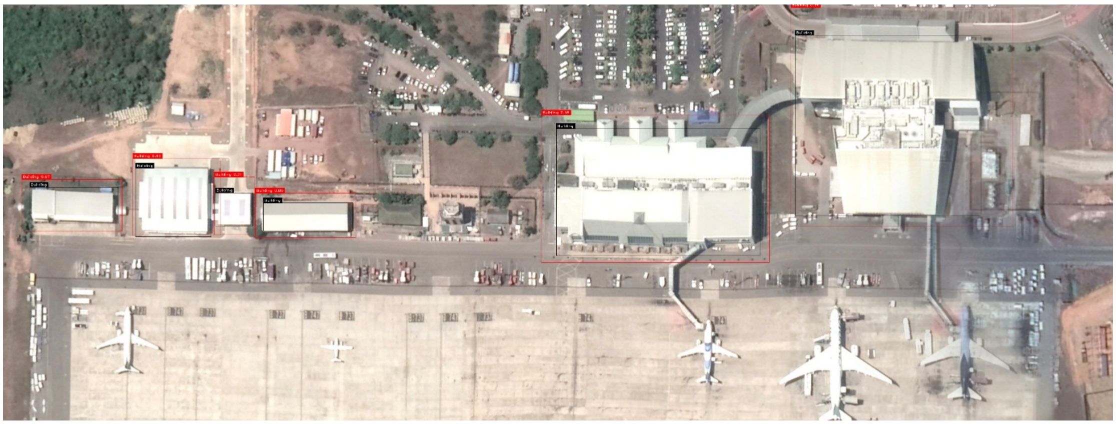


Figure 4.17: Example images of the result using the original YOLOv3 with the Jet saliency map (1375 x 515 pixels) [24]

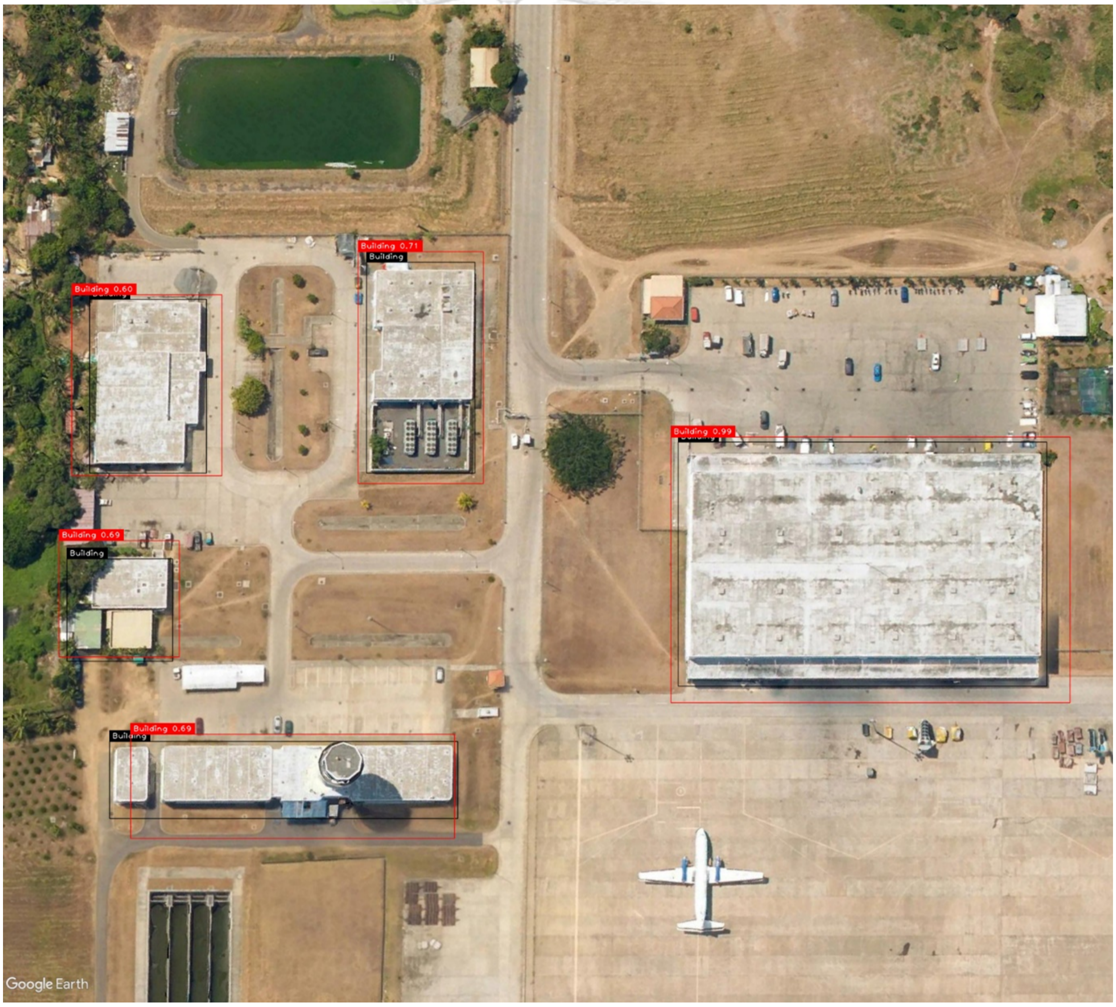


Figure 4.18: Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 1239 pixels)

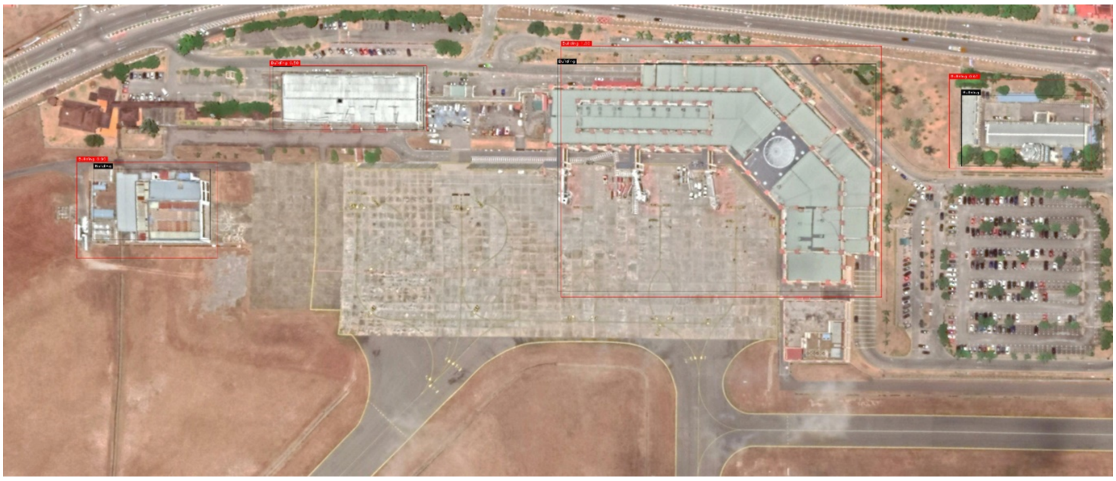


Figure 4.19: Example images of the result using the original YOLOv3 with the Jet saliency map (1375 x 584 pixels)

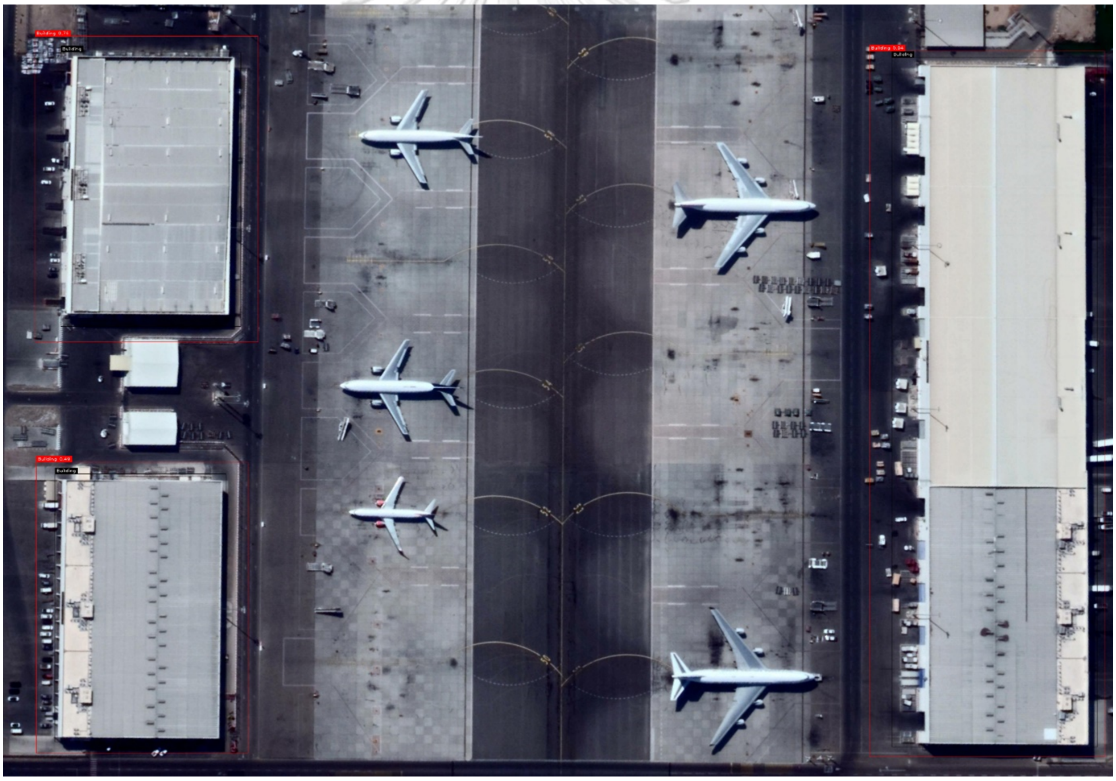


Figure 4.20: Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 958 pixels)

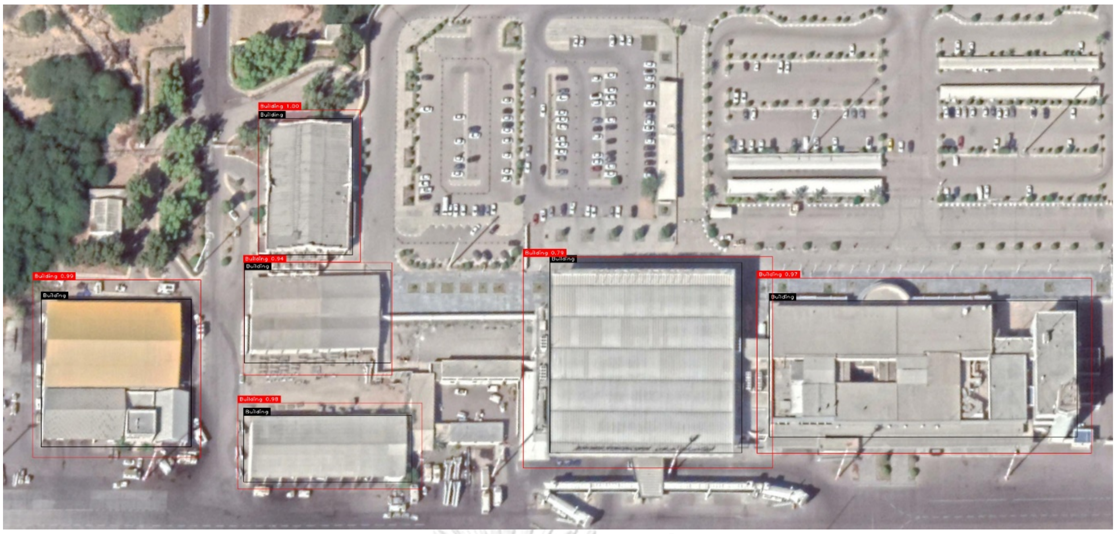


Figure 4.21: Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 651 pixels)

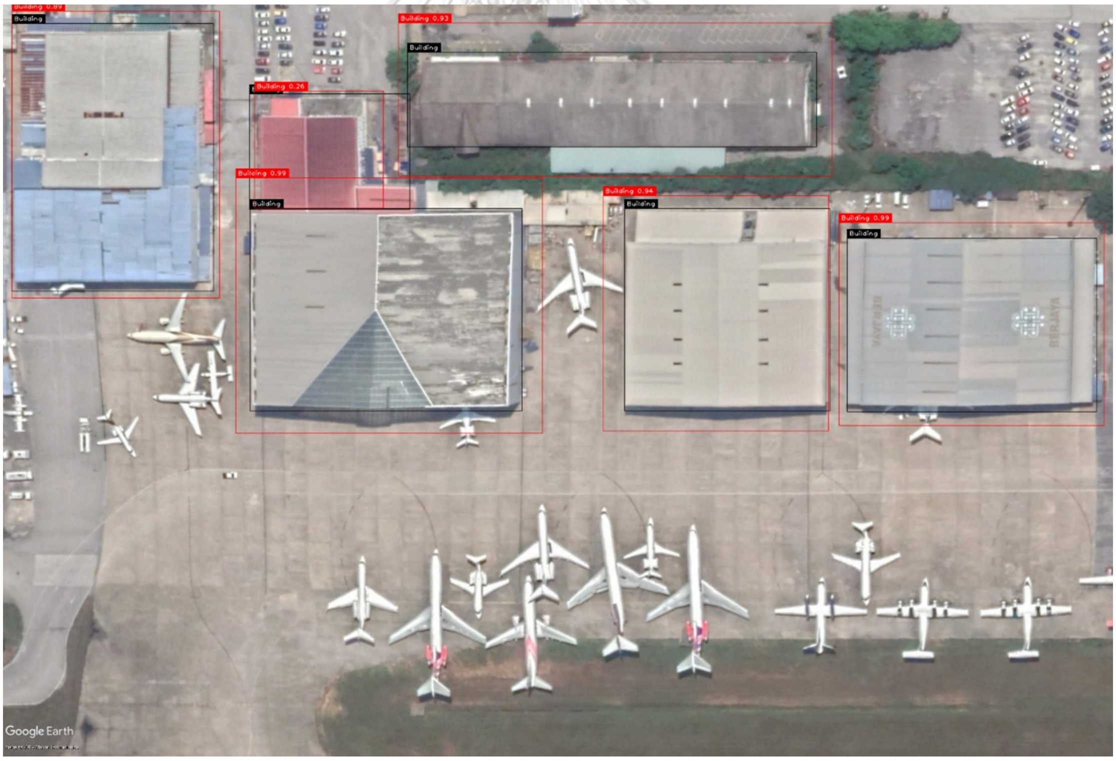


Figure 4.22: Example images of the result using the original YOLOv3 with the Jet saliency map (1377 x 932 pixels)

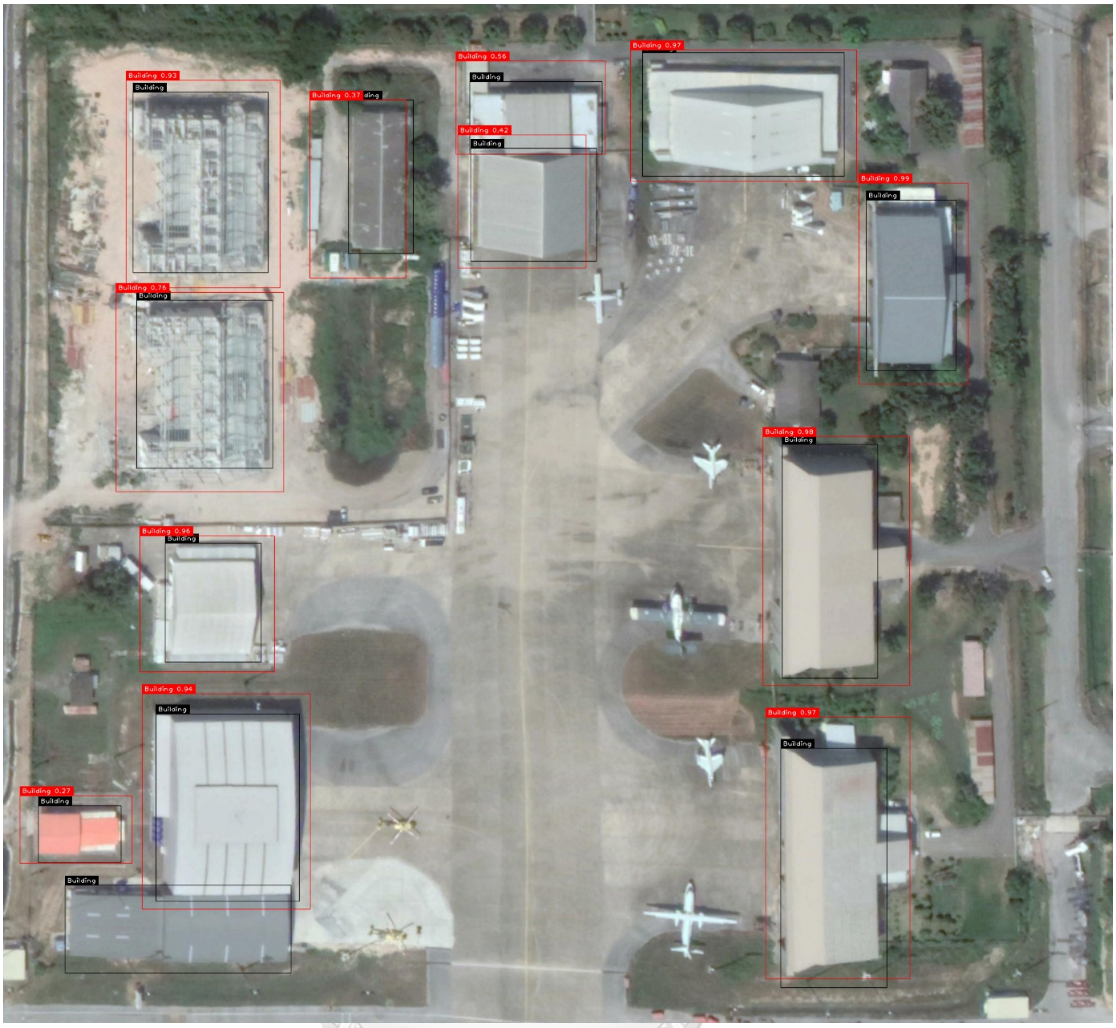


Figure 4.23: Example images of the result using the original YOLOv3 with the Jet saliency map (1384 x 1271 pixels)

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

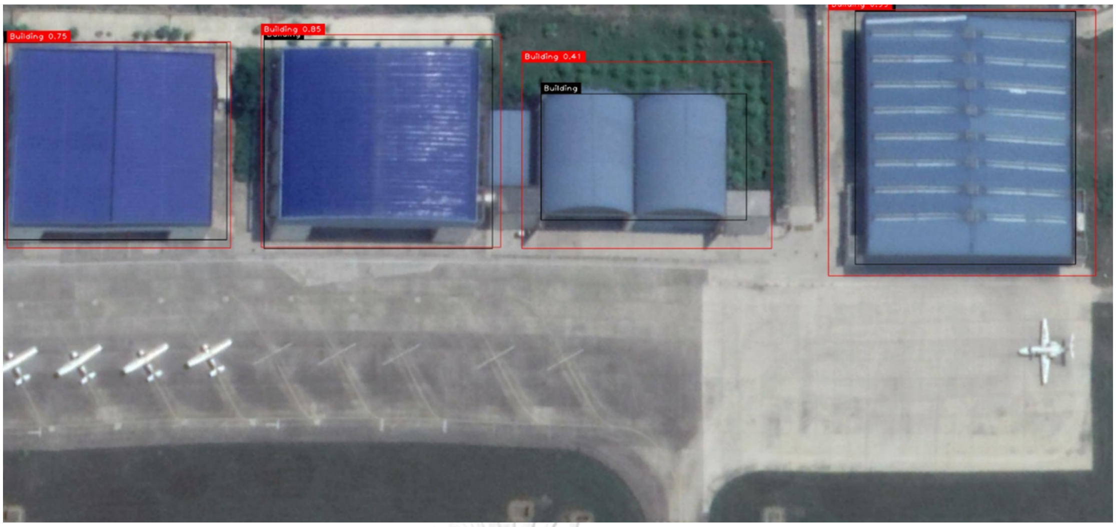


Figure 4.24: Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 643 pixels)

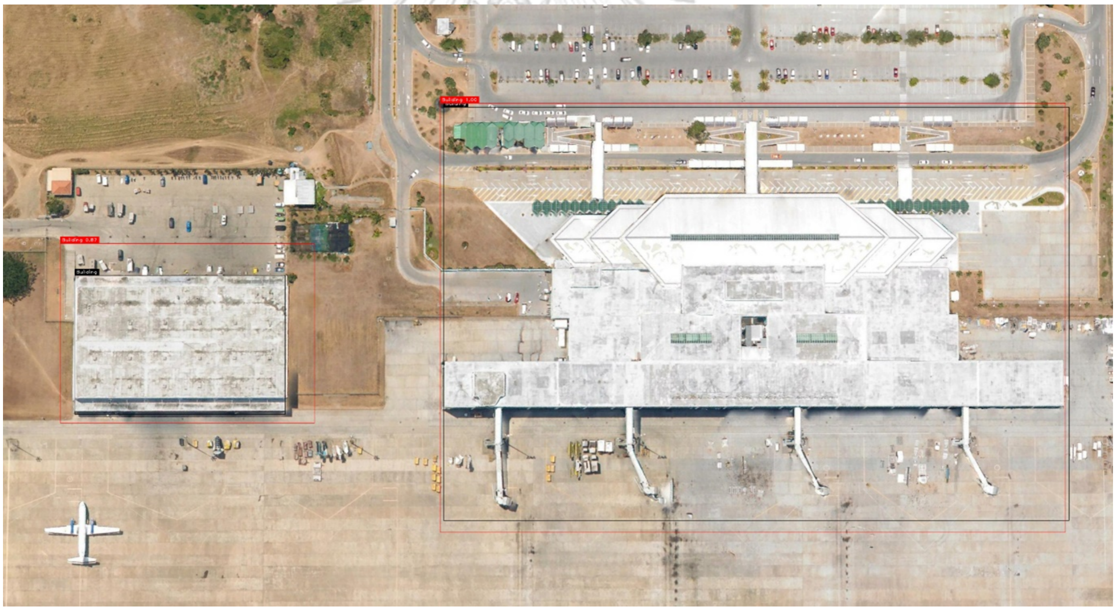


Figure 4.25: Example images of the result using the original YOLOv3 with the Jet saliency map (1378 x 747 pixels)

CHAPTER V

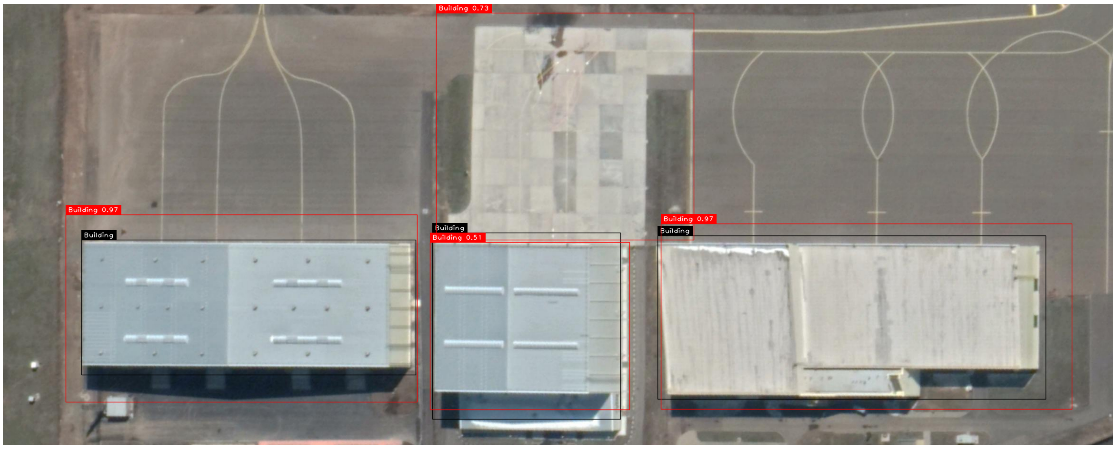
CONCLUSIONS

In this thesis, we have proposed a model for the detection of buildings at airports. The major idea behind this work starts with combining the 1D saliency map image with the 3D RGB image to acquire a novel image in 4D. The YOLOv3 model has also been improved to manipulate the complexity, noise, and small objects in the image. We have also been able to separate the foreground from the background in the images to easily detect the buildings of interest. From experiments, we found that the saliency map process can be applied to improve the model efficiently. It produces highly accurate results that can be seen in chapter IV. However, our proposed model has some limitations, i.e., it can be inaccurate in terms of detection when the buildings have a shape similar to other objects, or when the background color or objects is similar to the color of the target buildings as shown in Figure 5.1. Additionally, our model also consumes more time than the original model. Therefore, in our future work, we expect to handle these issues, as well as extend the model to detect objects in other scopes.

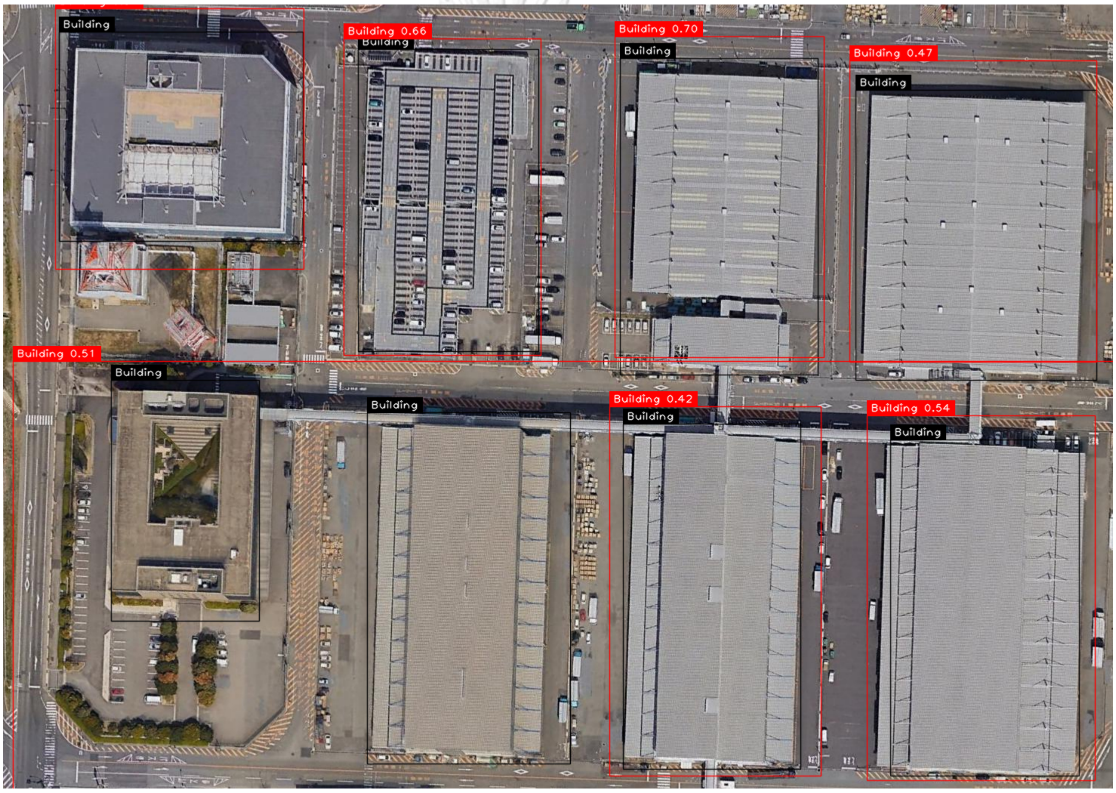
5.1 Future Work

In this work, the proposed method aims to detect airport buildings from remote sensing images as much as possible. However, our proposed method has some limitations when the other objects look like similar to the airport buildings which are caused to incorrect detection. Therefore, in future works, we will find some ways to decrease this proposed method's false positive. We attempt to improve the Jet saliency map for reducing noise better, the detector is then a more accurate and efficient detector. Besides, we will do more experiments about a dataset with a depth channel of the object in the image.

Chulalongkorn University



(a) Detecting objects rather than target buildings



(b) Incomplete detection of the target buildings

Figure 5.1: Example of failure detection [24]

REFERENCES

[1] Google, *Google Earth*, 2019. https://www.google.co.th/intl/th/earth.

[2] UCEI, *Unity Computer Education Institute Sagar*, 2021. https://unitycomputer.websites.co.in/.

[3] I. News, *Some permanent resident applications now exempt from biometrics requirement*, 2021. https://www.immigratetocanada.com/some-permanent-resident-applications-now-exempt-from-biometrics-requirement/.

[4] M. A. Reddy, *Satellite Imagery*, 2021. https://uoqasim.edu.iq/e_Learning/lec_file/Lecture2.pdf.

[5] T. komentar, *Robot Firefighter*, 2021. https://rangkairobotic.blogspot.com/2018/10/robot-firefighter.html.

[6] H. Mishra, *Write a Simple Steganography Program Using Python*, 2021. https://fullstackfeed.com/write-a-simple-steganography-program-using-python/.

[7] Sfjohnso, *Exploring Color Space*, 2021. https://www.instructables.com/Exploring-Color-Space/.

[8] Pngitem, *Hexagonal cone hsv color model*, 2021. https://www.pngitem.com/middle/ThRhmTT_hexagonal-cone-hsv-color-model-hd-png-download/.

[9] N. Promrit, *Introduction to Deep Learning (Machine Learning Pipeline)*, 2021. https://blog.pjjop.org/deep-learning/.

[10] O. Forge, *Colormap*, 2021. https://octave.sourceforge.io/octave/function/jet.html.

[11] E. van der Velden, “Cmasher: Scientific colormaps for making accessible, informative and’cmashing’plots,” *arXiv preprint arXiv:2003.01069*, 2020.

[12] K. Anton, *Heaviside step function in asy-code from wiki*, 2021. https://artofproblemsolving.com/community/c2083h1565545_heaviside_step_function_in_asycode_from_wiki.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

[13] R. Nagyfi, *The differences between Artificial and Biological Neural Networks*, 2021. `https://towardsdatascience.com/the-differences-between-artificial-and-biological-neural-networks-a8b46db828b7`.

[14] N. Phongchit, *Convolutional Neural Network*, 2021. `https://medium.com/@natthawatphongchit`.

[15] V. ary team, *Boost the world: cat detection*, 2021. `http://www.vision-ary.net/2015/03/boost-the-world-cat-faces/`.

[16] R. Zhu, S. Zhang, X. Wang, L. Wen, H. Shi, L. Bo, and T. Mei, "Scratchdet: Training single-shot object detectors from scratch," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2268–2277, 2019.

[17] D. Gutierrez, *Overview of the YOLO Object Detection Algorithm*, 2021. `https://opendatascience.com/overview-of-the-yolo-object-detection-algorithm/`.

[18] ProgrammerSought, *The whole network of hematemesis: analysis of the principle of YOLO v3*, 2021. `https://www.programmersought.com/article/96417192502/`.

[19] R. F. de Azevedo Kanehisa and A. de Almeida Neto, "Firearm detection using convolutional neural networks.," in *ICAART (2)*, pp. 707–714, 2019.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[21] A. Musib, *Playing with YOLO v1 on Google Colab*, 2021. `https://www.analyticsvidhya.com/blog/2020/08/playing-with-yolo-v1-on-google-colab/`.

[22] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[23] H. Zhao, Y. Zhou, L. Zhang, Y. Peng, X. Hu, H. Peng, and X. Cai, "Mixed yolov3-lite: A lightweight real-time object detection method," *Sensors*, vol. 20, no. 7, p. 1861, 2020.

[24] N. Pumpong, P. Boonserm, K. Kobayashi, and N. Cooharojananone, "Building detection in airports through remote sensing image using yolov3 with jet saliency map," in *2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA)*, pp. 491–496, IEEE, 2021.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

[25] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” in *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6, Ieee, 2017.

[26] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-based convolutional networks for accurate object detection and segmentation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 142–158, 2015.

[27] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *arXiv preprint arXiv:1506.01497*, 2015.

[28] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[29] Z. Zheng, Y. Liu, C. Pan, and G. Li, “Application of improved yolo v3 in aircraft recognition of remote sensing images,” *Electronics Optics & Control*, vol. 26, no. 4, pp. 28–32, 2019.

[30] Y. Sun, N. Cooharojananone, and H. Ochiai, “Aircraft detection based on saliency map and convolution neural network,” in *23rd International Computer Science and Engineering Conference (ICSEC)*, pp. 48–53, IEEE, 2019.

[31] T. Lilek and N. Cooharojananone, “Aircraft detection from remote sensing images using single shot scale-invariant face detector with viridis saliency map,” in *2020 International Conference on Mathematics and Computers in Science and Engineering (MACISE)*, pp. 171–174, IEEE, 2020.

[32] X. Ren, S. Du, and Y. Zheng, “Parallel rcnn: A deep learning method for people detection using rgb-d images,” in *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1–6, IEEE, 2017.

[33] F. Crameri, G. E. Shephard, and P. J. Heron, “The misuse of colour in science communication,” *Nature communications*, vol. 11, no. 1, pp. 1–10, 2020.

[34] P. Wang, Z. Li, Y. Hou, and W. Li, “Action recognition based on joint trajectory maps using convolutional neural networks,” in *Proceedings of the 24th ACM international conference on Multimedia*, pp. 102–106, 2016.

[35] L. Li, L. Qin, Z. Xu, Y. Yin, X. Wang, B. Kong, J. Bai, Y. Lu, Z. Fang, Q. Song, *et al.*, “Using artificial intelligence to detect covid-19 and community-acquired pneumonia based on pulmonary ct: evaluation of the diagnostic accuracy,” *Radiology*, vol. 296, no. 2, pp. E65–E71, 2020.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

[36] G. A. Miller, “Wordnet: a lexical database for english,” *Communications of the ACM*, 1995.

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[38] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, “Going deeper in spiking neural networks: Vgg and residual architectures,” *Frontiers in neuroscience*, vol. 13, p. 95, 2019.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

VITA

Name	Mr. Noppadon Pumpong
Date of Birth	August 31, 1995
Place of Birth	Uthai Thani, Thailand
Education	B.Sc. (Mathematics, First Class Honours), Kasetsart University, 2017
Scholarship	Development and Promotion of Science and Technology Talents Project (DPST) Scholarships
Publications	N. Pumpong, P. Boonserm, K. Kobayashi, and N. Cooharojananone, “Building detectionin airports through remote sensing image using yolov3 with jet saliency map,” in 2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA), pp. 491–496, IEEE, 2021.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY