



มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระบรมราชูปถัมภ์
ภาควิชาคอมพิวเตอร์ศึกษา คณะครุศาสตร์อุตสาหกรรม



เอกสารประกอบการอบรม

พื้นฐานหัวข้อในการอบรม

- Introduction to Image Processing
- Object Detection
- Optical Character Recognition
- Face Recognition

จัดทำโดย
วิทยากรฝึกอบรม IMAGE RECOGNITION WITH PYTHON



สารบัญ

1.ปัญญาประดิษฐ์ (AI : Artificial Intelligence).....	1
เหตุใดปัญญาประดิษฐ์จึงมีความสำคัญ	1
Mind Map ของ AI	2
Machine Learning	2
Computer Vision	3
เป้าหมายของ Computer vision	3
การประยุกต์ใช้ Computer Vision.....	4
ตัวอย่างการใช้ Computer Vision.....	4
การทำงานของ Computer Vision.....	5
Digital image (ภาพดิจิทัล).....	5
Pixel	6
Binary Image.....	6
Grayscale Image	6
RGB Image	7
Image Processing	8
เทคนิคที่ใช้ในการประมวลผลภาพ	8
ขั้นตอนการทำงานของ Computer Vision.....	10
ประเภทของอัลกอริทึมการจำแนกประเภท	12
โครงข่ายประสาทเทียม (Artificial neural networks: ANN).....	13
Convolutional Neural Network (CNN).....	15
ขั้นตอนการทำ CNN มี 4 ขั้นตอน	16
2.การจำแนกตรวจจับวัตถุในรูปภาพด้วย Yolo	19
Yolo คืออะไร	19
ตัวอย่างการทำ Object Detection ด้วย yolo	20
ขั้นตอนของ Non-maximum Suppression	23
ภาพรวมขั้นตอนของการ Object Detection ด้วย Yolo.....	24
การระบุคลาสให้แต่ละชิ้นส่วนของชิ้นสีเหลี่ยม	24
เปรียบเทียบ Yolo กับรูปแบบการตรวจจับกับอัลกอริทึมอื่น ๆ	26
ตัวอย่างการนำไปใช้งาน.....	27
การติดตั้ง Anaconda.....	28
ติดตั้ง PowerShell Prompt	30
Jupyter	31

Opencv.....	35
NUMPY	36
YOLO.....	37
COCO	42
เริ่มการเขียนโค้ด	47
3. Optical Character Recognition.....	52
ประเภทของเทคนิคการรู้จำตัวอักษรด้วยแสง.....	52
การรู้จำตัวอักษรแบบออนไลน์ (On-line Character Recognition).....	53
การเรียนรู้ตัวอักษรแบบออฟไลน์ (Off-line Character Recognition)	53
ขั้นตอนเทคนิคการรู้จำตัวอักษรด้วยแสง (Optical Character Recognition)	54
ขบวนการประมวลผลขั้นต้น (Pre-Processing)	54
การรู้จำ (Recognition)	58
ขบวนการประมวลผลขั้นสุดท้าย (Post-Processing).....	59
เครื่องมือ OCR Open source.....	59
Tesseract OCR	59
โครงสร้างการประมวลผลปกติ (เวอร์ชันก่อนหน้า).....	60
โครงสร้างการประมวลผลเสริมด้วยเทคนิค LSTM (เวอร์ชัน 4)	60
ขั้นตอนการรู้จำของ Tesseract.....	60
การติดตั้ง Tesseract.....	61
การสกัดข้อความจากรูปภาพ	63
การสกัดข้อความแบบตามเวลาจริง.....	64
4. Face Recognition	65
ประวัติของระบบจดจำใบหน้า	65
การตรวจหาใบหน้า	66
วิธีการทำงานของ Face Recognition.....	66
Modern Face Recognition with Deep Learning	66
Histogram of Oriented Gradients (HOG).....	66
ประโยชน์ของ Face Recognition.....	70
การติดตั้ง Face Recognition.....	71
Work shop 1 face_rec_img.....	73
Work shop 2 face_rec_camara.....	76

สารบัญรูปภาพ

FIGURE 1 MIND MAP ARTIFICIAL INTELLIGENCE	2
FIGURE 2 การทำงานของ COMPUTER VISION	5
FIGURE 3 ภาพ BINARY หรือ ภาพขาว-ดำ	6
FIGURE 4 ภาพ GRayscale	6
FIGURE 5 ภาพ RGB.....	7
FIGURE 6 GREYSCALE IMAGE (ภาพระดับสีเทา)	7
FIGURE 7 ภาพที่มีนุชย์มองเห็น และภาพที่คอมพิวเตอร์เห็น	8
FIGURE 8 PIPELINE COMPUTER VISION	10
FIGURE 9 INPUT DATA.....	10
FIGURE 10 การประมวลผลภาพ	10
FIGURE 11 กระบวนการ FEATURE EXTRACTION	11
FIGURE 12 โครงข่ายประสาทเทียม (ARTIFICIAL NEURAL NETWORKS: ANN)	13
FIGURE 13 การคำนวณผลรวมของอินพุต ด้วยฟังก์ชันการแปลง	14
FIGURE 14 CONVOLUTIONAL NEURAL NETWORK (CNN)	15
FIGURE 15 การแบ่งรูปภาพออกเป็น MATRIX	15
FIGURE 16 CONVOLUTION	16
FIGURE 17 การทำการคูณเมตริกซ์	16
FIGURE 18 การสกัด FEATURE	16
FIGURE 19 MAX POOLING	17
FIGURE 20 FLATTENING	17
FIGURE 21 FULL CONNECTION	18
FIGURE 22 FAST SINGLE-SHOT DETECTION	19
FIGURE 23 อัลกอริทึม YOLO	19
FIGURE 24 ลักษณะการทำงานต่าง ๆ	20
FIGURE 25 กระบวนการ INTERSECTION OVER UNION (IoU).....	23
FIGURE 26 ภาพรวมขั้นตอนของการ OBJECT DETECTION ด้วย YOLO	24
FIGURE 27 การเปรียบเทียบโมเดลต่าง ๆ ที่สามารถนำมาราทำ OBJECT DETECTION	26
FIGURE 28 การตรวจจับหนอกนิรภัยและการใช้อาวุธปืน เพื่อเตือนภัยเหตุกรรม	27
FIGURE 29 การสร้างระบบตรวจจับบุคคลแบบเวลาจริง	27
FIGURE 30 กระบวนการประมวลผล YOLO	51
FIGURE 31 เทคนิคการรู้จำตัวอักษรด้วยแสง	52
FIGURE 32 ประเภทของเทคนิคการรู้จำตัวอักษรด้วยแสง	52

FIGURE 33 การรู้จำตัวอักษรแบบออนไลน์ (ON-LINE CHARACTER RECOGNITION)	53
FIGURE 34 ตัวพิมพ์แบบฟอนต์เฉพาะ	53
FIGURE 35 ลายมือเขียนแบบตัวโดด	54
FIGURE 36 ตัวอักษรลายมือแบบเขียนต่อเนื่อง	54
FIGURE 37 ภาพที่เกิดปัญหาสัญญาณรบกวน	55
FIGURE 38 พังค์ชัน THRESHOLD 5 แบบ	55
FIGURE 39 เครื่องมือ OCR OPEN SOURCE	59
FIGURE 40 กระบวนการทำงานของ TESSERACT OCR	59
FIGURE 41 โครงสร้างการประมวลผลปกติ	60
FIGURE 42 โครงสร้างการประมวลผลเสริมด้วยเทคนิค LSTM	60
FIGURE 43 ขั้นตอนการรู้จำของ TESSERACT	60
FIGURE 44 เทคโนโลยีการจดจำใบหน้า	65
FIGURE 45 FACE RECOGNITION SYSTEM	66
FIGURE 46 กระบวนการ FACE RECOGNITION	66
FIGURE 47 การนำเข้ารูปภาพเพื่อประมวลผลสำหรับ HISTOGRAM OF ORIENTED GRADIENTS (HOG)	67
FIGURE 48 การค้นหาใบหน้า HISTOGRAM OF ORIENTED GRADIENTS (HOG)	67
FIGURE 49 ผลลัพธ์จากการค้นหาตำแหน่งใบหน้าด้วย HISTOGRAM OF ORIENTED GRADIENTS (HOG)	68
FIGURE 50 การกำหนดจุดสำคัญบนใบหน้า (FACE LANDMARKS) การกำหนดจุดแบบ 5 จุด (ซ้าย) การกำหนดจุดแบบ 68 จุด (ขวา)	68
FIGURE 51 ค่าคุณลักษณะใบหน้าที่ได้ 128 มิติ	68
FIGURE 52 กระบวนการ TRIPLET LOSS เพื่อการรู้จำใบหน้า	69
FIGURE 53 ผลลัพธ์ที่ได้จากการรู้จำใบหน้า	69
FIGURE 54 FACE RECOGNITION FROM CONVOLUTIONAL NEURAL NETWORK (CNN)	70

1. ปัญญาประดิษฐ์ (AI : Artificial Intelligence)

ปัญญาประดิษฐ์ (AI: Artificial Intelligence) คือ การเรียนรู้ที่เอาประสบการณ์ เอ้าข้อมูลหรือสิ่งที่ได้รับรู้มาก่อนในอดีตมาวิเคราะห์ว่าในอนาคตจะมีข้อมูลที่มีลักษณะคล้ายกันเข้ามาแล้วคำตอบมันคืออะไร แล้วนำข้อมูลที่มีอยู่มาใส่เข้าไปและเอาผลลัพธ์ที่รู้คำตอบอยู่แล้วมาสอน AI หลังจากนั้นเอ้าข้อมูลที่มีเงื่อนไขต่างกันเข้าไปใหม่เพื่อให้ AI เริ่มเรียนรู้ว่าสิ่งที่ใส่เข้ามาคำตอบคืออะไร แต่ทั้งนี้ทั้งนั้น ข้อมูลต้องมีมากพอที่จะทำให้ AI เอ้าไปวิเคราะห์ว่าอะไรจริงหรือไม่จริง เมื่อไหร่ก็ตามที่ AI ได้ข้อมูลมากขึ้น การฝึกฝนก็จะแม่นยำขึ้นเท่านั้น ที่สำคัญข้อมูลที่ได้จะต้องไม่มีความผิดพลาดในข้อมูล หากมีข้อมูลผิดพลาดอยู่ AI ก็จะเรียนรู้ความผิดพลาดตรงนั้นด้วย

เหตุใดปัญญาประดิษฐ์จึงมีความสำคัญ

- AI มีการเรียนรู้ซ้ำ ๆ ได้อย่างอัตโนมัติและศึกษาผ่านข้อมูลเหล่านั้น แต่ AI นั้นก็มีความแตกต่างจากทุนนิยมหรืออุปกรณ์อัตโนมัติ แทนที่จะประมวลผลงานแบบมนุษย์ AI สามารถประมวลผลในงานซ้ำ ๆ ที่มีปริมาณมากด้วยความเที่ยงตรงและมีประสิทธิภาพผ่านระบบคอมพิวเตอร์ สำหรับการประมวลผลการทำงานอัตโนมัติด้วยวิธีนี้ ยังจำเป็นที่จะต้องใช้มนุษย์ในการติดตั้งระบบและป้อนคำสั่งที่เหมาะสมด้วยเช่นกัน
- AI เพิ่มความชาญฉลาด แก่สิ่งที่มีอยู่เดิม โดยทั่วไปจะไม่มีการจำหน่วย AI ในรูปแบบแอปพลิเคชันเดียว หากแต่จะใช้ประสิทธิภาพของ AI ในการพัฒนาสิ่งที่มีอยู่เดิม หากจะยกตัวอย่างนั้นก็คือ Siri ที่ได้รับการติดตั้งเพิ่มใน iOS ใหม่ ๆ ของ Apple เป็นต้น
- AI เรียนรู้จากอัลกอริทึมการเรียนรู้แบบก้าวหน้า (progressive) อัลกอริทึมจะกล้ายเป็นตัวแยกประเภทหรือพยากรณ์ ดังนั้นอัลกอริทึมจะสามารถเรียนรู้วิธีการเล่นมากruk และเรียนรู้ว่าควรจะเดินหมายตัวใดในตาถัดไป ซึ่งแบบจำลองประเภทนี้จะได้รับการปรับให้ดีขึ้นเมื่อได้รับข้อมูลใหม่ กระบวนการส่งค่าย้อนกลับ คือเทคนิคนึงของ AI ใน การปรับแต่งแบบจำลองผ่านการฝึกฝนและข้อมูลเพิ่มเมื่อผลลัพธ์ครั้งแรกยังไม่ถูกต้องนัก
- AI จะวิเคราะห์ข้อมูลมากกว่าและลึกกว่า โดยใช้เครือข่ายประสาทเทียม (neural network) ที่มีหลายชั้น จำเป็นต้องใช้ข้อมูลปริมาณมากในการพัฒนาด้านการเรียนรู้เชิงลึกของแบบจำลองเนื่องจากแบบจำลองเหล่านี้จะเรียนรู้จากข้อมูลโดยตรง ยิ่งป้อนข้อมูลบริมามากขึ้นเท่าใด แบบจำลองก็จะยิ่งก่อให้เกิดความแม่นยำมากขึ้นเท่านั้น

- AI สามารถสร้างความแม่นยำอย่างเหลือเชื่อ ผ่านเครือข่ายประสาทเทียม (neural network) ยกตัวอย่าง เช่น การติดต่อกับ Alexa Google Search และ Google Photos ล้วนใช้เทคนิคการเรียนรู้เชิงลึก (deep learning) ทั้งนั้น และโปรแกรมเหล่านี้ยังมีความแม่นยำมากยิ่งขึ้นตามการใช้งานที่เพิ่มขึ้น
- AI สามารถใช้ประโยชน์อย่างสูงสุดจากข้อมูลที่มี เมื่อวัลกอริทึมสามารถเรียนรู้ได้ด้วยตัวเอง ข้อมูล ก็จะกลายเป็นทรัพย์สินทางปัญญาอันมีค่า ความลับซ่อนอยู่ในข้อมูลนั้นเอง เพียงแค่สามารถประยุกต์ใช้ AI เพื่อถึงทำความลับนั้นออกมานะ เนื่องจากบทบาทของข้อมูลนั้นบ่งบอกความสำคัญมากกว่าที่เคยเป็นมา มันสามารถก่อให้เกิดความได้เปรียบทางการแข่งขัน หากมีข้อมูลที่ดีที่สุดในอุตสาหกรรม ที่มีการแข่งขันกัน แม้ว่าต่างคนจะใช้เทคนิคกลวิธีที่เหมือนกัน แต่ผู้ซึ่งมีข้อมูลที่ดีที่สุดย่อมเป็นผู้ชนะ

Mind Map ของ AI

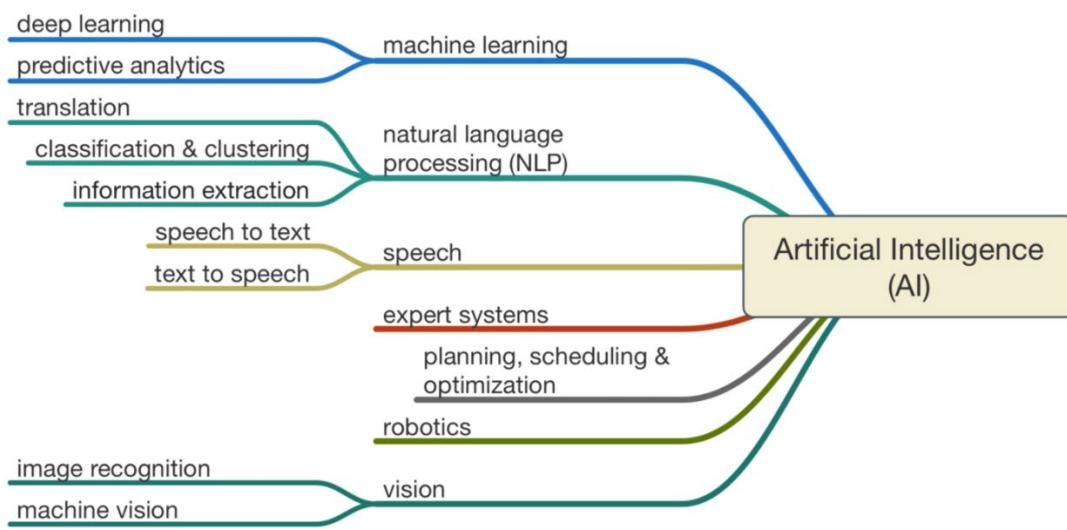


Figure 1 Mind Map Artificial Intelligence

จากที่เห็นอยู่นี่เป็น Mind Map ของ AI จะประกอบไปด้วย

Machine Learning

คือ การทำให้ระบบคอมพิวเตอร์เรียนรู้ได้ด้วยตนเองโดยใช้ข้อมูล AI ใช้ Machine Learning ในการสร้างความฉลาด มักจะใช้เรียนรู้โดยที่ไม่ได้มาจากเรียนรู้ของ AI ที่ไม่ได้เกิดจากการเขียนโดยใช้มนุษย์ มนุษย์มีหน้าที่เพียงเขียนโปรแกรมให้ AI หรือเครื่องจักรเรียนรู้จากข้อมูลเท่านั้น ที่เหลือเครื่องจัดการเอง ใน Machine Learning จะประกอบด้วย

1. Deep Learning – คือวิธีการเรียนรู้แบบอัตโนมัติด้วยการเลียนแบบการทำงานของโครงข่ายประสาทของมนุษย์ โดยนำระบบโครงข่ายประสาทมาช้อนกันหลายชั้นและทำการเรียนรู้ข้อมูลตัวอย่าง
2. Predictive Analysis – คือการวิเคราะห์แบบพยากรณ์ เพื่อทำนายสิ่งที่กำลังจะเกิดขึ้นหรืออนาคตจะเกิดขึ้น โดยใช้ข้อมูลที่ได้เกิดขึ้นแล้วกับแบบจำลองทางสถิติหรือ AI ต่าง ๆ

3. Natural Language Processing – การประมวลผลภาษาธรรมชาติ ช่วยให้คอมพิวเตอร์สามารถเข้าใจ ตลอดจนตีความและใช้งานภาษาปกติที่มนุษย์ใช้สื่อสารได้
4. Translation - คือการแปลข้อความจากภาษาของมนุษย์ภาษาหนึ่ง ไปยังอีกภาษาหนึ่ง โดยอัตโนมัติ
5. Classification & Clustering - Classification และ Clustering เป็น Model ที่ใช้เพื่อจัดกลุ่มของข้อมูล แต่มีแนวทางในการใช้งานและผลลัพธ์ที่แตกต่างกันอย่างสิ้นเชิง
6. Information Extraction - กระบวนการสกัดข้อมูลที่มีโครงสร้างที่ต้องการออกมาจากแหล่งข้อมูลที่ไม่มีโครงสร้างโดยอัตโนมัติ
7. Speech to Text - เป็นกระบวนการแปลงจากเสียงพูดเป็นข้อความ
8. Text to Speech - เป็นกระบวนการแปลงจากข้อความเป็นเสียงพูด
9. Expert System – คือระบบผู้เชี่ยวชาญ
10. Planning, Scheduling & Optimization - เป็นการทำให้เครื่องสามารถตัดสินใจและเลือกการดำเนินงานที่บรรลุเป้าหมายอย่างมีประสิทธิภาพ
11. Robotics – เป็นสาขาวิชาที่พัฒนาเครื่องยนต์ให้มีรูปร่างและเคลื่อนไหวได้แตกต่างกันไปตามวัตถุประสงค์การใช้งาน

Computer Vision

เป็นแขนงหนึ่งของวิทยาการปัญญาประดิษฐ์หรือ AI ซึ่งทำการฝึกสอนคอมพิวเตอร์ และระบบให้สามารถเข้าใจและตอบสนองต่อข้อมูลภาพได้อย่าง ชาญฉลาด ด้วยภาพดิจิทัลจากกล้องถ่ายภาพและวิดีโอต่าง ๆ และแบบจำลอง deep learning นั้น อุปกรณ์ต่าง ๆ จะสามารถเรียนรู้ที่จะระบุและทราบถึงวัตถุต่าง ๆ จากนั้นจะสามารถทำการตอบสนองต่อสิ่งที่มัน "มองเห็น" ได้ต่อไป

เป้าหมายของ Computer vision

- การตรวจจับ การแบ่งขอบเขต การระบุตำแหน่ง และการจดจำ วัตถุจากภาพ เช่น ใบหน้าของมนุษย์
- การประเมินผลสำหรับการแบ่งลักษณะของวัตถุในภาพ หรือ การเปรียบเทียบวัตถุ
- การเปรียบเทียบวัตถุในมุมมองต่าง ๆ ของรูปภาพ หรือวัตถุนั้น ๆ
- การเชื่อมโยงมุมมองต่าง ๆ ของรูปภาพ เพื่อสร้างแบบจำลองสามมิติ ของรูปภาพนั้น ๆ โดยแบบจำลองเหล่านั้นอาจจะนำมาใช้ในการสร้างต้นแบบ หุ่นยนต์ AI ใน การค้นหาวัตถุเหล่านั้น ด้วยรูปภาพ จำเป็นต้องมีฐานข้อมูลขนาดใหญ่

ระบบ AI ในปัจจุบันนี้ มีประสิทธิภาพสูง และสามารถดำเนินการต่อยอดจากผลลัพธ์ที่ได้รับ และนำข้อมูลจากการทำความเข้าใจภาพมาใช้ให้เกิดประโยชน์ต่อไปได้ ซึ่งมีรูปแบบของเทคโนโลยี Computer Vision หลายรูปแบบ และมีการใช้งานในหลายสถานการณ์ตามไปด้วย

การประยุกต์ใช้ Computer Vision

1. Image segmentation - คือการแยกส่วนของภาพออกเป็นหลาย ๆ ส่วนหรือชิ้นของค์ประกอบอยู่ ๆ เพื่อพิจารณาแยกส่วนกัน
2. Object detection - หรือการตรวจหาวัตถุแบบเฉพาะเจาะจงในภาพแต่ละภาพ ซึ่งมีการทำงานในระดับสูงที่สามารถระบุวัตถุหลายชิ้นในภาพเดียวกันได้
3. Face recognition - หรือการจำจำใบหน้า เป็นรูปแบบการระบุวัตถุชิ้นสูงที่มีได้ทำแค่การระบุว่า มีใบหน้าของมนุษย์อยู่ในภาพเท่านั้น แต่ยังสามารถแยกแยะบุคคลแต่ละบุคคลออกจากกันและระบุบุคคลที่จะจะได้ออกด้วย
4. Edge detection - เป็นเทคนิคการระบุหาขอบหรือมุมของวัตถุ หรือภาพทิวทัศน์ เพื่อให้ทราบได้ง่ายขึ้นว่าองค์ประกอบในภาพมีสิ่งใดบ้าง
5. Pattern detection - คือการระบุวัตถุจากรูปทรง หรือสี หรือสิ่งบ่งชี้ต่าง ๆ ที่พบรูปในภาพ ที่เป็นรูปแบบเดียวกันซ้ำ ๆ สำหรับวัตถุประเภทนั้น ๆ
6. Image classification - ทำงานด้วยการจัดกลุ่มภาพออกเป็นหมวดหมู่ต่าง ๆ
7. Feature matching - เป็นรูปแบบหนึ่งของการตรวจหารูปแบบหรือ pattern detection ที่ระบุจุดที่เหมือนหรือคล้ายคลึงกันในภาพต่าง ๆ เพื่อจัดหมวดหมู่แก้วัตถุและภาพเหล่านั้น

ตัวอย่างการใช้ Computer Vision

ระบบดูแลการจราจรบนท้องถนน โดยการนับจำนวนรถบนท้องถนนในภาพถ่ายด้วยกล้องวงจรปิด ในแต่ละช่วงเวลา

ระบบเก็บข้อมูลรถที่เข้าและออกอาคาร โดยใช้ภาพถ่ายของป้ายทะเบียนรถเพื่อประโยชน์ในด้านความปลอดภัย โดยนำข้อมูลภาพป้ายทะเบียนของรถยนต์ ที่ได้จากการจราจรปิดประเภท LPR Camera (กล้องสำหรับจับภาพป้ายทะเบียนโดยเฉพาะ) มาทำการวิเคราะห์ต่อจดจำป้ายทะเบียน

การนำไปใช้ประโยชน์ด้านการแพทย์ เครื่อง MRI และเครื่องอัลตราซาวด์ — เป็นเครื่องมือทางการแพทย์ สามารถตรวจดูอวัยวะในร่างกาย ซึ่งภาพที่ได้จากการ MRI จะได้ภาพออกมาในรูปแบบ 3 มิติ ที่ผ่านเทคนิคการประมวลผลภาพต่าง ๆ

ระบบดูแลการจราจรบนท้องถนน โดยการนับจำนวนรถบนท้องถนนในภาพถ่ายด้วยกล้องวงจรปิด ในแต่ละช่วงเวลา

ระบบแปลภาษา จะช่วยแปลข้อความหรือคำพูดตามความหมาย ในรูปแบบที่เป็นธรรมชาติมากขึ้น ซึ่งจะใช้บริบทที่กว้างขึ้นเพื่อช่วยให้แปลได้ตรงกับความหมายที่สุดระบบจะจัดเรียงและปรับให้ตรงหรือใกล้เคียงกับภาษาพูดของมนุษย์มากที่สุด การแปลข้อความหลาย ๆ ย่อหน้าหรือทักษะจะอ่านแล้วเข้าใจได้ง่ายขึ้น

ระบบตรวจจับใบหน้า โปรแกรมตรวจจับใบหน้าจะทำหน้าที่วิเคราะห์ตรวจสอบและจดจำใบหน้าที่ทำงานร่วมกับกล้องวงจรปิด IP Camera คุณภาพสูง โดยลักษณะการทำงานระบบจะทำการเปรียบเทียบ

ใบหน้าของบุคคล ที่ผ่านเข้ามายังกล้องที่ได้กำหนดและตั้งค่าเอาไว้ จากนั้นจะทำการนำภาพใบหน้าดังกล่าวมาเปรียบเทียบกับภาพบุคคลในฐานข้อมูล

การทำงานของ Computer Vision

Computer vision เป็นศูนย์กลางของนวัตกรรมระดับแนวหน้ามาก many รวมถึงรถยนต์ไร้คนขับ โดรน เทคโนโลยีการจดจำใบหน้า และอื่น ๆ อีกมากมาย ด้วยความก้าวหน้าของ AI

ภาพด้านล่างแสดงถึงวิธีการทำงานของคอมพิวเตอร์วิทัค์น์เมื่อเปรียบเทียบกับวิธีที่มนุษย์ประมวลผล การป้อนข้อมูลด้วยภาพ

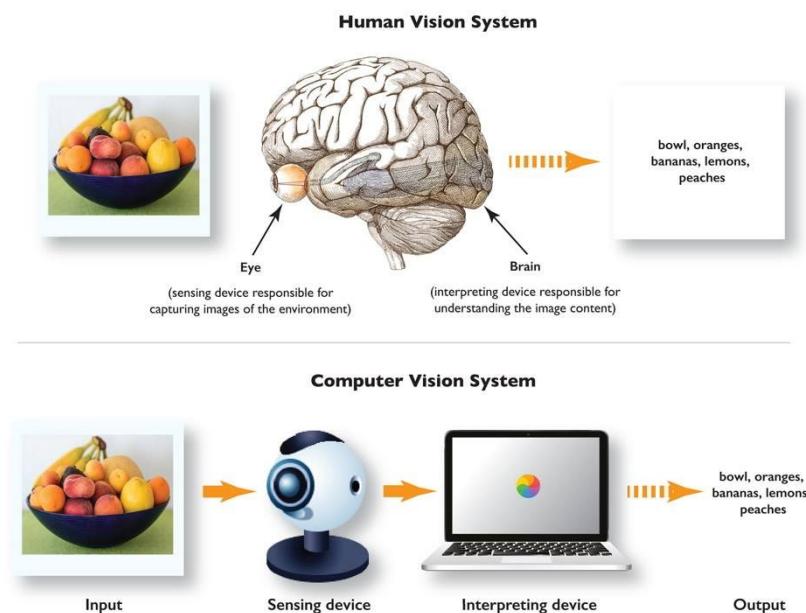


Figure 2 การทำงานของ Computer Vision

การประยุกต์ใช้ Computer Vision แตกต่างกันไป แต่ระบบการมองเห็นที่นำไปใช้ลำดับขั้นตอนที่แตกต่างกันที่คล้ายกันในการประมวลผลและวิเคราะห์ข้อมูลภาพ สิ่งเหล่านี้เรียกว่าท่อส่งวิสัยทัศน์ และแอปพลิเคชันการมองเห็นจำนวนมากเริ่มต้นด้วยการรับภาพและข้อมูล จากนั้นประมวลผลข้อมูลนั้น ดำเนินการตามขั้นตอนการวิเคราะห์และการรับรู้ และสุดท้ายก็ทำการคาดคะเนตามข้อมูลที่ได้จากการ

Digital image (ภาพดิจิทัล)

ภาพดิจิทัล คือ ข้อมูลรูปภาพที่ถูกจัดเก็บในรูปแบบดิจิทัล ซึ่งสามารถทำการเปลี่ยนแปลง แก้ไข ประมวลผล หรือถ่ายโอนได้โดยใช้ระบบคอมพิวเตอร์ ภาพดิจิทัลจะมีรูปแบบการเก็บเป็นเมทริกซ์ ซึ่งจะมีการจัดเก็บภาพแต่ละชนิดต่างกัน โดยแบ่งชนิดของภาพได้ ดังนี้

Pixel

พิกเซล (Pixel) คือ จุดที่เป็นจุดภาพที่แสดงบนหน้าจอ และสามารถรวมกันให้เกิดเป็นภาพขึ้นมา แต่ละ จุดจะมีสีที่แตกต่างกันไป ยิ่งละเอียดมากจะทำให้ภาพนั้นชัดเจนมากยิ่งขึ้น แนะนำว่าความใหญ่ของภาพก็ ใหญ่ขึ้นด้วยเช่นเดียวกัน พิกเซลมากไม่ได้หมายความว่าภาพนั้นจะคมชัดเสมอไป แต่ภาพที่มีพิกเซลมาก จะ สามารถถ่ายภาพได้มีความละเอียดสูงได้

ค่าพิกเซล 2, 3, 4, หรือ 5 ล้านพิกเซล หรืออีน ๆ ตัวเลขพกนี้คืออัตราส่วนของความละเอียดของ เม็ดสีต่างๆที่รวมกัน ในขนาดความกว้างและความสูง เช่น 2 ล้านพิกเซล จะเท่ากับความละเอียด 1080P คือ ภาพที่มีขนาด 1920H x 1080P หรือขนาดค่าความละเอียดของภาพ คำนวนได้จาก H (แนวอน) x P (แนวตั้ง)

พิกเซล 1 พิกเซล ถ้าเป็นภาพสีจะมี 3 เลเยอร์หรือชั้นซ้อนทับกัน ก็คือ สีแดง สีเขียว และสีฟ้า ซึ่งใน แต่ละสีมันจะแทนด้วยค่าได้ตั้งแต่ 0 - 255 ค่า เช่น สีขาวมี 3 เลเยอร์ คือ 255 255 255 สีดำ = 0 0 0 สีแดง = เลเยอร์เรด 255 เลเยอร์กรีนและบลู = 0 เป็นต้น

Binary Image

ใบหนารีในทางดิจิตอลหมายถึงว่ามีเพียง 2 สถานะคือ 0 และ 1 ซึ่งภาพใบหนารีเป็นรูปที่ใช้เนื้อที่เพียง 1 บิต จะมีแค่ความเข้ม 2 ค่าเท่านั้นคือ 0 และ 1 เป็นรูปที่ใช้เนื้อที่เพียง 1 บิตหมายความว่า พิกเซลใดที่มีค่าเป็น 0 ก็จะหมายถึงว่าพิกเซลนั้นจะแสดงสีดำ พิกเซลใดที่มีค่าเป็น 1 จะหมายถึงว่าพิกเซลนั้นจะแสดงสีขาว

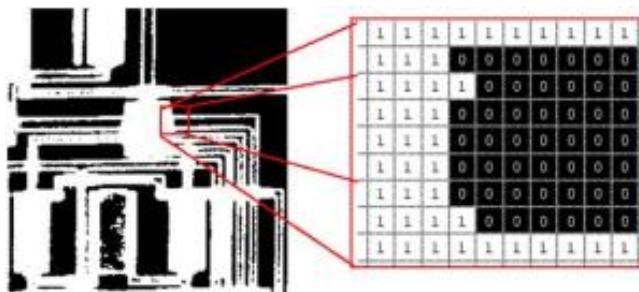


Figure 3 ภาพ Binary หรือ ภาพขาว-ดำ

Grayscale Image เป็นรูปที่เก็บโดยใช้รูปแบบของอาร์เรย์ 2 มิติ โดยค่าที่เก็บจะมีค่าอยู่ในช่วง ๆ หนึ่ง ซึ่งระดับของสีขึ้นอยู่กับขนาดของบิตที่ใช้เก็บค่าสี

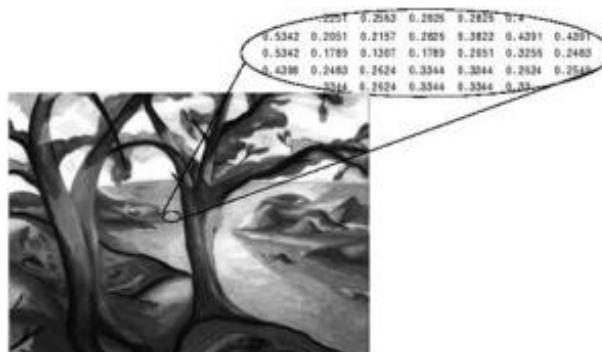


Figure 4 ภาพ Grayscale

RGB Image

“RGB” คือสีของรูปภาพที่เกิดจากการผสมสีของแสง 3 สี คือ แดง (Red) เขียว (Green) และน้ำเงิน (Blue)



Figure 5 ภาพ RGB

ตัวอย่างการนำรูปไปประมวลผล

Grayscale image (32 x 16)

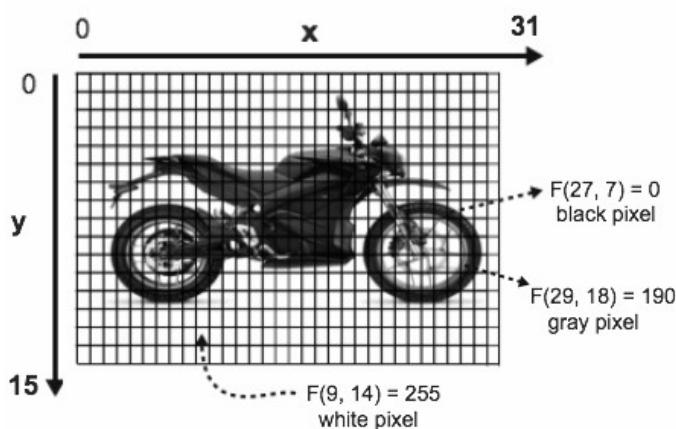


Figure 6 Greyscale image (ภาพระดับสีเทา)

ในภาพเป็นภาพ Grayscale แต่ละพิกเซลมีค่าที่แสดงถึงความเข้มของแสงบนพิกเซลเฉพาะนี้ ค่าพิกเซลแตกต่างกันไปตั้งแต่ 0 ถึง 255 เนื่องจากค่าพิกเซลแสดงถึงความเข้มของแสง ค่า 0 หมายถึงพิกเซล มืด (สีดำ) 255 จึงเป็นความสว่าง (สีขาว) และค่าที่อยู่ระหว่างนั้นแสดงถึงความเข้มของระดับสีเทา

จำไว้ว่า 0 หมายถึงสีดำและ 255 หมายถึงสีขาว ภาพโทนสีเทามีเพียงหนึ่งเลเยอร์ ในขณะที่ภาพสีมี RGB สามเลเยอร์ (แดง เขียว น้ำเงิน)

ในรูปภาพสามารถแสดงเป็นฟังก์ชันของตัวแปรสองตัวคือ X และ Y ซึ่งกำหนดพื้นที่สองมิติ ภาพด้านบนมีขนาด 32×16 ซึ่งหมายความว่าขนาดของรูปภาพกว้าง 32 พิกเซลและสูง 16 พิกเซล แกน X เริ่มจาก 0 ถึง 31 และแกน Y ตั้งแต่ 0 ถึง 16 โดยรวมแล้ว รูปภาพมี $32 \times 16 = 512$ พิกเซล



```

25 43 11 04 70 87 12 31 43 10 05 77 12 06 45 09 29 30 02
56 22 75 03 22 96 45 12 23 03 77 67 81 45 22 04 90 22 21
32 45 41 91 87 62 35 02 00 11 62 25 43 11 04 70 87 12 61
31 43 10 05 77 12 06 45 09 29 30 56 22 75 03 22 96 45 05
12 23 03 77 67 81 45 22 04 90 22 32 45 41 91 87 62 35 44
02 00 11 62 25 43 11 04 70 87 12 31 43 10 05 77 12 06 10
45 09 29 30 56 22 75 03 22 96 45 12 23 03 77 67 81 45 55
22 04 90 22 32 45 41 91 87 62 35 02 00 11 62 25 43 11 80
04 70 87 12 31 43 10 05 77 12 06 45 09 29 30 56 22 75 08
03 22 96 45 12 23 03 77 67 81 45 22 04 90 22 32 45 41 99
91 87 62 35 02 00 11 62 22 01 00 72 65 23 01 00 22 04 30
90 22 32 45 41 91 87 62 35 02 00 11 62 25 43 11 04 70 42
87 12 31 43 10 05 77 12 06 45 09 29 30 56 22 75 03 22 91
96 45 12 23 03 77 67 81 45 22 04 90 22 32 45 41 91 87 40
62 35 02 00 11 62 22 01 00 72 65 23 01 00 56 22 75 03 67
22 96 45 12 23 03 77 67 81 45 22 04 90 22 32 45 41 91 22

```

What we see

What computers see

Figure 7 ภาพที่มนุษย์มองเห็น และภาพที่คอมพิวเตอร์เห็น

Image Processing

เทคนิคที่ใช้ในการประมวลผลภาพ

1. ปรับปรุงคุณภาพของภาพ (Image Enhancement) — เป็นกระบวนการในการแปลงข้อมูลภาพตัวเลข เพื่อที่จะสร้างภาพที่เน้นรายละเอียดที่ต้องการ หรือปรับพิสัยของโทนแสงที่ต้องการของภาพ เมื่อ เปรียบเทียบกับข้อมูลหรือรายละเอียดอื่นๆ ของภาพ
2. การกรองภาพหรือการกำจัดสัญญาณรบกวนออกจากภาพ (Image Filters) — คือการนำภาพไปผ่านตัวกรองสัญญาณเพื่อให้ได้ภาพผลลัพธ์ออกมา ภาพผลลัพธ์ที่ได้จะมีคุณสมบัติแตกต่างจากภาพเริ่มต้น วัตถุประสงค์หลักของการกรองข้อมูลภาพคือการเน้น (enhance) หรือลดthon (attenuate) คุณสมบัติบางประการของภาพ เพื่อให้ได้ภาพที่มีคุณสมบัติตามต้องการ
3. การซ่อนหัวภาพ (Image Registration) — เป็นวิธีการนำข้อมูลของสองภาพหรือมากกว่า มารวมกันเพื่อให้เกิดภาพใหม่ที่มีข้อมูลภาพสมบูรณ์มากขึ้น โดยภาพใหม่ที่ได้นี้ จะเป็นการรวมตัวกันของข้อมูลหรือรายละเอียดในแต่ละภาพที่นำมา fusion กัน มีวัตถุประสงค์เพื่อให้ได้ภาพที่มีรายละเอียดและข้อมูลที่เพียงพอสำหรับการนำไปใช้
4. การคืนสภาพของภาพ (Image Restoration) — การทำให้ภาพคืนสู่สภาพเดิมหรือการปรับปรุงภาพให้เหมาะสมกับการมองเห็น
5. การแบ่งส่วนภาพ (Image Segmentation) — เป็นวิธีการแบ่งส่วนใดส่วนหนึ่งของภาพที่เราสนใจออกมาจากภาพที่เราต้องการ ซึ่งการแบ่งส่วนภาพนี้ โดยส่วนใหญ่แล้วจะเป็นขั้นตอนเบื้องต้นและสำคัญอย่างมากของการประมวลผลภาพทางการแพทย์ เนื่องจากภาพทางการแพทย์ที่ได้จากเครื่องถ่ายภาพแบบต่าง ๆ นั้น โดยปกติมักจะมีองค์ประกอบอื่น ๆ ที่อยู่ใกล้เคียงกับ

อวัยวะที่ทำถ่ายภาพมา เช่น เนื้อยื่อ กระดูก อวัยวะข้างเคียง หรือแม้กระทั่งสัญญาณรบกวน (Noise) ที่ขึ้นในขณะถ่ายภาพ ด้วยเหตุนี้ การวิเคราะห์เฉพาะอวัยวะที่ต้องการ

6. การหาขอบภาพในวัตถุ (Image Segmentation and EdgDeTecion) — เป็นการหาเส้นรอบวัตถุที่อยู่ในภาพ เมื่อทราบเส้นรอบวัตถุ เราจะสามารถคำนวณพื้นที่ (ขนาด) หรือรู้จำชนิดของวัตถุนั้นได้ อย่างไรก็ตาม การหาขอบภาพที่ถูกต้องสมบูรณ์ไม่ใช่เป็นเรื่องง่ายโดยเฉพาะอย่างยิ่ง การหาขอบของภาพที่มีคุณภาพต่ำ มีความแตกต่างระหว่างพื้นหน้ากับพื้นหลังน้อย หรือมีความสว่างไม่สม่ำเสมอทั่วภาพ ขอบภาพเกิดจากความแตกต่างของความเข้มแสงจากจุดหนึ่งไปยังอีกจุดหนึ่ง หากต่างนี้มีค่ามาก ขอบภาพก็จะเห็นได้ชัด ถ้าความแตกต่างมีค่าน้อยขอบภาพก็จะไม่ชัดเจน
7. การบีบอัดภาพ (Image Compression)
 - 7.1. การบีบอัดแบบไม่มีการสูญเสียรายละเอียดข้อมูล (Lossless compression) ค่าความสว่างของแต่ละจุดภาพจะยังคงอยู่เหมือนเดิมทุกประการ หรือไม่มีการเปลี่ยนแปลงค่าของแต่ละจุดภาพ ซึ่งการบีบอัดวิธินี้จะอาศัยเทคนิคการจัดเก็บข้อมูลเชิงเลขในการลดขนาดของข้อมูล
 - 7.2. การบีบอัดแบบสูญเสียรายละเอียดข้อมูล (Lossy compression) วิธีการนี้จะมีการเปลี่ยนแปลงค่าความสว่างของจุดภาพนั้นหมายความว่า วิธีการนี้ไม่เหมาะสมสำหรับข้อมูลภาพที่ต้องมีการจำแนกข้อมูล (Classification)
8. การสร้างภาพ 3 มิติ (3D Image Reconstruction) — คือการนำภาพหลายภาพในวัตถุชิ้นเดียวกัน และหาจุดเชื่อมต่อและนำมาประกอบกันให้เป็น 3 มิติ

ขั้นตอนการทำงานของ Computer Vision

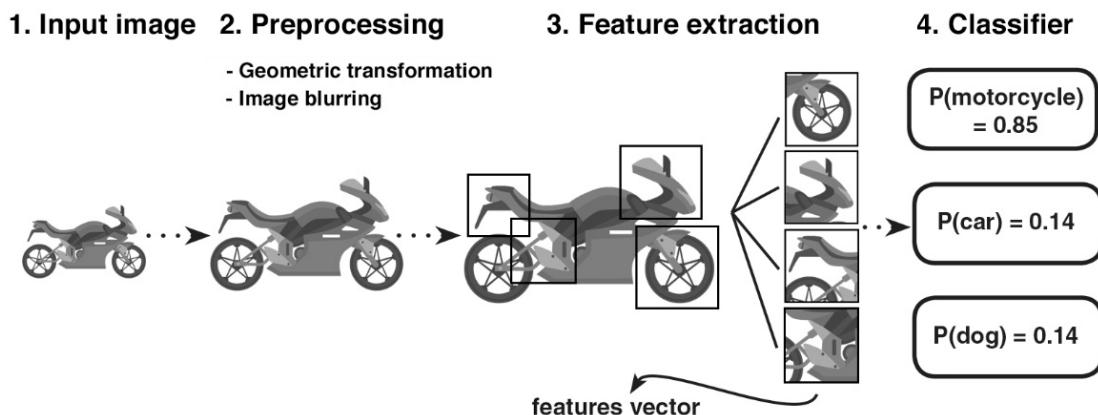


Figure 8 Pipeline Computer Vision

1. Input data คือมิวเตอร์รับอินพุตด้วยภาพจากอุปกรณ์สร้างภาพ เช่น กล้อง โดยทั่วไปจะบันทึกเป็นรูปภาพหรือลำดับของรูปภาพที่สร้างวิดีโอ



Figure 9 Input data

2. Preprocessing

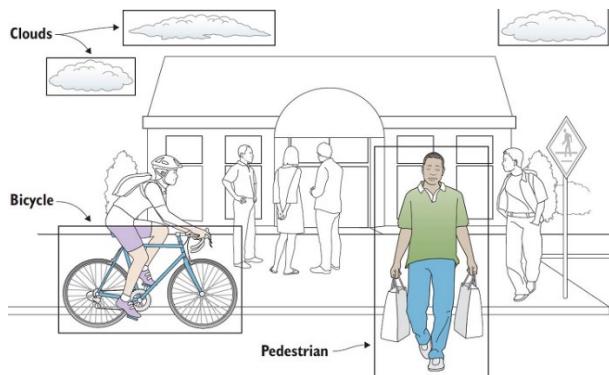


Figure 10 การประมวลผลภาพ

จากรูปที่เห็นอยู่ ในหนึ่งภาพมีอะไรหลายอย่างมาก เช่น ก้อนเมฆ คนปั่นจักรยาน คนถือถุงกระดาษ ร้านค้าหรืออาคาร คนยืนอยู่ร้านค้า ป้ายจราจร ต้นไม้ ฯลฯ

สมมติว่าโจทย์ต้องการเพียงแค่ก้อนเมฆเพียงอย่างเดียวการพิริโพรเซสคือเตรียมข้อมูลที่ต้องการไปวิเคราะห์ หากจะเอาแค่ก้อนเมฆ ก็ตัดเอามาแค่เพียงก้อนเมฆหากจะเอาจักรยานไปวิเคราะห์ก็ตัดให้เหลือเพียงแค่จักรยานหรือกำจัดสิ่งที่ไม่ต้องการหรือเลือกปอนอยู่ออกไปให้เหลือเพียงสิ่งที่ต้องการจริง ๆ

อย่างที่พูดไปข้างต้นเกี่ยวกับ Image processing วิธีนี้จะเอา Image processing มาใช้ เช่น ปรับภาพ เปลี่ยนจากภาพสีเป็นโนนสีเทา การตัดส่วนที่ไม่จำเป็นออกรวมไปถึงการปรับขนาดรูปภาพ เปลี่ยนรูปร่าง

หรือกำหนดมาตรฐานแต่ละภาพเท่านั้น เช่น ทำให้มีขนาดเท่ากัน จากนั้นจะสามารถเปรียบเทียบและวิเคราะห์เพิ่มเติมในลักษณะเดียวกันได้ เป็นต้น

3. Feature Extraction

จะเป็นการแยกคุณสมบัติ หรือคุณลักษณะคือสิ่งที่ช่วยกำหนดวัตถุบางอย่าง และมักจะเป็นข้อมูลเกี่ยวกับรูปร่างหรือสีของวัตถุ ตัวอย่างเช่น คุณลักษณะบางอย่างที่แยกแยะรูปร่างของล้อรถจักรยานยนต์ ไฟหน้า บังโคลน และอื่นๆ ผลลัพธ์ของกระบวนการนี้คือเวกเตอร์คุณลักษณะซึ่งเป็นรายการของรูปร่างที่ไม่ซ้ำกันซึ่งระบุวัตถุ

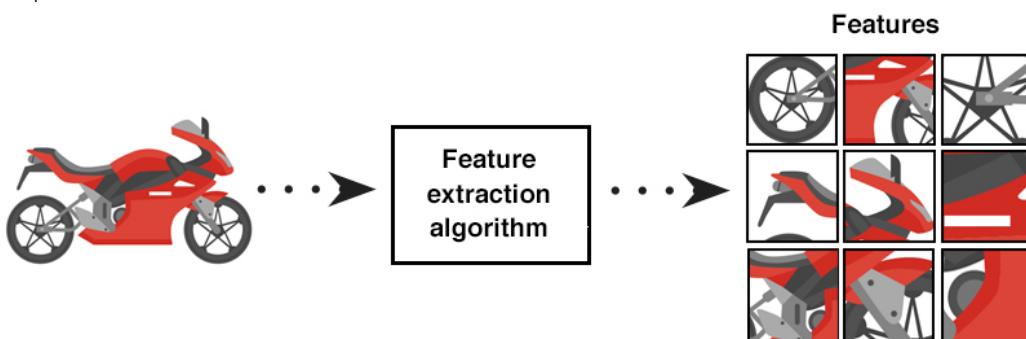


Figure 11 กระบวนการ Feature Extraction

4. Classifier

การจำแนกข้อมูล (Classification) ที่ซึ่งจะทำการสร้างโมเดลหรือตัวจำแนกข้อมูล (Classifier) เพื่อท่านายหมวดหมู่ของข้อมูล ขั้นตอนนี้ดูเวลาเตอร์คุณลักษณะจากขั้นตอนก่อนหน้าและคาดการณ์ผลลัพธ์ของรูปภาพ ผลลัพธ์ของกระบวนการนี้คือความน่าจะเป็นของแต่ละคลาส ดังที่เห็นในรูปข้างต้น สุนัขมีโอกาสสืบอยู่ที่สูงที่ 1% ในขณะที่มีความเป็นไปได้ 85% ที่จะเป็นมอเตอร์ไซค์ จะเห็นได้ว่าถึงแม้ไม่มีโมเดลจะสามารถทำนายระดับที่เหมาะสมได้ด้วยความน่าจะเป็นสูง แต่ก็ยังสับสนเล็กน้อยในการแยกแยะระหว่างรถยนต์และรถจักรยานยนต์ เนื่องจากคาดการณ์ว่ามีโอกาส 14% ที่ภาพนี้จะเป็นภาพของรถยนต์ เนื่องจากว่าเป็นมอเตอร์ไซค์ จึงสามารถพูดได้ว่าอัลกอริทึมการจำแนกประเภท ML นั้นแม่นยำถึง 85% เพื่อปรับปรุงความแม่นยำนี้ อาจต้องทำเพิ่มเติมจากขั้นตอนที่ 1 (รับภาพการฝึกเพิ่มเติม) หรือขั้นตอนที่ 2 (การประมวลผลเพิ่มเติมเพื่อขัดสัญญาณรบกวน) หรือขั้นตอนที่ 3 (แยกคุณลักษณะที่ดีกว่า) หรือขั้นตอนที่ 4 (เปลี่ยนอัลกอริทึมตัวแยกประเภท)

ประเภทของอัลกอริทึมการจำแนกประเภท



1. Logistic Regression

การวิเคราะห์การคัดถ่ายโลจิสติก เป็นการวิเคราะห์เพื่อทำนายโอกาสที่จะเกิด โดยอาศัยสมการโลจิสติกที่สร้างขึ้นจากชุดตัวแปรทำนาย ที่เป็นตัวแปรที่มีข้อมูลอยู่ ในระดับช่วงเป็นอย่างน้อย โดยที่ระหว่างตัวแปรทำนายจะต้องมีความสัมพันธ์กันต่ำและในการวิเคราะห์ จะต้องใช้ขนาดตัวแปรทำนายไม่ต่ำกว่า 30 ตัวแปร

2. Decision Trees

เป็นการเรียนรู้ที่ใช้จำแนกประเภทของตัวอย่าง โดยระบบจะเรียนรู้จากตัวอย่างที่สอนให้ แล้วเก็บความรู้ที่ได้ในรูปแบบของต้นไม้ตัดสินใจ ซึ่งอาจถูกแปลงไปเป็นอีกภูมิที่หนึ่ง นิยมใช้มากใน Machine Learning

3. Support Vector Machines

เป็นอัลกอริทึมที่สามารถนำมาช่วยแก้ปัญหาการจำแนกข้อมูล ใช้ในการวิเคราะห์ข้อมูลและจำแนกข้อมูล โดยอาศัยหลักการของการหาสัมประสิทธิ์ของสมการเพื่อสร้างเส้นแบ่งแยกกลุ่มข้อมูลที่ถูกป้อนเข้าสู่กระบวนการสอนให้ระบบเรียนรู้ โดยเน้นไปยังเส้นแบ่งแยกและกลุ่มข้อมูลได้ดีที่สุด

4. K-Nearest Neighbour (KNN)

เป็นวิธีการจำแนกประเภทข้อมูลวิธีหนึ่ง โดยจัดว่าเป็นการจำแนกแบบมีผู้ฝึกสอน (Supervised Machine Learning Algorithm) หรือ ทราบคำตอบอยู่แล้ว จากนั้นจะใช้โมเดลในการจำแนกประเภท ข้อมูลจากข้อมูลที่รู้คำตอบ

5. Naive Bayes Classifier

เป็น Model หนึ่งในการแบ่งกลุ่มที่ต้องการโดยใช้ความน่าจะเป็นที่ซื่อว่า Naive Bayes ยกตัวอย่าง เช่น ต้องการแบ่งกลุ่มว่าคนไข้ที่เข้ามานั้นเป็นไข้หวัดใหญ่หรือไม่ ซึ่งจะต้องถามอาการคนไข้มาให้ได้มากที่สุด ว่าอาการเป็นอย่างไร และถึงจะคาดคะเนจากข้อมูลอาการที่ได้ว่ามีความน่าจะเป็นไข้หวัดใหญ่เท่าไหร่ ซึ่งความน่าจะเป็นดังกล่าวจะใช้ Naive Bayes ในการหาค่าความน่าจะเป็น

6. Convolutional Neural Network (CNN)

โครงข่ายประสาทแบบคอนволูชัน โดยจะจำลองการทำงานของเห็นของมนุษย์ที่มองพื้นที่เป็นที่อยู่ ๆ และนำกลุ่มของพื้นที่อยู่ ๆ มาพسانกัน เพื่อดูว่าสิ่งที่เห็นอยู่เป็นอะไรกันแน่

โครงข่ายประสาทเทียม (Artificial neural networks: ANN)

ระบบคอมพิวเตอร์จากไมเดลทางคณิตศาสตร์ เพื่อจำลองการทำงานของโครงข่ายประสาทชีวภาพที่อยู่ในสมองของสิ่งมีชีวิต โครงข่ายประสาทเทียมสามารถเรียนรู้ที่จะทำงานที่มีขอบหมายได้ จากการเรียนรู้ผ่านตัวอย่าง โดยไม่ถูกโปรแกรมด้วยกฎเกณฑ์ตายตัวแบบระบบอัตโนมัติ ยกตัวอย่างเช่น ในการประมวลผลภาพ คอมพิวเตอร์ที่ทำงานด้วยระบบโครงข่ายประสาทเทียมจะเรียนรู้การจำแนกรูปภาพแมวได้จากการให้ตัวอย่างรูปภาพที่กำกับโดยผู้เขียนโปรแกรมว่า “เป็นแมว” หรือ “ไม่เป็นแมว” จากนั้นนำผลลัพธ์ที่ได้ไปใช้ระบุภาพแมว

ในตัวอย่างรูปภาพอื่น ๆ โปรแกรมโครงข่ายประสาทเทียมสามารถแยกแยะรูปภาพแมวได้โดยปราศจากการความรู้ก่อนหน้า ว่า “แมว” คืออะไร อาทิ แมวมีขน มีหูแหลม มีเขี้ยว มีหาง แทนที่จะใช้ความรู้ดังกล่าว โครงข่ายประสาทเทียมทำการระบุตัวแมวโดยอัตโนมัติด้วยการระบุลักษณะเฉพาะ จากชุดตัวอย่างที่เคยได้ประมวลผล

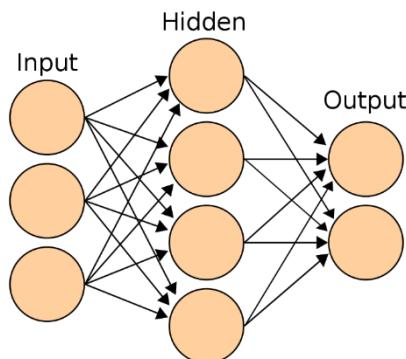


Figure 12 โครงข่ายประสาทเทียม (Artificial neural networks: ANN)

การประมวลผล ของโครงข่ายประสาทเทียมเกิดขึ้นในหน่วยประมวลผลย่อย เรียกว่า โนนด (node) ซึ่งโนนดเป็นการจำลองลักษณะการทำงานมาจากการเซลล์การส่งสัญญาณ ระหว่างโนนดที่เชื่อมต่อกัน จำลองมาจากการเชื่อมต่อของไบโปรส แล้วแกนประสาทในระบบประสาทของสมองมนุษย์ภายในโนนด จุดเชื่อมต่อแต่ละจุด มีความคล้ายคลึงกับจุดประสาทประสาท (Synapses) ในสมอง มีความสามารถในการส่งสัญญาณไปยังเซลล์ประสาทเซลล์อื่น ๆ ที่เชื่อมต่อกับมัน

Node ANN

ในการสร้างระบบโครงข่ายประสาทเทียม เอาร์พุตของแต่ละเซลล์ประสาทจะมาจากการคำนวณผลรวมของอินพุต ด้วยฟังก์ชันการแปลง (transfer function) ซึ่งทำหน้าที่รวมค่าเชิงตัวเลขจากเอาร์พุตของเซลล์ประสาทเทียม และทำการตัดสินใจว่าจะส่งสัญญาณเอาร์พุตออกไปในรูปใด ฟังก์ชันการแปลงอาจเป็นฟังก์ชันเส้นตรงหรือไม่ก็ได้ โครงข่ายประสาทเทียม ประกอบไปด้วย จุดเชื่อมต่อ (Connections) ซึ่งสามารถ

เรียกสั้น ๆ ได้ว่า เอจ (Edge), เมื่อโครงข่ายประสาทมีการเรียนรู้ จะเกิดค่าน้ำหนักขึ้น, ค่าน้ำหนัก (weights) คือ สิ่งที่ได้จากการเรียนรู้ของโครงข่ายประสาทเทียม หรือเรียกอีกอย่างหนึ่งว่า ค่าความรู้ (knowledge) ค่านี้ จะถูกเก็บเป็นทักษะเพื่อใช้ในการจัดจำข้อมูลอื่น ๆ ที่อยู่ในรูปแบบเดียวกัน

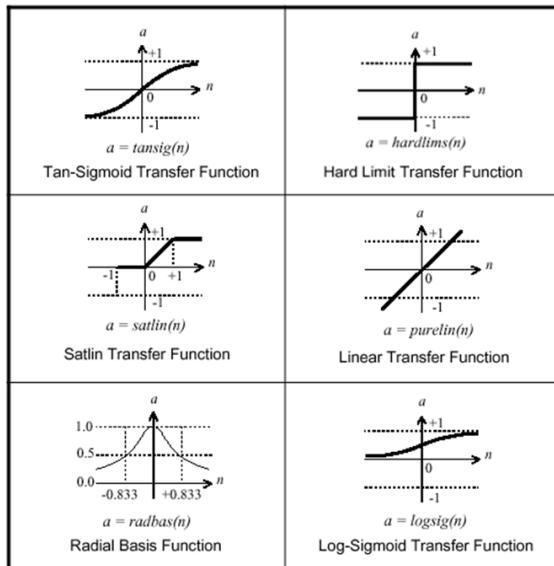


Figure 13 การคำนวณผลรวมของอินพุต ด้วยฟังก์ชันการแปลง

Transfer Function

จุดประสงค์ของการสร้างโครงข่ายประสาทเทียม มีหลากหลายจุดประสงค์ เช่น การแก้ปัญหาแบบเดียวกับที่สมองมนุษย์สามารถทำได้ หรือ การทำงานที่เฉพาะเจาะจง, ปัจจุบัน มีการประยุกต์ใช้โครงข่ายประสาทเทียมกับงานหลากหลายรูปแบบ อาทิ คอมพิวเตอร์วิทยา, การรู้จำคำพูด, การแปลงภาษา, การกรองเนื้อหาโดยอัลกอริتمีเดีย, การเล่นเกม, การวินิจฉัยโรค และกิจกรรมบางอย่างที่ไม่คิดว่าปัญญาประดิษฐ์จะทำได้ เช่น การวาดภาพ, การประพันธ์เพลง และ การประพันธ์บทกวี เป็นต้น

Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) เป็นเครือข่ายประสาทเทียมประเภทหนึ่งที่ใช้ในการจดจำ และการประมวลผลภาพที่ออกแบบมาโดยเฉพาะเพื่อประมวลผลข้อมูลพิเศษการวิเคราะห์รูปภาพที่มีนุ่มนิ่ม มองเห็น โดยจะแบ่งรูปภาพออกเป็นพื้นที่ย่อยๆ เป็น Pixel แต่ละอันเพื่อทำการวิเคราะห์ Metric ของรูปภาพ โดยถ้าเป็นรูปภาพสีขาวดำ จะเป็นแบบ Matrix 2x2 ถ้าเป็นภาพสีจะเป็น Matrix 3x3

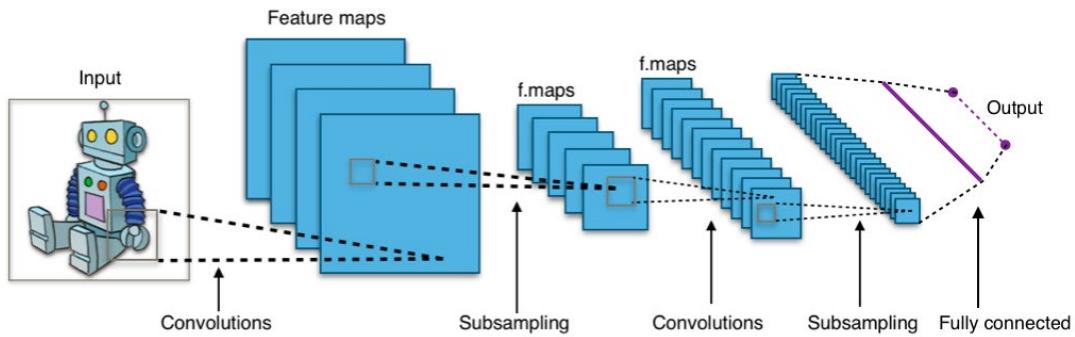


Figure 14 Convolutional Neural Network (CNN)

ตัวอย่างวิธีการแบ่งรูปภาพออกเป็น Metric

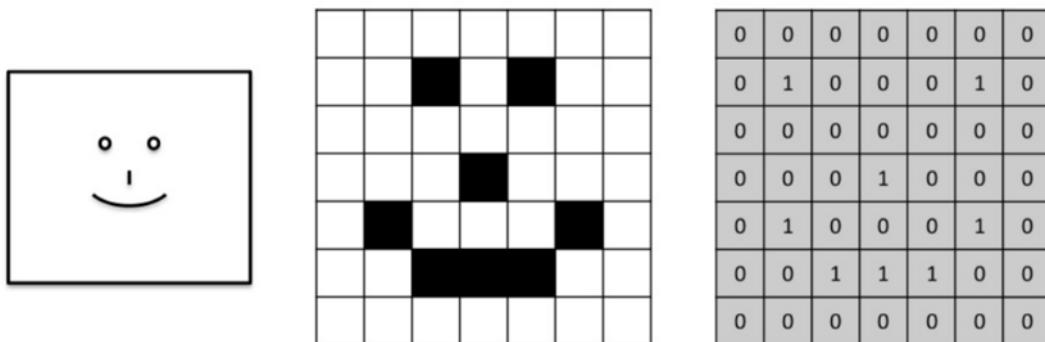


Figure 15 การแบ่งรูปภาพออกเป็น Matrix

ขั้นตอนการทำ CNN มี 4 ขั้นตอน

Step 1: Convolution

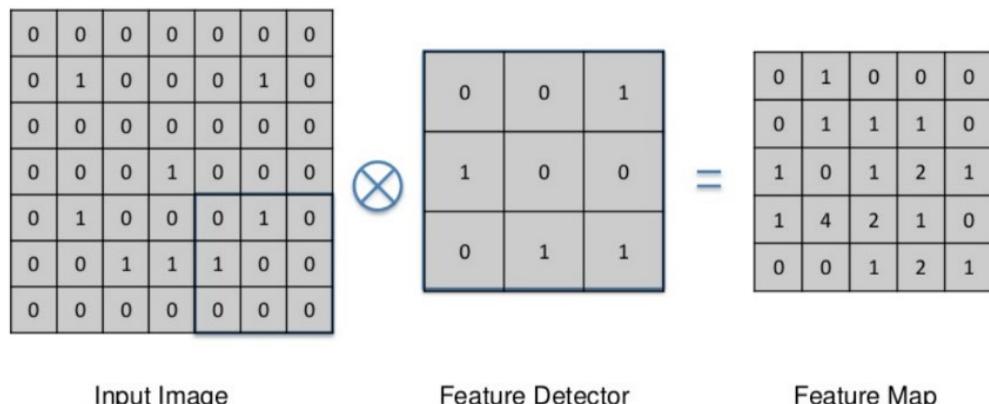


Figure 16 Convolution

1.1. การทำการคูณแมทริกซ์ ระหว่าง Input Image กับ Feature Detector ทำให้ได้ Feature Map

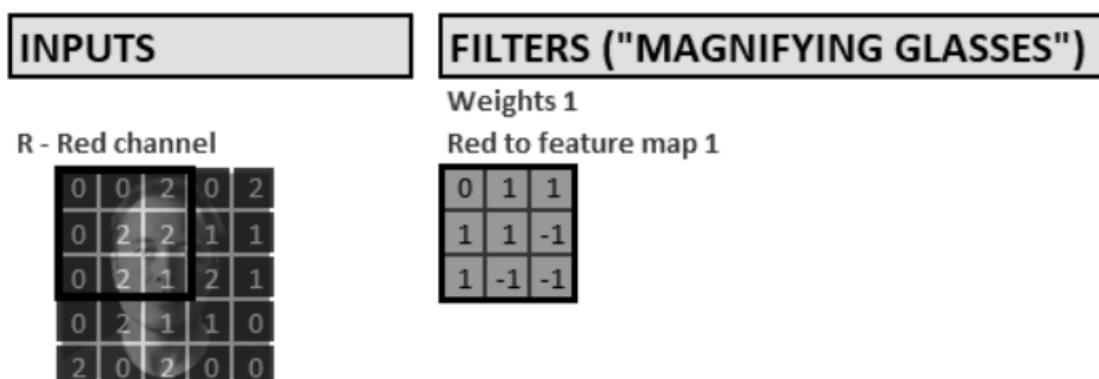


Figure 17 การทำการคูณแมทริกซ์

1.2. ขั้นตอนการสกัด Feature ทำได้โดย นำค่าที่อยู่ใน Input มาทำการคูณแบบ Matrix กับ ค่า Weight

CONV 1									
<div style="display: flex; align-items: center; justify-content: space-between;"> <div style="flex: 1;"> <p>From Red</p> <table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td>-1</td><td>4</td><td>2</td></tr> <tr><td>2</td><td>4</td><td>4</td></tr> <tr><td>4</td><td>3</td><td>7</td></tr> </table> </div> <div style="flex: 1; text-align: right;"> <p>-1</p> $\begin{aligned} &= (0 \times 0) + (0 \times 1) + (2 \times 1) + &= 0 + 0 + 2 + &= 2 + \\ &+ (0 \times 1) + (2 \times 1) + (2 \times -1) + &0 + 2 + (-2) + &0 + \\ &+ (0 \times 1) + (2 \times -1) + (1 \times -1) &0 + (-2) + (-1) &-3 \end{aligned}$ </div> </div> <p style="text-align: center; margin-top: 10px;">9 multiplications for 1 layer</p>	-1	4	2	2	4	4	4	3	7
-1	4	2							
2	4	4							
4	3	7							

Figure 18 การสกัด Feature

Step 2: Max Pooling

Max Pooling: คือ ตัวกรอกที่ใช้ในการเลือกค่าที่มากที่สุดในบริเวณแมทริกซ์เดียวกัน

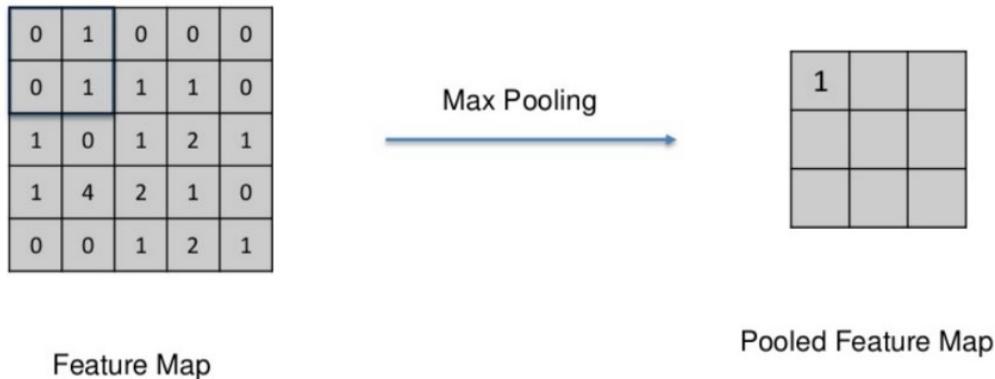


Figure 19 Max Pooling

Step 3: Flattening

Flattening เป็นการทำ Pooling Feature Map ที่ได้ทำเป็นคอลัมน์เดียวกัน เพื่อความสะดวกในการวิเคราะห์ข้อมูล

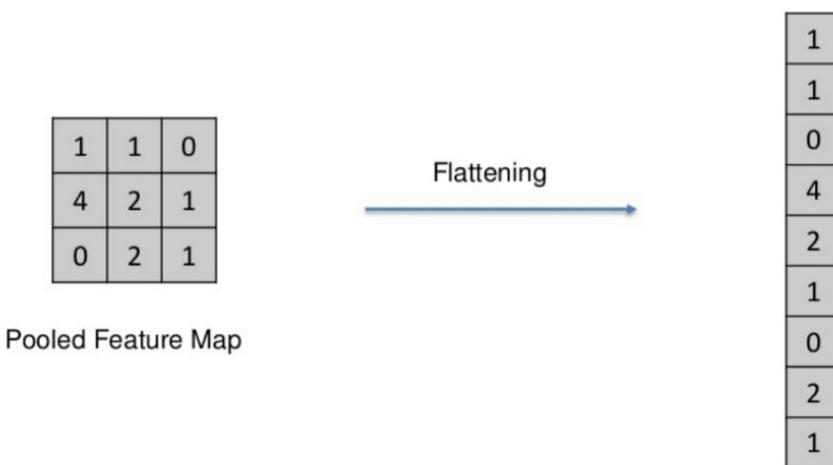


Figure 20 Flattening

Step 4: Full Connection

Full Connection คือ การนำ Flattening ที่ได้มานเข้าสู่โมเดล Deep Learning

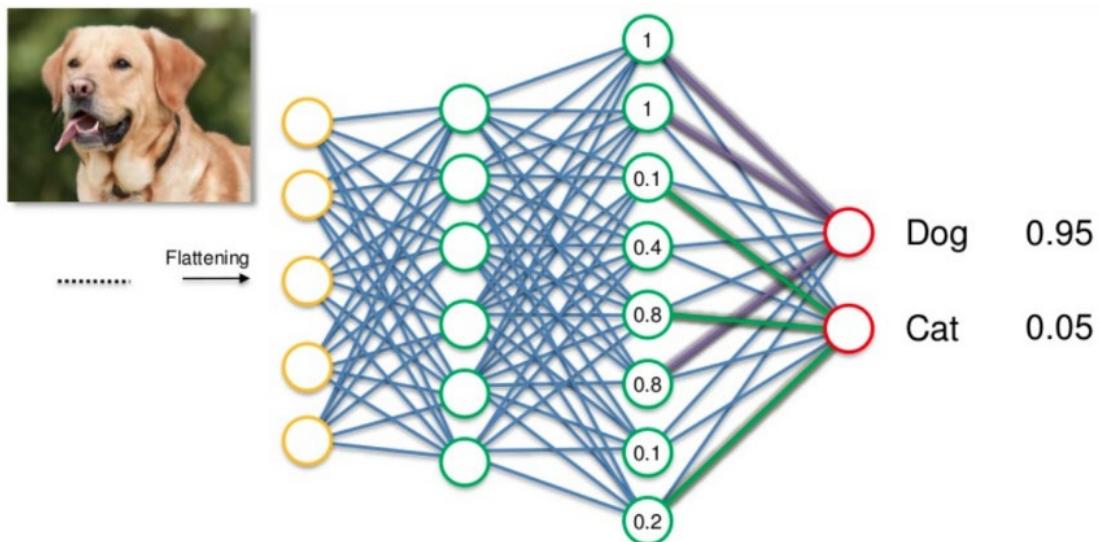


Figure 21 Full Connection

2. การจำแนกตรวจจับวัตถุในรูปภาพด้วย Yolo

Yolo คืออะไร

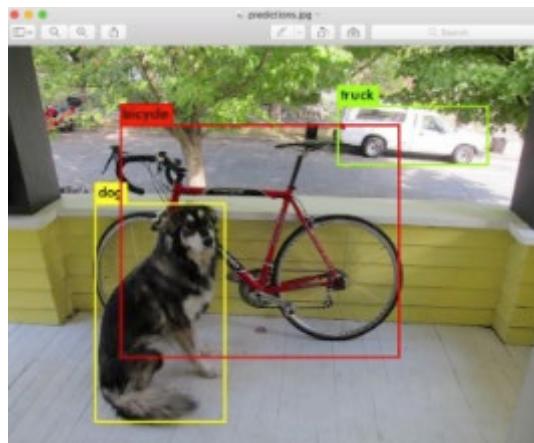


Figure 22 Fast single-shot detection

YOLO เป็นการรู้จำวัตถุที่ใช้วิธีการที่เรียกว่า “Fast single-shot detection” ซึ่งจะเป็นวิธีการที่สามารถตรวจจับวัตถุได้จากการส่งผ่านรูปภาพเข้าไปในระบบเพียงครั้งเดียว

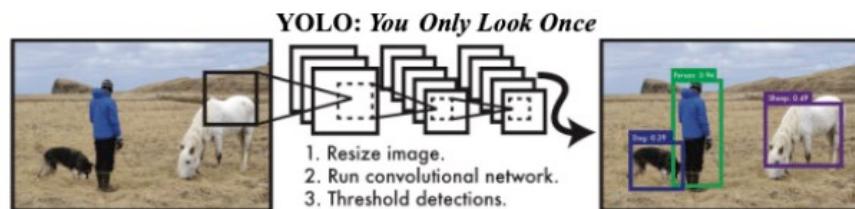


Figure 23 อัลกอริทึม YOLO

ในอัลกอริทึม YOLO การดำเนินการจำแนกวัตถุว่าเป็นชนิดอะไร (classification) และดำเนินการหาตำแหน่งของวัตถุ (localization) โดยใช้กรอบล้อมวัตถุ (Bounding Box) จะทำไปพร้อม ๆ กัน กรรมวิธีของ YOLO ไม่ได้พิจารณาดำเนินการจากภาพทั้งภาพ แต่จะแบ่งภาพออกเป็นส่วน ๆ ซึ่งวิธีการแบบนี้ส่งผลดีในด้านความเร็วของการประมวลผล YOLO มีขั้นตอนการทำงานโดยรวม

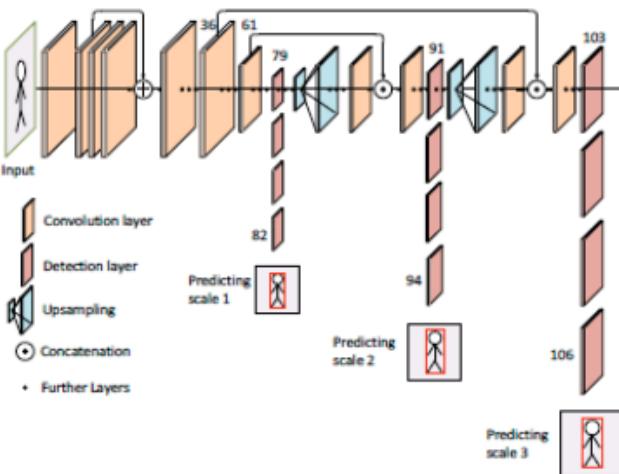


Figure 24 ลักษณะการทำงานต่าง ๆ

ลักษณะการทำงานต่าง ๆ ดังนี้ ชั้นของคอนволูชัน ชั้นของการเพิ่มอัตราสุ่ม ชั้นของการทำนายผล ดัง Figure 24 ลักษณะการทำงานต่าง ๆ ซึ่งเป็นโครงสร้างสถาปัตยกรรมของอัลกอริทึม YOLO V3 ทั้งนี้ อัลกอริทึม YOLO จะมีสถาปัตยกรรมแบบ Tiny ซึ่งมีการลดTHONจำนวนชั้นบางชั้นออกไป ทำให้โครงสร้างมี ความซับซ้อนน้อยกว่าอัลกอริทึม YOLO แบบสมบูรณ์ โดยส่งผลดีในด้านความเร็วในการประมวลผล แต่ความ ถูกต้องแม่นยำจะลดลงในงานนี้ได้เลือกใช้ Tiny YOLOv3 ซึ่งเป็นรุ่นที่มีการพัฒนาประสิทธิภาพในด้านความ ถูกต้องแม่นยำให้มากขึ้นจากรุ่นเดิมโดย Tiny YOLOv3 มีจำนวนชั้นของโครงสร้างเครือข่ายประสาทเทียม (Layers) จำนวน 24 ชั้น ซึ่งมีจำนวนน้อยกว่าอัลกอริทึมแบบ YOLO ประเภทสมบูรณ์ที่มีจำนวน 106 ชั้น ทำให้ลดเวลาและลดการใช้ทรัพยากรในการประมวลผลลงได้อย่างมาก

ตัวอย่างการทำ Object Detection ด้วย yolo

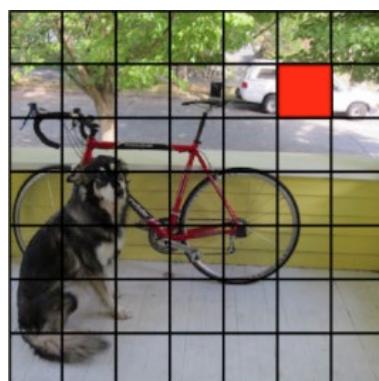
1. ภาพ Input เข้ามา



2. ทำการตัดภาพเป็นชิ้นสี่เหลี่ยม



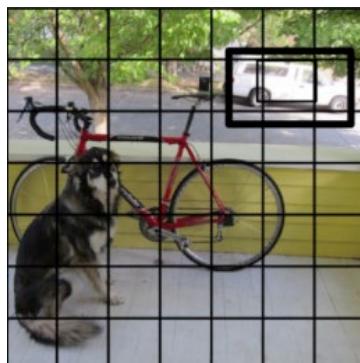
3. ชิ้นสี่เหลี่ยมที่ตัดออกจะทำการนำ回去คำนวณความเชื่อมั่นสิ่งที่จะเป็น



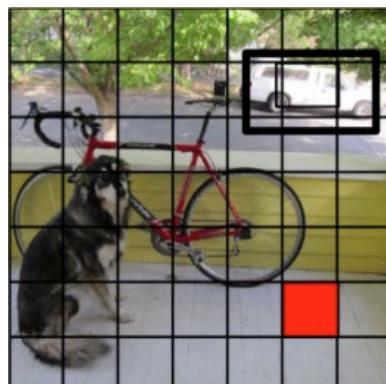
4. ทำการตีกรอบสิ่งที่ตรวจจับได้



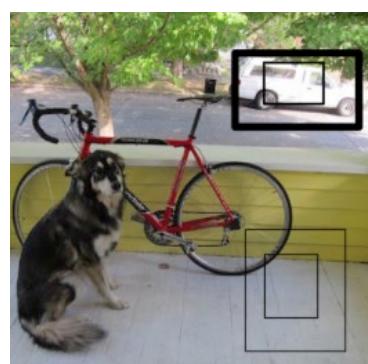
5. ใน 1 ภาพจะมี 2 กระบวนการคือกระบวนการตัดเป็นชิ้นสี่เหลี่ยมและนำ回去คำนวณการตีกรอบให้กับวัตถุที่เจอ



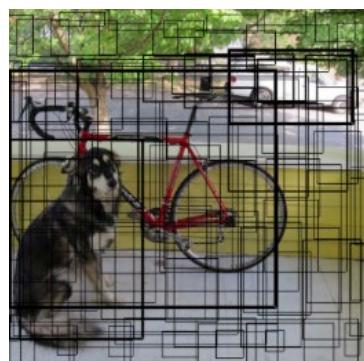
6.ภาพแสดงถึงคลาสที่ทำนายได้กับกรอบสีเหลืองที่เจอจากกระบวนการคุณโนลูชั่น



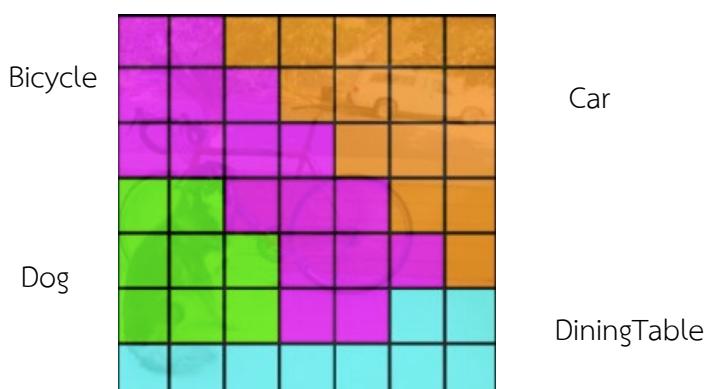
7.ความหมายของสีเหลืองแสดงถึงค่าของความเชื่อมั่นที่ตรวจเจอ



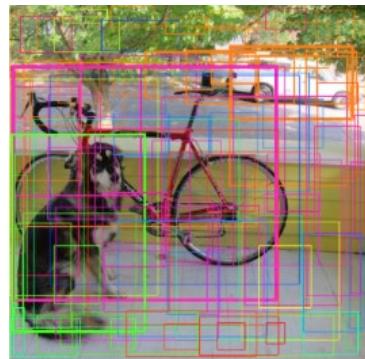
8.ผลลัพธ์จากการตรวจเช็คทั้งหมดในภาพ



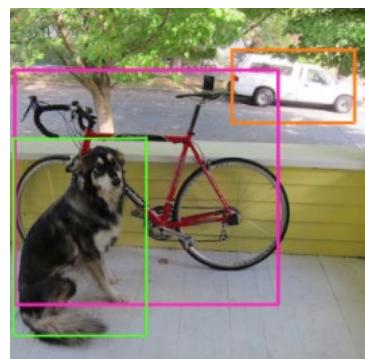
9.ภาพที่แบ่งออกเป็นสีกับคลาสที่ตรวจเจอ



10. กระบวนการจากขั้นตอนที่ 8 กับขั้นตอนที่ 9 รวมกัน



11. จากขั้นตอนที่ 10 ผ่านกระบวนการ Non-maximum Suppression จะได้ผลลัพธ์ดังภาพ



ขั้นตอนของ Non-maximum Suppression

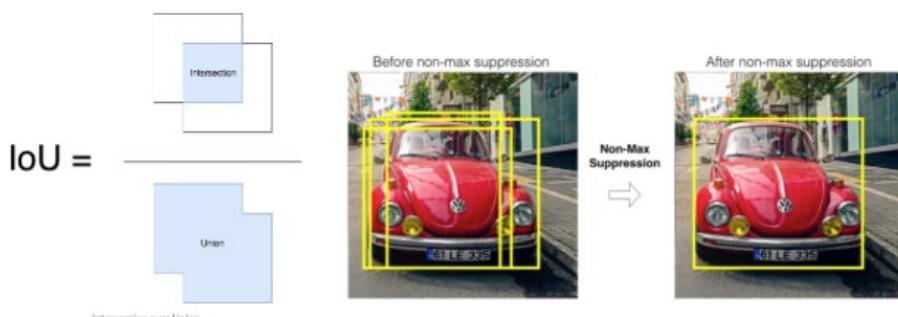


Figure 25 กระบวนการ Intersection over Union (IoU)

กระบวนการ Intersection over Union (IoU) ซึ่งคำนวณจากอัตราส่วนของพื้นที่ซ้อนทับ (ของพื้นที่ที่ทำนายกับพื้นที่จริง) กับ พื้นที่รวม (ของพื้นที่ที่ทำนายกับพื้นที่จริง) ในการทำนายกรอบล้อมวัตถุจะได้ข้อมูลเป็นชุดข้อมูลประเภทอาร์เรย์ ซึ่งประกอบด้วยข้อมูลการมืออยู่จริงของวัตถุ ตำแหน่งและขนาดของกรอบล้อมวัตถุ และชนิดของวัตถุ

ภาพรวมขั้นตอนของการ Object Detection ด้วย Yolo

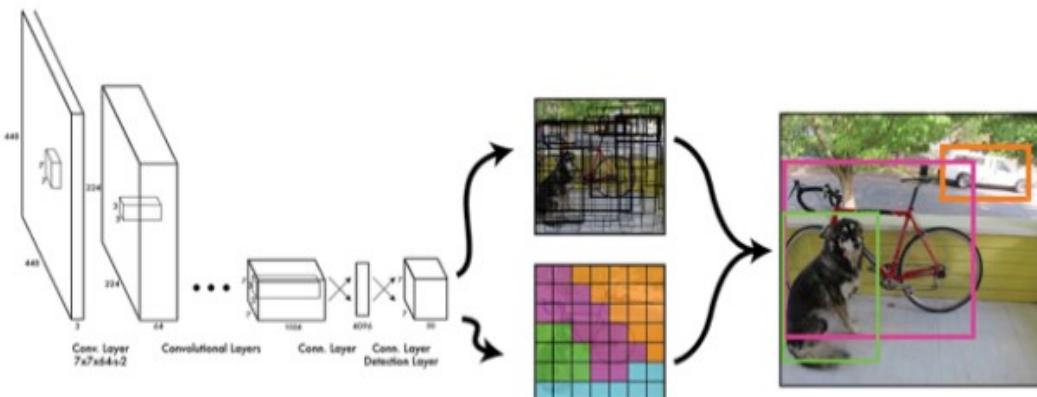
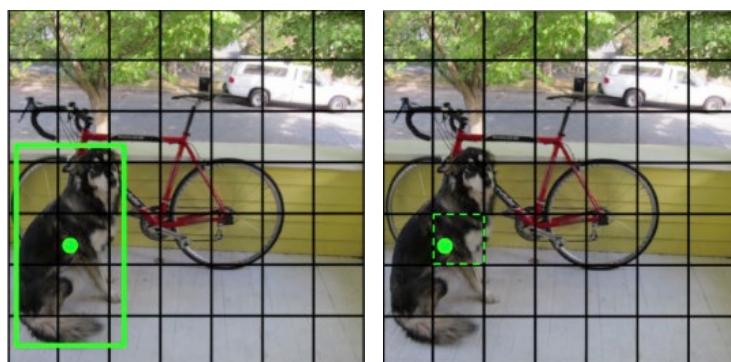


Figure 26 ภาพรวมขั้นตอนของการ Object Detection ด้วย Yolo

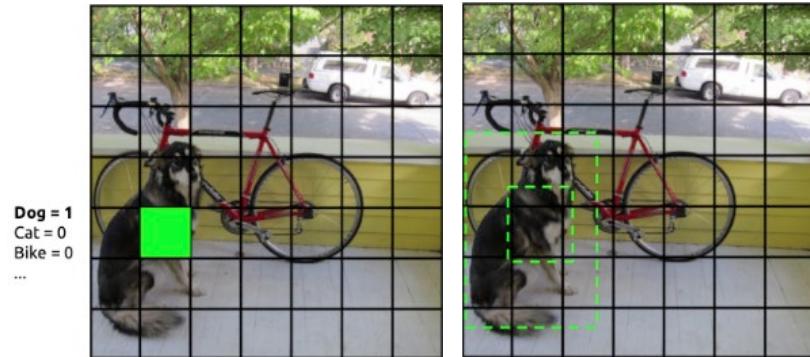
เมื่อมีรูปภาพเข้ามา ก็จะผ่านกระบวนการโครงสร้างข่ายประสาทเทียมแบบคอนโวลูชันซึ่งจะผ่านชั้นต่าง ๆ ที่เป็นอัลกอริทึมของ yolo และจากนั้นจะทำการแบ่งภาพออกเป็น 2 ส่วนคือส่วนแรกคือส่วนที่อัลกอริทึมตรวจพบวัตถุและตีกรอบสีเหลี่ยมไว้ตามค่าความเชื่อมั่นที่เจอสัมพันธ์กับความหนาของกรอบสีเหลี่ยม ส่วนที่สองคือการตัดภาพออกเป็นสีเหลี่ยมจัตุรัสจากนั้นกำหนดคลาสเอาไว้ตามตำแหน่งต่าง ๆ เมื่อกำหนดเสร็จสิ้น จะนำส่วนแรกและส่วนที่สองมาร่วมเข้าด้วยกันทั้งสองส่วน ผลลัพธ์ที่ได้ส่วนแรกนั้นกรอบสีเหลี่ยมแต่ละอันก็จะมีคลาสของตัวเอง ขั้นตอนเดียวมาก็จะต้องผ่านกระบวนการที่เรียกว่า Non-maximum Suppression ก็จะได้ผลลัพธ์ที่ต้องการ

การระบุคลาสให้แต่ละชิ้นส่วนของชิ้นสีเหลี่ยม

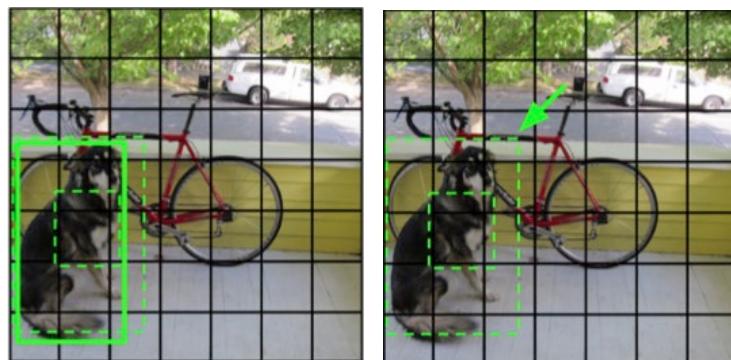
1. ในระหว่างขั้นตอนการฝึกฝน



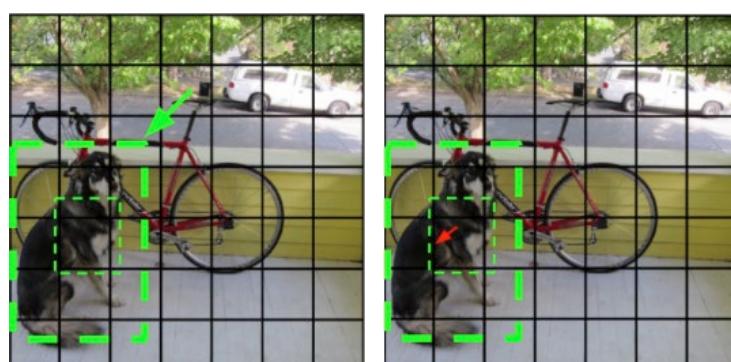
2. ระบบจะทำการเทียบค่าความเชื่อมั่นในช่องนั้น ๆ จากนั้นจะนับช่องรอบ ๆ เพื่อหากรอบของ object โดยที่เมื่อได้กรอบแล้วจะเข้าดูว่าค่าความเชื่อมั่นมากกว่าช่อง หรือ กรอบในนั้นใหม่ ถ้าใช้ก็เพิ่มค่าความเชื่อมั่นกับกรอบปัจจุบัน จากนั้นลบกรอบด้านนอก



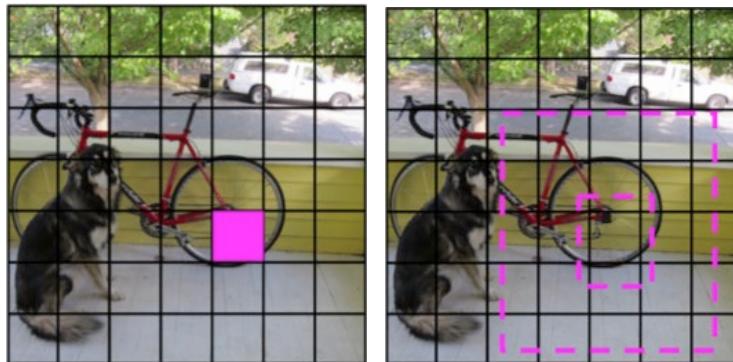
3. เมื่อทำการให้ค่าความเชื่อมั่น จะได้กรอบรอบ ๆ object ออกมานิ่งๆ ก็จะมีกรอบนั้นทับซ้อน



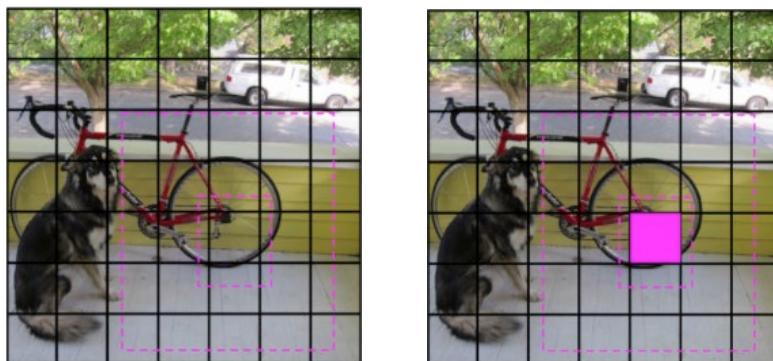
4. ขั้นตอนนี้จะทำการเช็คกันระหว่างกรอบที่มีอยู่โดยเพิ่มค่าความเชื่อมั่นของกรอบที่ดีที่สุดแล้วลดของกรอบที่น้อยกว่า



5. ในบางจุดที่ไม่มีค่าของไหร่หรือไม่สามารถลบออกได้ว่ามันคืออะไรนั้น ระบบก็จะทำการลดค่าความเชื่อมั่นไปเรื่อยๆ



6. ทำการลดไปเรื่อยๆ จนถึงขั้นระบุว่าไม่มี class นั้นใน coco data set ที่มีอยู่



เปรียบเทียบ Yolo กับรูปแบบการตรวจจับกับอัลกอริทึมอื่น ๆ

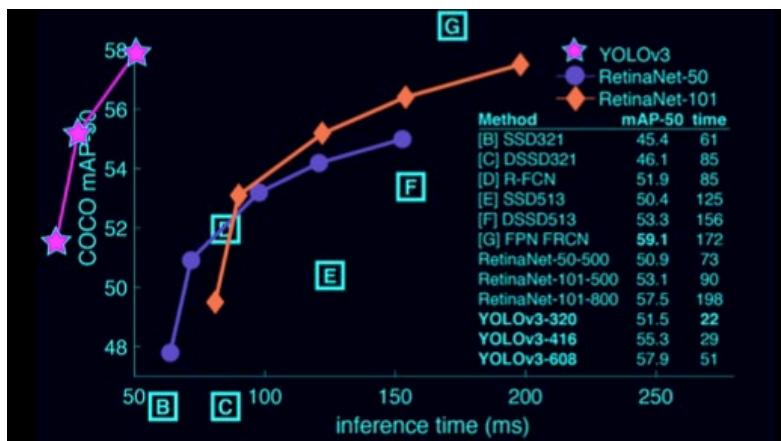


Figure 27 การเปรียบเทียบโมเดลต่างๆ ที่สามารถนำมาทำ object detection

ดังภาพข้างต้นคือการเปรียบเทียบโมเดลต่างๆ ที่สามารถนำมาทำ object detection ก็จะมี yolov3 / retinanet 50 / retinanet 101 จะเห็นได้ว่า yolov3 ที่ค่าตอบสนองต่อวินาทีการคาดเดานี้เร็วมาก เร็วกว่าสองตัวที่นำมาเปรียบเทียบคือ retinanet 50 และ retinanet 101

ตัวอย่างการนำไปใช้งาน

1. การตรวจจับหมวดนิรภัยและการใช้อาวุธปืน เพื่อเตือนภัยเหตุโจรอกรรรม จากภาพกล้องวงจรปิดแบบเวลาจริง¹



Figure 28 การตรวจจับหมวดนิรภัยและการใช้อาวุธปืน เพื่อเตือนภัยเหตุโจรอกรรรม

2. การสร้างระบบตรวจจับบุคคลแบบเวลาจริงราคาประหยัด บน Raspberry Pi โดยประยุกต์อัลกอริทึม Tiny YOLO V3²

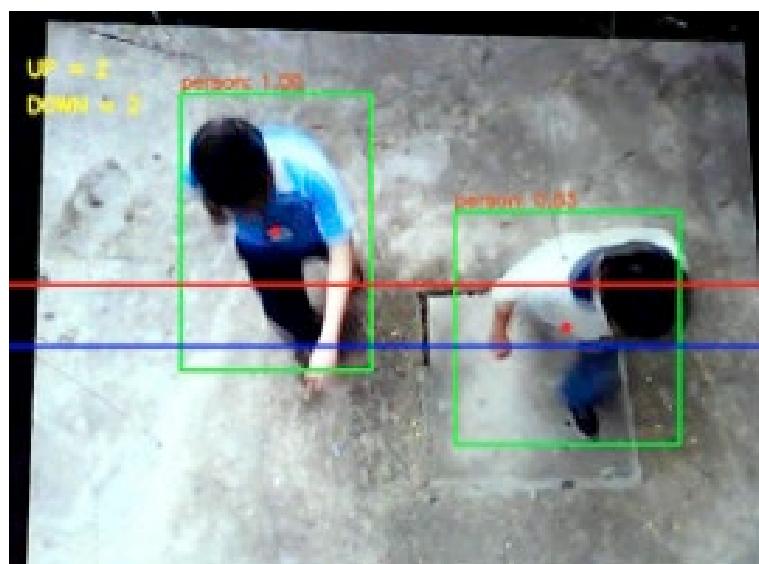


Figure 29 การสร้างระบบตรวจจับบุคคลแบบเวลาจริง

¹ งานวิจัย คุณยังยุทธ ละมูลมณฑ และคุณธนาสัย สุคนธ์พันธ์

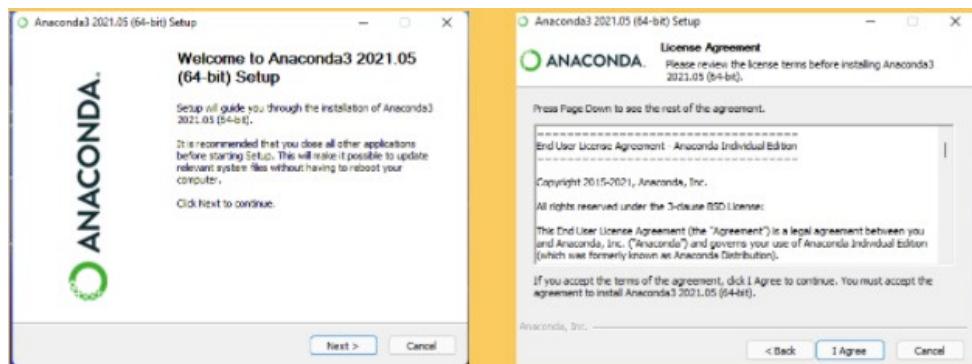
² งานวิจัย คุณปราโมทย์ ปัญญาโต และคุณนลิน สีดาห้าว

การติดตั้ง Anaconda

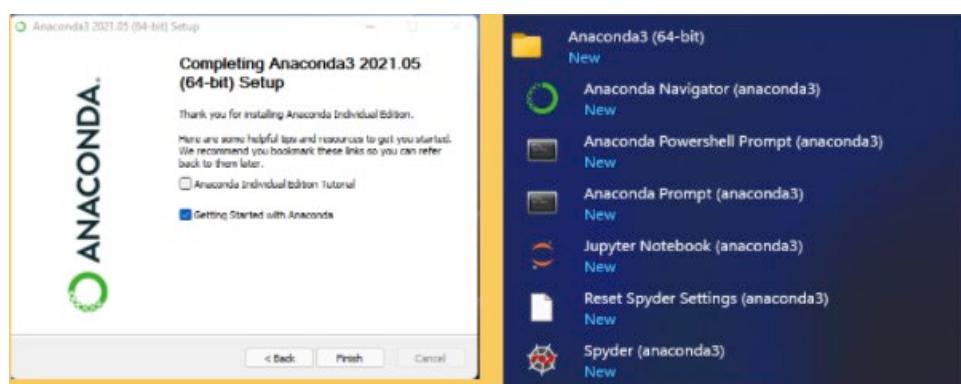
- ให้ Download Anaconda จากลิ้งค์: <https://www.anaconda.com/products/individual#Downloads>



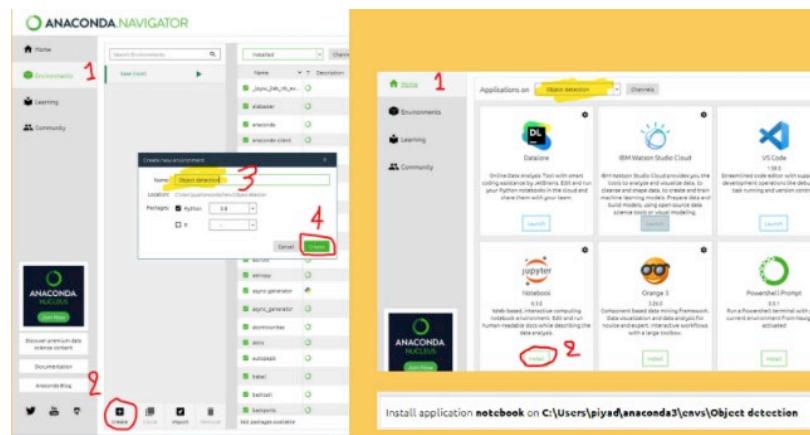
- กด install ตามภาพ



- ติดตั้งเสร็จสนิท

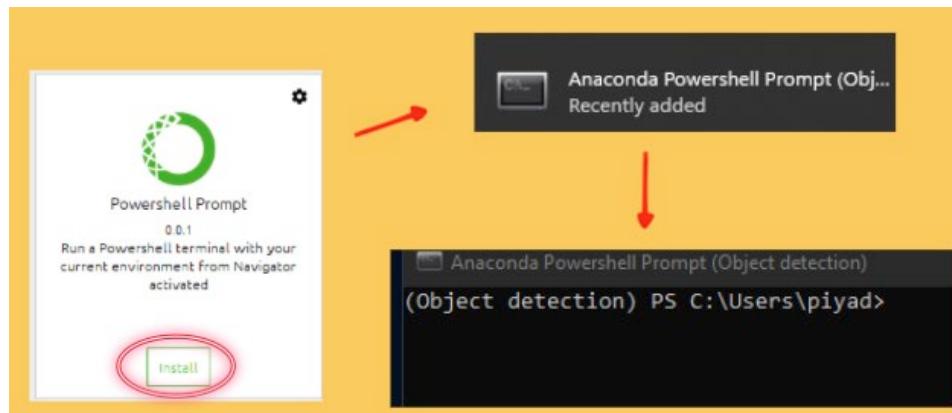


4. สร้าง Envoriment สำหรับการใช้งาน

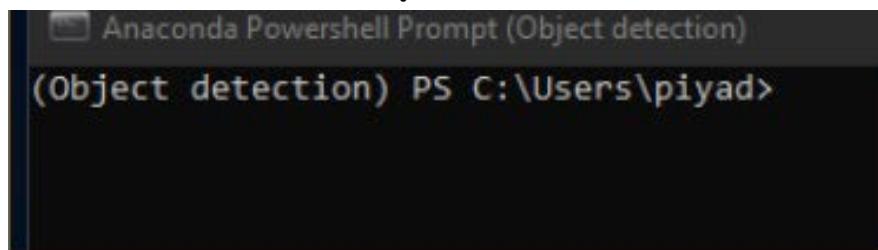


ติดตั้ง PowerShell Prompt

1. กดปุ่ม install ตามภาพด้านล่าง



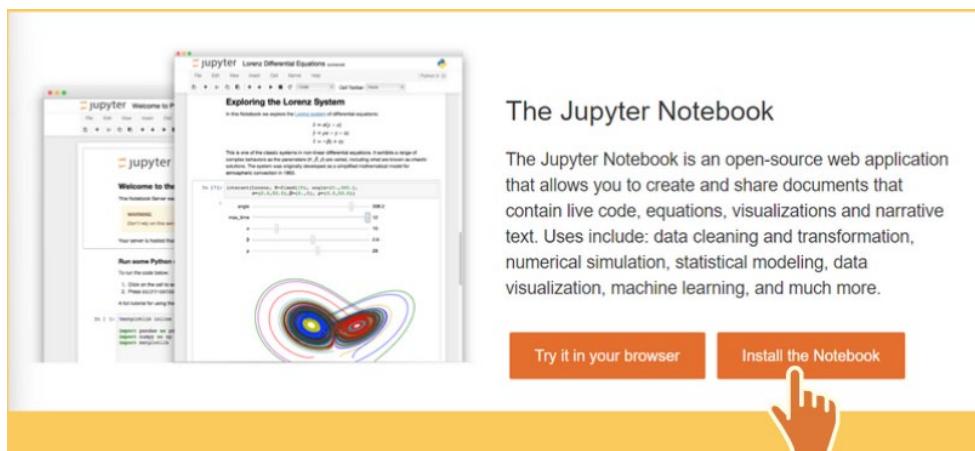
2. หน้าตาของโปรแกรมที่จะแสดงออกมารูปแบบนี้



Jupyter



1. กดที่ install the Notebook (กรณีไม่สามารถติดตั้งจาก Anaconda ได้)



The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

[Try it in your browser](#) [Install the Notebook](#)

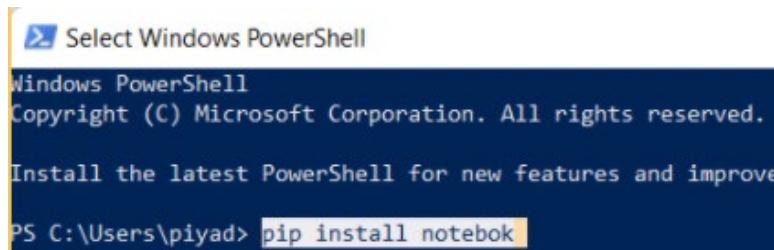
2. คัดลอกคำสั่งที่จะถูกใช้ในการติดตั้ง “`pip install notebook`”\

Installation with pip

If you use `pip`, you can install it with:

```
pip install notebook
```

3. นำไปพิมพ์ลงใน powershell ที่เปิดเอาไว้



```
PS C:\Users\piyad> pip install notebook
```



4. รอให้ระบบทำการติดตั้ง

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSW

PS C:\Users\piyad> pip install notebook
Collecting notebook
  Using cached notebook-6.4.2-py3-none-any.whl (9.7 kB)
Collecting nbformat
  Using cached nbformat-5.1.3-py3-none-any.whl (178 kB)
Collecting ipython_genutils
  Using cached ipython_genutils-0.2.0-py3-none-any.whl (26 kB)
Collecting traitlets>=4.2.1
  Using cached traitlets-5.0.5-py3-none-any.whl (180 kB)
Collecting send2trash>=1.5.0
  Using cached send2trash-1.7.1-py3-none-any.whl (17 kB)
Collecting jupyter_core>=4.6.1
  Using cached jupyter_core-4.7.1-py3-none-any.whl (82 kB)
Collecting nbconvert
  Using cached nbconvert-6.1.0-py3-none-any.whl (551 kB)
Collecting pyzmq>=17
  Using cached pyzmq-22.2.1-cp39-cp39-win_amd64.whl (1.0 MB)
Collecting ipykernel
  Using cached ipykernel-6.0.3-py3-none-any.whl (122 kB)
Collecting prometheus_client
  Using cached prometheus_client-0.11.0-py2.py3-none-any.whl (56 kB)
Collecting jinja2
  Using cached jinja2-3.0.1-py3-none-any.whl (133 kB)
Collecting jupyter-client>=5.3.4
  Using cached jupyter_client-6.1.12-py3-none-any.whl (112 kB)
Collecting tornado>=6.1
  Using cached tornado-6.1-cp39-cp39-win_amd64.whl (422 kB)
Collecting argon2-cffi<20.1.0
  Using cached argon2_cffi-20.1.0-cp39-cp39-win_amd64.whl (42 kB)
Collecting terminado>=0.8.3
```

5. เลือกตำแหน่งที่ต้องการเปิด notebook ไว้ใช้งาน

```
PS C:\Users\piyad> D:
PS D:>
```

6. ใช้คำสั่งในการเรียกใช้งานตัว jupyter notebook

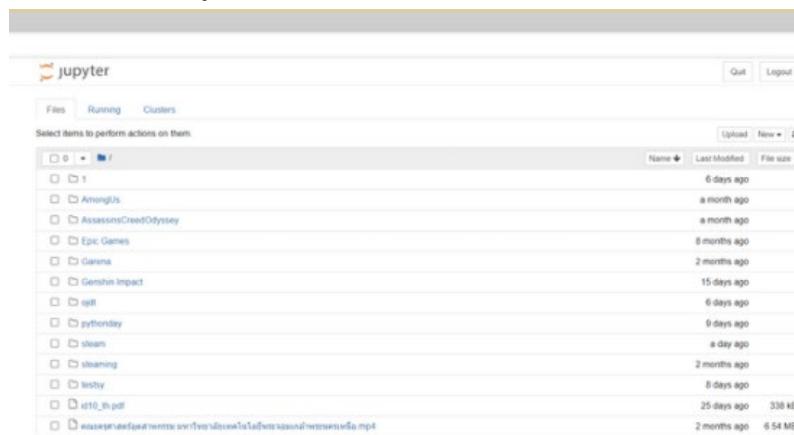
```
PS C:\Users\piyad>
PS C:\Users\piyad> D:
PS D:> jupyter notebook
```

7. รอให้ระบบทำการเรียกการใช้งาน Jupyter notebook

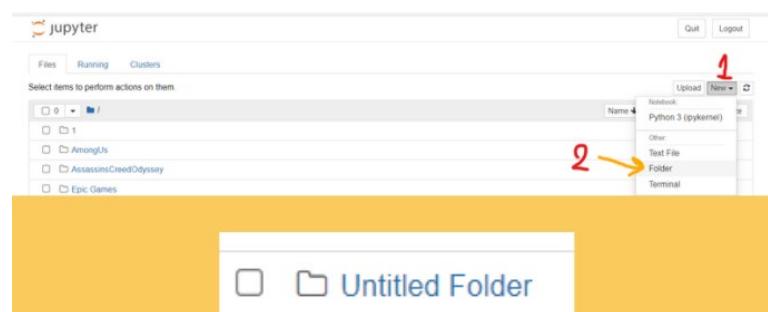
```
PS C:\Users\piyad> jupyter notebook
[I 23:31:08.050 NotebookApp] Serving notebooks from local directory: C:\Users\piyad
[I 23:31:08.051 NotebookApp] Jupyter Notebook 6.4.2 is running at:
[I 23:31:08.051 NotebookApp] http://localhost:8888/?token=105905fb70fe955708c74f25352f4d5ebac9abdf94fe673
[I 23:31:08.051 NotebookApp] or http://127.0.0.1:8888/?token=105905fb70fe955708c74f25352f4d5ebac9abdf94fe673
[I 23:31:08.051 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 23:31:08.096 NotebookApp]

To access the notebook, open this file in a browser:
  file:///C:/Users/piyad/AppData/Roaming/jupyter/runtime/nbserver-13704-open.html
Or copy and paste one of these URLs:
  http://localhost:8888/?token=105905fb70fe955708c74f25352f4d5ebac9abdf94fe673
or http://127.0.0.1:8888/?token=105905fb70fe955708c74f25352f4d5ebac9abdf94fe673
```

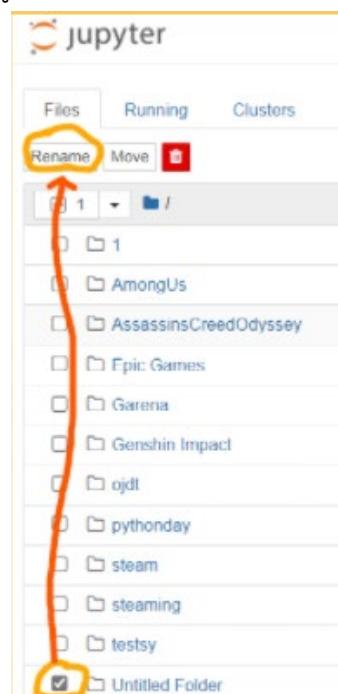
8. หน้าตาของโปรแกรมจะถูกแสดงทาง Browser ของเครื่อง



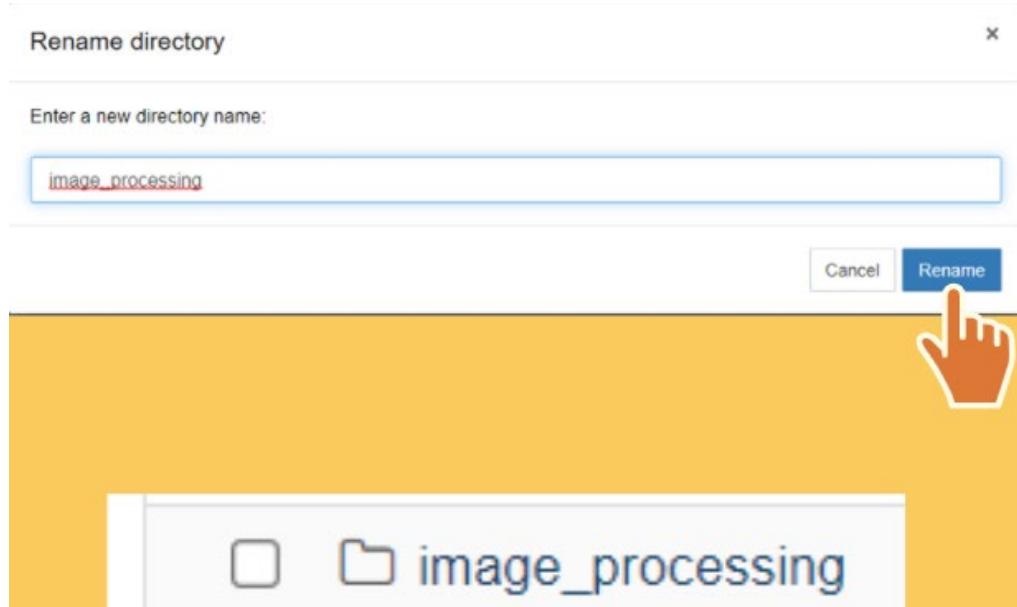
9. การสร้าง Folder สำหรับพื้นที่งานที่จะใช้งาน



10. การเปลี่ยนชื่อ Folder ที่คุณสร้างขึ้นมา



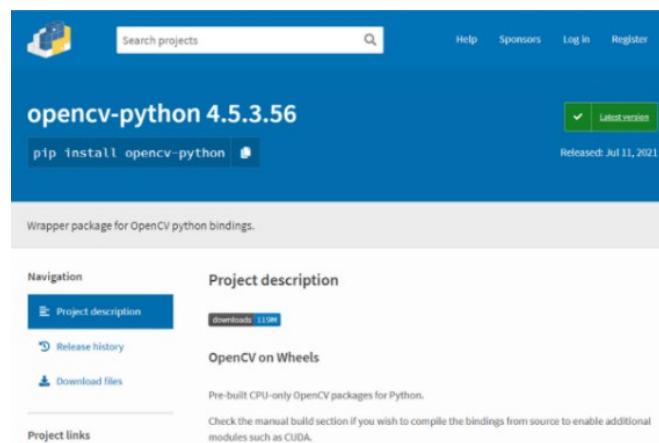
11. ตั้งชื่อแล้วกด Rename



Opencv

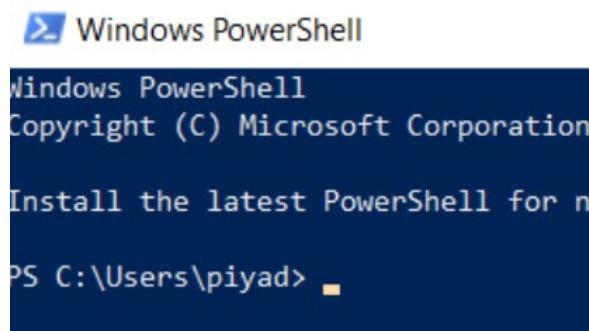


1. ทำการติดตั้ง OpenCV

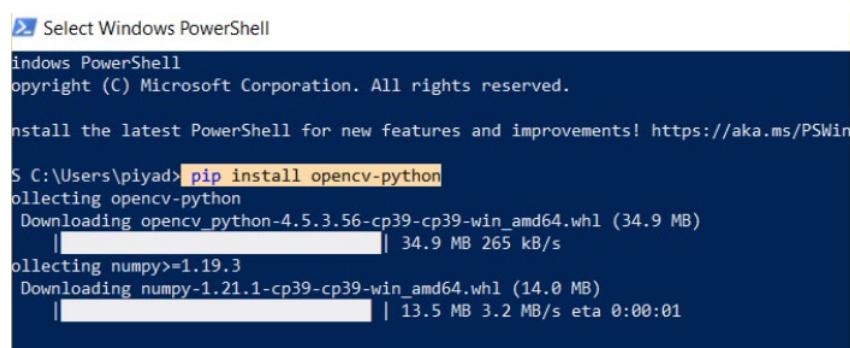


The screenshot shows the PyPI project page for 'opencv-python 4.5.3.56'. At the top, there's a search bar and navigation links for Help, Sponsors, Log In, and Register. Below the title, there's a green button labeled 'Latest version' with a checkmark. To its right, it says 'Released: Jul 11, 2021'. The main content area contains the command 'pip install opencv-python' and a note: 'Wrapper package for OpenCV python bindings.' On the left, there's a 'Navigation' sidebar with 'Project description' (selected), 'Release history', 'Download files', and 'Project links'. On the right, there's a 'Project description' section with a note about pre-built CPU-only OpenCV packages for Python.

2. ทำการเปิด powershell เพิ่มขึ้นมาอีก 1 หน้าต่าง



3. รอให้ระบบทำการติดตั้ง



The screenshot shows a Windows PowerShell window with a yellow vertical bar highlighting the command line. The command 'pip install opencv-python' is being run. The output shows the download of 'opencv_python-4.5.3.56-cp39-cp39-win_amd64.whl' (34.9 MB) at 34.9 MB 265 kB/s, and the download of 'numpy-1.19.3' (14.0 MB) at 13.5 MB 3.2 MB/s eta 0:00:01.

```
PS C:\Users\piyad> pip install opencv-python
Collecting opencv-python
  Downloading opencv_python-4.5.3.56-cp39-cp39-win_amd64.whl (34.9 MB)
    |████████| 34.9 MB 265 kB/s
Collecting numpy>=1.19.3
  Downloading numpy-1.19.3-py3-none-any.whl (14.0 MB)
    |████████| 13.5 MB 3.2 MB/s eta 0:00:01
```

NUMPY



- ติดตั้ง NUMPY โดยใช้ pip install numpy

INSTALLING NUMPY

The only prerequisite for installing NumPy is Python itself. If you don't have Python yet and want the simplest way to get started, we recommend you use the [Anaconda Distribution](#) - it includes Python, NumPy, and many other commonly used packages for scientific computing and data science.

NumPy can be installed with `conda`, with `pip`, with a package manager on macOS and Linux, or [from source](#). For more detailed instructions, consult our [Python and NumPy installation guide](#) below.

CONDA

If you use `conda`, you can install NumPy from the `defaults` or `conda-forge` channels:

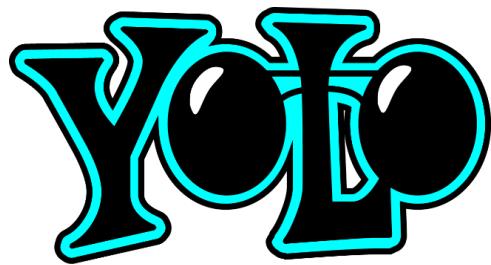
```
# Best practice, use an environment rather than install in the base env
conda create -n my-env
conda activate my-env
# If you want to install from conda-forge
conda config --env --add channels conda-forge
# The actual install command
conda install numpy
```

PIP

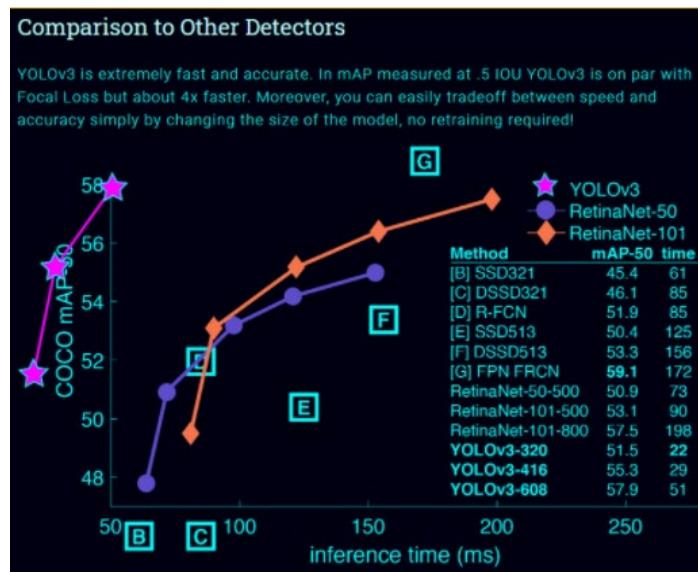
If you use `pip`, you can install NumPy with:

```
pip install numpy
```

YOLO



1. พิจารณาว่าจะใช้ YOLO ตัวไหนในการทำงาน (ในที่นี้ YOLO-tiny)

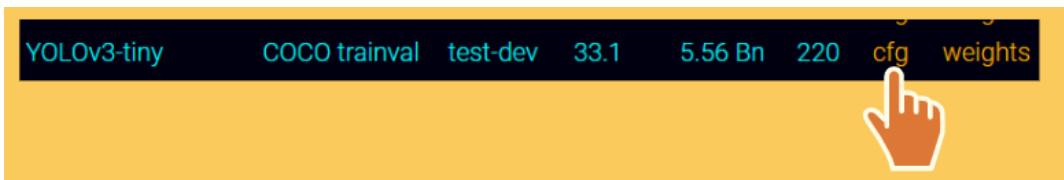


2. เลือกตัวที่เราต้องการใช้งานจากเว็บ

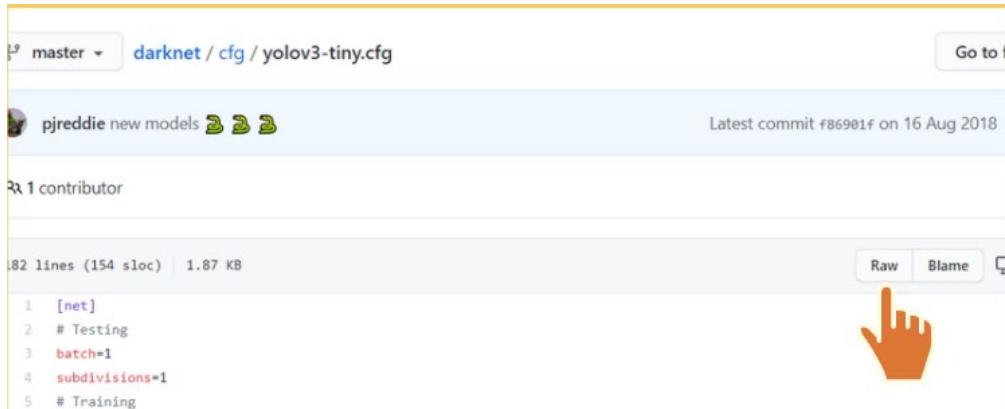
Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

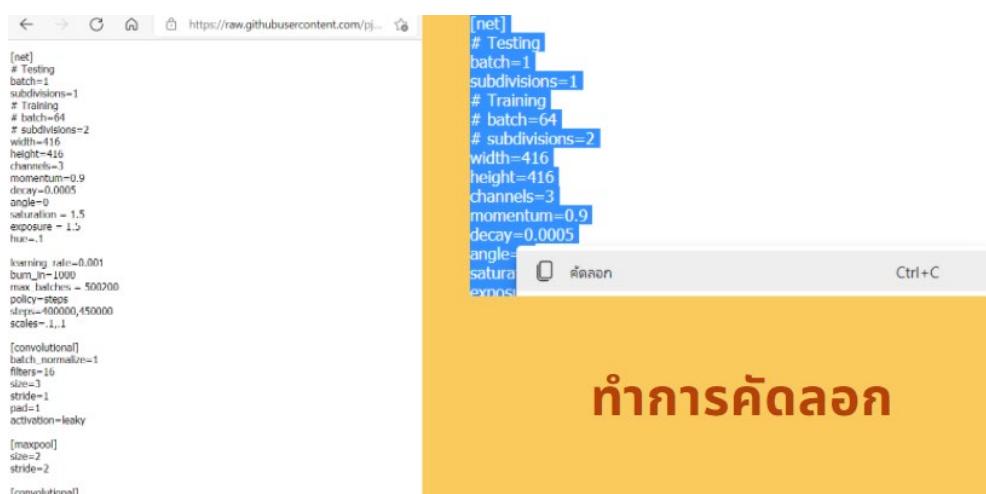
3. เลือก download ไฟล์ cfg



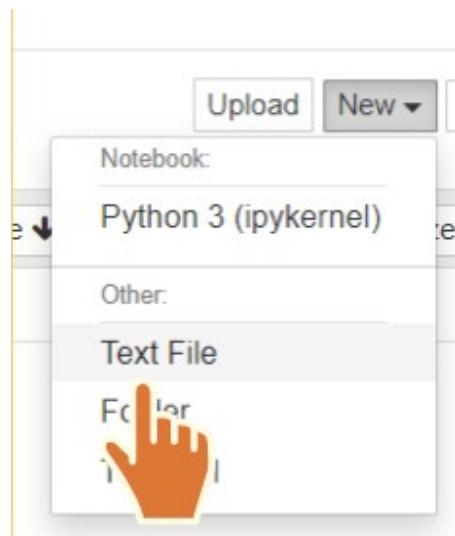
4. เลือกที่ RAW



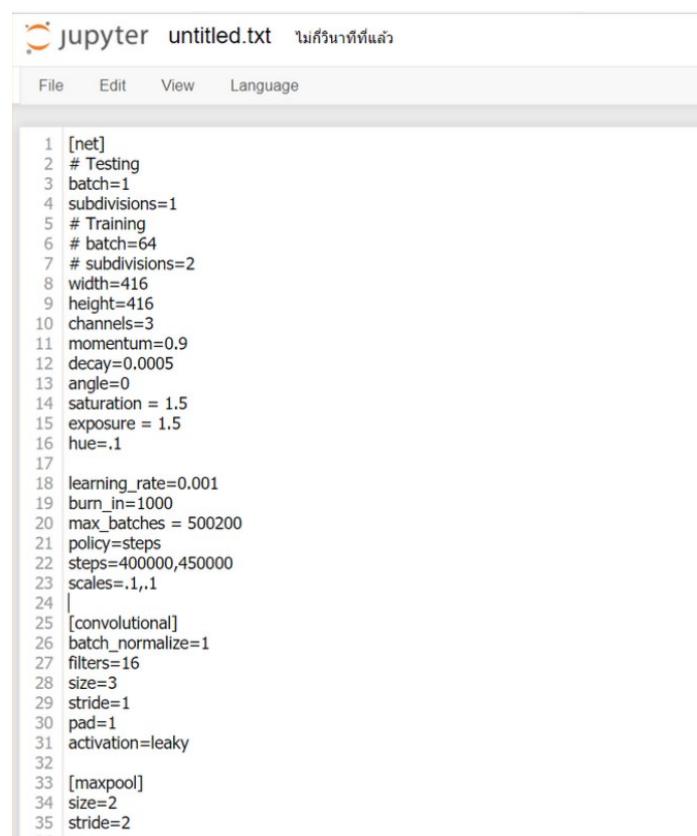
5. คัดลอกข้อความภายใต้ทั้งหมด



6. กลับมาที่ jupyter book แล้วทำการสร้าง text file

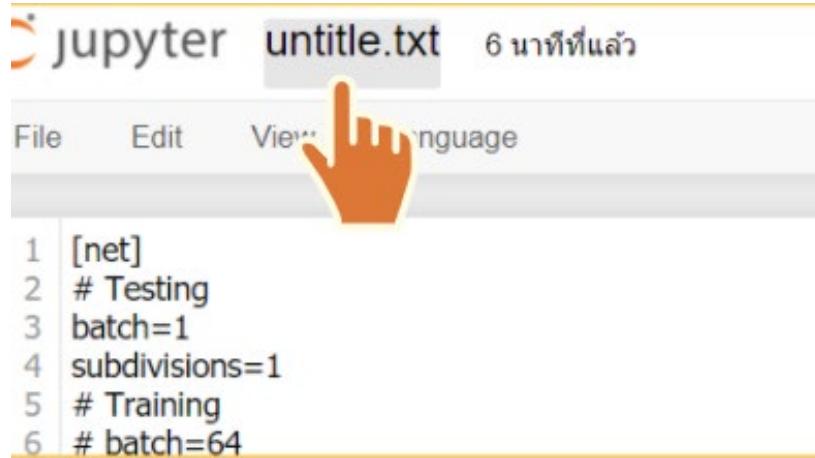


7. นำมารวังไว้ที่ text file ที่สร้างเอาไว้



```
1 [net]
2 # Testing
3 batch=1
4 subdivisions=1
5 # Training
6 # batch=64
7 # subdivisions=2
8 width=416
9 height=416
10 channels=3
11 momentum=0.9
12 decay=0.0005
13 angle=0
14 saturation = 1.5
15 exposure = 1.5
16 hue=.1
17
18 learning_rate=0.001
19 burn_in=1000
20 max_batches = 500200
21 policy=steps
22 steps=400000,450000
23 scales=.1,.1
24 |
25 [convolutional]
26 batch_normalize=1
27 filters=16
28 size=3
29 stride=1
30 pad=1
31 activation=leaky
32
33 [maxpool]
34 size=2
35 stride=2
36
```

8. แก้ไขข้อไฟล์เป็น yolov3.cfg

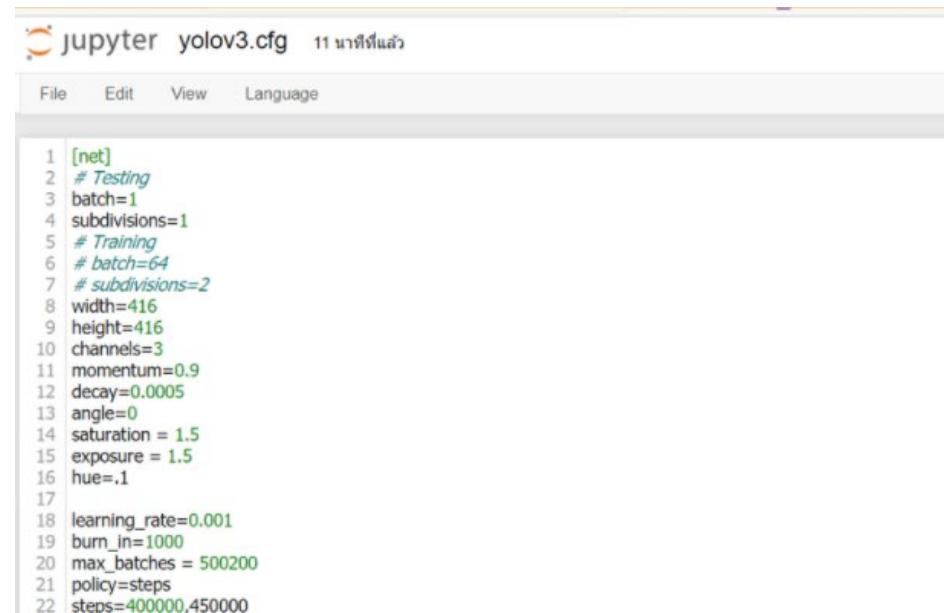


```

1 [net]
2 # Testing
3 batch=1
4 subdivisions=1
5 # Training
6 # batch=64

```

9. หน้าตาของไฟล์ที่จะได้ออกมา

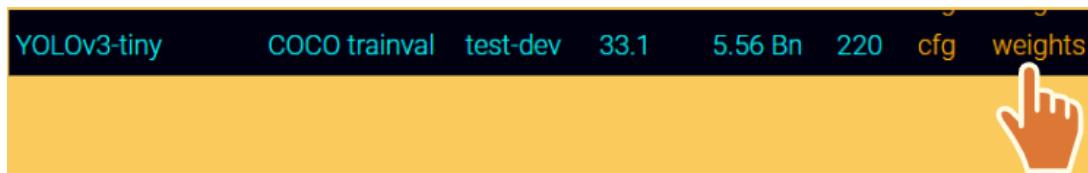


```

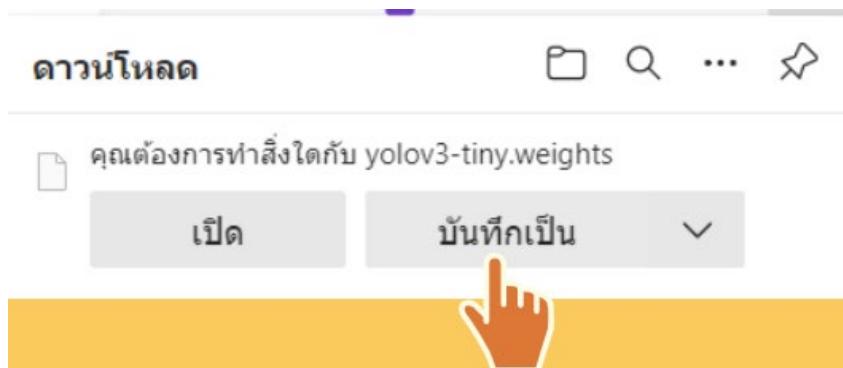
1 [net]
2 # Testing
3 batch=1
4 subdivisions=1
5 # Training
6 # batch=64
7 # subdivisions=2
8 width=416
9 height=416
10 channels=3
11 momentum=0.9
12 decay=0.0005
13 angle=0
14 saturation = 1.5
15 exposure = 1.5
16 hue=.1
17
18 learning_rate=0.001
19 burn_in=1000
20 max_batches = 500200
21 policy=steps
22 steps=400000,450000

```

10. download ไฟล์ weights



11. กด บันทึกเป็น หรือ save as



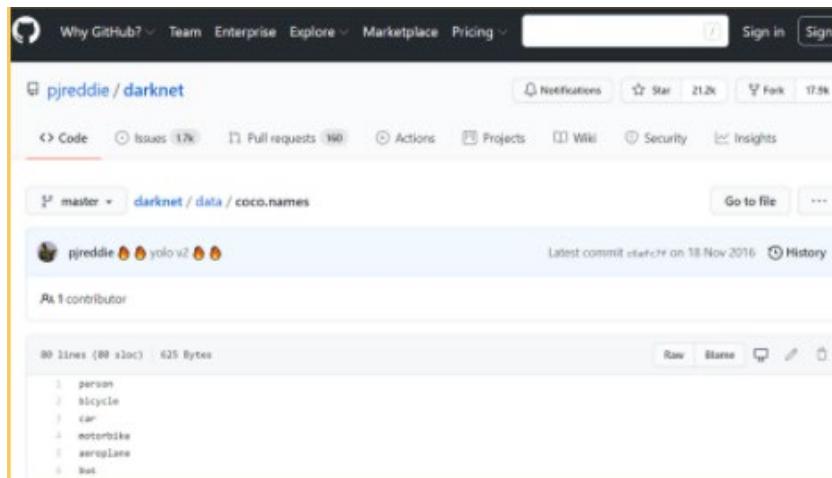
12. ไฟล์ที่จะต้องมีในตอนนี้



COCO



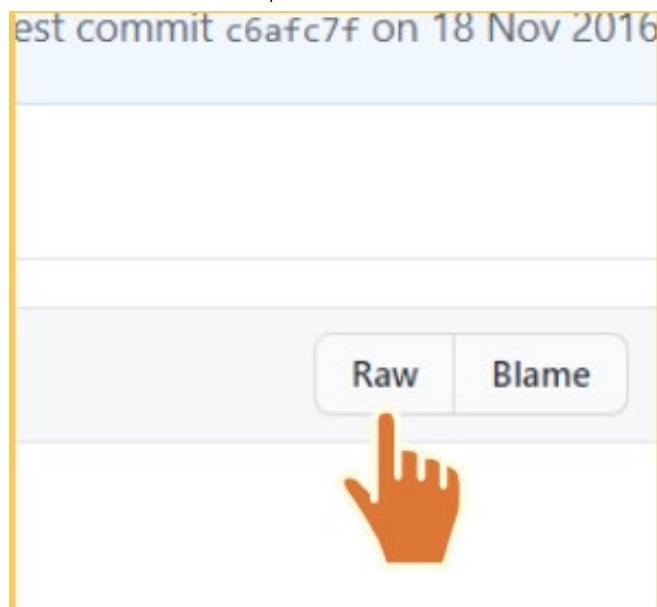
- สามารถเข้าไปได้ที่ [https://github.com/pjreddie/darknet/blob/master/
data/coco.names](https://github.com/pjreddie/darknet/blob/master/data/coco.names)



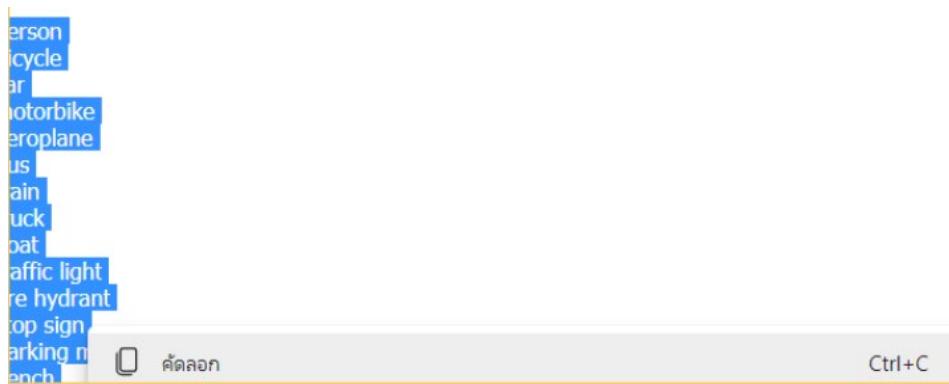
A screenshot of a GitHub repository page for 'pjreddie / darknet'. The repository has 17k issues, 160 pull requests, 21.2k stars, 17.5k forks, and 11 contributors. The 'coco.names' file is selected, showing its content. The file contains the following text:

```
person
bicycle
car
motorbike
aeroplane
bus
```

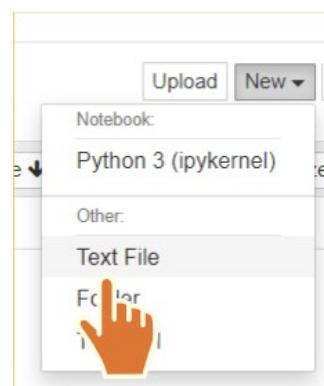
- ทำการ download มาด้วยการกดที่ปุ่ม RAW



3. คัดลอกจาก notepad



4. สร้างไฟล์ text file ขึ้นมา



5. วางข้อความที่เราคัดลอกมาลงไป

jupyter untitled.txt ไม่รีบราที่แล้ว

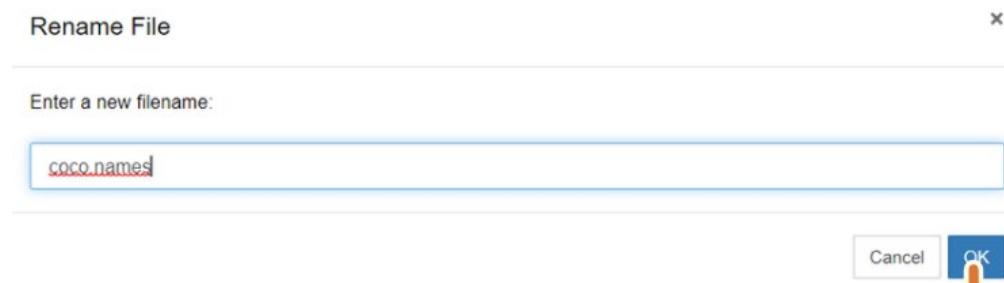
```
File Edit View Language
1 person
2 bicycle
3 car
4 motorbike
5 aeroplane
6 bus
7 train
8 truck
9 boat
10 traffic light
11 fire hydrant
12 stop sign
13 parking meter
14 bench
15 bird
16 cat
17 dog
18 horse
```



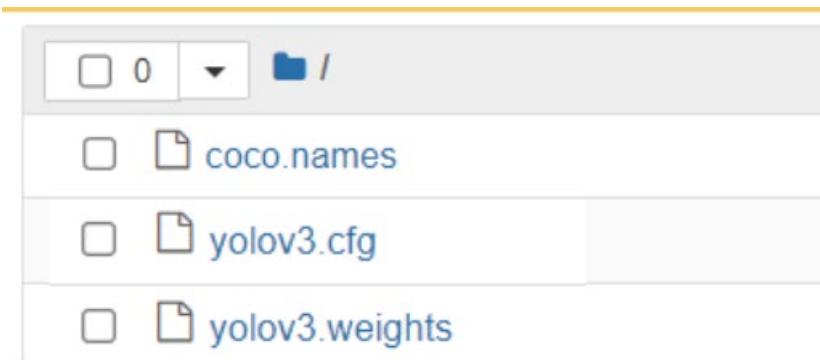
6. เปลี่ยนชื่อไฟล์ เป็น coco.names

```
jupyter untitled.txt 1 นาทีที่แล้ว
File Edit View Language
1 person
2 bicycle
3 car
4 motorbike
5 aeroplane
6 bus
7 train
8 truck
9 boat
10 traffic light
11 fire hydrant
```

7. คลิก OK



8. ไฟล์ที่เราต้องมีในตอนนี้



9. ไฟล์ภาพที่ได้ทำการเตรียมไว้ให้ผู้รวมอบรมสามารถนำรูปอื่นมาใช้งานได้



10. ไฟล์ที่จะมีอยู่

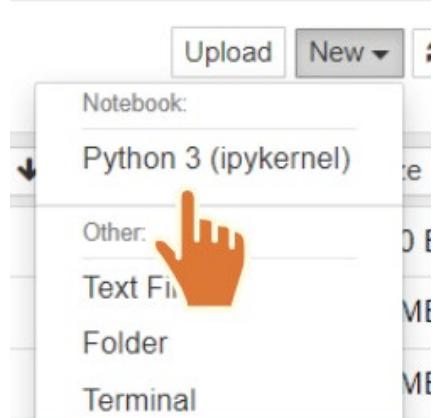
 coco.names

 example.png

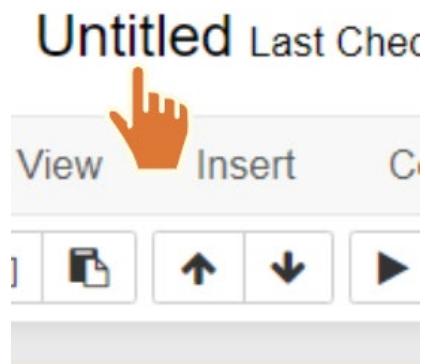
 yolov3-tiny.weights

 yolov3.cfg

11. สร้างไฟล์ที่เราจะทำการเขียนโค้ดลงไว้



12. กดที่ชื่อของไฟล์เพื่อเปลี่ยนชื่อ

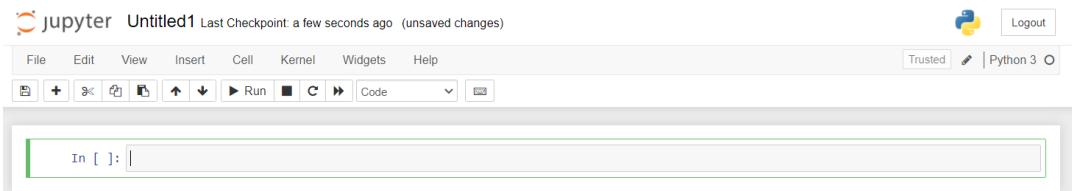


13. หลังจากตั้งชื่อแล้วให้ทำการ กดที่ rename



เริ่มการเขียนโค้ด

1. ไปที่หน้าต่างไฟล์ Python ที่สร้างขึ้นมา



2. นำเข้า Library ต่าง ๆ

```
import numpy as np
import cv2
import time
```

3. ทำการสร้างตัวแปรเพื่อเก็บค่าที่จะถูกนำเข้ามา

```
INPUT_FILE = 'example.jpg'
LABELS_FILE = 'coco.names'
CONFIG_FILE = 'yolov3.cfg'
WEIGHTS_FILE = 'yolov3.weights'
CONFIDENCE_THRESHOLD = 0.3
```

4. คำสั่งให้นำ LABEL FILE ซึ่งตอนนี้เป็น COCO.NAMES ให้ระบบนำไปแกะออกมานำไปบรรจุใน LABEL โดยให้เรียกเป็นบรรทัดเอ้าไว้

```
LABELS = open(LABELS_FILE).read().strip().split("\n")
```

5. คำสั่งให้สร้างค่า seed ที่ถูกสุ่มขึ้นมาจากนั้นนำไปสู่สีที่แตกต่างกัน สำหรับนำไปแสดงเป็นกรอบรอบขึ้นวัตถุที่ถูก Detection

```
np.random.seed(4)
COLORs = np.random.randint(0,255,size = (len(LABELS),3),dtype = "uint8")
```

6. คำสั่งให้สร้างตัวแปร net ขึ้นมาเพื่อนำค่า CONFIG และ WEIGHT ส่งไปยังเซิร์ฟเวอร์ของ Darknet เพื่อนำมารunning ในโปรแกรมต่อไป

```
net = cv2.dnn.readNetFromDarknet(CONFIG_FILE, WEIGHTS_FILE)
```

7. คำสั่งสร้างตัวแปร image ขึ้นมาเพื่อค่าของรูปภาพตัวอย่างที่นำเข้ามาคำสั่งสร้างตัวแปร (H, W) เพื่อเก็บความสูงและความกว้างของภาพไว้

```
image = cv2.imread(INPUT_FILE)
(H, W) = image.shape[:2]
```

8. คำสั่งกำหนดให้ Output เฉพาะส่วนที่ต้องการจาก YOLO

```
In = net.getLayerNames()  
  
In = [In[i[0]-1]for i in net.getUnconnectedOutLayers()]
```

9. สร้างตัวแปร blob นำไปปลดขนาดรูปภาพที่นำเข้ามาทดสอบให้เหลือ 416*416 ให้สลับสีฟ้ากับน้ำเงิน ให้ส่งค่าไป net.setInput และเริ่มนับเวลาเพื่อที่จะคำนวณว่า YOLO ใช้เวลาในการคำนวณไปเท่าไหร่

```
blob = cv2.dnn.blobFromImage(image, 1/255.0,(416,416),swapRB = True , crop =  
False)
```

```
net.setInput(blob)  
  
start = time.time()  
  
layerOutputs = net.forward(In)  
  
end = time.time()
```

10. แสดงค่าเวลาที่ถูกใช้ไปในการคำนวณ YOLO ขึ้นมาในช่อง OUTPUT

```
print("[INFO] YOLO took {:.6f}seconds".format(end - start))
```

11. ประการตัวแปร Array ทั้งหมด 3 ตัวคือ boxes, confidences, classIDs

```
boxes = []  
  
confidences = []  
  
classIDs = []
```

12. คำสั่งวนซ้ำแต่ละเอาร์พุตของเลเยอร์

```
for output in layerOutputs:
```

13. คำสั่งวนซ้ำการตรวจจับแต่ละครั้ง

```
for detection in output:
```

14. คำสั่ง แยก class ID และ ค่า confidence (เช่น ความน่าจะเป็น) ของการตรวจจับทุกปัจจุบัน

```
scores = detection[5:]  
  
classID = np.argmax(scores)  
  
confidence = scores[classID]
```

15. หากค่า confidence ที่คาดการณ์ว่าตรวจจับ มีค่ามากกว่าเกณฑ์ความน่าจะเป็นวัดถูกในภาพ

```
if confidence > CONFIDENCE_THRESHOLD:
```

16. ปรับขนาด bounding box รูปโดยสัมพันธ์กับขนาดของรูปภาพ YOLO จะส่งตำแหน่งตรงกลาง bounding box (x, y) ของกรอบ ล้อมรอบตามด้วยความกว้างและความสูงของกล่อง

```
box = detection[0:4] * np.array([W, H, W ,H])
(centerX, centerY, width , height) = box.astype("int")
```

17. ใช้จุดตำแหน่งตรงกลาง bounding box (x, y) - ที่มีการ detection เพื่อหามุมด้านบนและด้านซ้ายของ bounding box

```
x = int(centerX - (width /2))
```

```
y = int(centerY - (height /2))
```

18. อัปเดตรายการ bounding box confidence และ classID

```
boxes.append([x ,y ,int(width), int(height)])
```

```
confidences.append(float(confidence))
```

```
classIDs.append(classID)
```

19. คำสั่งสร้างตัวแปร idxs เก็บค่า boxes, confidence, CONFIDENCE_THRESHOLD, CONFIDENCE_THRESHOLD

```
idxs = cv2.dnn.NMSBoxes(boxes, confidences,
CONFIDENCE_THRESHOLD,CONFIDENCE_THRESHOLD)
```

20. คำสั่งถ้ามีการตรวจสอบว่ามีการตรวจพบ idxs อย่างน้อยหนึ่งรายการ

```
if len(idxs)>0:
```

21. คำสั่งวนรอบข้อด้านซ้ายที่เราเก็บไว้ใน idxs

```
for i in idxs.flatten():
```

22. แยกค่า bounding box กล่องขอบเขต

```
(x ,y) = (boxes[i][0], boxes[i][1])
```

```
(w ,h) = (boxes[i][2], boxes[i][3])
```

23. สร้างกล่องสีที่เป็น output ออกมาที่ภาพ

```
color = [int(c)for c in COLORs[classIDs[i]]]
```

```
cv2.rectangle(image, (x, y),(x+w,y+h),color,2)
```

```
text = "{}:{:.4f}{}".format(LABELS[classIDs[i]],confidences[i])
```

```
cv2.putText(image, text, (x, y - 5 ),cv2.FONT_HERSHEY_SIMPLEX,0.5,
color,2).
```

24. แสดงผลภาพที่ได้ออกมาในหน้าต่างใหม่

```
cv2.imshow('Image',image)
```

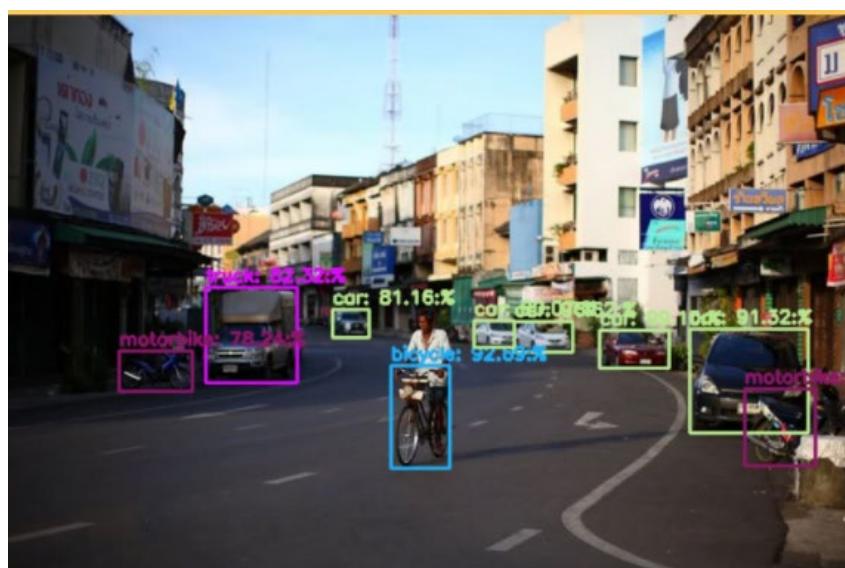
```
cv2.waitKey(0)
```

```
cv2.destroyAllWindows()
```

25. กด RUN เพื่อทดสอบ



26. ภาพที่ได้ออกมาจากการทำงาน



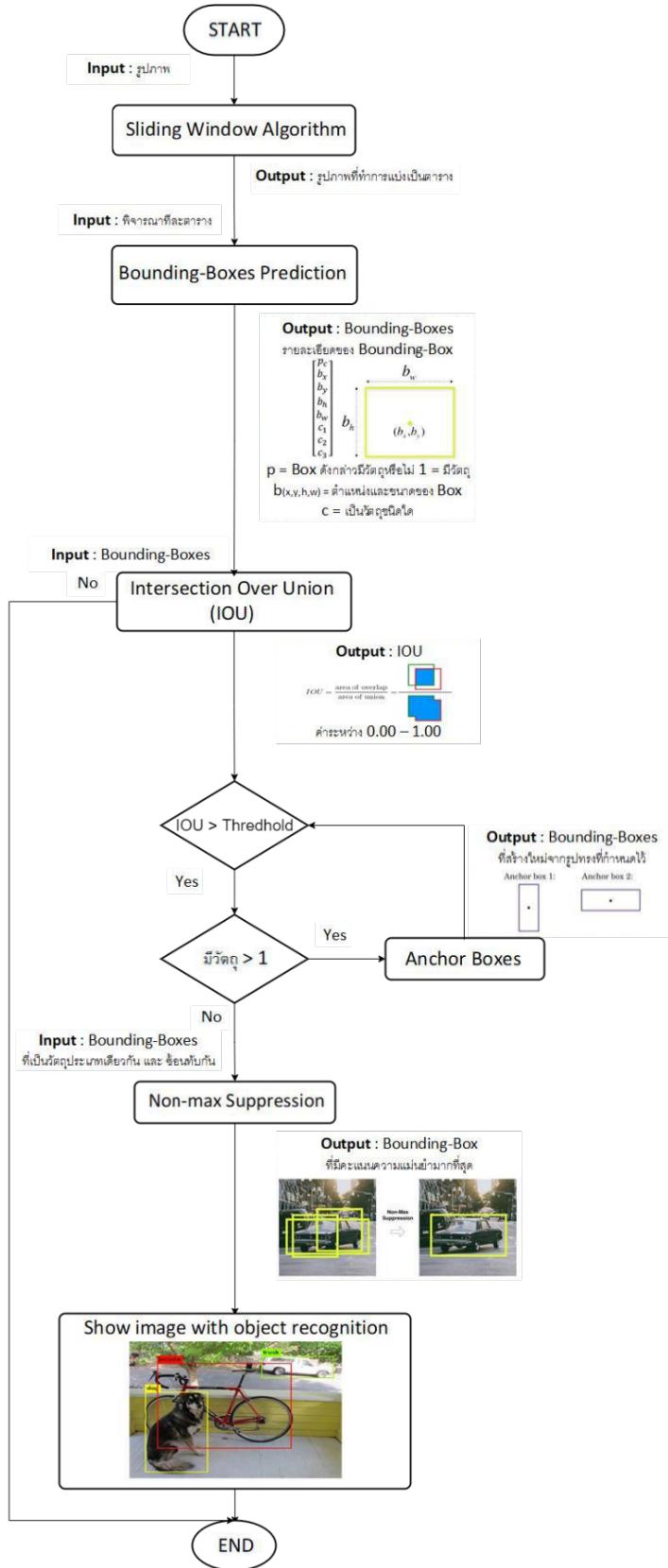


Figure 30 กระบวนการประมวลผล YOLO

3. Optical Character Recognition

เทคนิคการรู้จำตัวอักษรด้วยแสง (Optical Character Recognition: OCR) เป็นการแปลงภาพทางวิเล็กทรอนิกส์กล่าว คือ เป็นการรู้จำตัวอักษรภายในภาพ หรือข้อความภายในภาพ ทำการประมวลผลด้วยการสแกนด้วยเครื่องสแกน หรืออัลกอริทึมในการรู้จำ การประยุกต์ใช้เทคนิคการรู้จำตัวอักษรเมื่อผลการรู้จำตัวอักษรเป็นตัวอักษรหรือข้อความที่สามารถแก้ไขได้ (Text) ซึ่งการรู้จำตัวอักษรสามารถจำแนกได้เป็นกลุ่ม ๆ ตามลักษณะของตัวอักษรที่นำมาประมวลผล หรือแหล่งที่มาของตัวอักษร ซึ่งการรู้จำตัวอักษรสามารถแบ่งเป็น 2 ประเภท คือ การรู้จำแบบออนไลน์ และการรู้จำตัวอักษรแบบออฟไลน์



Figure 31 เทคนิคการรู้จำตัวอักษรด้วยแสง

ประเภทของเทคนิคการรู้จำตัวอักษรด้วยแสง

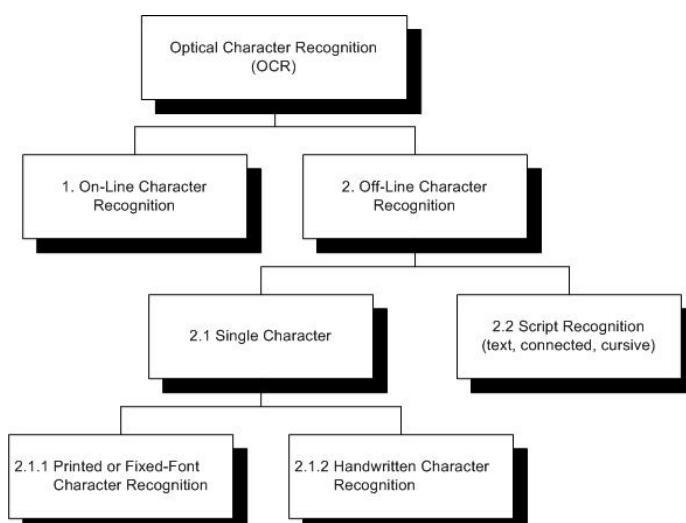


Figure 32 ประเภทของเทคนิคการรู้จำตัวอักษรด้วยแสง

การรู้จำตัวอักษรแบบออนไลน์ (On-line Character Recognition)

เป็นการประมวลผลตัวอักษรด้วยปากาอิเล็กทรอนิกส์ที่นิยมใช้ภายในมือถือ หรือเว็บไซต์ต่าง ๆ การประมวลผลจะใช้การลากจากจุดหนึ่งไปยังจุดหนึ่ง ตามลักษณะของการเขียนตัวอักษร ซึ่งเป็นการประมวลผลที่ซับซ้อนมากกว่าการรู้จำแบบอฟไลน์ เนื่องจากเป็นการวิเคราะห์ข้อมูลแบบทิศทางการเขียน การลากเส้น และความเร็วของการลากเส้น ทำให้การประมวลผลในส่วนนี้มีความยาก



Figure 33 การรู้จำตัวอักษรแบบออนไลน์ (On-line Character Recognition)

การเรียนรู้ตัวอักษรแบบอฟไลน์ (Off-line Character Recognition)

การรู้จำตัวอักษรแบบอฟไลน์ เป็นการประมวลผลตัวอักษรด้วยเครื่องสแกน แบบพิมพ์หรือเทคนิคการจดจำ ซึ่งเป็นการประมวลผลตัวอักษร ได้ทั้งแบบเดี่ยว แบบติดกันเป็นกลุ่มคำของตัวอักษร โดยสามารถจำแนกการประมวลผลตามลักษณะของตัวอักษรดังนี้

ตัวอักษรแบบโดด (Single Character) เป็นการประมวลผลตัวอักษรที่มีอินพุตเป็นภาพของตัวอักษรที่เป็นตัวเดี่ยว ๆ มีระยะห่างจากกัน ไม่เชื่อมติดกับตัวอักษรตัวอื่น ๆ ซึ่งมักประยุกต์ใช้กับการรู้จำตัวอักษรภาษาอังกฤษหรือตัวเลข สามารถแบ่งออกเป็น 2 กลุ่ม คือ

1. **การรู้จำตัวพิมพ์แบบฟอนต์เฉพาะ (Printed Fixed-Font Character Recognition)** เป็นการรู้จำพื้นฐานที่นิยมใช้โดยง่ายที่สุด แต่มักจะมีข้อเสียที่เกิดจากการรู้จำตัวอักษรที่มีสัญญาณรบกวนสูงคุณภาพของตัวอักษรต่า



Figure 34 ตัวพิมพ์แบบฟอนต์เฉพาะ

2. การรู้จำลายมือเขียนแบบตัวโดด (Isolated Handprint Character Recognition) การประมวลผลในส่วนนี้เป็นการรู้จำตัวอักษรด้วยลายมือเขียน มีระยะห่างจากกัน ไม่ติดหรือทับซ้อนกับตัวอักษรตัวอื่น ๆ โดยเขียนทิลละตัวแยกจากกันชัดเจน เช่น เบอร์มือถือ ซึ่งเป็นตัวเลขจากลายมือเขียน เป็นต้น การประมวลผลที่มีความยากขึ้นอีกรอบจากตัวอักษรแบบโดด เนื่องจากลายมือของผู้เขียนตัวอักษรมีเมื่อนอกัน มีความหลากหลายที่แตกต่างกันทำให้มีความผิดพลาดในการประมวลผลเนื่องจากโครงสร้างของตัวอักษรในการประมวลผลไม่เป็นที่แน่นอน

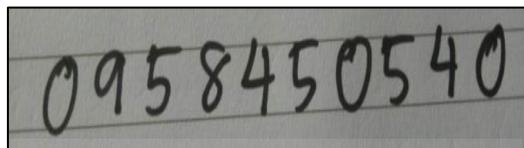


Figure 35 ลายมือเขียนแบบตัวโดด

ตัวอักษรลายมือแบบเขียนต่อเนื่อง (Script recognition) การประมวลผลในส่วนนี้เป็นการรู้จำที่ยกที่สุดเนื่องจากลักษณะการเขียนด้วยลายมือที่ไม่มีขอบเขตในการเขียน ทั้งยังเป็นการประมวลผลโดยตัวอักษรเขียนได้อย่างต่อเนื่อง เป็นตัวอักษรที่มีเส้นลากเชื่อมติดกัน ตัวติดกัน และมีโครงสร้างของตัวอักษรที่แตกต่างกัน ไม่กำหนดพ่อนต์ของตัวอักษร มีความแตกต่างของลายมือ

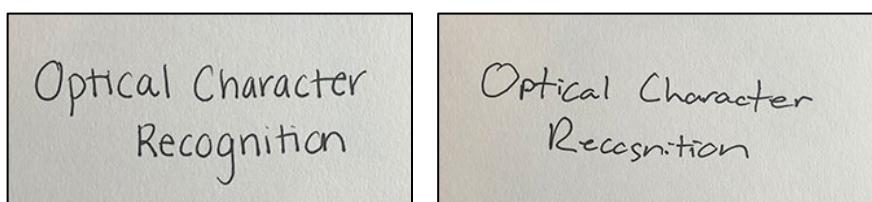


Figure 36 ตัวอักษรลายมือแบบเขียนต่อเนื่อง

ขั้นตอนเทคนิคการรู้จำตัวอักษรด้วยแสง (Optical Character Recognition)

การประมวลผลหรือขั้นตอนในการจัดจำตัวอักษร มีการประมวลผลหลักอยู่ 3 ขั้นตอน ดังต่อไปนี้

ขั้นตอนการประมวลผลขั้นต้น (Pre-Processing)

การประมวลผลขั้นต้นเป็นการประมวลผลขั้นแรกของเทคนิคโอีชาร์ (OCR) เป็นการเข้าถึงตัวอักษร และการปรับปรุงภาพ โดยทำการตรวจสอบภาพที่ก่อนนำไปรู้จำ สามารถแยกย่อยขั้นตอนการแปลงภาพได้หลายขั้นตอน ขึ้นอยู่กับการประมวลผลภาพซึ่งเรียกว่ากระบวนการประมวลผลขั้นต้นว่า Pre-Processing เช่น

การลดสัญญาณรบกวน (Noise Filtering) เป็นการกรองข้อมูลภาพที่มีสิ่งแทรกซ่อนที่ไม่ต้องการ โดยมีจุดประสงค์เพื่อลดความผิดพลาด สิ่งรบกวน ภายในภาพ เช่น รอยชุดปีด หรือการเติมเต็มความคอมชัดของภาพเพื่อให้สามารถเพิ่มประสิทธิผลของการรู้จำมากยิ่งขึ้น

Noise คือ จุดรบกวนในภาพที่เกิดจาก การปรับเพิ่มค่าความไวแสง เมื่อเราเพิ่มค่าความไวแสง ของกล้องดิจิตอลก็เป็นการสั่งให้เพิ่มกระแสไฟฟ้าบนตัวเซนเซอร์ให้สูงขึ้น ก็เพื่อเพิ่มความไวต่อแสงให้มากขึ้น สิ่งที่ตามมาคือสัญญาณรบกวนที่เพิ่มสูงขึ้นตามไปด้วย ซึ่งกรณีที่ค่าความไวแสงต่ำ มักไม่ค่อยจะปรากฏให้เห็น ซึ่ง Noise จะแสดงให้เห็นเป็นจุดรบกวนเม็ดเล็ก ๆ สีต่าง ๆ ในเงามืด หรือรายละเอียดต่าง ๆ ของภาพ

โดยยิ่งเราร่างภาพซึ่งใช้ค่าเพิ่มค่าความไวแสง สูง ๆ เท่าใด ยิ่งจะประภูมิ Noise มาเข้าแท่นนั้น แน่นอนส่งผลโดยตรงกับคุณภาพของภาพถ่ายเราได้

ปัญหา Noise หรือสัญญาณรบกวนนั้นถือเป็นเรื่องปกติที่มา กับกล้องดิจิตอลอยู่แล้ว เพราะมันเกิดจากสัญญาณรบกวนจากการแสไฟจากกล้องอยู่แล้ว โดยเฉพาะอย่างยิ่งการใช้ค่าความไวแสงสูง ๆ ในยุคปัจจุบันนี้ กล้องจะมีวงจรลดสัญญาณรบกวนอยู่แล้ว และยังมีฟังก์ชันลดสัญญาณรบกวนให้ผู้ใช้กล้องสามารถปรับเพิ่มเติมได้อีก ทำให้กล้องรุ่นใหม่ ๆ สามารถถ่ายภาพได้โดยใช้ค่า ISO ที่สูง ๆ ได้สบาย ๆ โดยไม่มีประภูมิ Noise ออกมาก

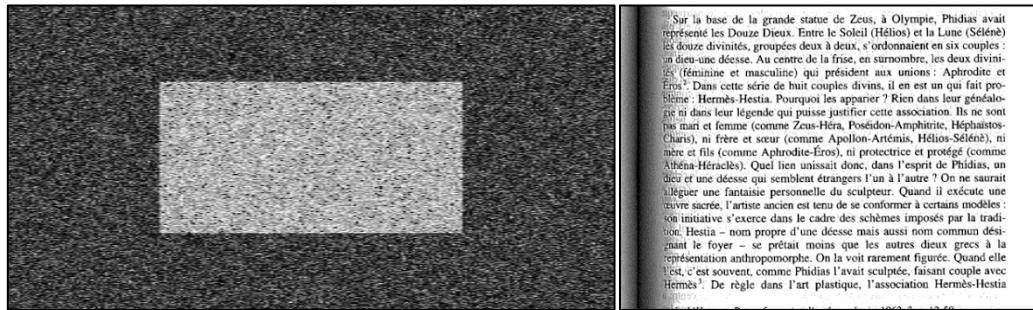


Figure 37 ภาพที่เกิดปัญหาสัญญาณรบกวน

พึงขั้น cv2.threshold() ใช้คัดกรองภาพโดยพิจารณาความสว่าง เพื่อที่จะคัดเอาหรือปรับค่าบางส่วน

พึงขั้น Threshold มี 5 แบบ

1. Threshold Binary
2. Threshold Binary, Inverted
3. Truncate
4. Threshold to Zero
5. Threshold to Zero, Inverted

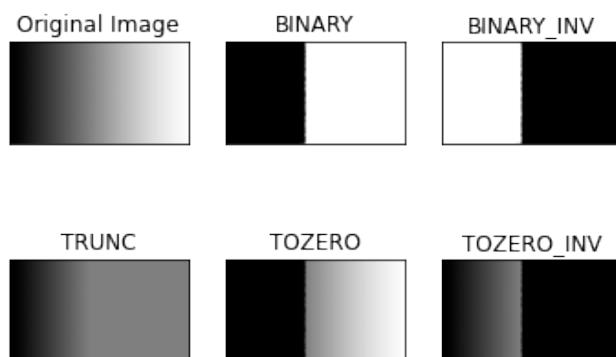
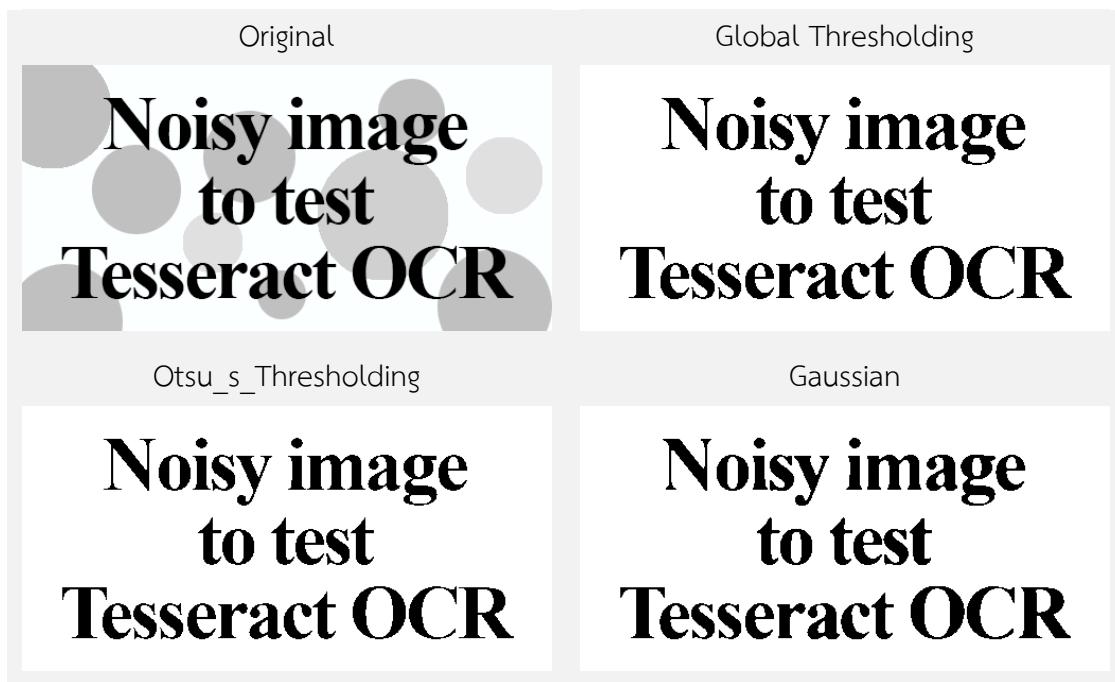
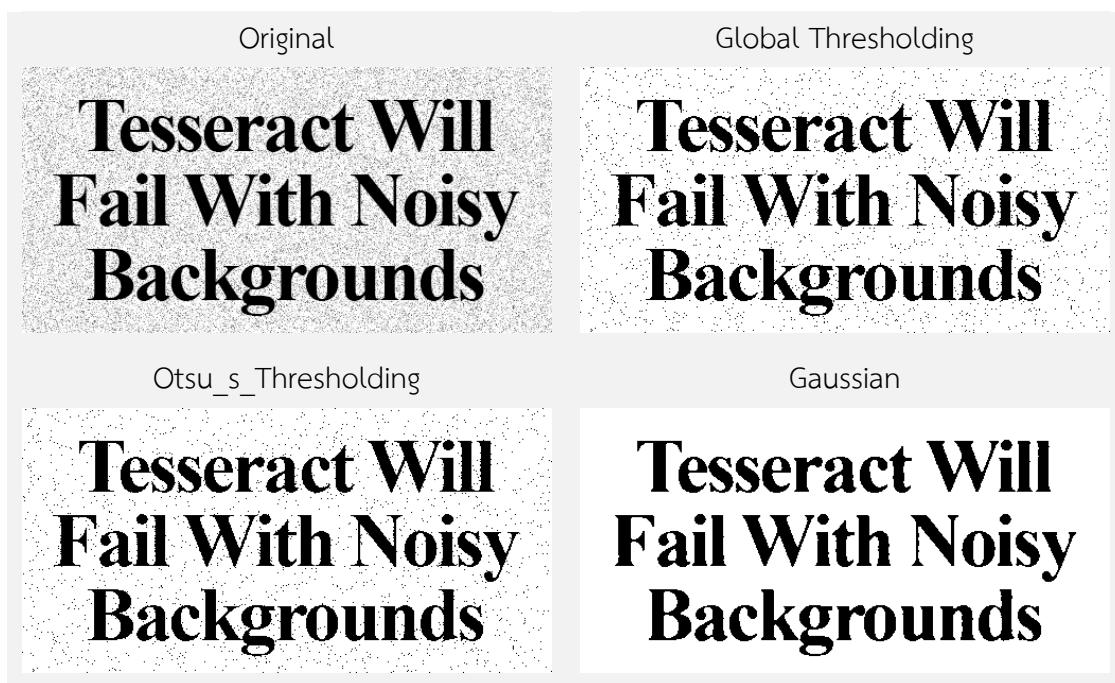


Figure 38 พึงขั้น Threshold 5 แบบ

ตัวอย่างการลดสัญญาณรบกวน

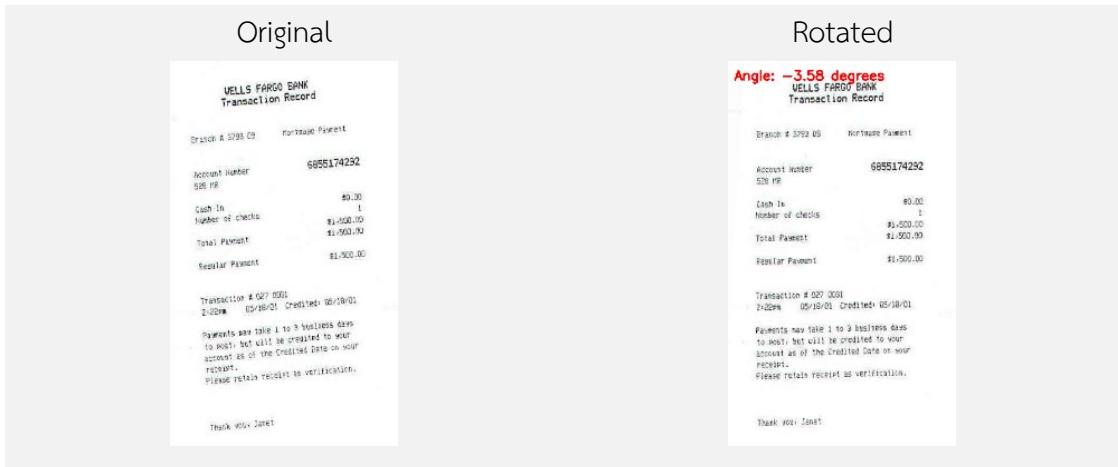


ตัวอย่างการลดสัญญาณรบกวน

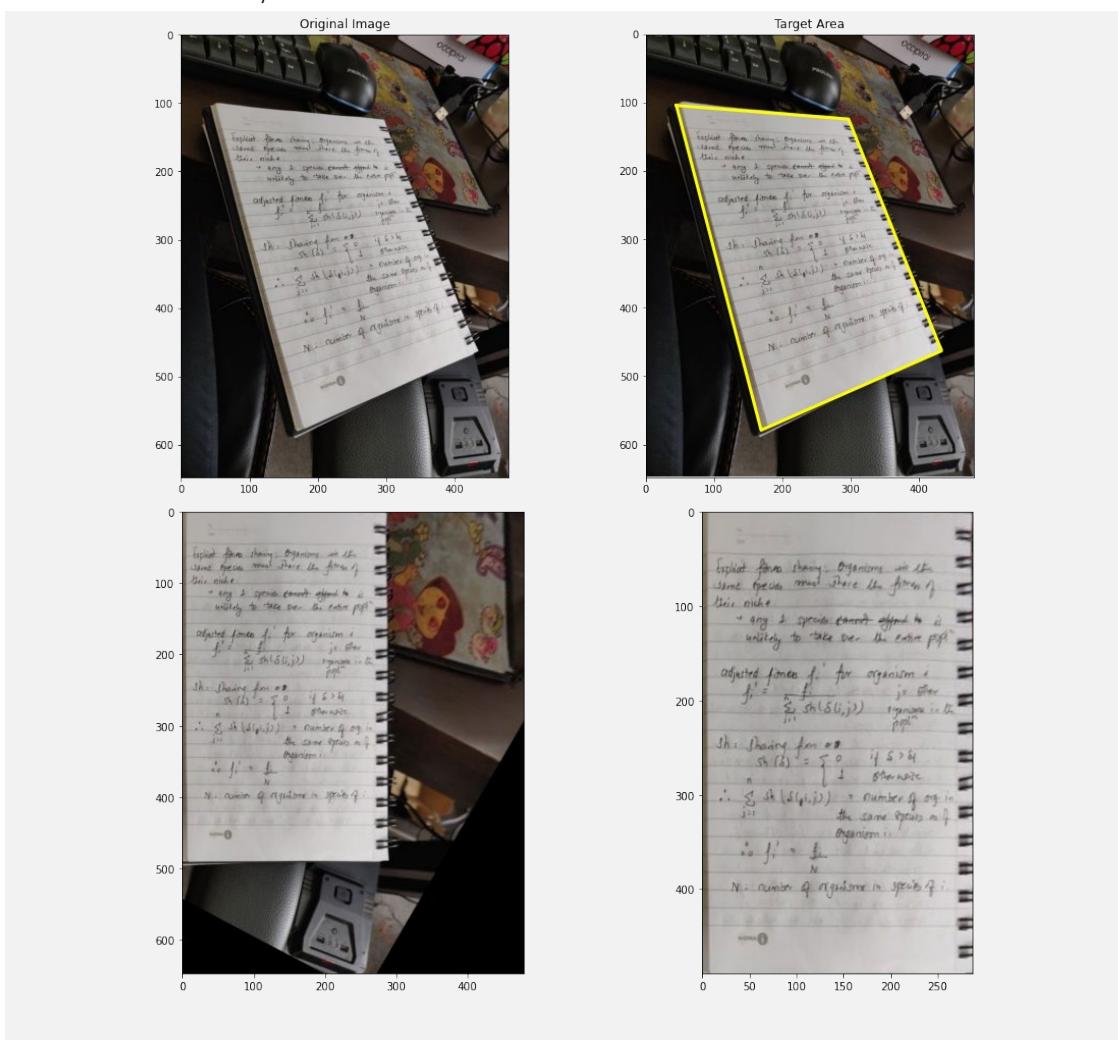


Skew correction ตรวจหาค่าความเอียงและการหมุนกลับของภาพเอกสาร

ตัวอย่างการการหมุนกลับของภาพ



ตัวอย่างการการหมุนกลับของภาพ



การรู้จำ (Recognition)

การรู้จำเป็นส่วนที่สำคัญที่สุด เป็นการระบุหรือตัดสินใจว่ารูปตัวอักษรผ่านกระบวนการนี้ใช้ตัวอักษรนั้น ๆ หรือไม่ ซึ่งวิธีการรู้จำตัวอักษรมีหลากหลายที่สามารถนำมาใช้งานได้จริง ซึ่งมีความแม่นยำ ความเร็ว และการทำงานที่แตกต่างกันขึ้นอยู่กับผู้พัฒนา สามารถจำแนกเทคนิคการรู้จำพื้นฐานได้ 4 กลุ่ม ดังนี้

วิธีทางสถิติ (Statistical Approach) เป็นวิธีการที่ประยุกต์ใช้หลักการทางสถิติ โดยการคาดการความน่าจะเป็นของตัวเลขรวมถึงการใช้ฟังก์ชันการแจกแจงความน่าจะเป็นในการระบุตัวอักษร โดยใช้ภาพอินพุตที่ผ่านกระบวนการปรับปรุงภาพ สกัดลักษณะเด่นของตัวอักษร และประมาณผลตัวอักษรโดยใช้ความน่าจะเป็นว่าเป็นตัวอักษรตัวใด ๆ แล้วนำผลลัพธ์ในการรู้จำมาหาค่าความน่าจะเป็นของตัวอักษรที่ใกล้เคียงมากที่สุดแล้วแสดงผลลัพธ์เป็นตัวอักษrnนั้น ๆ

วิธีการวิเคราะห์ทางโครงสร้าง (Structural Analysis) คือ การวิเคราะห์ โครงสร้างภาพตัวอักษร โดยใช้โครงสร้างพื้นฐานของภาพตัวอักษร สกัดลักษณะเด่นของตัวอักษร โครงสร้างของตัวอักษร เช่น เส้นตรงบน ล่าง วงกลม วงรี เส้นตัด จุดเชื่อม เป็นต้น ในขั้นตอนการรู้จำจะใช้การวิเคราะห์ลักษณะของตัวอักษร เช่น พ่อร์มอล

แกรมมาแมชชีน (formal grammar machine) โครงสร้างกราฟ หรือโครงสร้างต้นไม้เป็นต้น เพื่อเป็นอัลกอริทึมในการตัดสิน เทคนิคการวิเคราะห์มักจะใช้กับความหลากหลายของตัวอักษร เทคนิคนี้ขึ้นอยู่กับการสร้างรูปแบบโครงสร้างและการวิเคราะห์โครงสร้าง

โครงข่ายประสาทเทียม (Neural Network) เป็นเทคนิคใหม่ที่ได้รับการพัฒนาเป็นอย่างมาก เนื่องจากประสิทธิผลความแม่นยำสูง ถูกนำไปใช้ในงานหลาย ๆ ด้าน โครงข่ายประสาทเทียมเป็นเทคนิคที่เลียนแบบการทำงานของสมองมนุษย์ มีโครงข่ายความจำอย่างจำนวนมากในการแบ่งโครงสร้างเป็นฐานข้อมูลความรู้ เป็นการแบ่งเลยอร์หรือรูปแบบในการประมวลผลเป็นชั้น ๆ เพื่อกันหาเลยอร์ที่ถูกต้องของข้อมูลนั้น ๆ

วิธีการรู้จำด้วยการเปรียบเทียบเทียบคู่รูปแบบ (Template Matching) เป็นวิธีการรู้จำตัวอักษรแบบแรกเริ่มที่นำมาประยุกต์ใช้กับการรู้จำด้วยอักษร โดยมีหลักการพื้นฐาน คือ การสร้างแม่แบบหรือที่เรียกว่า template เพื่อนำมาอ่านและเปรียบเทียบความคล้ายคลึงกันของตัวอักษร โดยกำหนดตำแหน่งของพิกเซลตัวอักษรแล้วเปรียบเทียบหาความแตกต่างระหว่างตัวอักษรแต่ละตัวจากนั้นก็จะรู้ว่าเป็นรหัสตัวอักษรอะไร โดยใช้ค่าต่างระดับในการตัดสินใจ วิธีนี้มีข้อเสียที่ความเปลี่ยนแปลงต่อข้อมูลที่แตกต่างกัน ขนาด ความเอียงของตัวอักษรซึ่งในการเปรียบเทียบแบบมี 2 ลักษณะ คือ การเปรียบเทียบโครงสร้างของภาพ และเปรียบเทียบหน้าหนักของภาพ

ขบวนการประมวลผลขั้นสุดท้าย (Post-Processing)

การประมวลผลขั้นตอนสุดท้าย คือ การตรวจสอบและแก้ไขผลจากการรู้จำรูปตัวอักษรที่ผ่านกระบวนการรู้จำเมื่อผลเป็นรหัสตัวอักษร ซึ่งอาจจะเป็นตัวอักษรที่ถูกต้องหรือผิดพลาดก็เป็นได้ ดังนั้นขั้นตอนสุดท้ายเป็นการตรวจสอบความถูกต้องของการรู้จำ โดยการแสดงผลการรู้จำเป็นตัวอักษรที่สามารถแก้ไขได้ ส่วนนี้มักเป็นอัลกอริทึมในการตรวจสอบความถูกต้อง เช่น ไวยากรณ์ภาษา เป็นต้น

เครื่องมือ OCR Open source

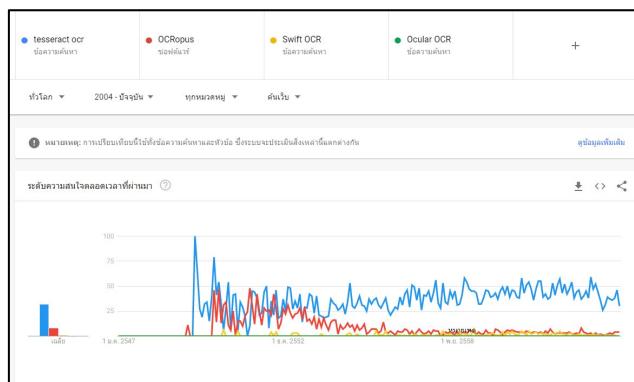


Figure 39 เครื่องมือ OCR Open source

Tesseract เป็นเอ็นจีนการรู้จำอักษรด้วยแสงสำหรับระบบปฏิบัติการต่าง ๆ เป็นซอฟต์แวร์เสรี ปล่อยให้สัญญาอนุญาต Apache เดิมที่พัฒนาโดย Hewlett-Packard เป็นซอฟต์แวร์ที่เป็นกรรมสิทธิ์ในทศวรรษ 1980 มันถูกปล่อยออกมานเป็นโอเพนซอร์สในปี 2005 และการพัฒนาได้รับการสนับสนุนจาก Google ตั้งแต่ปี 2006 Tesseract ได้รับการพิจารณาให้เป็นหนึ่งในเอ็นจีน OCR โอเพนซอร์สที่แม่นยำที่สุดที่มีอยู่

OCRopus ได้รับการออกแบบมาโดยเฉพาะเพื่อใช้ในโครงการแปลงหนังสือเป็นดิจิทัลจำนวนมาก เช่น Google Books, Internet Archive หรือห้องสมุด รองรับภาษาและแบบอักษรจำนวนมาก อย่างไรก็ตาม สามารถใช้กับแอปพลิเคชันเดสก์ท็อปและสำนักงาน และแอปพลิเคชันสำหรับผู้พิการทางสายตา

Tesseract OCR

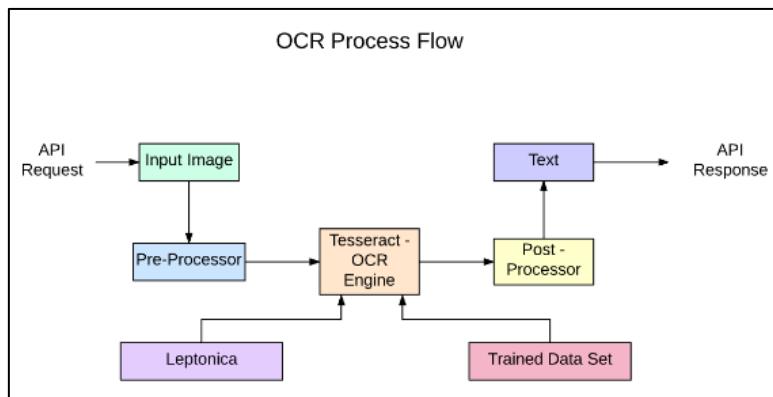


Figure 40 กระบวนการทำงานของ Tesseract OCR

Tesseract เป็นเอ็นจีนการรู้จำข้อความโดยเพ่นซอร์ส (OCR) ซึ่งอยู่ภายใต้ลิขสิทธิ์ Apache 2.0 สามารถใช้โดยตรงหรือใช้ API เพื่อแยกข้อความที่พิมพ์ออกจากรูปภาพ รองรับหลายภาษา Tesseract ไม่มี GUI ในตัว Tesseract เข้ากันได้กับภาษาการเขียนโปรแกรมและเฟรมเวิร์กต่าง ๆ สามารถใช้กับการวิเคราะห์เค้าโครงที่มีอยู่เพื่อจัดจำข้อความภายใต้เอกสารขนาดใหญ่ หรือใช้ร่วมกับตัวตรวจจับข้อความภายนอกเพื่อจดจำข้อความจากรูปภาพ

โครงสร้างการประมวลผลปกติ (เวอร์ชันก่อนหน้า)

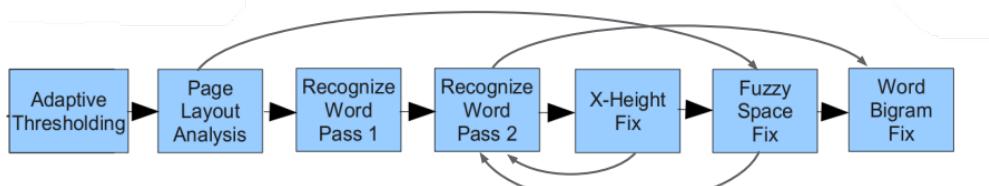


Figure 41 โครงสร้างการประมวลผลปกติ

โครงสร้างการประมวลผลเสริมด้วยเทคนิค LSTM (เวอร์ชัน 4)

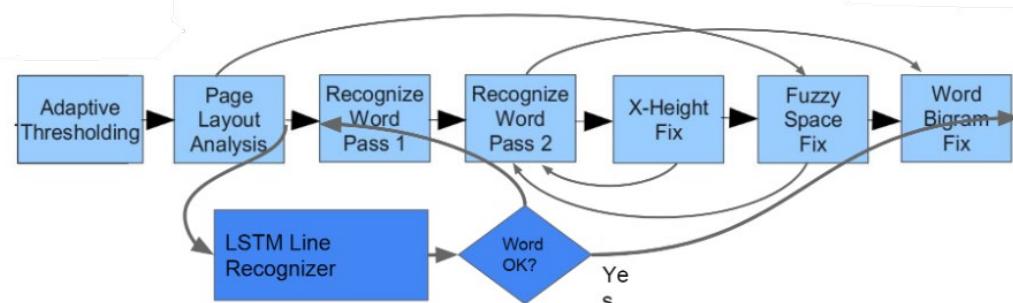


Figure 42 โครงสร้างการประมวลผลเสริมด้วยเทคนิค LSTM

ขั้นตอนการรักษาของ Tesseract

Tesseract Word Recognizer

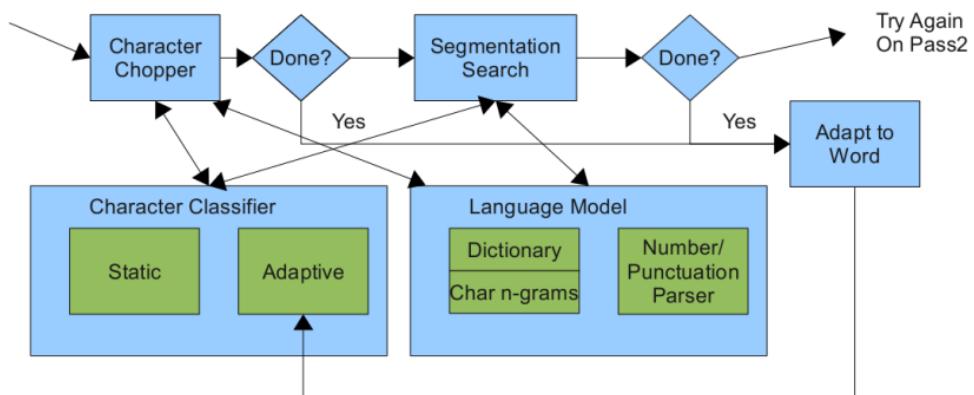
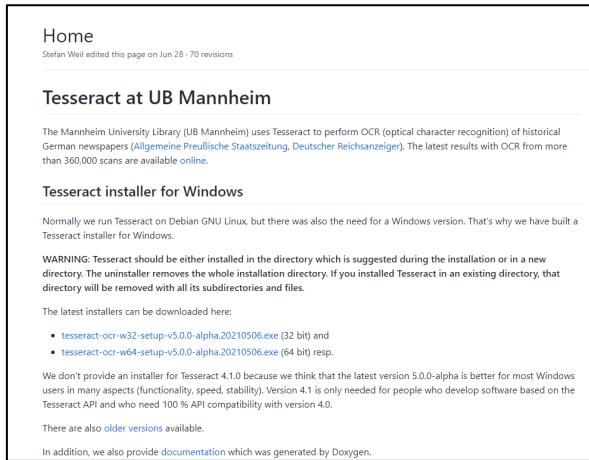


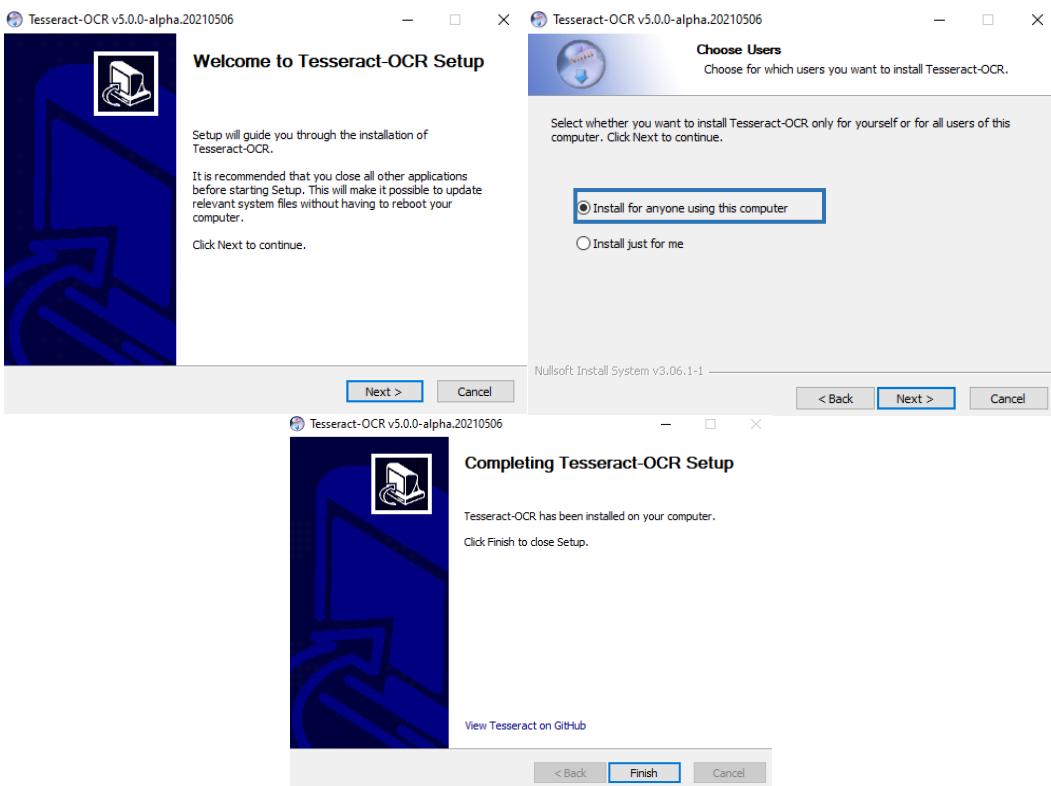
Figure 43 ขั้นตอนการรุ้งจำของ Tesseract

การติดตั้ง Tesseract

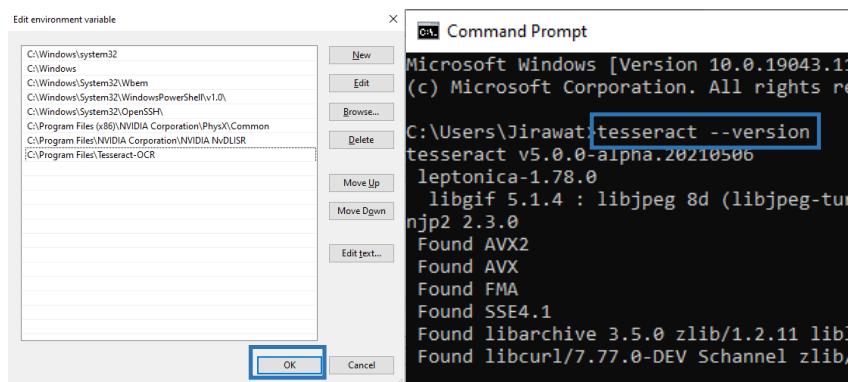
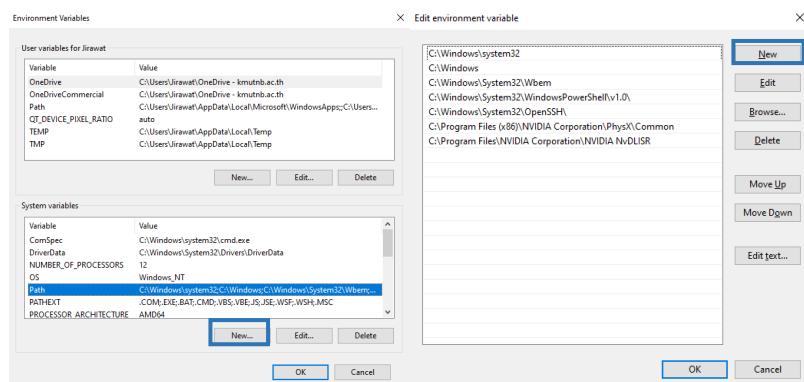
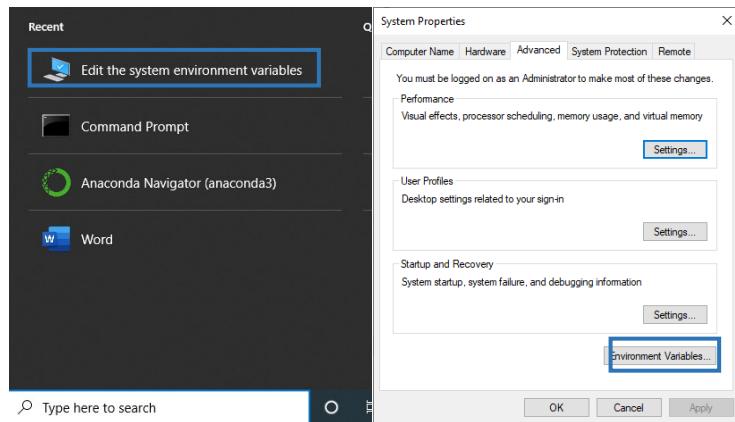
1.ดาวน์โหลดไฟล์ไลบรารี <https://github.com/UB-Mannheim/tesseract/wiki>



2.ทำการติดตั้งไลบรารี



3. ตั้งค่า Path ไฟล์



การสกัดข้อความจากรูปภาพ

```
import cv2 # ได้จากการติดตั้ง Library OpenCV โดย pip install opencv-python
import pytesseract # ติดตั้ง pytesseract เพื่อให้ Python สามารถเรียกใช้ Tesseract OCR ได้
โดย pip install pytesseract

img = cv2.imread("./File/Welcome.jpg") # ใช้ OpenCV ในการอ่านรูปภาพ welcome.jpg
cv2.imshow("Image", img) # ใช้ OpenCV ในการแสดงรูปภาพ welcome.jpg
cv2.waitKey(0) # เป็นคำสั่งหน่วงเวลาตามที่กำหนดไว้ต่อหน่วยมิลลิวินาที ถ้าใช้ 0 หน่วงค้างไว้
imgchar = pytesseract.image_to_string(img, lang= 'tha+eng') # ใช้ Tesseract OCR ในการแปลงรูปภาพเป็น
ข้อความ
print(imgchar) # แสดงข้อความที่อ่านได้

imgbox = pytesseract.image_to_boxes(img, lang= 'tha+eng') # จะได้ข้อมูลเป็นกล่องขอบเขตสำหรับอักขระแต่
ละตัวที่ตราชพบ
print(imgbox) # แสดงข้อมูลอักขระที่ตราชพบ
imgboxsplit = imgbox.splitlines()
print(imgboxsplit)

imgH, imgW, _ = img.shape # imgH จะเก็บค่าความสูงของภาพ และ imgW จะเก็บค่าความกว้างของภาพ
for boxes in imgboxsplit: # splitlines คือนำค่าที่ขึ้นบรรทัดใหม่มาเก็บไว้ในรูปแบบ List และวนอ่านค่าซ้ำมาเก็บไว้
ในตัวแปร boxes
    boxes = boxes.split(' ') # แยก字符串โดยใช้ ' '
    print(boxes) # แสดง List ของ boxes
    x, y, w, h = int(boxes[1]), int(boxes[2]), int(boxes[3]), int(boxes[4]) # นำค่าใน Index ของ boxes มาเก็บไว้
    ในตัวแปร x, y, w และ h
    cv2.rectangle(img, (x, imgH-y), (w, imgH-h), (0, 0, 255), 3) # ใช้ OpenCV ในการสร้างกรอบสี่เหลี่ยม
    cv2.putText(img, boxes[0], (x, imgH-y+30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2) # เป็นการ
    วาดอักขระลงในรูป โดย cv2.putText(รูปภาพ, ข้อความ, ตำแหน่ง, ชนิดของ Font, ขนาดของ Font, สี, ความหนาของ
    เส้นที่ใช้วาดอักขระ)
    cv2.putText(img, imgchar, (int(imgH/50), int(imgW/20)), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 0), 2) # บ
    ปันการวาดอักขระลงในรูป โดย cv2.putText(รูปภาพ, ข้อความ, ตำแหน่ง, ชนิดของ Font, ขนาดของ Font, สี, ความ
    หนาของเส้นที่ใช้วาดอักขระ)
    cv2.imshow("Result", img) # แสดงรูปภาพ
    cv2.waitKey(0) # เป็นคำสั่งหน่วงเวลาตามที่กำหนดไว้ต่อหน่วยมิลลิวินาที ถ้าใช้ 0 หน่วงค้างไว้
    cv2.destroyAllWindows() # เป็นฟังก์ชันที่ใช้ในการปิดหน้าจอการทำงาน
```

การสกัดข้อความแบบตามเวลาจริง

```
import cv2 # ได้จากการติดตั้ง Library OpenCV โดย pip install opencv-python
import pytesseract # ติดตั้ง pytesseract เพื่อให้ Python สามารถ
                   เรียกใช้ Tesseract OCR ได้ โดย pip install pytesseract

cap = cv2.VideoCapture(0) # cv2.VideoCapture(ลำดับของกล้อง)
if not cap.isOpened(): # ตรวจสอบว่ากล้องพร้อมใช้งานหรือไม่
    print("Cannot open webcam") # ถ้าไม่จะแสดงข้อความ Cannot open video

while cap.isOpened(): # ถ้ากล้องยังใช้งานอยู่ให้นำลุปต่อไป
    ret, frame = cap.read() # จับภาพแบบ frame ต่อ frame
    imgH, imgW, _ = frame.shape # imgH จะเก็บค่าความสูงของ frame และ imgW จะเก็บค่าความกว้าง
                                 ของ frame
    imgchar = pytesseract.image_to_string(frame) # ใช้ Tesseract OCR ในการแปลงภาพที่จับได้ต่อ frame เป็น
                                                 ข้อความ
    imgboxes = pytesseract.image_to_boxes(frame) # จะได้ข้อมูลเป็นกล่องขอบเขตสำหรับอักขระแต่ละตัวที่
                                                 ตรวจพบ

    for boxes in imgboxes.splitlines(): # splitlines คือคำที่ขึ้นบรรทัดใหม่มาเก็บไว้ในรูปแบบ List และวนอ่าน
                                         คำข้ามมาเก็บไว้ในตัวแปร boxes
        boxes = boxes.split(' ') # แยกสตริงโดยใช้ ' '
        x, y, w, h = int(boxes[1]), int(boxes[2]), int(boxes[3]), int(boxes[4]) # นำค่าใน Index ของ boxes มา
                           เก็บไว้ในตัวแปร x, y, w และ h
        cv2.rectangle(frame, (x, imgH-y), (w, imgH-h), (0, 0, 255), 3) # ใช้ OpenCV ในการสร้างกรอบสีเหลือง
        cv2.putText(frame, boxes[0], (x, imgH-y+30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2) # เป็น
                                         การวาดอักขระลงในรูป โดย cv2.putText(รูปภาพ, ข้อความ, ตำแหน่ง, ชนิดของ Font, ขนาดของ Font, สี, ความหนา
                                         ของเส้นที่ใช้วาดอักขระ)

        cv2.putText(frame, imgchar, (int(imgH/50), int(imgW/20)), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255,
                                         255), 2) # เป็นการวาดอักขระลงในรูป โดย cv2.putText(รูปภาพ, ข้อความ, ตำแหน่ง, ชนิดของ Font, ขนาด
                                         ของ Font, สี, ความหนาของเส้นที่ใช้วาดอักขระ)

        cv2.imshow("Result", frame) # แสดงผลของ frame
        if cv2.waitKey(1000) & 0xFF == ord('q'): # หน่วงเวลา 1 วินาทีและตรวจสอบว่ามีการกดปุ่ม q ที่คีย์บอร์ด
                                         หรือไม่
            break # ถ้าใช้ก็จบการวนลูปของ while
cap.release() # ทิ้ง Task ที่กำลังทำงานอยู่ออกไปให้หมด
cv2.destroyAllWindows() # เป็นฟังก์ชันที่ใช้ในการปิดหน้าจอการทำงาน
```

4. Face Recognition

ระบบการรู้จำใบหน้า หรือ ระบบการจดจำใบหน้า (อังกฤษ: Face Recognition system) คือ ระบบการตรวจหาใบหน้าของมนุษย์และปรับภาพใบหน้าโดยอัตโนมัติ กรอบจะปรากฏขึ้นบนใบหน้าที่ถูกตรวจจับ และไฟกัส สี และค่าการวัดแสงจะถูกปรับโดยอัตโนมัติ นอกจากนั้นเมื่อบันทึกด้วยคุณภาพแบบ HD เทคโนโลยีการบีบอัดจะจัดสรรความจุของข้อมูลให้ลดลง แต่ได้ข้อมูลที่เป็นประโยชน์มากขึ้นเพื่อปรับคุณภาพของภาพ ข้อมูลที่ได้จะถูกนำมาเปรียบเทียบกับข้อมูลตัวอย่างที่เก็บบันทึกไว้ จะจะหันไปหน้า หรือเพียงบางส่วน ขึ้นกับชนิดของวิธีแยกเอกสารลักษณ์ใบหน้า ระบบการรู้จำใบหน้าเป็นส่วนหนึ่งของ เทคโนโลยีปัญญาประดิษฐ์ในส่วนเนื้อหาของเรื่อง การรับรู้ของเครื่อง (Machine perception)

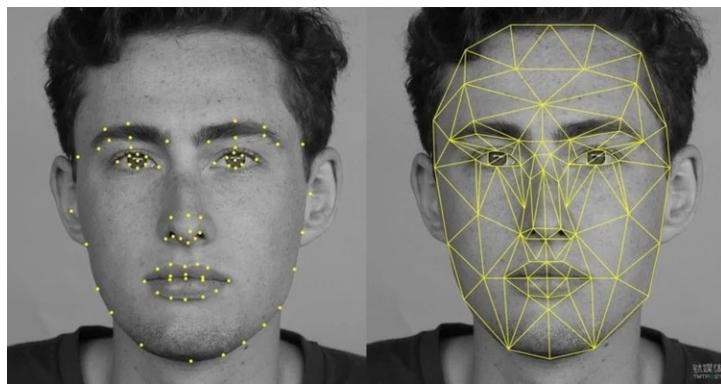


Figure 44 เทคโนโลยีการจดจำใบหน้า

ประวัติของระบบจดจำใบหน้า

จุดกำเนิดของเทคโนโลยี Face Recognition ปี 1960s สามารถจำแนกภาพ ใบหน้าแบบ Manua ผ่านการใช้งาน RAND Tablet สามารถใช้ในการบันทึกและจำแนกอัตลักษณ์ต่าง ๆ บนใบหน้า อาทิ ตา หู จมูก ปาก และริมฝีปาก ปี 1970s พัฒนาความแม่นยำให้มีความแม่นยำมากขึ้น โดยระบุอัตลักษณ์เจาะจง 21 จุด อาทิ สีผิว ความหนาของริมฝีปาก แต่ระบบยังต้องใช้การคำนวณวัดและระบุตำแหน่งจากผู้ใช้ เช่นเดิม 1980s / 90s ใช้การวิเคราะห์ องค์ประกอบหลักกับใช้เทคนิคพีชคณิตเชิงเส้นทั่วไป เพื่อแก้ไขจุดบกพร่องเดิม ที่ต้องคำนวณวัดค่าและระบุตำแหน่งแบบ Manua 1990s / 2000s โปรแกรม FERET สร้างฐานข้อมูลภาพใบหน้า ในชุดทดสอบมีภาพนิ่ง 2,413 ภาพ คิดเป็น 856 คน การทำภาพทดสอบสำหรับการจดจำใบหน้าได้สร้างแรงบันดาลใจให้เกิดนวัตกรรมและส่งผลให้เทคโนโลยีการจดจำใบหน้ามีประสิทธิภาพมากขึ้น 2000s มีคนก่อวัลเรื่องสิทธิส่วนบุคคล และความปลอดภัย รัฐบาลสหรัฐฯ จึงบังคับใช้กฎหมายสำหรับข้อมูลที่ใช้ในการปรับใช้เทคโนโลยีการจดจำใบหน้า 2006 อัลกอริทึมการจดจำใบหน้าล่าสุดที่มีอยู่ มีการใช้ภาพใบหน้าความแม่นยำมากกว่าอัลกอริทึมการจดจำใบหน้าที่ผ่านมา 2010-ปัจจุบัน เริ่มใช้ฟังก์ชันการจดจำใบหน้าที่ช่วยระบุผู้คนที่ใบหน้าอาจมีรูปภาพที่ผู้ใช้ร่วมกับการจดจำใบหน้าให้เข้ากับชีวิตประจำวันของเรา

การตรวจหาใบหน้า

การตรวจหาตำแหน่งของใบหน้า เป็นจุดเริ่มต้นของระบบการรู้จำใบหน้า การตรวจหาใบหน้า เป็นการตรวจหาใบหน้าทั่ว ๆ ไป ยังไม่สามารถแยกแยะได้ว่า หน้าไหนเป็นหน้าของใคร วิธีที่นิยมคือ วิธีของวีโอล่า โจนส์ พัฒนาขึ้นเมื่อปี พ.ศ. 2544 โดยพอล วีโอล่า และไมเคิล โจนส์ อัลกอริธึมของวีโอล่า โจนส์ นี้ถูกบรรจุไว้ใน OpenCV ในชื่อ cvHaarDetectObjects() การตรวจหาใช้วิธีตรวจหาตำแหน่งของ จมูก และ ตาทั้งสองข้าง ในรูปของเมตริก แนวตั้งและแนวนอน

วิธีการทำงานของ Face Recognition

How Facial Recognition Systems work



Figure 45 Face Recognition system

Face Recognition system การดำเนินงานของระบบตรวจสอบใบหน้าจะแบ่งออกเป็น 2 ส่วน คือ ส่วนที่เรียกว่า Face Detection และ Face Recognition ซึ่งอาจทำให้คนสับสนได้ว่าทั้งสองแบบคือสิ่งเดียวกัน ทำหน้าที่เหมือนกัน แต่ที่จริงแล้วมันมีความต่างในจุดประสงค์ของการใช้งาน

การตรวจจับใบหน้า Face Detection ไม่ได้เป็นส่วนที่วิเคราะห์ผลข้อมูล เพราะมันเป็นกระบวนการค้นหาใบหน้าของบุคคลจากภาพหรือวีดีโอ ซึ่งจะสามารถระบุได้ว่าในเฟรมนั้น ๆ มีบุคคลปรากฏขึ้นมากี่คนแต่จะยังระบุไม่ได้ละเอียดว่าเป็นใคร

ส่วนการจดจำใบหน้า Face Recognition จะเป็นขั้นตอนหลังจากที่มีการส่งข้อมูลเข้าไปเปรียบเทียบ กับข้อมูลใน data base ที่มีอยู่ เพื่อให้สามารถระบุได้ว่าบุคคลผู้นั้นเป็นใคร

Modern Face Recognition with Deep Learning

สำหรับการรู้จำใบหน้าด้วยไลบรารี face_recognition จะมีกระบวนการในการทำงานดังนี้

Histogram of Oriented Gradients (HOG)

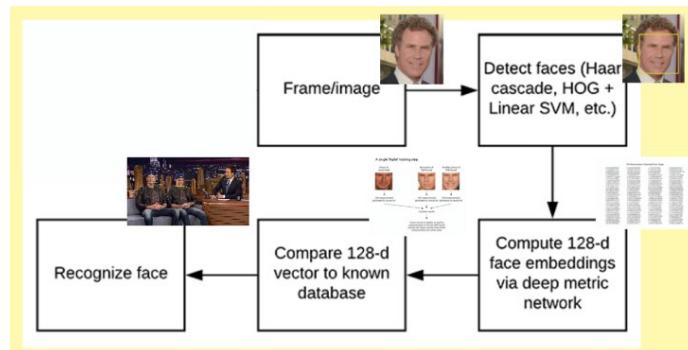


Figure 46 กระบวนการ Face Recognition

จากภาพที่ 46 เป็นกระบวนการของการจดจำใบหน้าโดยแบ่งออกเป็น 5 ขั้นตอน ได้แก่

1. การนำเข้ารูปภาพ
2. การค้นหาตำแหน่งใบหน้า
3. การแปลงค่า 128 มิติจากใบหน้าโดยโครงข่ายประสาทเทียม
4. การเปรียบเทียบค่าที่ได้จากการแปลงค่ากับฐานข้อมูล
5. ผลลัพธ์ที่ได้ การรู้จำใบหน้า

ขั้นที่ 1 การนำเข้ารูปภาพ โดยในการค้นหาใบหน้าในรูปภาพ เริ่มต้นด้วยการทำให้รูปภาพเป็นขาวดำ ดังภาพที่ 47 เพราะไม่ต้องการข้อมูลสีเพื่อค้นหาใบหน้า และจะได้ความเร็วในการประมวลผลรูปภาพ



Figure 47 การนำเข้ารูปภาพเพื่อประมวลผลสำหรับ Histogram of Oriented Gradients (HOG)

ขั้นที่ 2 การค้นหาใบหน้าเพื่อการตัดส่วนที่ไม่จำเป็นในการรู้จำใบหน้า เช่น พื้นหลัง วัตถุอื่นๆ ที่ไม่ใช่ใบหน้า โดย HOG จะทำการคำนวณพิกเซล โดยการคูนเมทริกของภาพ ผลลัพธ์ที่ได้จากการคำนวณจะเป็นจำนวนการไถ่ระดับสีที่ซึ่งในแต่ละทิศทางหลัก (จำนวนจุดขึ้น ชี้ขึ้นขวา ชี้ไปทางขวา ฯลฯ) จากนั้นจะแทนที่สีเหลี่ยมจตุรัสนั้นในภาพด้วยทิศทางลูกศรที่แข็งแกร่งที่สุด ผลลัพธ์ที่ได้คือการเปลี่ยนภาพตันฉบับให้กลายเป็นภาพจำลองที่เรียบง่าย ซึ่งรวมรวมโครงสร้างพื้นฐานของใบหน้า โดยผลลัพธ์ที่ได้แสดงทิศทางและขอบของรูปภาพ และนำผลที่ได้มาเปรียบเทียบกับโมเดลที่ฝึกจากใบหน้าอื่นจำนวนมากเพื่อค้นหาตำแหน่งของใบหน้า โดยการค้นหาส่วนที่คล้ายกัน ดังภาพที่ 48

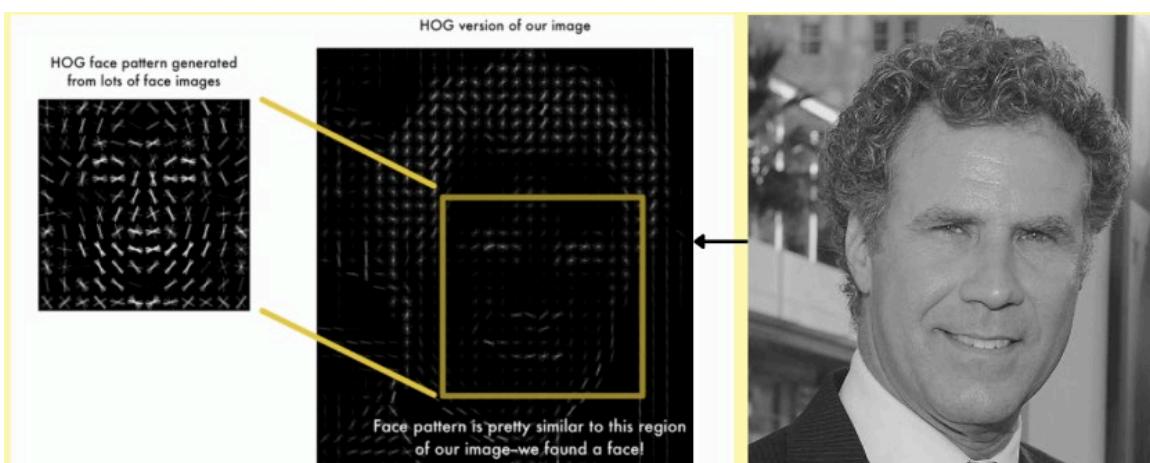


Figure 48 การค้นหาใบหน้า Histogram of Oriented Gradients (HOG)



Figure 49 ผลลัพธ์จากการค้นหาตำแหน่งใบหน้าด้วย Histogram of Oriented Gradients (HOG)

ขั้นที่ 3 การแปลงค่า 128 มิติจากใบหน้าโดยใช้ขั้นตอนนี้ เริ่มจากการแปลงรูปภาพเป็นค่าที่มีสกัดคุณลักษณะ (Feature Extractions) จากใบหน้า 128 มิติ โดยโมเดล Dlib จะทำการกำหนดจุดแลนด์มาร์คของใบหน้าที่ได้จากการค้นหาตำแหน่งใบหน้าที่ โดยมีอยู่การกำหนดจุดแลนด์มาร์ค 2 แบบ คือ กำหนดจุดใบหน้า 5 จุด และ 68 จุด ดังภาพที่ 50 ผลลัพธ์ที่ได้ คือ ตำแหน่งของตาและปากของใบหน้า เพื่อทำการหมุนปรับขนาด และตัดภาพ ให้ดวงตาและปากอยู่ตรงกลางของภาพมากที่สุด โดยพยายามรักษาไม่ให้ภาพบิดเบี้ยว น้อยที่สุด โดยใช้การแปลงภาพเพื่อจราณเท่านั้น เช่น การหมุนและสเกลที่รักษาเส้นคู่ขนาน (called affine transformations) หลังจากนั้นจะเข้าสู่กระบวนการแปลงใบหน้าที่ได้เป็นค่า 128 มิติดังภาพที่ 51 โดยโครงข่ายประสาทเทียมแบบ convolutional neural network : CNN

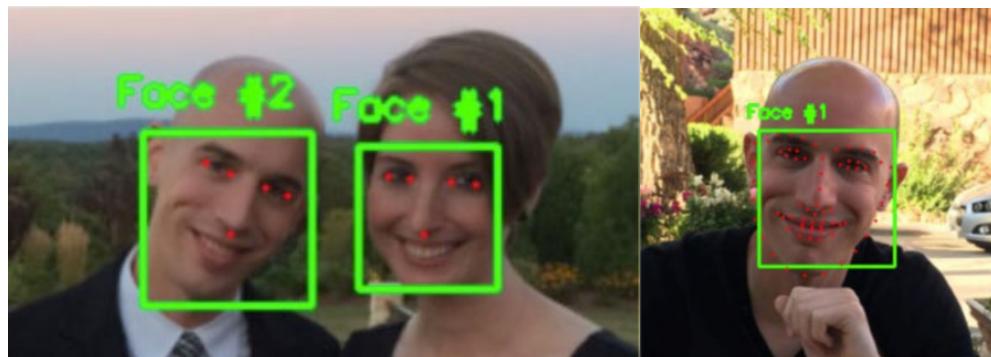


Figure 50 การกำหนดจุดสำคัญบนใบหน้า (face landmarks) การกำหนดจุดแบบ 5 จุด (ซ้าย) การกำหนดจุดแบบ 68 จุด (ขวา)

128 Measurements Generated from Image		
Input Image		
	→	
0.007493042033608	0.04523252433984	0.073208412617114
0.1250884674129	-0.06000917911519	0.08017605215807
0.030819439718723	-0.01981477253139	-0.0005216327845118
0.03605059068403	0.065554238855939	-0.1318951100111
-0.097486683401871	0.1226262897253	0.073130601544
0.036640171166509	0.039750303916929	-0.005957510539889
0.1811310447140	0.144111000000000	0.04327451234599
-0.040640540029538	-0.019015879929907	-0.020598002707070
-0.12567175924778	-0.10558545013666	0.07818105528817
-0.061418771743774	-0.074287034577171	-0.076589616525173
0.048741490771574	0.0081761881224811	0.12369473318058
-0.21050991947651	0.14748643765606	0.05641842260568
0.0610674074045	0.113027630000000	0.08972754760258
0.06198994072915	0.19372989484114	-0.022389197481632
0.10904195904723	0.084853030741215	0.02096049556136
-0.019414527341723	0.0064811296761036	0.21180312335491
0.15245945751667	-0.16582328081131	-0.05059439810049
-0.142547775558491	-0.007077775558491	-0.072376452386379
0.0893146905121613	-0.056797775558491	0.0344219947737379
0.0879411095905	0.11479432267934	-0.070000000000000
-0.021407851949334	0.14841195949971	-0.098621491730213
-0.01829890441656	0.04852542480866	0.013955107890069
-0.01014151386917	-0.051016297191381	-0.14132921397688
0.000188740000003	-0.000188740000003	0.0050111928275228
0.09813885733007	-0.09813885733007	-0.09813885733007
-0.024310374802351	-0.114437922983535	-0.0394010760232987
-0.057223934860223	0.01468386967351	0.043765489012003
0.023635015061468	-0.081752359867096	-0.012062264469002
-0.099809731383324	0.0370203558953	0.012774495407930
0.0240337680839002	-0.094388014247417	0.01638788878918
0.051597066223621	-0.10034311562777	-0.040977258235216

Figure 51 ค่าคุณลักษณะใบหน้าที่ได้ 128 มิติ

ขั้นที่ 4 การเปรียบเทียบค่าที่ได้จากการแปลงค่ากับฐานข้อมูล โดยจะนำค่าที่ได้ 128 มิติจากใบหน้ามาเปรียบเทียบโดยใช้กระบวนการ Triplet loss function โดยกระบวนการนี้มีจุดเด่นคือ ลดระยะเวลาในการคำนวณและลดปัญหาในการสร้างคลาสจากใบหน้ากรณีใบหน้าเพิ่ม จากแบบปกติที่จะใช้ Bottleneck layer รองสุดท้ายในการแยกใบหน้าใบการแยกใบหน้า โดยใช้ Machine Learning เช่น SVM และ KNN ในการทำงาน กระบวนการ Triplet จะประกอบด้วย 3 ส่วนคือ Anchor : ตัวอย่างตั้งต้นที่ใช้ในการเปรียบเทียบ Positive : ตัวอย่างของใบหน้าเดียวกับ Anchor และ Negative : ตัวอย่างของใบหน้าที่ต่างกับ Anchor โดยมีหลักการคือ ลด distance ระหว่าง anchor และ positive (เป็นบุคคลเดียวกัน) และเพิ่ม distance ระหว่าง anchor และ negative (เป็นคนละคนกัน) ดังภาพที่ 52 จะได้ผลลัพธ์ของการรู้จำใบหน้ากรณีที่ค่าใกล้กันมีโอกาสที่จะเป็นบุคคลเดียวกัน โดยการกำหนดค่า TOLERANCE หรือค่าที่เรากำหนดขึ้น

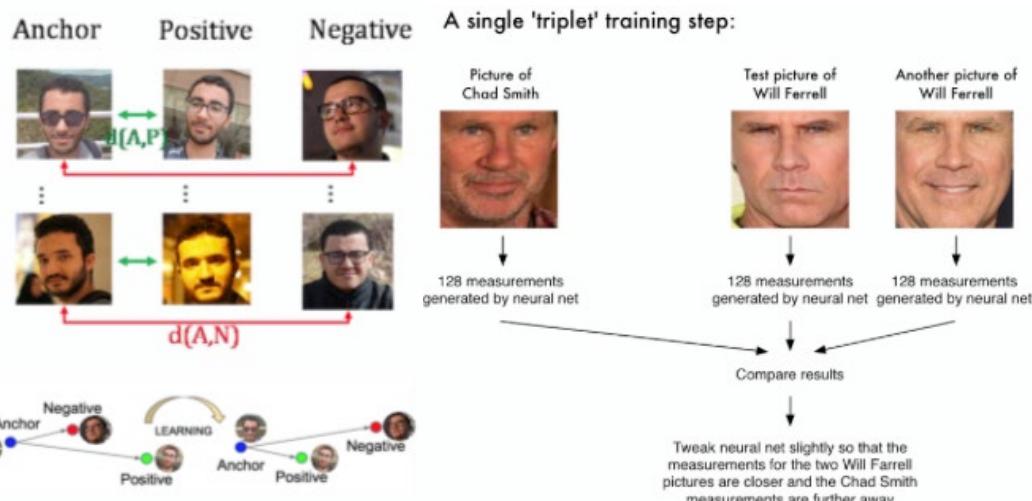


Figure 52 กระบวนการ Triplet loss เพื่อการรู้จำใบหน้า

ขั้นที่ 5 การนำผลลัพธ์มาแสดงผลเป็นระบบบรู๊ฟใบหน้า



Figure 53 ผลลัพธ์ที่ได้จากการรู้จำใบหน้า

Convolutional Neural Network (CNN)

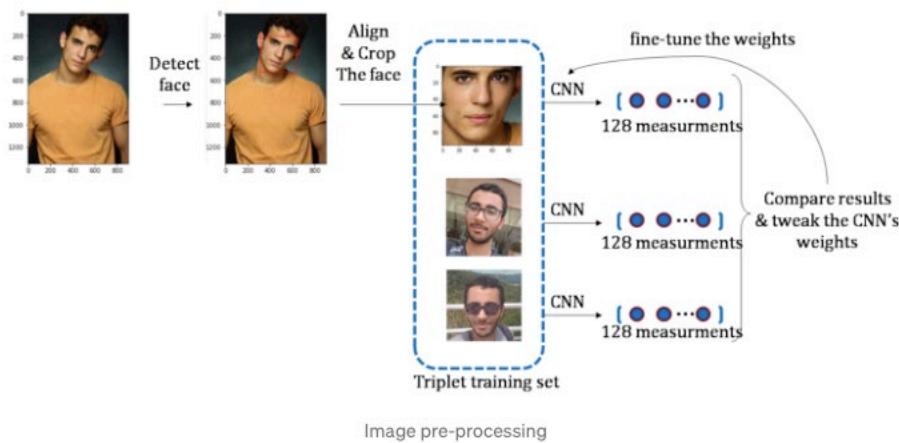


Figure 54 Face Recognition from Convolutional Neural Network (CNN)

จากรูปภาพที่ 54 การทำระบบรู้จำใบหน้าโดยเป็นโมเดล cnn เปรียบเทียบใบหน้าของบุคคลโดยโมเดล CNN ของ Dlib จะมีค่าความแม่นยำสูงเมื่อเทียบกับการตรวจจับใบหน้าด้วย HOG โดย จะมีการระบุใบหน้าในตำแหน่งของรูปภาพ ก่อนจะถูกส่งไปยังโครงข่ายประสาทเทียมแบบคอนโวลูชัน เพื่อให้ได้ความแม่นยำสูงในการจดจำใบหน้า โดย Dlib จะมีการฝึกโมเดลในการแปลงค่าใบหน้า การค้นหาจุดสังเกตบนใบหน้า และทำการแปลงเป็นจุดพิกัดอ้างอิง โดยจะเป็นการใช้ CNN เพื่อแยกเวกเตอร์ 128 มิติ (128-dimensional) และทำการเปรียบเทียบใบหน้าด้วยการเปรียบเทียบใบหน้าโดย triplet ในกระบวนการระยะห่างใบหน้า เพื่อค้นหาใบหน้าในฐานข้อมูลและแสดงผลต่อไป

ประโยชน์ของ Face Recognition

เห็นได้ชัดว่าไฟเจอร์นี้จะเน้นไปที่การจดจำใบหน้าของบุคคลที่ปรากฏบนกล้อง CCTV ดังนั้น จึงถูกนำมาใช้อย่างแพร่หลายได้กับระบบปรักษาความปลอดภัย อุปกรณ์สแกนใบหน้าควบคุมการเปิดปิดประตู หรือระบบลงทะเบียนนักงาน นอกจากนั้น ระบบสแกนใบหน้าตามอาคาร สำนักงานต่าง ๆ ที่เราคุ้นชินกัน ก็เกิดจากการปรับใช้เทคโนโลยีนี้ ซึ่งเป็นตัวอย่างที่เห็นภาพได้จำกัดมาก เพราะเมื่อมีบุคคลที่มีข้อมูลอยู่ในระบบหรือได้ผ่านการเก็บข้อมูลเรียบร้อยแล้ว เข้ามาสแกนใบหน้า ระบบจะส่งภาพที่ตรวจจับได้มาเปรียบเทียบกับฐานข้อมูลเพื่อระบุว่าใบหน้าที่ตรวจจับได้ตรงกับบุคคลใด เพื่อคัดกรองคนเข้าอาคาร ระบบจดจำใบหน้ายังสามารถปรับใช้ตามความต้องการของแต่ละธุรกิจ แต่ละอุตสาหกรรม และยังสามารถปรับใช้ภายในครัวเรือนได้ด้วย

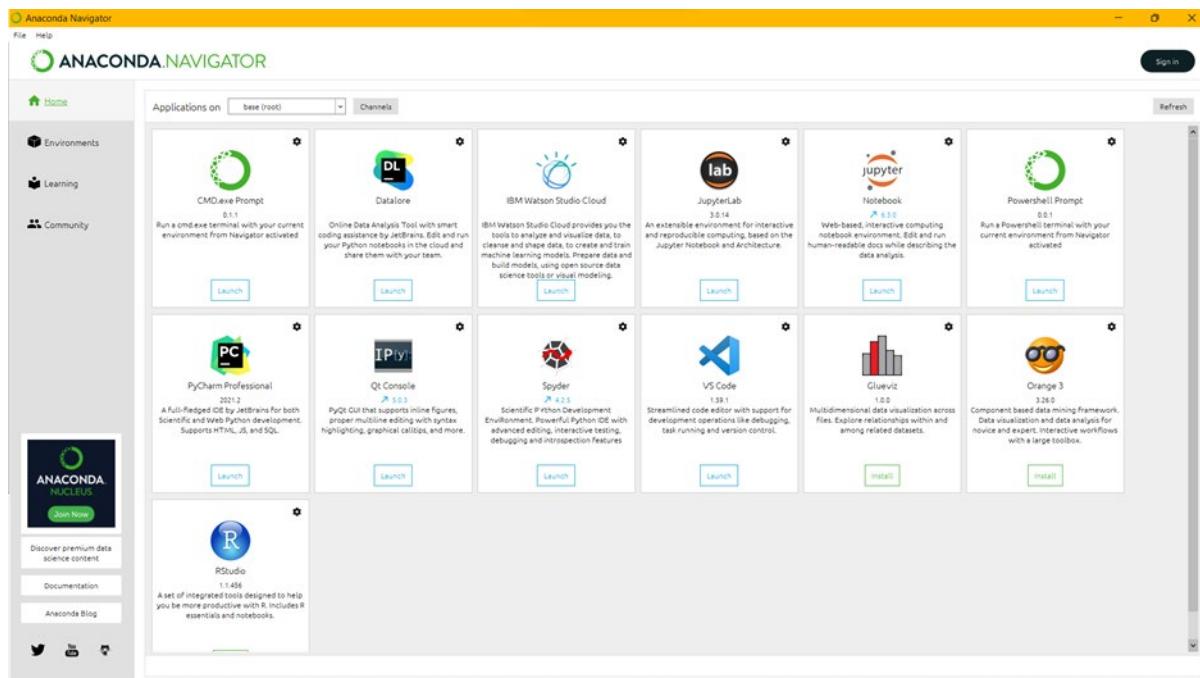
การติดตั้ง Face Recognition

1. ให้ทำการดาวน์โหลดไฟล์ face_recognition.yml สำหรับการติดตั้ง ตามลิงค์ด้านล่าง

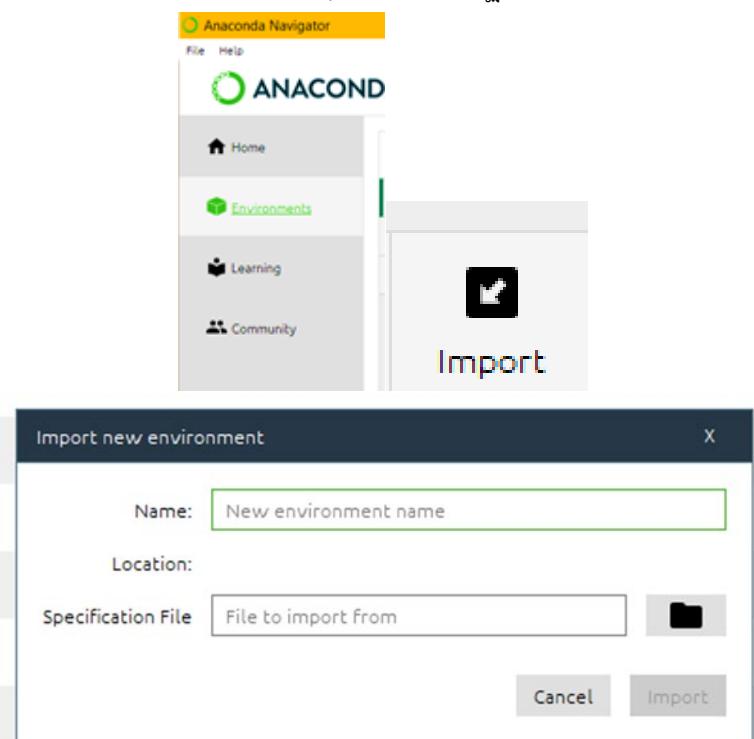


shorturl.asia/SqExp

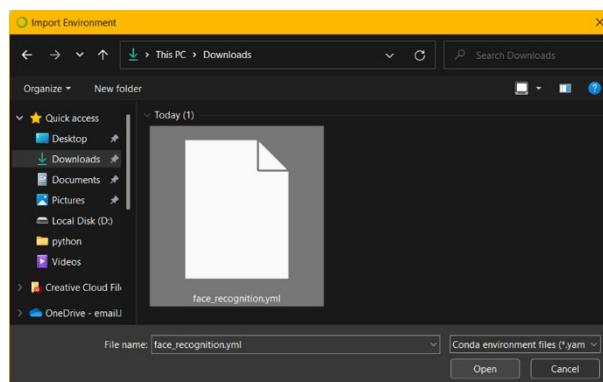
2. เปิดโปรแกรม Anaconda



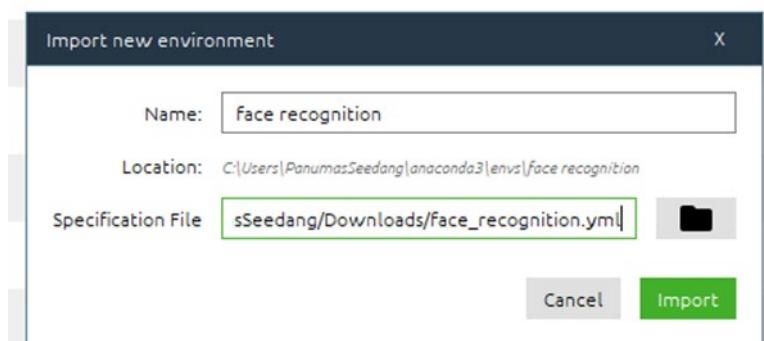
3. คลิกเลือก Environments > เลือก Import จะปรากฏหน้าต่างสำหรับเลือกไฟล์ที่ดาวน์โหลดไว้



4. คลิกเลือกไฟล์ face_recognition.yml ที่ทำการดาวน์โหลดมา



5. คลิกปุ่ม Import



Work shop 1 face_rec_img

ให้ทำการดาวน์โหลดไฟล์ ตามลิงค์ด้านล่าง

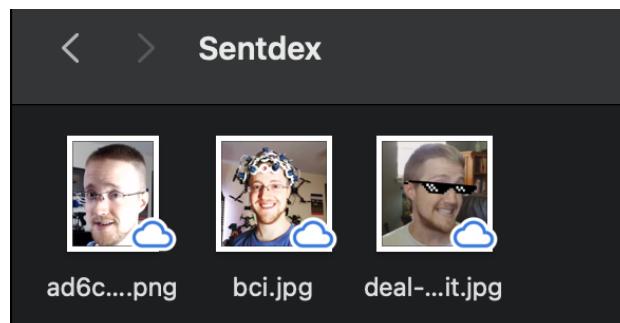


shorturl.asia/SqExp

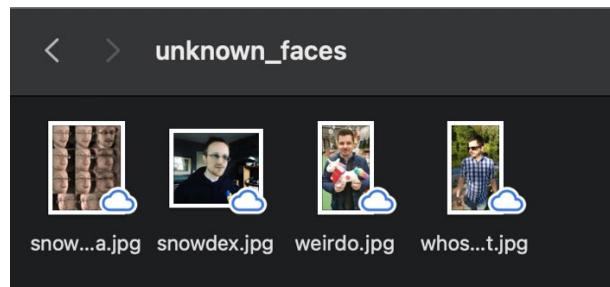
หลังจากนั้นให้ทำการย้ายตำแหน่งไฟล์ไปที่โปรเจคในตำแหน่งที่ต้องการทำโดยจะมีไฟล์ทั้งหมด 3 ส่วนคือ

- known_faces ภายในจะประกอบด้วยโฟลเดอร์ที่เป็นชื่อของบุคคลที่ต้องการสร้างฐานข้อมูล โดยภายในแต่ละโฟลเดอร์มีภาพอย่างน้อยสามภาพเพื่อความแม่นยำ เช่น

1.1. Sentdex



- unknown_faces ภายในจะประกอบด้วยภาพของข้อมูลที่เราต้องการตรวจสอบหรือรู้จำใบหน้า



- Fac_rec_img.py สำหรับการเขียนโปรแกรม

ไฟล์ Fac_rec_img.py

```
#เรียกใช้งาน library
import face_recognition
import os
import cv2

#สร้างตัวแปรสำหรับการจัดการข้อมูล
KNOWN_FACES_DIR = 'known_faces'
UNKNOWN_FACES_DIR = 'unknown_faces'
TOLERANCE = 0.6
FRAME_THICKNESS = 3
FONT_THICKNESS = 2
MODEL = 'cnn' # hog or cnn

#สร้าง feature selection ของภาพที่เป็นส่วนของฐานข้อมูลเพื่อใช้สำหรับการรู้จำใบหน้า
print('Loading known faces...')
known_faces = []
known_names = []
for name in os.listdir(KNOWN_FACES_DIR):
    for filename in os.listdir(f'{KNOWN_FACES_DIR}/{name}'):
        image=face_recognition.load_image_file(f'{KNOWN_FACES_DIR}/{name}/{filename}')
        encoding = face_recognition.face_encodings(image)[0]
        known_faces.append(encoding)
        known_names.append(name)

#สร้าง feature selection ของภาพที่เป็นส่วนของภาพที่ต้องการตรวจสอบใบหน้า
print('Processing unknown faces...')
for filename in os.listdir(UNKNOWN_FACES_DIR):
    print(f'Filename {filename}', end="")
    image = face_recognition.load_image_file(f'{UNKNOWN_FACES_DIR}/{filename}')
    locations = face_recognition.face_locations(image, model=MODEL)
    encodings = face_recognition.face_encodings(image, locations)
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
```

#เปรียบเทียบใบหน้าที่ได้จากการอ่านข้อมูลกับภาพที่ต้องการตรวจสอบ

```
for face_encoding, face_location in zip(encodings, locations):
    results = face_recognition.compare_faces(known_faces, face_encoding,
TOLERANCE)
    match = None
    if True in results:
        match = known_names[results.index(True)]
        print(f' - {match} from {results}'')
```

#การลากกรอบให้กับภาพที่จะแสดง

```
top_left = (face_location[3], face_location[0])
bottom_right = (face_location[1], face_location[2])
color = [0, 255, 0]
cv2.rectangle(image, top_left, bottom_right, color, FRAME_THICKNESS)
```

#การใส่ข้อความสำหรับแสดงชื่อ

```
top_left = (face_location[3], face_location[2])
bottom_right = (face_location[1], face_location[2] + 22)
cv2.rectangle(image, top_left, bottom_right, color, cv2.FILLED)
cv2.putText(image, match, (face_location[3] + 10, face_location[2] + 15),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (200, 200, 200), FONT_THICKNESS)
```

#การแสดงผลรูปภาพ

```
cv2.imshow(filename, image)
cv2.waitKey(0)
#cv2.destroyAllWindows()
```

Work shop 2 face_rec_camara

ให้ทำการปรับเปลี่ยนข้อมูลจากการดึงข้อมูลของไฟล์ unknown_faces เป็นการใช้กล้องในการเปรียบเทียบแทนโดยเปลี่ยนตามตำแหน่งต่อไปนี้

ไฟล์ Fac_rec_camara.py

```
#สร้างตัวแปรสำหรับการจัดการข้อมูล
KNOWN_FACES_DIR = 'known_faces'
#UNKNOWN_FACES_DIR = 'unknown_faces'      ทำการ comment ไว้
TOLERANCE = 0.6
FRAME_THICKNESS = 3
FONT_THICKNESS = 2
MODEL = 'cnn'                      # hog or cnn
```

ให้ทำการปรับเปลี่ยนข้อมูลจากการดึงข้อมูลของไฟล์ unknown_faces เป็นการใช้กล้องในการเปรียบเทียบแทนโดยเปลี่ยนตามตำแหน่งต่อไปนี้

ไฟล์ Fac_rec_camara.py

```
#สร้าง feature selection ของภาพที่เป็นส่วนของภาพที่ต้องการตรวจสอบใบหน้า
print('Processing unknown faces...')
#for filename in os.listdir(UNKNOWN_FACES_DIR): ทำการ comment ไว้
#    print(f'Filename {filename}', end='')

#image      =      face_recognition.load_image_file(f'{UNKNOWN_FACES_DIR}\
/{filename}')
```

while True:

```
ret, image = video.read()
locations = face_recognition.face_locations(image, model=MODEL)
encodings = face_recognition.face_encodings(image, locations)
#image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR) ทำการ comment ไว้
```

#การแสดงผลรูปภาพ

```
cv2.imshow(filename, image)
#cv2.destroyWindow(filename)
#cv2.waitKey(10000)           ทำการ comment ไป
if cv2.waitKey(1) & 0xFF == ord("q"):
    break
```