มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ภาควิชาคอมพิวเตอร์ศึกษา คณะครุศาสตร์อุตสาหกรรม

เอกสารประกอบการอบรม

พื้นฐานหัวข้อในการอบรม

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

จัดทำโดย
วิทยากรฝึกอบรม IMAGE RECOGNITION WITH PYTHON

สารบัญ

สารบัญรูปภาพ

1.ปัญญาประดิษฐ์ (AI : Artificial Intelligence)

ปัญญาประดิษฐ์ (AI : Artificial Intelligence) เป็นระบบประมวลผลที่มีต้นแบบมาจากโครงข่ายประสาทของมนุษย์สามารถเรียนรู้และเพิ่มประสิทธิภาพการประมวลผลได้ตามจำนวนข้อมูลที่เพิ่มขึ้นผ่านกระบวนการเรียนรู้ด้วยตนเอง ซึ่งสามารถจดจำ คิด วิเคราะห์เรียนรู้และเชื่อมโยงข้อมูลต่างๆที่ซับซ้อนได้อย่างรวดเร็ว (Deep Learning) เสมือนระบบสมองของมนุษย์ จึงอาจเรียกได้ว่า “สมองกลอัจฉริยะ” ดังนั้น AI จึงถือเป็นเทคโนโลยีที่ร้อนแรงที่สุดในปัจจุบันและเข้ามามีบทบาทสำคัญต่อการใช้ชีวิตการทำงาน รวมถึงการนำมาใช้ในการเสริมศักยภาพทางธุรกิจและอุตสาหกรรม ซึ่งจะสามารถส่งผลต่อการเจริญเติบโต ทางด้านเศรษฐกิจอย่างยั่งยืนของประเทศ สำหรับปัญญาประดิษฐ์หรือ Artificial Intelligence นั้น เราสามารถแยกออกได้เป็น 2 คำ ได้แก่

“Artificial” มีความหมายว่า สิ่งที่ไม่มีชีวิต ถูกสร้างหรือสังเคราะห์ขึ้นโดยมนุษย์

“Intelligence” มีความหมายว่า ความฉลาด ความคิดคำนวณที่จะนำไปสู่ผลสำเร็จ

โดยทั่วไปแล้วศาสตร์ทางปัญญาประดิษฐ์จะเป็นสาขาในด้านวิทยาการคอมพิวเตอร์และวิศวกรรมเป็นหลัก แต่บางครั้งก็ยังรวมไปถึงศาสตร์ในด้านอื่นๆ อีกด้วย ไม่ว่าจะเป็นทางด้านจิตวิทยา ปรัชญา หรือแม้แต่ชีววิทยาศาสตร์ ทางด้านปัญญาประดิษฐ์นั้นไม่มีข้อกำหนดหรือรูปแบบที่ชัดเจนในการกำหนดความฉลาดของเครื่องจักร เนื่องจากเราไม่สามารถนำมาเปรียบเทียบกับความฉลาดของมนุษย์ที่มีการเปลี่ยนแปลงอยู่ตลอดเวลาได้ และสามารถเข้าใจกลไกเพียงบางส่วนเท่านั้น

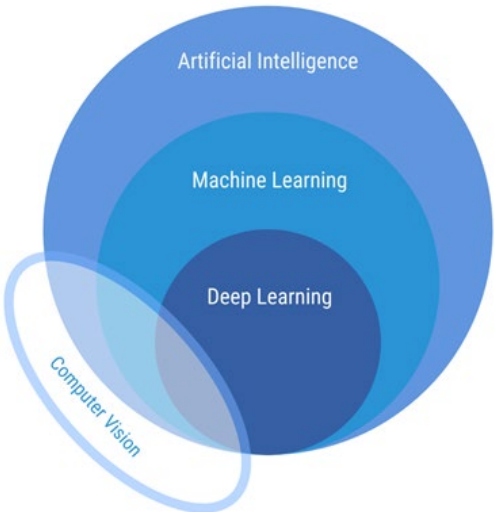


Figure 1 *Machine Learning and Deep Learning are subsets of Artificial Intelligence.*

Machine Learning

Machine Learning คือ ส่วนการเรียนรู้ของเครื่อง ถูกใช้งานเสมือนเป็นสมองของ AI (Artificial Intelligence) อาจพูดได้ว่า AI ใช้ Machine Learning ในการสร้างความฉลาด มักจะใช้เรียกโมเดลที่เกิดจากการเรียนรู้ของปัญญาประดิษฐ์ ไม่ได้เกิดจากการเขียนโดยใช้มนุษย์ มนุษย์มีหน้าที่เขียนโปรแกรมให้ AI (เครื่อง) เรียนรู้จากข้อมูลเท่านั้น ที่เหลือเครื่องจัดการเอง

Machine Learning เรียนรู้จากสิ่งที่มนุษย์ส่งเข้าไปกระตุ้น แล้วจดจำเอาไว้เป็นมันสมอง ส่งผลลัพธ์ออกมาเป็นตัวเลข หรือ code ที่ส่งต่อไปแสดงผล หรือให้เจ้าตัว AI นำไปแสดงการกระทำ Machine Learning เองสามารถเอาไปใช้งานได้หลายรูปแบบ ต้องอาศัยกลไกที่เป็นโปรแกรม หรือเรียกว่า Algorithm ที่มีหลากหลายแบบ โดยมี Data Scientist เป็นผู้ออกแบบ หนึ่งใน Algorithm ที่ได้รับความนิยมสูง คือ Deep Learning ซึ่งถูกออกแบบมาให้ใช้งานได้ง่าย และประยุกต์ใช้ได้หลายลักษณะงาน อย่างไรก็ตาม ในการทำงานจริง Data Scientist จำเป็นต้องออกแบบตัวแปรต่างๆ ทั้งในตัวของ Deep Learning เอง และต้องหา Algorithm อื่นๆ มาเป็นคู่เปรียบเทียบ เพื่อมองหา Algorithm ที่เหมาะสมที่สุดในการใช้งานจริง

Deep Learning

Machine Learning คือ ส่วนการเรียนรู้ของเครื่อง ถูกใช้งานเสมือนเป็นสมองของ AI (Artificial Intelligence) อาจพูดได้ว่า AI ใช้ Machine Learning ในการสร้างความฉลาด มักจะใช้เรียกโมเดลที่เกิดจากการเรียนรู้ของปัญญาประดิษฐ์ ไม่ได้เกิดจากการเขียนโดยใช้มนุษย์ มนุษย์มีหน้าที่เขียนโปรแกรมให้ AI (เครื่อง) เรียนรู้จากข้อมูลเท่านั้น ที่เหลือเครื่องจัดการเอง

Computer Vision

เป็นแขนงหนึ่งของวิทยาการปัญญาประดิษฐ์หรือ AI ซึ่งทำการฝึกฝนคอมพิวเตอร์ และระบบให้สามารถเข้าใจและตอบสนองต่อข้อมูลภาพได้อย่าง ชาญฉลาด ด้วยภาพดิจิทัลจากกล้องถ่ายภาพและวิดีโอ 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 ของรูปภพนั้น ๆ โดยแบบจำลองเหล่านั้นอาจจะนำมาใช้ในการสร้างต้นแบบ หุ่นยนต์ AI ในการค้นหาวัตถุเหล่านั้นด้วยรูปภาพ จำเป็นต้องมีฐานข้อมูลขนาดใหญ่

ระบบ AI ในปัจจุบันนั้น มีประสิทธิภาพสูง และสามารถดำเนินการต่อยอดจากผลลัพธ์ที่ได้รับ และนำข้อมูลจากการทำความเข้าใจภาพมาใช้ให้เกิดประโยชน์ต่อไปได้ ซึ่งมีรูปแบบของเทคโนโลยี Computer Vision หลายรูปแบบ และมีการใช้งานในหลายสถานการณ์ตามไปด้วย

การประยุกต์ใช้ Computer Vision

1. Image segmentation - คือการแยกส่วนของภาพออกเป็นหลาย ๆ ส่วนหรือชิ้นองค์ประกอบย่อย ๆ เพื่อพิจารณาแยกส่วนกัน
2. Object detection - หรือการตรวจหาวัตถุแบบเฉพาะเจาะจงในภาพแต่ละภาพ ซึ่งมีการทำงานในระดับสูงที่สามารถระบุวัตถุหลายชิ้นในภาพเดียวกันได้
3. Face recognition - หรือการจดจำใบหน้า เป็นรูปแบบการระบุวัตถุขั้นสูงที่มิได้ทำแค่การระบุว่ามีใบหน้าของมนุษย์อยู่ในภาพเท่านั้น แต่ยังสามารถแยกแยะบุคคลแต่ละบุคคลออกจากกันและระบุบุคคลที่เจาะจงได้อีกด้วย
4. Edge detection - เป็นเทคนิคการระบุหาขอบหรือมุมของวัตถุ หรือภาพทิวทัศน์ เพื่อให้ทราบได้ง่ายขึ้นว่าองค์ประกอบในภาพมีสิ่งใดบ้าง
5. Pattern detection - คือการระบุวัตถุจากรูปทรง หรือสี หรือสิ่งบ่งชี้ต่าง ๆ ที่พบในภาพ ที่เป็นรูปแบบเดียวกันซ้ำ ๆ สำหรับวัตถุประเภทนั้น ๆ
6. Image classification - ทำงานด้วยการจัดกลุ่มภาพออกเป็นหมวดหมู่ต่าง ๆ
7. Feature matching - เป็นรูปแบบหนึ่งของการตรวจหารูปแบบหรือ pattern detection ที่ระบุจุดที่เหมือนหรือคล้ายคลึงกันในภาพต่าง ๆ เพื่อจัดหมวดหมู่แก่วัตถุและภาพเหล่านั้น

ตัวอย่างการใช้ Computer Vision

ระบบดูแลการจราจรบนท้องถนน โดยการนับจำนวนรถบนท้องถนนในภาพถ่ายด้วยกล้องวงจรปิดในแต่ละช่วงเวลา

ระบบเก็บข้อมูลรถที่เข้าและออกอาคาร โดยใช้ภาพถ่ายของป้ายทะเบียนรถเพื่อประโยชน์ในด้านความปลอดภัย โดยนำข้อมูลภาพป้ายทะเบียนของรถยนต์ ที่ได้จากกล้องวงจรปิดประเภท LPR Camera (กล้องสำหรับจับภาพป้ายทะเบียนโดยเฉพาะ) มาทำการวิเคราะห์ตรวจหาป้ายทะเบียน

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

ระบบดูแลการจราจรบนท้องถนน โดยการนับจำนวนรถบนท้องถนนในภาพถ่ายด้วยกล้องวงจรปิดในแต่ละช่วงเวลา

ระบบแปลภาษา จะช่วยแปลข้อความหรือคำพูดตามความหมาย ในรูปแบบที่เป็นธรรมชาติมากขึ้น ซึ่งจะใช้บริบทที่กว้างขึ้นเพื่อช่วยให้แปลได้ตรงกับความหมายที่สุดระบบจะจัดเรียงและปรับให้ตรงหรือใกล้เคียงกับภาษาพูดของมนุษย์มากที่สุด การแปลข้อความหลาย ๆ ย่อหน้าหรือบทความก็จะอ่านแล้วเข้าใจได้ง่ายขึ้น

ระบบตรวจจับใบหน้า โปรแกรมตรวจจับใบหน้าจะทำหน้าที่วิเคราะห์ตรวจสอบและจดจำใบหน้าที่ทำงานร่วมกับกล้องวงจรปิด IP Camera คุณภาพสูง โดยลักษณะการทำงานระบบจะทำการเปรียบเทียบใบหน้าของบุคคล ที่ผ่านเข้ามาในกล้องที่ได้กำหนดและตั้งค่าเอาไว้ จากนั้นจะทำการนำภาพใบหน้าดังกล่าวมาเปรียบเทียบกับภาพบุคคลในฐานข้อมูล

การทำงานของ Computer Vision

คอมพิวเตอร์วิทัศน์เป็นศูนย์กลางของนวัตกรรมระดับแนวหน้ามากมาย รวมถึงรถยนต์ไร้คนขับ โดรน เทคโนโลยีการจดจำใบหน้า และอื่นๆ อีกมากมาย ด้วยความก้าวหน้าของ AI

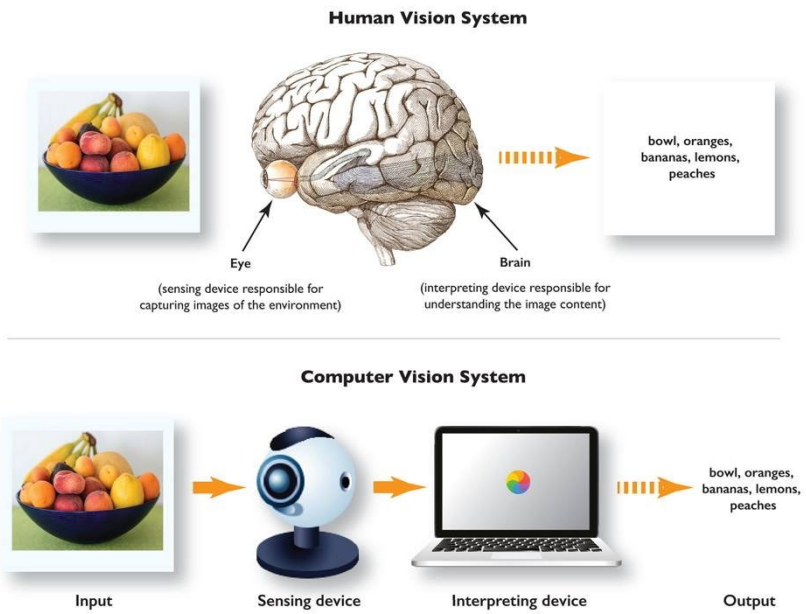


Figure 2 การทำงานของ Computer Vision

Computer Vision เมื่อเทียบกับวิธีที่มนุษย์ประมวลผลการป้อนข้อมูลด้วยภาพ

การประยุกต์ใช้ Computer Vision แตกต่างกันไป จากรูปจะได้เห็นกระบวนการการทำงาน 4 ขั้นตอนนั่นก็คือ Input data, preprocessing, feature extraction, ML model การประยุกต์ใช้ Computer Vision แตกต่างกันไป จากรูปภาพนี้ เราจะได้เห็นกระบวนการการทำงาน 4 ขั้นตอนนั่นก็คือ Input data, preprocessing, feature extraction, ML model

Digital image (ภาพดิจิทัล)

ภาพดิจิทัล คือ ข้อมูลรูปภาพที่ถูกจัดเก็บในรูปแบบดิจิทัล ซึ่งภาพสามารถทำการเปลี่ยนแปลง แก้ไข ประมวลผล หรือถ่ายโอนได้โดยใช้ระบบคอมพิวเตอร์ ภาพดิจิทัลจะมีรูปแบบการเก็บเป็นเมทริกซ์ ซึ่งจะมีการจัดเก็บภาพแต่ละชนิดต่างกัน โดยแบ่งชนิดของภาพได้ ดังนี้

Pixel

พิกเซล (Pixel) คือ จุดที่เป็นจุดภาพที่แสดงบนหน้าจอ และเอามารวมกันให้เกิดเป็นภาพขึ้นมา แต่ละจุดจะมีสีที่แตกต่างกันไป ยิ่งละเอียดมากจะทำให้ภาพนั้นชัดเจนมากยิ่งขึ้น แน่นอนว่าความใหญ่ของภาพก็ใหญ่ขึ้นด้วยเช่นเดียวกัน พิกเซลมากไม่ได้หมายความว่าภาพนั้นจะคมชัดเสมอไป แต่ภาพที่มีพิกเซลมาก จะสามารถถ่ายภาพได้มีความละเอียดสูงได้

ค่าพิกเซล 2, 3, 4, หรือ 5 ล้านพิกเซล หรืออื่น ๆ ตัวเลขพวกนี้คืออัตราส่วนของความละเอียดของเม็ดสีต่างๆที่รวมกัน ในขนาดความกว้างและความสูง เช่น 2 ล้านพิกเซล จะเท่ากับความละเอียด 1080P คือ ภาพที่มีขนาด 1920H x 1080P หรือขนาดค่าความละเอียดของภาพ คำนวนได้จาก H (แนวนอน) x P (แนวตั้ง)

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

Binary Image

ไบนารีในทางดิจิตอลหมายถึงว่ามีเพียง 2 สถานะคือ 0 และ 1 ซึ่งภาพไบนารี่เป็นรูปที่ใช้เนื้อที่เพียง 1 บิต จะมีแค่ความเข้ม 2 ค่าเท่านั้นคือ 0 และ 1 เป็นรูปที่ใช้เนื้อที่เพียง 1 บิตหมายความว่า พิกเซลใดที่มีค่าเป็น 0 ก็จะหมายถึงว่าพิกเซลนั้นจะแสดงสีดำ พิกเซลใดที่มีค่าเป็น 1 จะหมายถึงว่าพิกเซลนั้นจะแสดงสีขาว

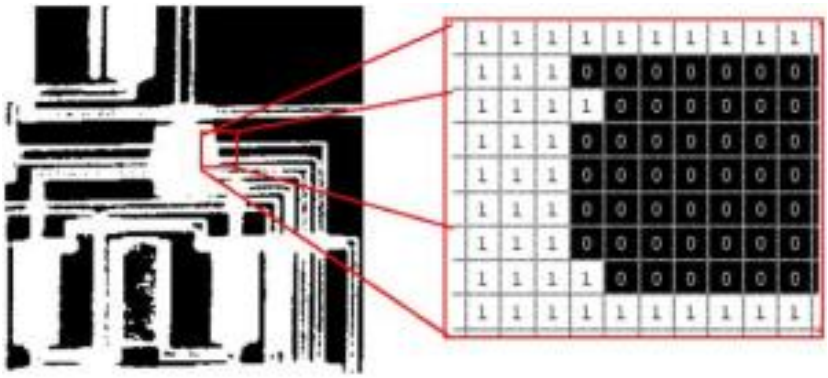


Figure 3 ภาพ Binary หรือ ภาพขาว-ดำ

RGB Image

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

Figure 4 ภาพ RGB

เป็นรูปที่เก็บโดยใช้รูปแบบของอาร์เรย์ 2 มิติ โดยค่าที่เก็บจะมีค่าอยู่ในช่วง ๆ หนึ่ง ซึ่งระดับของสีขึ้นอยู่กับขนาดของบิตที่ใช้เก็บค่าสี

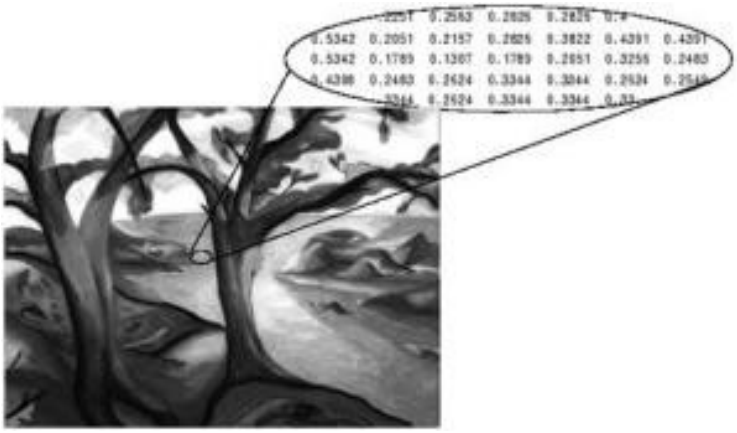


Figure 5 Grayscale

ตัวอย่างการนำรูปไปประมวลผล

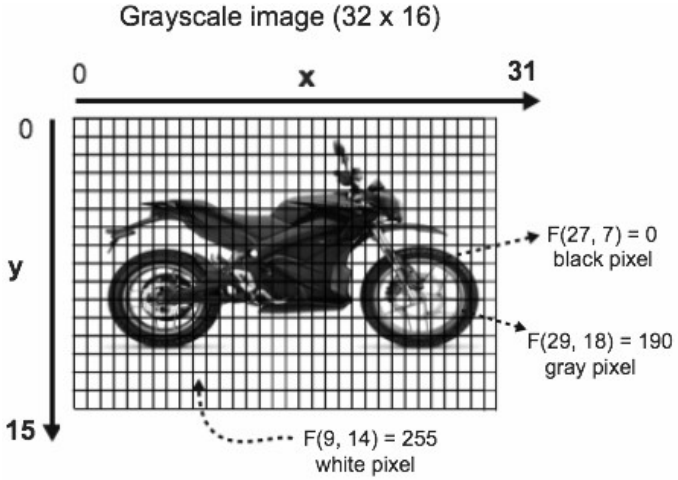


Figure 6 Greyscale image (ภาพระดับสีเทา)

ภาพระดับสีเทา) คือภาพขาว-ดำ-เทา โดยจะมีระดับความเข้มจากการแปลงสีเท่า คือ 0–255 ภาพเกรย์สเกล เกิดจากการแปลงภาพสี RGB มาเป็น Grayscale รูปภาพสามารถแสดงเป็นฟังก์ชันของตัวแปรสองตัวคือ X และ Y ซึ่งกำหนดพื้นที่สองมิติ ภาพด้านบนมีขนาด 32 x 16 ซึ่งหมายความว่าขนาดของรูปภาพกว้าง 32 พิกเซล และสูง 16 พิกเซล แกน X เริ่มจาก 0 ถึง 31 และแกน Y ตั้งแต่ 0 ถึง 16 (นับ 0 รวมไปด้วย)

โดยรวมแล้ว รูปภาพมี 32×16 = 512 พิกเซล ในภาพระดับสีเทานี้ แต่ละพิกเซลมีค่าที่แสดงถึงความเข้มของแสงบนพิกเซลเฉพาะรูปนี้มีค่าพิกเซลแตกต่างกันไปตั้งแต่ 0 ถึง 255 เนื่องจากค่าพิกเซลแสดงถึงความเข้มของแสง ค่า 0 จึงหมายถึงพิกเซลมืด (สีดำ) 255 จึงเป็นความสว่าง (สีขาว) และค่าที่อยู่ระหว่างนั้นแสดงถึงความเข้มของระดับสีเทา

เมื่อดูภาพ จะเห็นวัตถุ แนวนอน สี ฯลฯ แต่นี่ไม่ใช่กรณีของคอมพิวเตอร์ พิจารณาภาพด้านล่าง สมองของมนุษย์สามารถมองดูแล้วรู้ทันทีว่าเป็นภาพมอเตอร์ไซค์ สำหรับคอมพิวเตอร์ รูปภาพจะดูเหมือนเมทริกซ์ 2 มิติของค่าพิกเซลซึ่งแสดงถึงความเข้มของสเปกตรัมสี จะเห็นได้ว่าพิกเซลในเมทริกซ์แสดงถึงความเข้มของความสว่างในพิกเซลนั้น 0 หมายถึงสีดำและ 255 หมายถึงสีขาว

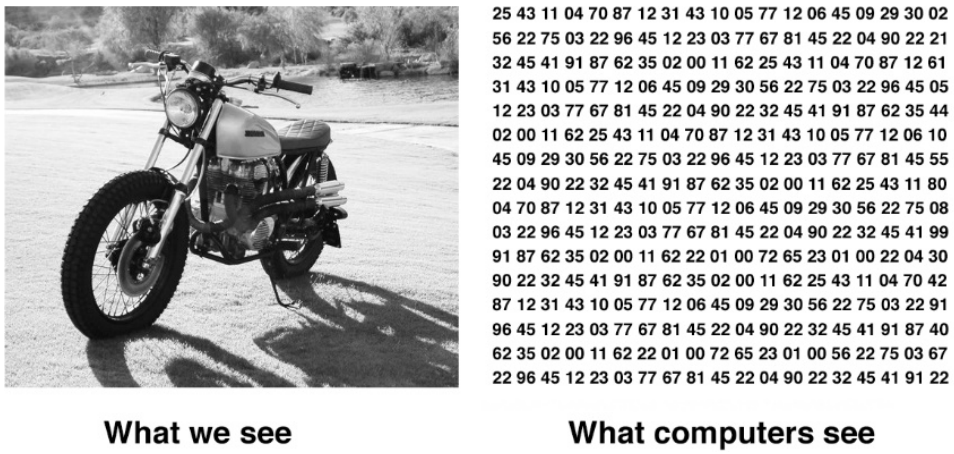


Figure 7 ภาพที่มนุษย์มองเห็น และภาพที่คอมพิวเตอร์เห็น

ขั้นตอนการทำงานของ Computer Vision

1. Input data คือการนำเข้าข้อมูล ข้อมูลในที่นี้เราจะหมายถึง Image คอมพิวเตอร์รับอินพุตด้วยภาพจากอุปกรณ์สร้างภาพ เช่น กล้องหรือสมาร์ทโฟน โดยทั่วไปจะบันทึกเป็นรูปภาพหรือลำดับของรูปภาพที่สร้างวิดีโอ และภาพที่ได้จากการถ่ายโดยกล้องหรือสมาร์ทโฟนเนี่ย เราจะเรียกว่าภาพดิจิทัล ซึ่งเป็นการแสดงผลภาพในลักษณะสองมิติในหน่วยที่เรียกว่าพิกเซล ภาพดิจิทัลจะมีรูปแบบการเก็บเป็นเมทริกซ์ ซึ่งจะมีการจัดเก็บภาพแต่ละชนิดต่างกัน ขึ้นอยู่กับระบบสีของภาพดังกล่าว
2. Preprocessing ในที่นี้หมายถึง การนำภาพมาประมวลผลหรือคิดคำนวณด้วยคอมพิวเตอร์ เพื่อให้ได้ข้อมูลที่เราต้องการทั้งในเชิงคุณภาพและปริมาณ โดยมีขั้นตอนต่าง ๆ ที่สำคัญ คือ การทำให้ภาพมีความคมชัดมากขึ้น การกำจัดสัญญาณรบกวนออกจากภาพ การแบ่งส่วนของวัตถุที่เราสนใจออกมาจากภาพ เพื่อนำภาพวัตถุที่ได้ไปวิเคราะห์หาข้อมูลเชิงปริมาณ เช่น ขนาด รูปร่าง และทิศทางการเคลื่อนของวัตถุในภาพ จากนั้นเราสามารถนำข้อมูลเชิงปริมาณเหล่านี้ไปวิเคราะห์ และสร้างเป็นระบบ เพื่อใช้ประโยชน์ในงานด้านต่างๆ เช่น ระบบรู้จำลายนิ้วมือเพื่อตรวจสอบว่าภาพลายนิ้วมือที่มีอยู่นั้นเป็นของผู้ใด ระบบตรวจสอบคุณภาพของผลิตภัณฑ์ในกระบวนการผลิตของโรงงานอุตสาหกรรม ระบบคัดแยกเกรดหรือคุณภาพของพืชผลทางการเกษตร ระบบอ่านรหัสไปรษณีย์อัตโนมัติ เพื่อคัดแยกปลายทางของจดหมายที่มีจำนวนมากในแต่ละวันโดยใช้ภาพถ่ายของรหัสไปรษณีย์ที่อยู่บนซอง ระบบเก็บข้อมูลรถที่เข้าและออกอาคารโดยใช้ภาพถ่ายของป้ายทะเบียนรถเพื่อประโยชน์ในด้านความปลอดภัย ระบบดูแลและตรวจสอบสภาพการจราจรบนท้องถนนโดยการนับจำนวนรถบนท้องถนนในภาพถ่ายด้วยกล้องวงจรปิดในแต่ละช่วงเวลา ระบบรู้จำใบหน้าเพื่อเฝ้าระวังผู้ก่อการร้ายในอาคารสถานที่สำคัญ ๆ หรือในเขตคนเข้าเมือง เป็นต้น จะเห็นได้ว่าระบบเหล่านี้จำเป็นต้องมีการประมวลผลภาพจำนวนมาก และเป็นกระบวนการที่ต้องทำซ้ำ ๆ กันในรูปแบบเดิมเป็นส่วนใหญ่ ซึ่งงานในลักษณะเหล่านี้ หากให้มนุษย์วิเคราะห์เอง มักต้องใช้เวลามากและใช้แรงงานสูง อีกทั้งหากจำเป็นต้องวิเคราะห์ภาพเป็นจำนวนมาก ผู้วิเคราะห์ภาพเองอาจเกิดอาการล้า ส่งผลให้เกิดความผิดพลาดขึ้นได้ ดังนั้นคอมพิวเตอร์จึงมีบทบาทสำคัญในการทำหน้าที่เหล่านี้แทนมนุษย์ อีกทั้งเป็นที่ทราบโดยทั่วกันว่า คอมพิวเตอร์มีความสามารถในการคำนวณและประมวลผลข้อมูจำนวนมหาศาลได้ในเวลาอันสั้น จึงมีประโยชน์อย่างมากในการเพิ่มประสิทธิภาพการประมวลผลภาพและวิเคราะห์ข้อมูลที่ได้จากภาพในระบบต่าง ๆ ดังกล่าวข้างต้น

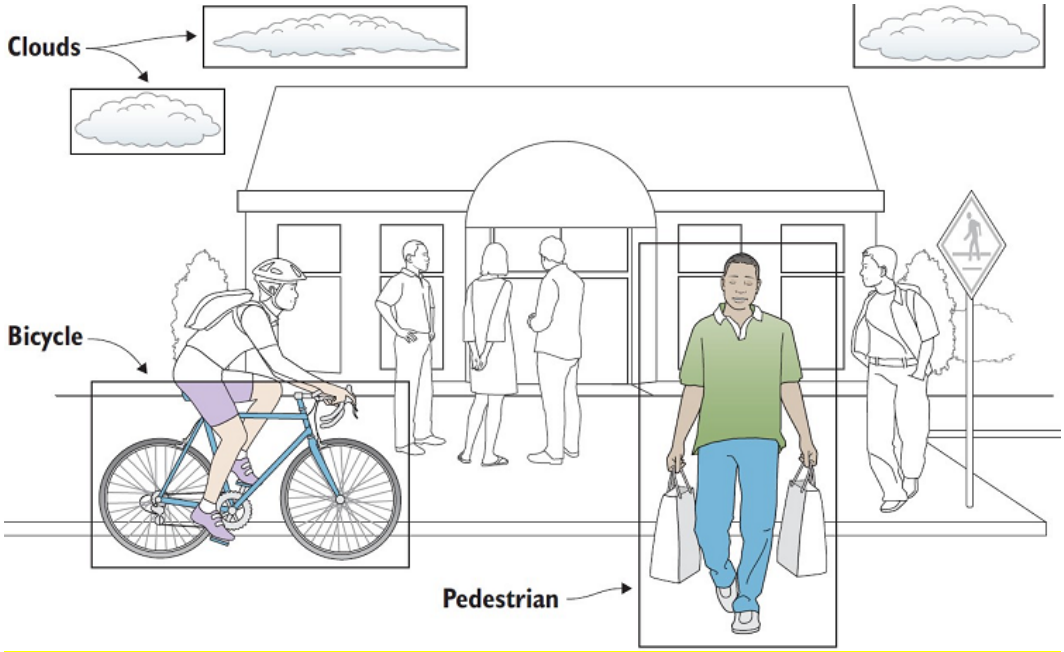


Figure 8 การประมวลผลภาพ

ะตัดให้เหลือเพียงแค่จักรยาน หรือจะบอกว่าระบบจะจำกำจัดสิ่งที่ไม่ต้องการหรือเจือปนอยู่ออกไปให้เหลือเพียงสิ่งที่ต้องการจริง ๆ

Image Processing

เทคนิคที่ใช้ในการประมวลผลภาพ

1. ปรับปรุงคุณภาพของภาพ (Image Enhancement) — เป็นกระบวนการในการแปลงข้อมูลภาพตัวเลข เพื่อที่จะสร้างภาพที่เน้นรายละเอียดที่ต้องการ หรือปรับพิสัยของโทนแสงที่ต้องการของภาพ เมื่อ เปรียบเทียบกับข้อมูลหรือรายละเอียดอื่นๆ ของภาพ
2. การกรองภาพหรือการกำจัดสัญญาณรบกวนออกจากภาพ (Image Filters) — คือการนำภาพไปผ่านตัวกรองสัญญาณเพื่อให้ได้ภาพผลลัพธ์ออกมา ภาพผลลัพธ์ที่ได้จะมีคุณสมบัติแตกต่างจากภาพเริ่มต้น วัตถุประสงค์หลักของการกรองข้อมูลภาพคือการเน้น (enhance) หรือลดทอน (attenuate) คุณสมบัติบางประการของภาพ เพื่อให้ได้ภาพที่มีคุณสมบัติตามต้องการ
3. การซ้อนทับภาพ (Image Registration)— เป็นวิธีการนำข้อมูลของสองภาพหรือมากกว่า มารวมกันเพื่อให้เกิดภาพใหม่ที่มีข้อมูลภาพสมบูรณ์มากขึ้น โดยภาพใหม่ที่ได้นี้ จะเป็นการรวมตัวกันของข้อมูลหรือรายละเอียดในแต่ละภาพที่นำมาผสานกัน มีวัตถุประสงค์เพื่อให้ได้ภาพที่มีรายละเอียดและข้อมูลที่เพียงพอสำหรับการนำไปใช้
4. การคืนสภาพของภาพ (Image Restoration) — การทำให้ภาพคืนสู่สภาพเดิมหรือการปรับปรุงภาพให้เหมาะสมกับการมองเห็น

5. การแบ่งส่วนภาพ (Image Segmentation) — เป็นวิธีการแบ่งส่วนใดส่วนหนึ่งของภาพที่เราสนใจออกมาจากภาพที่เราต้องการ ซึ่งการแบ่งส่วนภาพนี้ โดยส่วนใหญ่แล้วจะเป็นขั้นตอนเบื้องต้นและสำคัญอย่างมากของการประมวลผลภาพทางการแพทย์ เนื่องจากภาพทางการแพทย์ที่ได้จากเครื่องถ่ายภาพแบบต่าง ๆ นั้น โดยปกติมักจะมีองค์ประกอบอื่น ๆ ที่อยู่ใกล้เคียงกับอวัยวะที่ทำถ่ายภาพมา เช่น เนื้อเยื่อ กระดูก อวัยวะข้างเคียง หรือแม้กระทั่งสัญญาณรบกวน (Noise) ที่ขึ้นในขณะถ่ายภาพ ด้วยเหตุนี้ การวิเคราะห์เฉพาะอวัยวะที่ต้องการ
6. การหาขอบภาพในวัตถุ (Image Segmentation and EdgDeTection) — เป็นการหาเส้นรอบวัตถุที่อยู่ในภาพ เมื่อทราบเส้นรอบวัตถุ เราจะสามารถคำนวณหาพื้นที่ (ขนาด) หรือรู้จำชนิดของวัตถุนั้นได้ อย่างไรก็ตาม การหาขอบภาพที่ถูกต้องสมบูรณ์ไม่ใช่เป็นเรื่องง่ายโดยเฉพาะอย่างยิ่งการหาขอบของภาพที่มีคุณภาพต่ำ มีความแตกต่างระหว่างพื้นหน้ากับพื้นหลังน้อย หรือมีความสว่างไม่สม่ำเสมอทั่วภาพ ขอบภาพเกิดจากความแตกต่างของความเข้มแสงจากจุดหนึ่งไปยังอีกจุดหนึ่ง หากต่างนี้มีค่ามาก ขอบภาพก็จะเห็นได้ชัด ถ้าความแตกต่างมีค่าน้อยขอบภาพก็จะไม่ชัดเจน
7. การบีบอัดภาพ (Image Compression)
   7.1. การบีบอัดแบบไม่มีการสูญเสียรายละเอียดข้อมูล (Lossless compression) ค่าความสว่างของแต่ละจุดภาพจะยังคงอยู่เหมือนเดิมทุกประการ หรือไม่มีการเปลี่ยนแปลงค่าของแต่ละจุดภาพ ซึ่งการบีบอัดวิธีนี้จะอาศัยเทคนิคการจัดเก็บข้อมูลเชิงเลขในการลดขนาดของข้อมูล
   7.2. การบีบอัดแบบสูญเสียรายละเอียดข้อมูล (Lossy compression) วิธีการนี้จะมีการเปลี่ยนแปลงค่าความสว่างของจุดภาพนั่นหมายความว่า วิธีการนี้ไม่เหมาะสมสำหรับข้อมูลภาพที่ต้องมีการจำแนกข้อมูล (Classification)
8. การสร้างภาพ 3 มิติ (3D Image Reconstruction) — คือการนำภาพหลายๆมุมในวัตถุชิ้นเดียวกัน และหาจุดเชื่อมต่อและนำมาประกอบกันให้เป็น 3 มิติ

ในส่วนของ Image processing จะเอามาใช้ในขั้นตอน Preprocessing อย่างเช่น การปรับภาพเปลี่ยนจากภาพสีเป็นโทนสีเทา การตัดส่วนที่ไม่จำเป็นออกรวมไปถึงการปรับขนาดรูปภาพ เปลี่ยนรูปร่าง หรือกำหนดมาตรฐานแต่ละภาพเท่านั้น เช่น ทำให้มีขนาดเท่ากัน จากนั้นจะสามารถเปรียบเทียบและวิเคราะห์เพิ่มเติมในลักษณะเดียวกันได้ เป็นต้น

3. Feature Extraction

จะเป็นการแยกคุณสมบัติ หรือคุณลักษณะคือสิ่งที่ช่วยกำหนดวัตถุบางอย่าง และมักจะเป็นข้อมูลเกี่ยวกับรูปร่างหรือสีของวัตถุ ตัวอย่างเช่น คุณลักษณะบางอย่างที่แยกแยะรูปร่างของล้อรถจักรยานยนต์ ไฟหน้า บังโคลน และอื่นๆ ผลลัพธ์ของกระบวนการนี้คือเวกเตอร์คุณลักษณะซึ่งเป็นรายการของรูปร่างที่ไม่ซ้ำกันซึ่งระบุวัตถุ

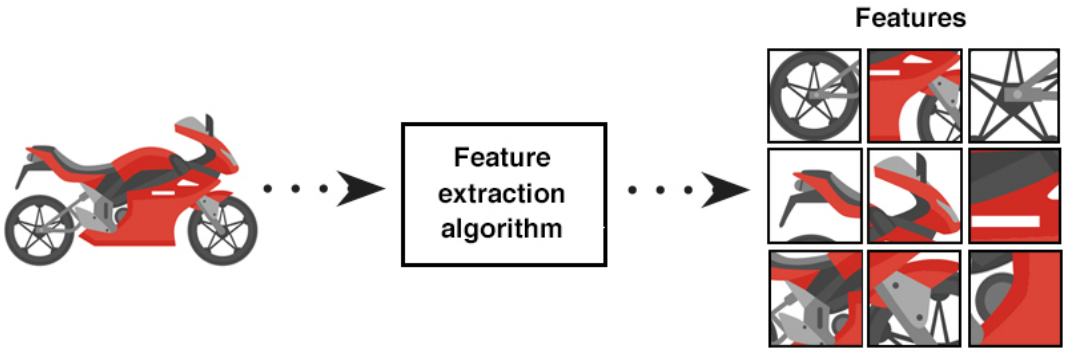


Figure 9 กระบวนการ Feature Extraction

การจำแนกข้อมูล (Classification) ที่ซึ่งจะทำการสร้างโมเดลหรือตัวจำแนกข้อมูล (Classifier) เพื่อทำนายหมวดหมู่ของข้อมูล ขั้นตอนนี้ดูเวกเตอร์คุณลักษณะจากขั้นตอนก่อนหน้าและคาดการณ์ผลลัพธ์ของรูปภาพ ผลลัพธ์ของกระบวนการนี้คือความน่าจะเป็นของแต่ละคลาส ดังที่เห็นในรูปข้างต้น สุนัขมีโอกาสน้อยที่สุดที่ 1% ในขณะที่มีความเป็นไปได้ 85% ที่จะเป็นมอเตอร์ไซค์ จะเห็นได้ว่าถึงแม้โมเดลจะสามารถทำนายระดับที่เหมาะสมได้ด้วยความน่าจะเป็นสูง

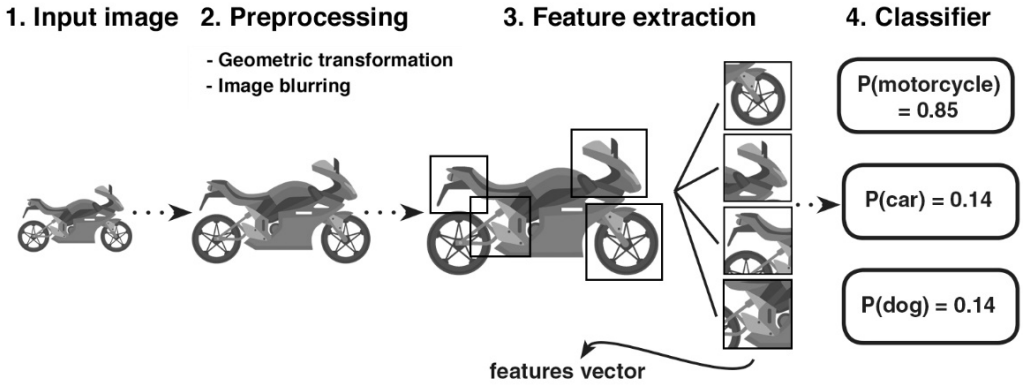


Figure 10 การ Classify

แต่ก็ยังสับสนเล็กน้อยในการแยกแยะระหว่างรถยนต์และรถจักรยานยนต์ เนื่องจากคาดการณ์ว่ามีโอกาส 14% ที่ภาพนี้จะเป็นภาพของรถยนต์ เนื่องจากรู้ว่าเป็นมอเตอร์ไซค์ จึงสามารถพูดได้ว่าอัลกอริทึมการจำแนกประเภท ML นั้นแม่นยำถึง 85% เพื่อปรับปรุงความแม่นยำนี้ อาจต้องทำเพิ่มเติมจากขั้นตอนที่ 1 (รับภาพการฝึกเพิ่มเติม) หรือขั้นตอนที่ 2 (การประมวลผลเพิ่มเติมเพื่อขจัดสัญญาณรบกวน) หรือขั้นตอนที่ 3 (แยกคุณลักษณะที่ดีกว่า) หรือขั้นตอนที่ 4 (เปลี่ยนอัลกอริทึมตัวแยกประเภท)

ประเภทของอัลกอริทึมการจำแนกประเภท

Figure 11 ประเภทของอัลกอริทึมการจำแนกประเภท

ตัวหนึ่งที่สามารถอธิบายได้ว่าทำไมถึงแบ่งเป็นคลาสนี้ ทำไมต้องเป็นคลาสนี้ สามารถอธิบายได้ด้วยรูปแบบของ “TREE” นั่นก็คือ มี Node แม่เป็นคนตั้งคำถามว่าใช่หรือไม่ Node ลูกตัวแรกอาจจะเป็นใช่ อีกตัวจะเป็นไม่ โดยปัจจัยสำคัญในการสร้างโมเดลนี้คือ “ความลึกของต้นไม้” ยิ่งต้นไม้ลึกหรือมีจำนวนชั้นที่มาก ก็จะถามได้ละเอียดมากขึ้นแต่ก็จะยิ่ง overfit มากขึ้น แต่ถ้าจำนวนชั้นที่น้อยไป ก็จะไม่แม่นยำพอจะใช้งาน

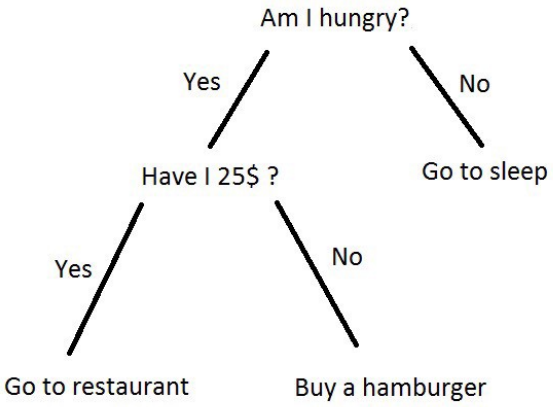


Figure 12 Decision Tree

K-Nearest Neighbour (KNN) เป็นวิธีการจำแนกประเภทข้อมูลวิธีหนึ่ง โดยจัดว่าเป็นการจำแนกแบบมีผู้ฝึกสอน (Supervised Machine Learning Algorithm) หรือทราบคำตอบอยู่แล้ว จากนั้นจะใช้โมเดลในการจำแนกประเภทข้อมูลจากข้อมูลที่รู้คำตอบ KNN เป็นวิธีการแบ่งคลาสสำหรับใช้จัดหมวดหมู่ข้อมูล (Classification) โดยใช้หลักการเปรียบเทียบข้อมูลที่สนใจกับข้อมูลอื่นว่ามีความคล้ายคลึงมากน้อยเพียงใด หากข้อมูลที่กำลังสนใจนั้นอยู่ใกล้ข้อมูลใดมากที่สุด ระบบจะให้คำตอบเป็นเหมือนคำตอบของข้อมูลที่อยู่ใกล้ที่สุดนั้นลักษณะการทำงานแบบไม่ได้ใช้ข้อมูลชุดเรียนรู้ (training data) ในการสร้างแบบจำลองแต่จะใช้ข้อมูลนี้มาเป็นตัวแบบจำลองเลย เทคนิคนี้จะตัดสินใจว่าข้อมูลมีความคล้ายคลึงหรือใกล้เคียงกับคลาสใด โดยการตรวจสอบข้อมูลบางจำนวนนั่นเอง

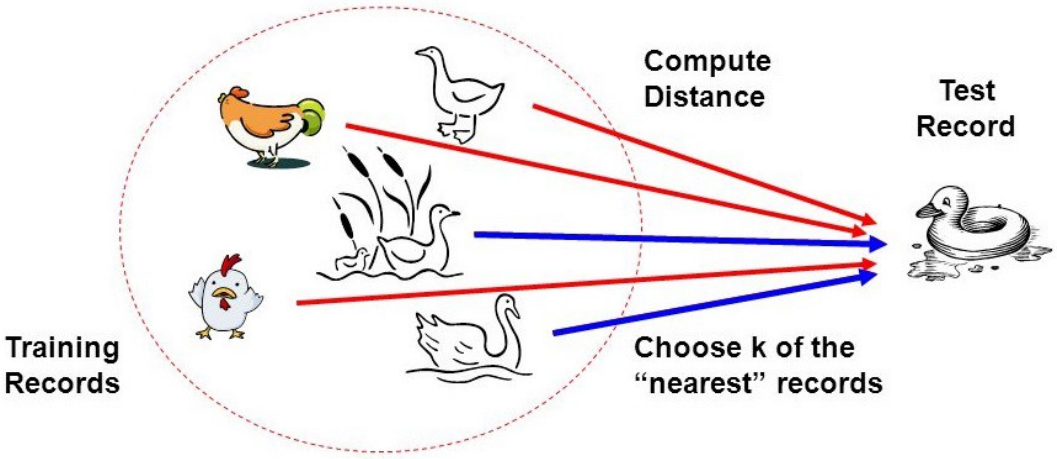


Figure 13 ตัวอย่าง KNN

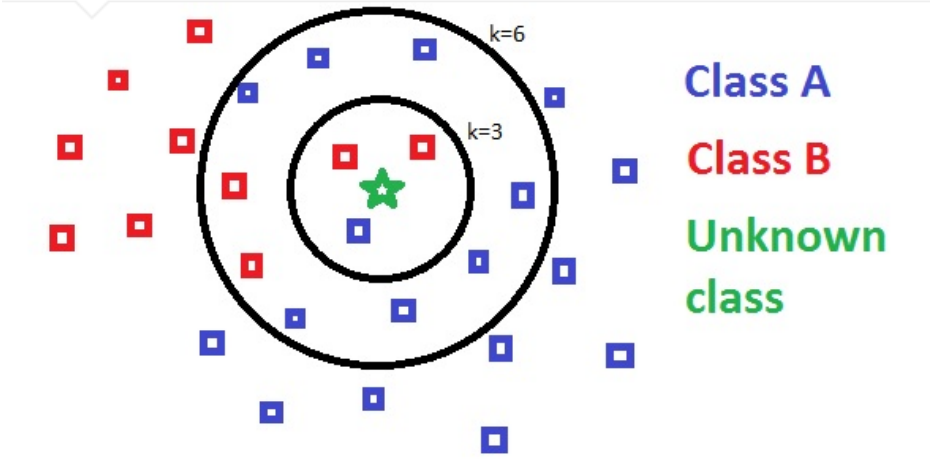


Figure 14 ตัวอย่าง KNN

Naive Bayes Classifier เป็น Model หนึ่งในการแบ่งกลุ่มที่ต้องการโดยใช้ความน่าจะเป็นที่ชื่อว่า เนอีฟเบย์ จากรูปตัวอย่างที่แสดง ตามที่ระบุไว้ วัตถุสามารถจำแนกได้เป็นสีเขียวหรือสีแดง งานของเนอีฟเบย์คือจัดประเภทเคสใหม่ ๆ เมื่อพวกเขามาถึง เช่น ตัดสินใจว่าเป็นเลเบลคลาสใด ตามออบเจกต์ที่ออกจากปัจจุบัน เนื่องจากมีวัตถุสีเขียว มากเป็นสองเท่าของสีแดง จึงมีเหตุผลที่จะเชื่อว่ากรณีใหม่ (ซึ่งยังไม่ได้รับการสังเกต) มีแนวโน้มที่จะมีสมาชิกเป็นสีเขียว มากกว่าสีแดงถึงสองเท่า ในการวิเคราะห์แบบเบย์ ความเชื่อนี้เรียกว่าความน่าจะเป็นก่อนหน้า ความน่าจะเป็นก่อนหน้านี้ขึ้นอยู่กับประสบการณ์ก่อนหน้านี้ ในกรณีนี้คือเปอร์เซ็นต์ของวัตถุสีเขียวและสีแดง และมักใช้เพื่อคาดการณ์ผลลัพธ์ก่อนที่จะเกิดขึ้นจริง

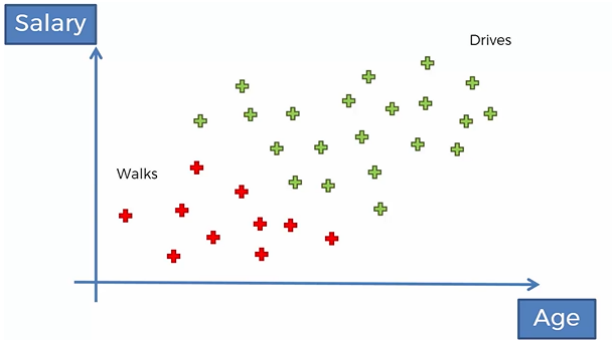


Figure 12 Naive Bayes

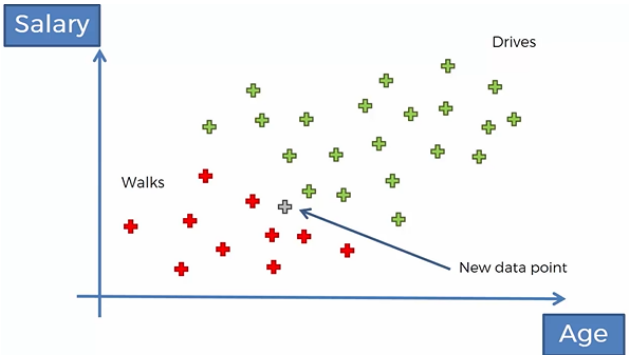


Figure 15 Naive Bayes

Logistic Regression การถดถอยโลจิสติกเป็นวิธีทางสถิติสำหรับการทำนายคลาสไบนารี ผลลัพธ์หรือตัวแปรเป้าหมายมีลักษณะเป็นสองขั้ว หมายความว่ามีเพียงสองคลาสที่เป็นไปได้ ตัวอย่างในชีวิตจริงของตัวอย่างการจัดหมวดหมู่คือการจัดหมวดหมู่อีเมลเป็นสแปมหรือไม่เป็นสแปม การจัดหมวดหมู่เนื้องอกเป็นมะเร็งหรือเป็นพิษเป็นภัย และจัดหมวดหมู่ธุรกรรมเป็นการปลอมแปลงหรือของแท้ คำตอบของปัญหาทั้งหมดเหล่านี้อยู่ในรูปแบบหมวดหมู่ นั่นคือใช่หรือไม่ใช่ การวิเคราะห์การถดถอยโลจิสติกเป็นการวิเคราะห์เพื่อทำนายโอกาสที่จะเกิด

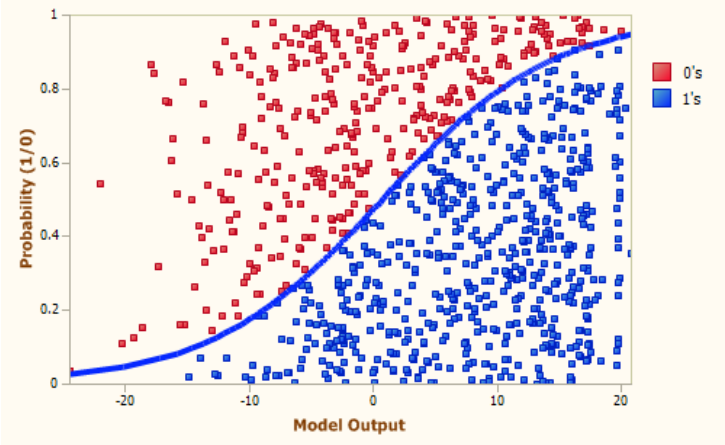


Figure 16 Logistic Regression

Support Vector Machines (SVM) เป็นอัลกอริทึมที่สามารถนำมาช่วยแก้ปัญหาการจำแนกข้อมูล ใช้ในการวิเคราะห์ข้อมูลและจำแนกข้อมูล โดยอาศัยหลักการของการหาสัมประสิทธิ์ของสมการเพื่อสร้างเส้นแบ่งแยกกลุ่มข้อมูลที่ถูกป้อนเข้าสู่กระบวนการสอนให้ระบบเรียนรู้ (Supervised) โดยเน้นไปยังเส้นแบ่งแยกแยะกลุ่มข้อมูลได้ดีที่สุด โดยการแบ่ง Class ของข้อมูลออกจากกัน ซึ่งสามารถใช้การแบ่งด้วยสมการเชิงเส้นได้ทั้ง Linear และ Non-Linear

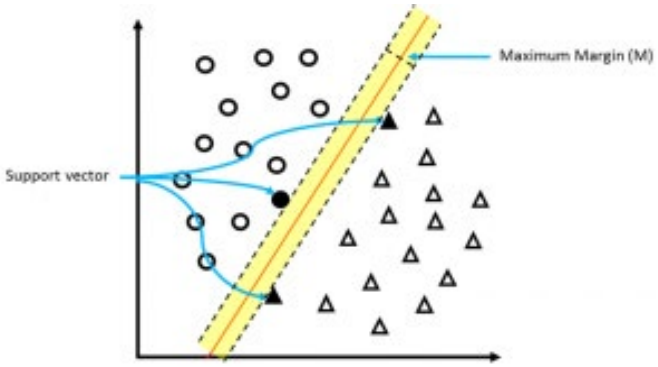


Figure 17 Support Vector Machines (SVM)

การเลือกวิธีการในการจำแนกข้อมูล

ในการเปรียบเทียบประสิทธิภาพของอัลกอริทึมนั้น ในการเลือกใช้เทคนิคอาจรู้เบื้องต้นว่า ข้อมูลที่จะนำไปประเมินเหมาะสมกับเทคนิคในการประเมินค่าความถูกต้องเทคนิคไหนเพื่อที่จะได้ผลลัพธ์ที่ดี และนำเทคนิคในการประเมินค่าความถูกต้องของข้อมูลมาใช้หลาย ๆ เทคนิคเพื่อนำมาเปรียบเทียบค่าความถูกต้องกัน ซึ่งการเปรียบเทียบด้วยเทคนิคหลาย ๆ เทคนิคจะทำให้เห็นว่าเทคนิคไหนที่มีค่าความถูกต้องและเกิดประสิทธิภาพมากที่สุดเมื่อนำไปใช้งาน และเมื่อนำไปประเมินแล้วโมเดลมีความแม่นยำมากกว่า 80% หรือ 90%+ ขึ้นไปถือว่าโมเดลใช้งานได้จริง ๆ

โครงข่ายประสาทเทียม (Artificial neural networks: ANN)

ระบบคอมพิวเตอร์จากโมเดลทางคณิตศาสตร์ เพื่อจำลองการทำงานโครงข่ายประสาทชีวภาพที่อยู่ในสมองของสิ่งมีชีวิต โครงข่ายประสาทเทียมสามารถเรียนรู้ที่จะทำงานที่มอบหมายได้ จากการเรียนรู้ผ่านตัวอย่าง โดยไม่ถูกโปรแกรมด้วยกฎเกณฑ์ตายตัวแบบระบบอัตโนมัติ ยกตัวอย่างเช่น ในการประมวลผลภาพ คอมพิวเตอร์ที่ทำงานด้วยระบบโครงข่ายประสาทเทียมจะเรียนรู้การจำแนกรูปภาพแมวได้จากการให้ตัวอย่างรูปภาพที่กำกับโดยผู้เขียนโปรแกรมว่า “เป็นแมว” หรือ “ไม่เป็นแมว” จากนั้นนำผลลัพธ์ที่ได้ไปใช้ระบุภาพแมว

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

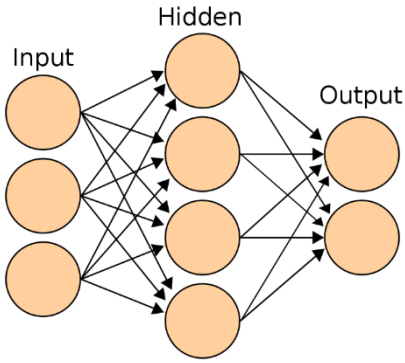


Figure 18 โครงข่ายประสาทเทียม (Artificial neural networks: ANN)

node)

ซึ่งโหนดเป็นการจำลองลักษณะการทำงานมาจากเซลล์การส่งสัญญาณ ระหว่างโหนดที่เชื่อมต่อกัน จำลองมาจากการเชื่อมต่อของใยประสาท และแกนประสาทในระบบประสาทของสมองมนุษย์ภายในโหนดจุดเชื่อมต่อแต่ละจุด มีความคล้ายคลึงกับจุดประสานประสาท (Synapses) ในสมอง มีความสามารถในการส่งสัญญาณไปยังเซลล์ประสาทเซลล์อื่น ๆ ที่เชื่อมต่อกับมัน

Node ANN

ในการสร้างระบบโครงข่ายประสาทเทียม เอาต์พุตของแต่ละเซลล์ประสาทจะมาจากการคำนวณผลรวมของอินพุต ด้วยฟังก์ชันการแปลง (transfer function) ซึ่งทำหน้าที่รวมค่าเชิงตัวเลขจากเอาต์พุตของเซลล์ประสาทเทียม แล้วทำการตัดสินใจว่าจะส่งสัญญาณเอาต์พุตออกไปในรูปใด ฟังก์ชันการแปลงอาจเป็นฟังก์ชันเส้นตรงหรือไม่ก็ได้ โครงข่ายประสาทเทียม ประกอบไปด้วย จุดเชื่อมต่อ (Connections) ซึ่งสามารถเรียกสั้น ๆ ได้ว่า เอจ (Edge), เมื่อโครงข่ายประสาทมีการเรียนรู้ จะเกิดค่าน้ำหนักขึ้น, ค่าน้ำหนัก (weights) คือ สิ่งที่ได้จากการเรียนรู้ของโครงข่ายประสาทเทียม หรือเรียกอีกอย่างหนึ่งว่า ค่าความรู้ (knowledge) ค่านี้จะถูกเก็บเป็นทักษะเพื่อใช้ในการจดจำข้อมูลอื่น ๆ ที่อยู่ในรูปแบบเดียวกัน

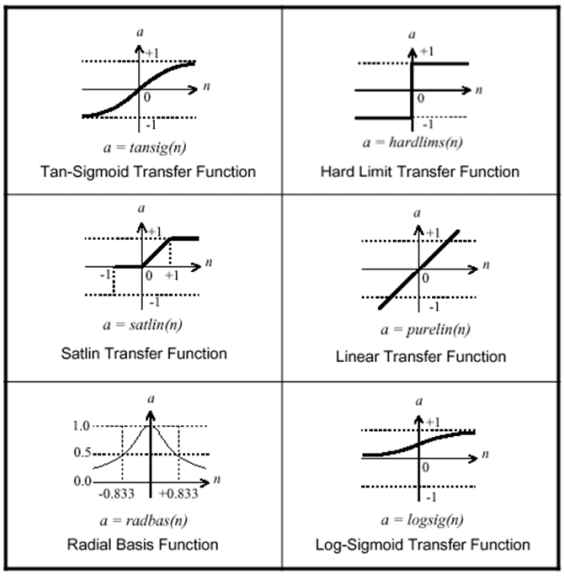


Figure 19 การคำนวณผลรวมของอินพุต ด้วยฟังก์ชันการแปลง

จุดประสงค์ของการสร้างโครงข่ายประสาทเทียม มีหลากหลายจุดประสงค์ เช่น การแก้ปัญหาแบบเดียวกับที่สมองมนุษย์สามารถทำได้ หรือ การทำงานที่เฉพาะเจาะจง, ปัจจุบัน มีการประยุกต์ใช้โครงข่ายประสาทเทียมกับงานหลากหลายรูปแบบ อาทิ คอมพิวเตอร์วิทัศน์, การรู้จำคำพูด, การแปลภาษา, การกรองเนื้อหาโซเชียลมีเดีย, การเล่นเกม, การวินิจฉัยโรค และกิจกรรมบางอย่างที่ไม่คิดว่าปัญญาประดิษฐ์จะทำได้ เช่น การวาดภาพ, การประพันธ์เพลง และ การประพันธ์บทกวี เป็นต้น

Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) เป็นเครือข่ายประสาทเทียมประเภทหนึ่งที่ใช้ในการจดจำและการประมวลผลภาพที่ออกแบบมาโดยเฉพาะเพื่อประมวลผลข้อมูลพิกเซลการวิเคราะห์รูปภาพที่มนุษย์มองเห็น โดยจะแบ่งรูปภาพออกเป็นพื้นที่ย่อยๆ เป็น Pixel แต่ละอันเพื่อทำการวิเคราะห์ Metric ของรูปภาพ โดยถ้าเป็นรูปภาพสีขาวดำ จะเป็น แบบ Matrix 2x2 ถ้าเป็นภาพสีจะเป็น Matrix 3x3

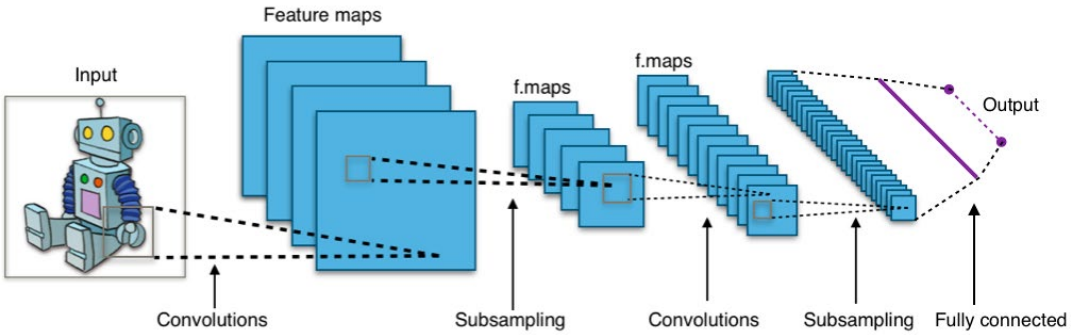
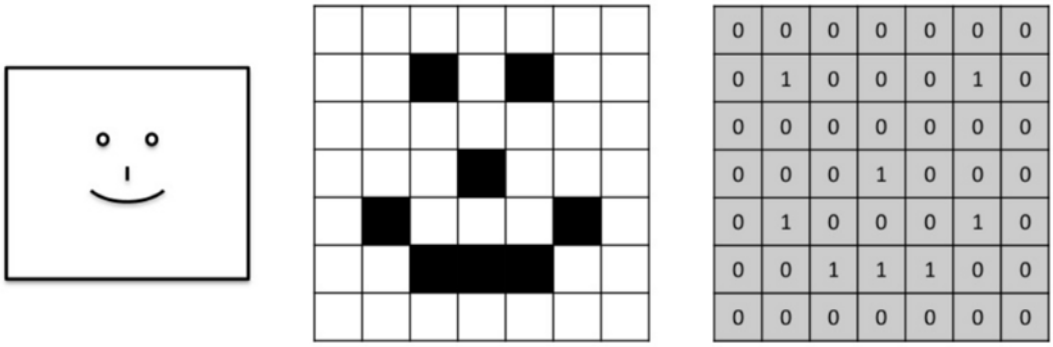


Figure 20 Convolutional Neural Network (CNN)

ตัวอย่างวิธีการแบ่งรูปภาพออกเป็น Metric



0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Figure 21 การแบ่งรูปภาพออกเป็น Matrix

ขั้นตอนการทำ CNN มี 4 ขั้นตอน

Step 1: Convolution

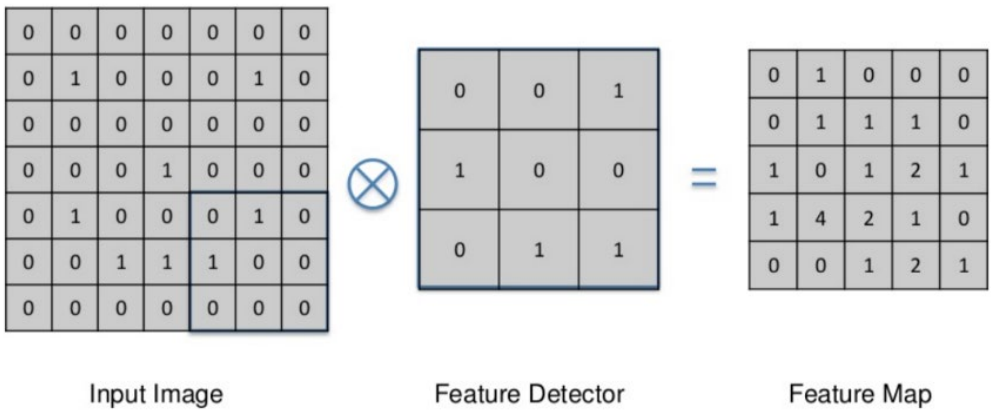


Figure 22 Convolution

การทำการคูณเมทริกซ์ ระหว่าง Input Image กับ Feature Detector ทำให้ได้ Feature Map

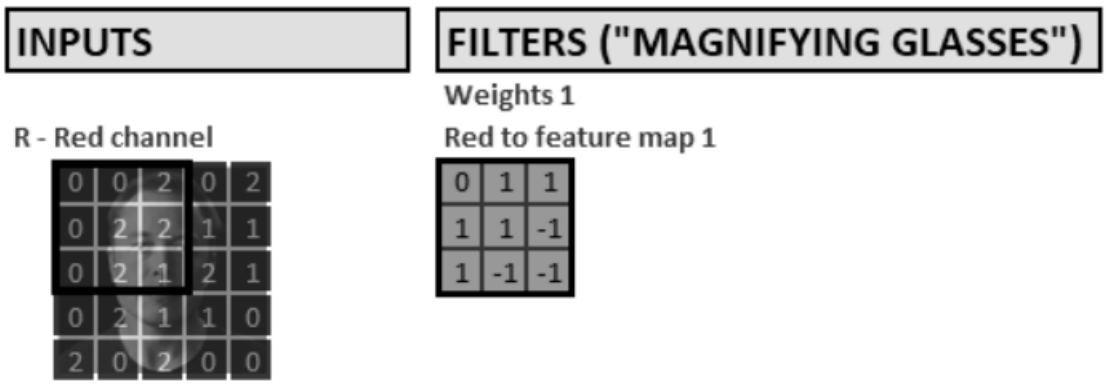


Figure 23 การทำการคูณเมทริกซ์

ขั้นตอนการสกัด Feature ทำได้โดย นำค่าที่อยู่ใน Input มาทำการคูณแบบ Matrix กับ ค่า Weight

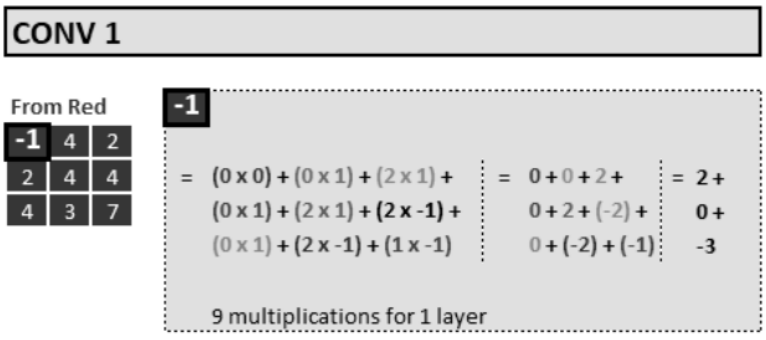


Figure 24 การสกัด Feature

Step 2: Max Pooling

Max Pooling: คือ ตัวกรอกที่ใช้ในการเลือกค่าที่มากที่สุดในบริเวณเมทริกซ์เดียวกัน

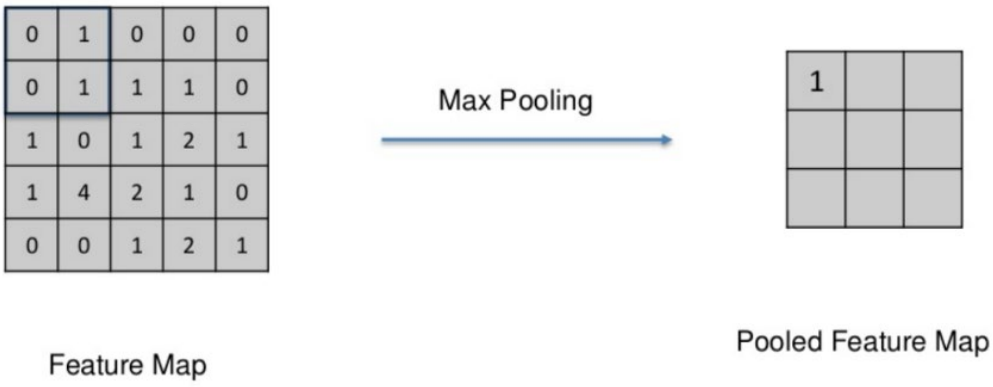


Figure 25 Max Pooling

Flattening เป็นการทำ Pooling Feature Map ที่ได้ทำเป็นคอลัมน์เดียวกัน เพื่อความสะดวกในการวิเคราะห์ข้อมูล

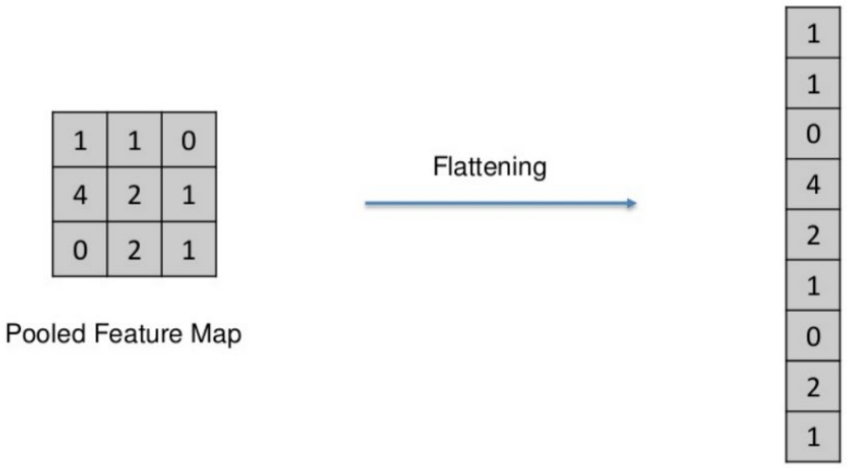


Figure 26 Flattening

Step 4: Full Connection

Full Connection คือ การนำ Flattening ที่ได้มาเข้าสู่โมเดล Deep Learning

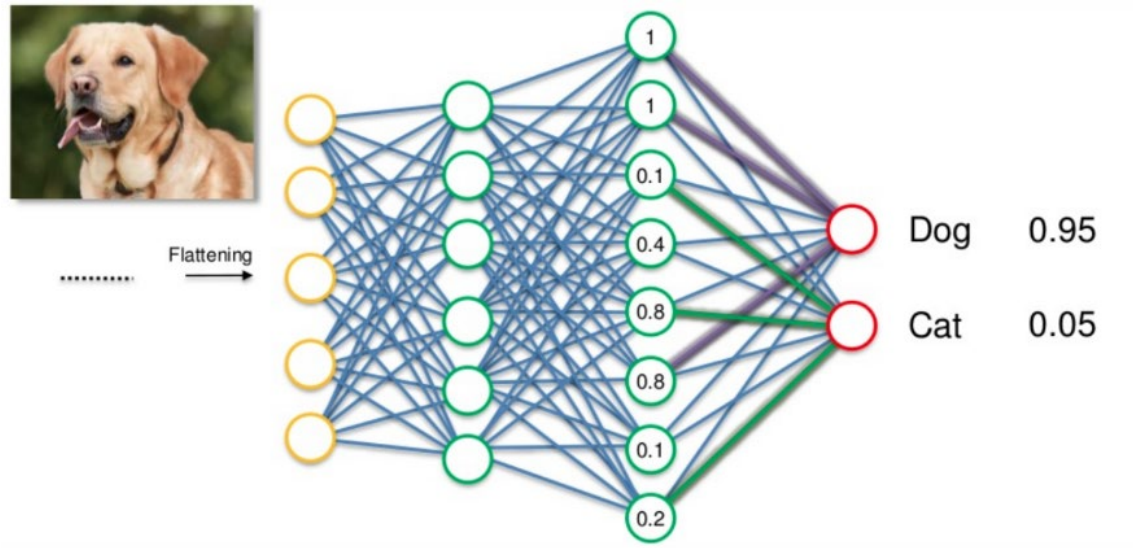


Figure 27 Full Connection

2.การจำแนกตรวจจับวัตถุในรูปภาพด้วย Yolo

Yolo คืออะไร

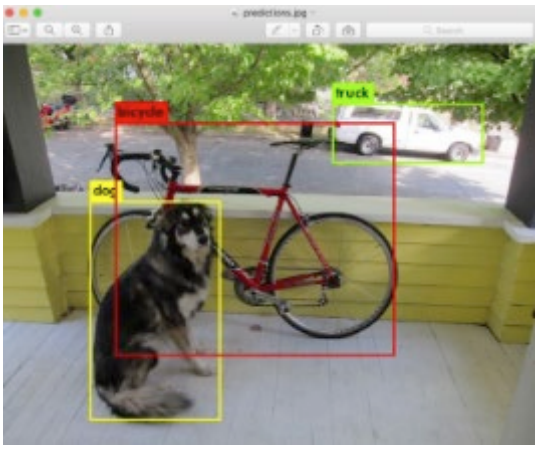


Figure 28 Fast single-shot detection

YOLO เป็นการรู้จำวัตถุที่ใช้วิธีการที่เรียกว่า “Fast single-shot detection” ซึ่งจะเป็นวิธีการที่สามารถตรวจจับวัตถุได้จากการส่งผ่านรูปภาพเข้าไปในระบบเพียงครั้งเดียว

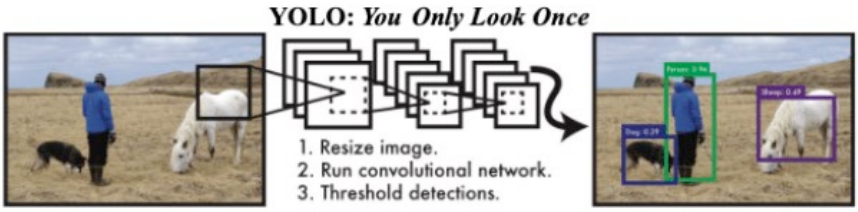


Figure 29 อัลกอริทึม YOLO

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

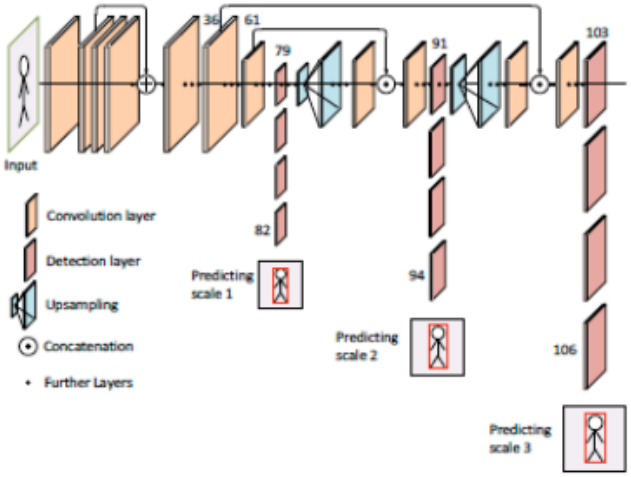


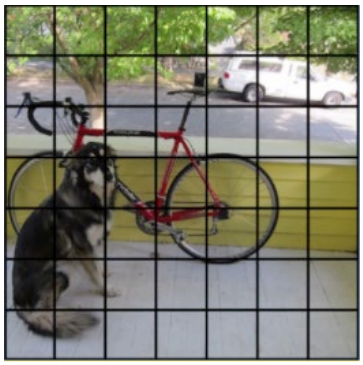
Figure 30 ลักษณะการทำงานต่าง ๆ

ลักษณะการทำงานต่าง ๆ ดังนี้ ชั้นของคอนโวลูชั่น ชั้นของการเพิ่มอัตราสุ่ม ชั้นของการทำนายผล ดัง Figure 24 ลักษณะการทำงานต่าง ๆ ซึ่งเป็นโครงสร้างสถาปัตยกรรมของอัลกอริทึม YOLO V3 ทั้งนี้ อัลกอริทึม YOLO จะมีสถาปัตยกรรมแบบ Tiny ซึ่งมีการลดทอนจำนวนชั้นบางชั้นออกไป ทำให้โครงสร้างมีความซับซ้อนน้อยกว่าอัลกอริทึม YOLO แบบสมบูรณ์ โดยส่งผลดีในด้านความเร็วในการประมวลผล แต่ความถูกต้องแม่นยำจะลดลงในงานนี้ได้เลือกใช้ Tiny YOLOv3 ซึ่งเป็นรุ่นที่มีการพัฒนาประสิทธิภาพในด้านความถูกต้องแม่นยำให้มากขึ้นจากรุ่นเดิมโดย Tiny YOLOv3 มีจำนวนชั้นของโครงสร้างเครือข่ายประสาทเทียม (Layers) จำนวน 24 ชั้น ซึ่งมีจำนวนน้อยกว่าอัลกอริทึมแบบ YOLO ประเภทสมบูรณ์ที่มีจำนวน 106 ชั้น ทำให้ลดเวลาและลดการใช้ทรัพยากรในการประมวลผลลงได้อย่างมาก

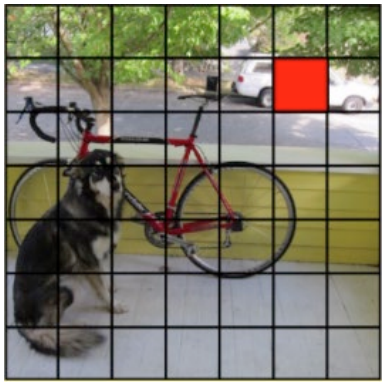
Object Detection ด้วย yolo

1.ภาพ Input เข้ามา

2.ทำการตัดภาพเป็นชิ้นสี่เหลี่ยม

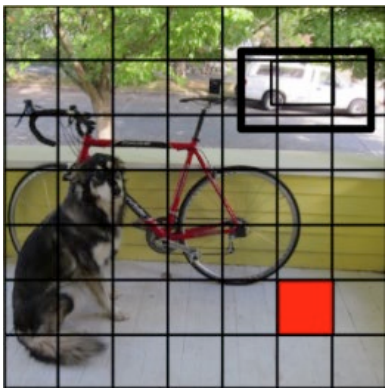


3.ชิ้นสี่เหลี่ยมที่ตัดออกมาจะทำการทำนายค่าของสี่เหลียมออกมาให้ค่าความเชื่อมั่นสิ่งที่จะเป็น

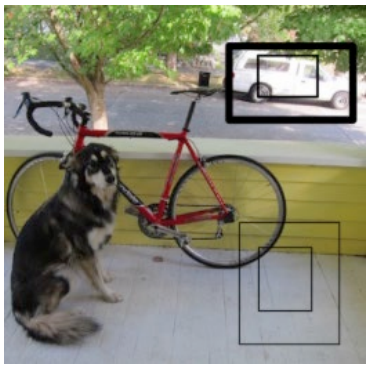


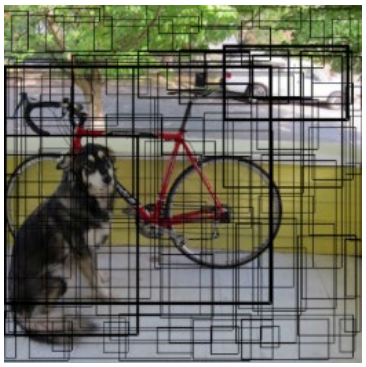
.ทำการตีกรอบสิ่งที่ตรวจจับได้

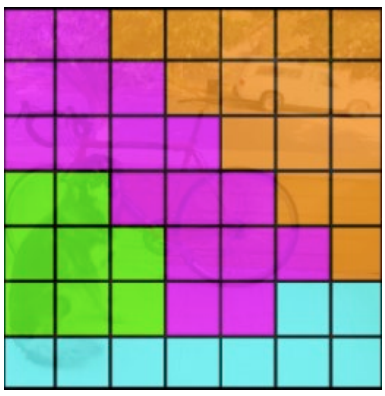
6.ภาพแสดงถึงคลาสที่ทำนายได้กับกรอบสี่เหลียมที่เจอจากกระบวนการคอนโวลูชั่น



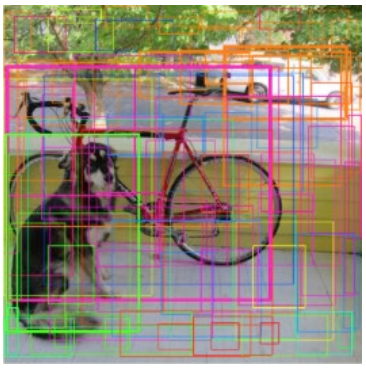
7.ความหนาของสี่เหลี่ยมแสดงถึงค่าของความเชื่อมั่นที่ตรวจเจอ



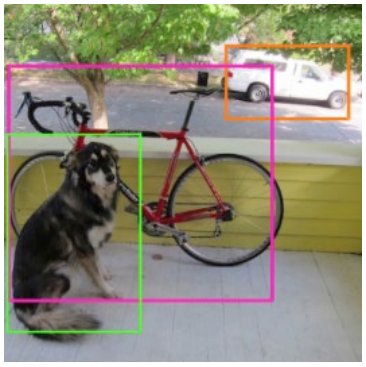




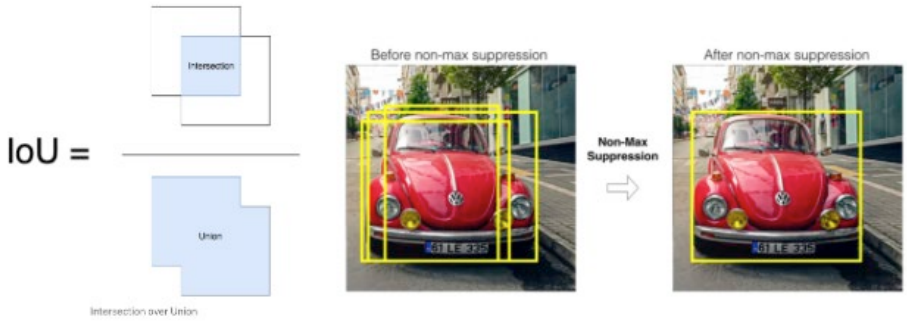
10.กระบวนการจากขั้นตอนที่ 8 กับขั้นตอนที่ 9 มารวมกัน



11.จากขั้นตอนที่ 10 ผ่านกระบวนการ Non-maximum Suppression จะได้ผลลัพธ์ดังภาพ



Non-maximum Suppression



กระบวนการ Intersection over Union (IoU)

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

ภาพรวมขั้นตอนของการ Object Detection ด้วย Yolo

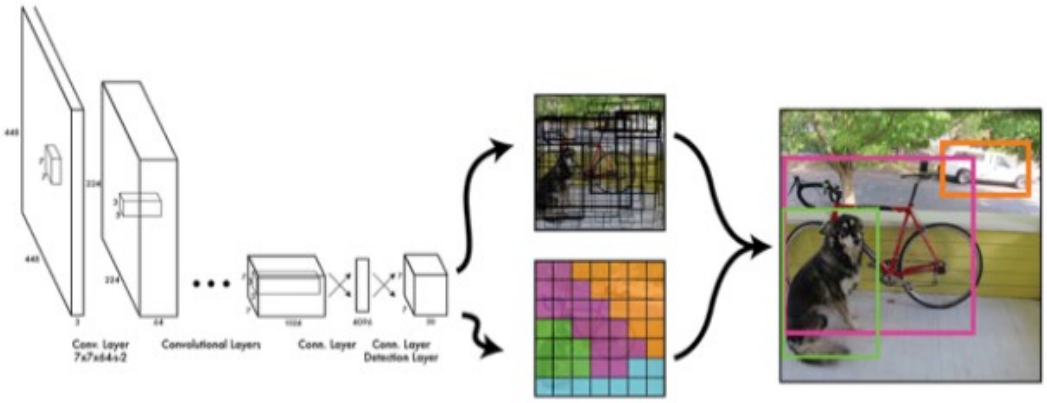
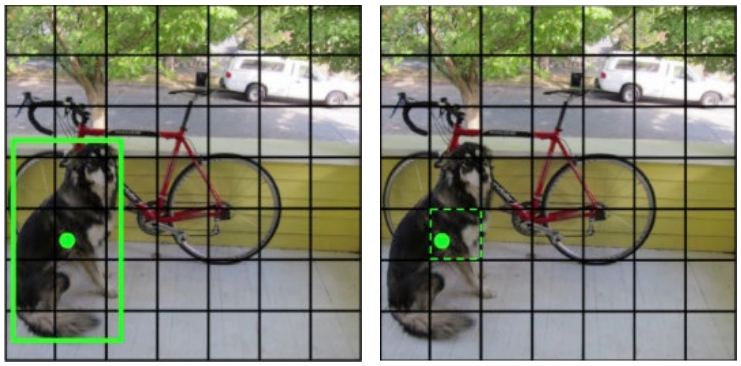


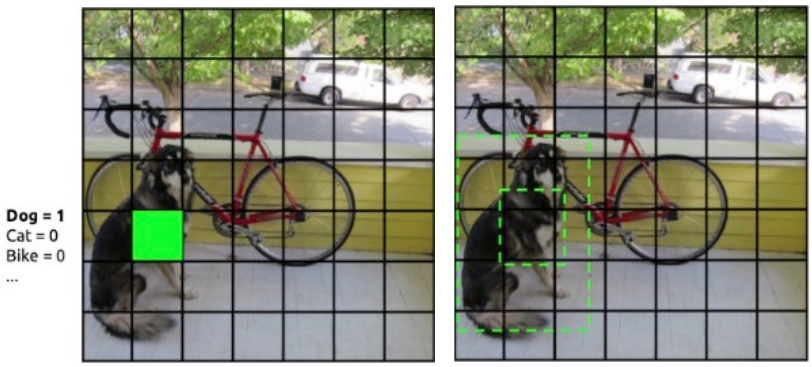
Figure 32 ภาพรวมขั้นตอนของการ Object Detection ด้วย Yolo

เมื่อมีรูปภาพเข้ามาก็จะผ่านกระบวนการโครงข่ายประสาทเทียมแบบคอนโวลูชั่นซึ่งจะผ่านชั้นต่าง ๆ ที่เป็นอัลกอริทึมของ yolo แล้วจากนั้นจะทำการแบ่งภาพออกเป็น 2 ส่วนคือส่วนแรกคือส่วนที่อัลกอรึทึมตรวจพบวัตถุละตีกรอบสี่เหลี่ยมไว้ตามค่าความเชื่อมันที่เจอสัมพันธ์กับความหนาของกรอบสี่เหลียม ส่วนที่สองคือการตัดภาพออกเป็นสี่เหลี่ยมจัตุรัสจากนั้นกำหนดคลาสเอาไว้ตามตำแหน่งต่าง ๆ เมื่อกำหนดเสร็จสิ้นจะนำส่วนแรกและส่วนที่สองมารวมเข้าด้วยกันทั้งสองส่วน ผลลัพธ์ที่ได้ส่วนแรกนั้นกรอบสี่เหลี่ยมแต่ละอันก็จะมีคลาสของตัวเอง ขั้นตอนถัดมาก็จะต้องผ่านกระบวนการที่เรียกว่า Non-maximum Suppression ก็จะได้ผลลัพธ์ที่ต้องการ

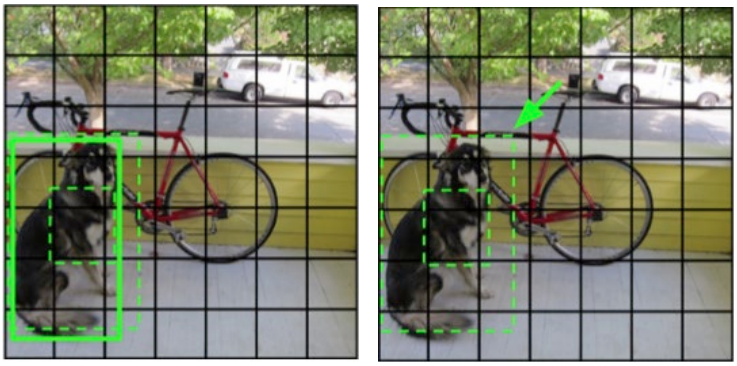
1.ในระหว่างขั้นตอนการฝึกฝน



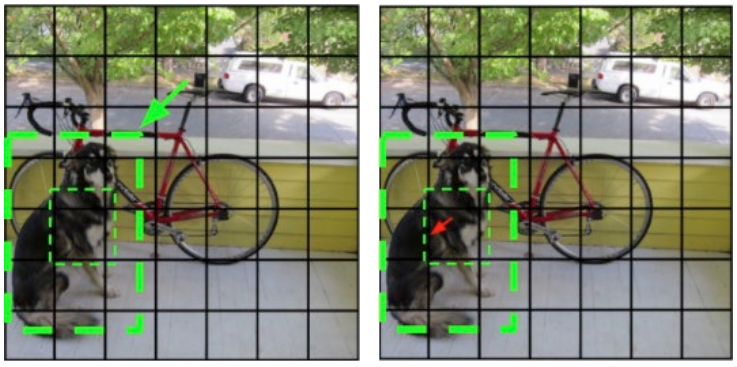
2.ระบบจะทำการเทียบค่าความเชื่อมั่นในช่องนั้น ๆ จากนั้นจะนับช่องรอบ ๆ เพื่อหากรอบของ object โดยที่เมื่อได้กรอบแล้วจะเช็คดูว่าค่าความเชื่อมั่นมากกว่าช่อง หรือ กรอบในนั้นไหม ถ้าใช้ก็เพิ่มค่าความเชื่อมั่นกับกรอบปัจจุบัน จากนั้นลบกรอบด้านในออก



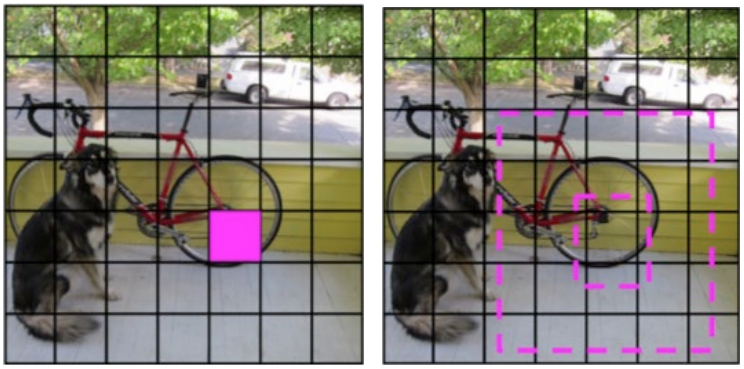
3.เมื่อทำการให้ค่าความเชื่อมั่น จะได้กรอบรอบ ๆ object ออกมาในทุกครั้งก็จะมีกรอบนั้นทับซ้อน



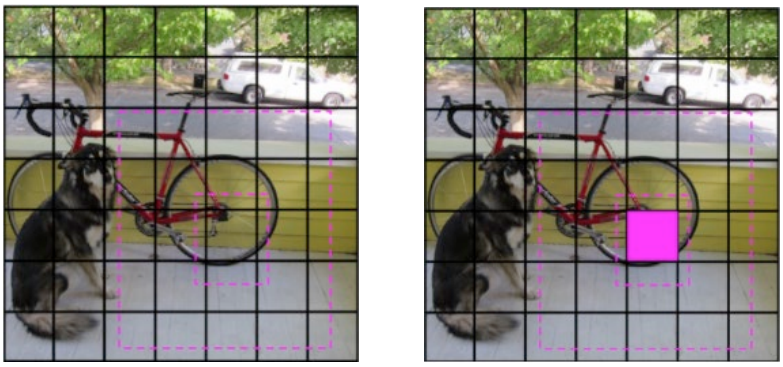
.ขั้นตอนนี้จะทำการเช็คกันระหว่างกรอบที่มีอยู่โดยเพิ่มค่าความเชื่อมั่นของกรอบที่ดีที่สุดแล้วลดของกรอบที่น้อยกว่า



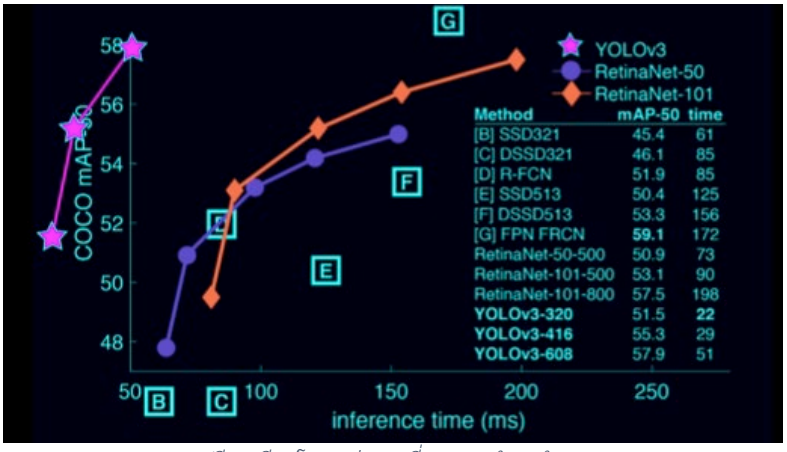
5.ในบางจุดที่ไม่มีค่าอะไรหรือไม่สามารถบอกได้ว่ามันคืออะไรนั้น ระบบก็จะทำการลดค่าความเชื่อมั่นไปเรื่อย ๆ



6.ทำการลดไปเรื่อย ๆ จนถึงขั้นระบุว่าไม่มี class นั้นใน coco data set ที่มีอยู่



Yolo กับรูปแบบการตรวจจับกับอัลกอริทึมอื่น ๆ



การเปรียบเทียบโมเดลต่าง ๆ ที่สามารถนำมาทำ object detection

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

ตัวอย่างการนำไปใช้งาน

1.การตรวจจับหมวกนิรภัยและการใช้อาวุธปืน เพื่อเตือนภัยเหตุโจรกรรม จากภาพกล้องวงจรปิดแบบเวลาจริง[1]

Figure 34 การตรวจจับหมวกนิรภัยและการใช้อาวุธปืน เพื่อเตือนภัยเหตุโจรกรรม

2.การสร้างระบบตรวจนับบุคคลแบบเวลาจริงราคาประหยัด บน Raspberry Pi โดยประยุกต์อัลกอริทึม Tiny YOLO V3[2]

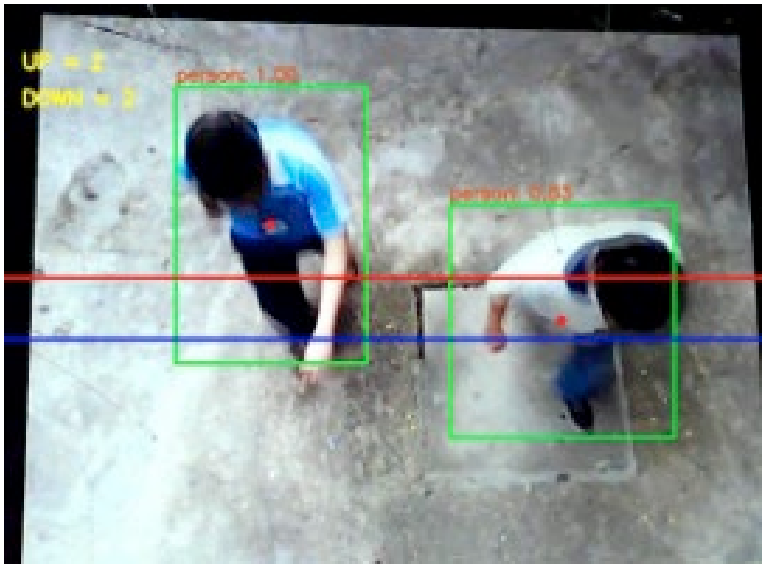


Figure 35 การสร้างระบบตรวจนับบุคคลแบบเวลาจริง

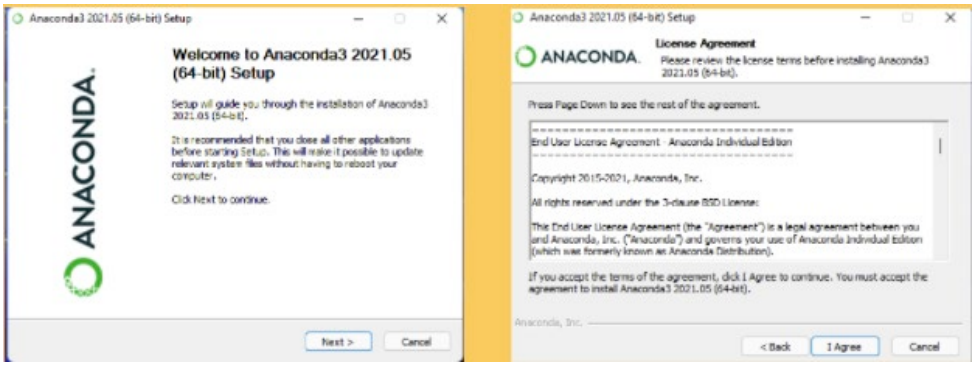
[1] งานวิจัย คุณยงยุทธ ละมูลมอญ และคุณธนาสัย สุคนธ์พันธ์

[2] งานวิจัย คุณปราโมทย์ ปัญญาโต และคุณนลิน สีดาห้าว

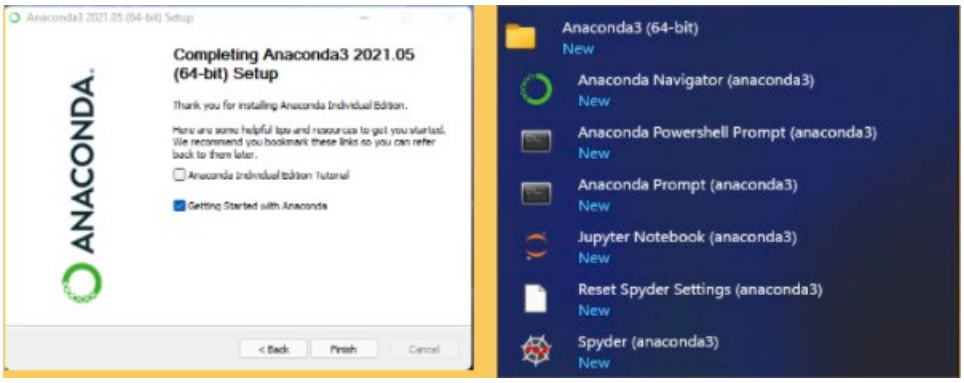
การติดตั้ง Anaconda

1. ให้ Download Anaconda จากลิงก์: https://www.anaconda.com/products/individual#Downloads

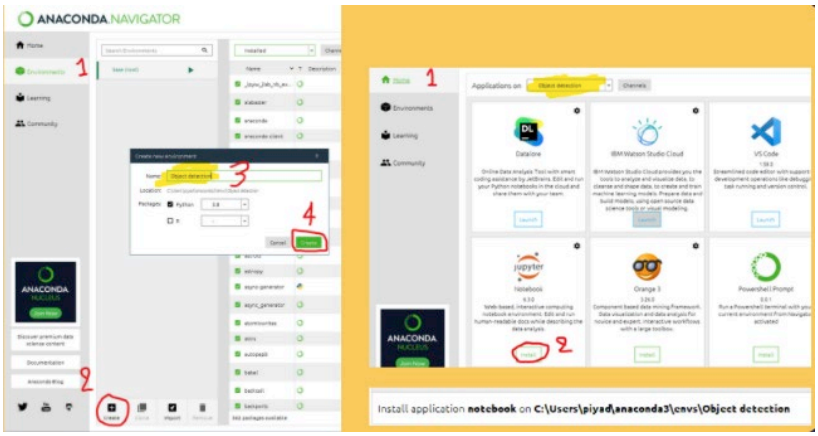
2. กด install ตามภาพ



ติดตั้งเสร็จสิ้น

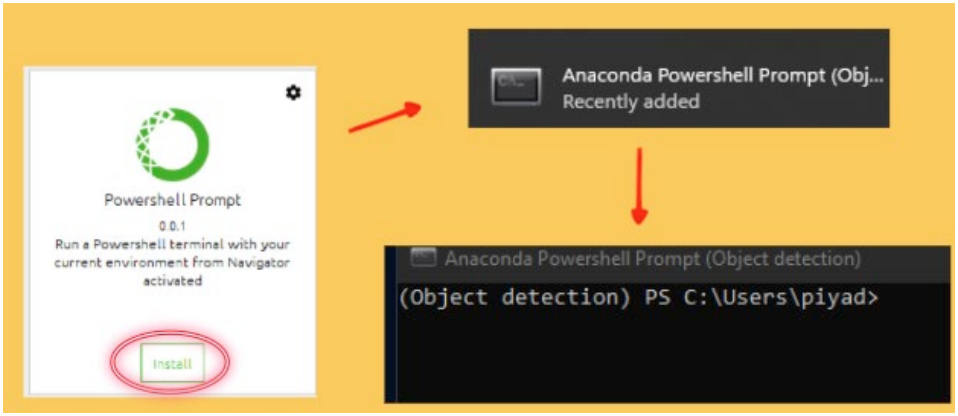


4. สร้าง Envoriment สำหรับการใช้งาน

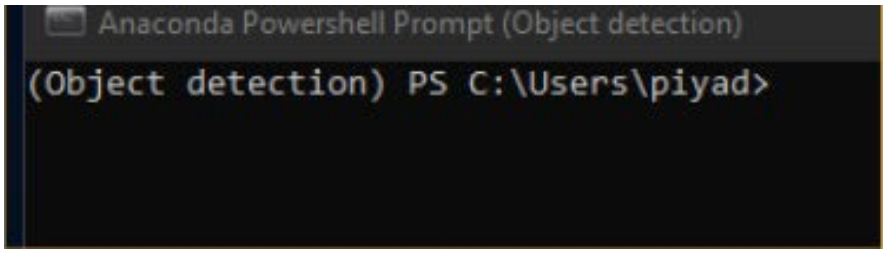


ติดตั้ง PowerShell Prompt

1. กดปุ่ม install ตามภาพด้านล่าง

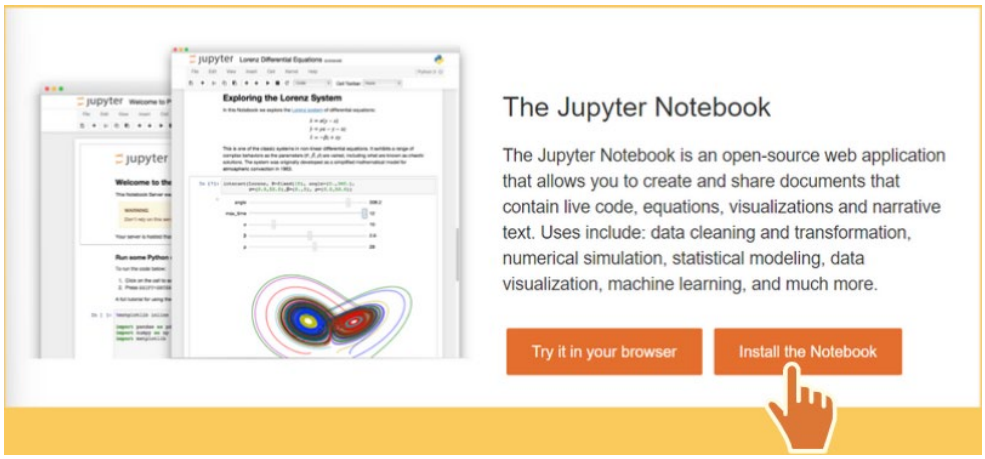


2. หน้าตาของโปรแกรมที่จะแสดงออกมารูปแบบนี้

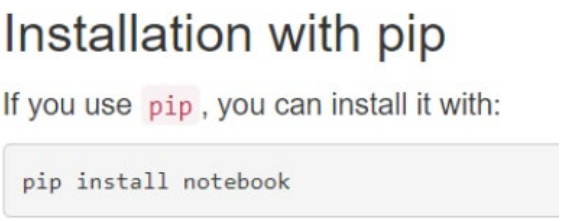


Jupyter

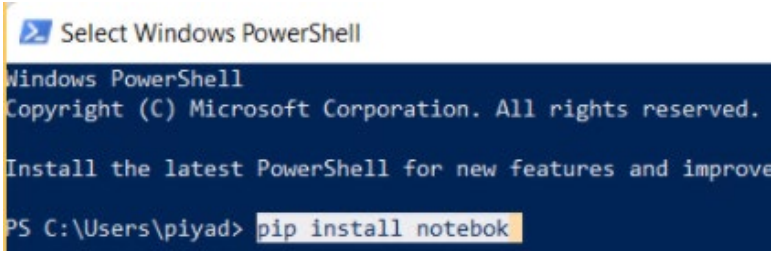
1. กดที่ install the Notebook (กรณีไม่สามารถติดตั้งจาก Anaconda ได้)



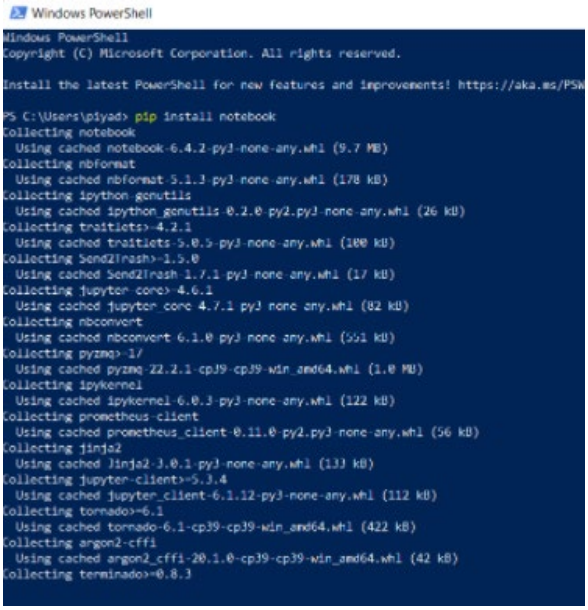
คัดลอกคำสั่งที่จะถูกใช้ในการติดตั้ง “pip install notebook”\



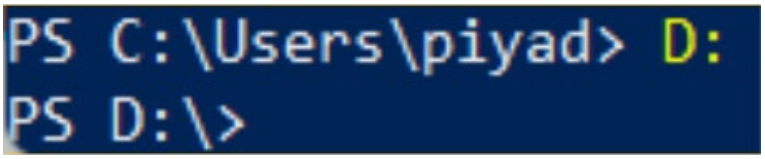
นำไปพิมพ์ลงใน powershell ที่เปิดเอาไว้



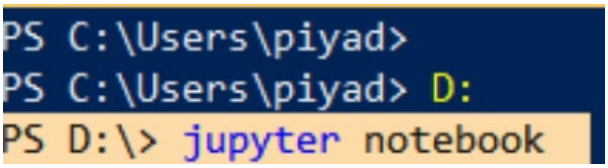
4. รอให้ระบบทำการติดตั้ง



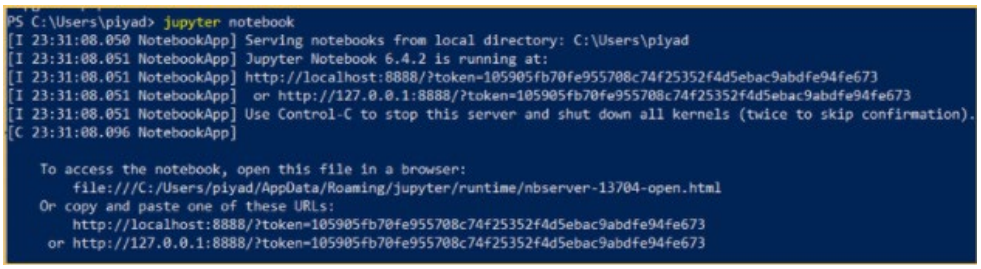
5. เลือกตำแหน่งที่ต้องการเปิด notebook ไว้ใช้งาน



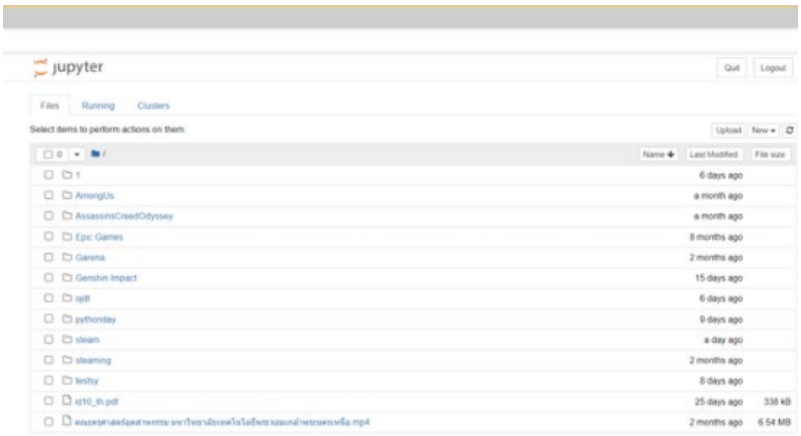
ใช้คำสั่งในการเรียกใช้งานตัว jupyter notebook



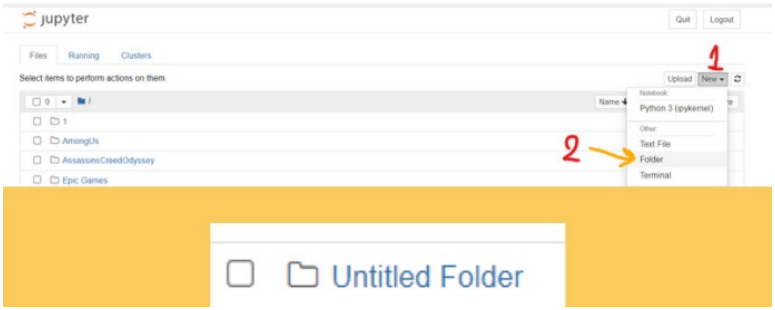
รอให้ระบบทำการเรียกการใช้งาน Jupyter notebook



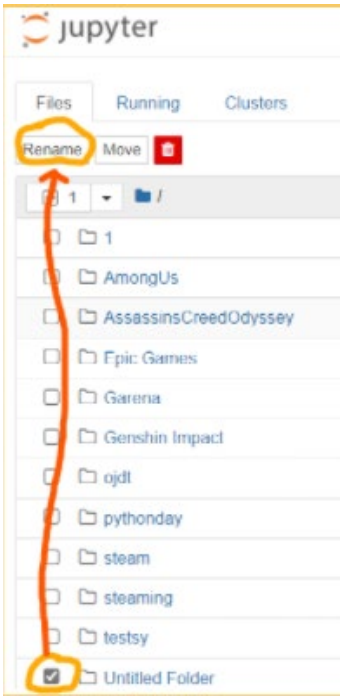
8. หน้าตาของโปรแกรมจะถูกแสดงทาง Browser ของเครื่อง



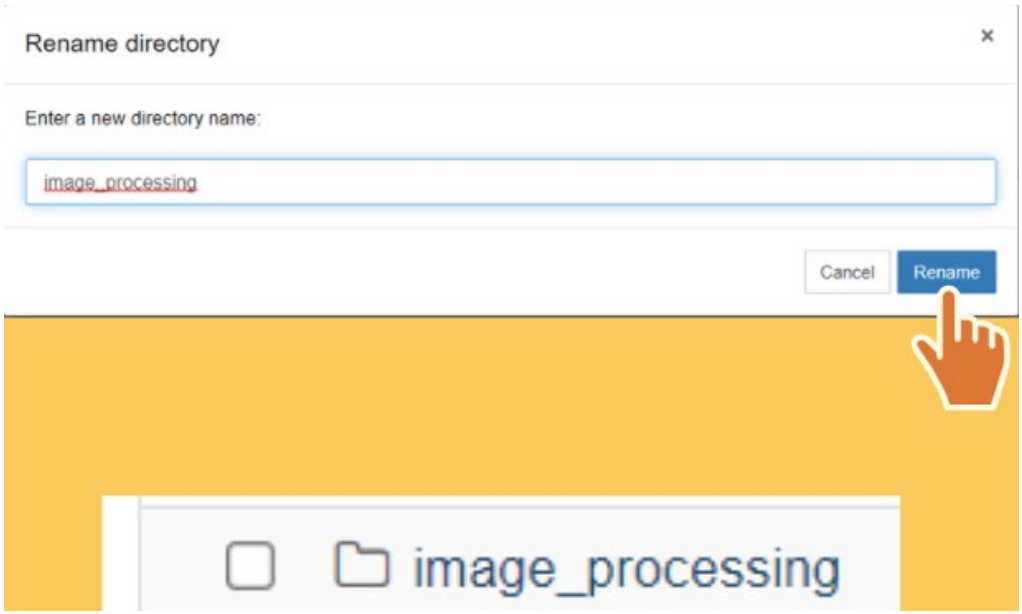
9. การสร้าง Folder สำหรับพื้นที่งานที่จะใช้งาน



การเปลี่ยนชื่อ Folder ที่ถูกสร้างขึ้นมา

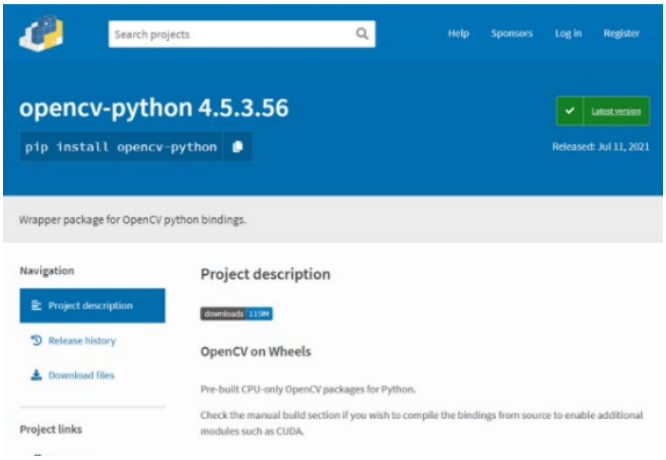


11. ตั้งชื่อแล้วกด Rename

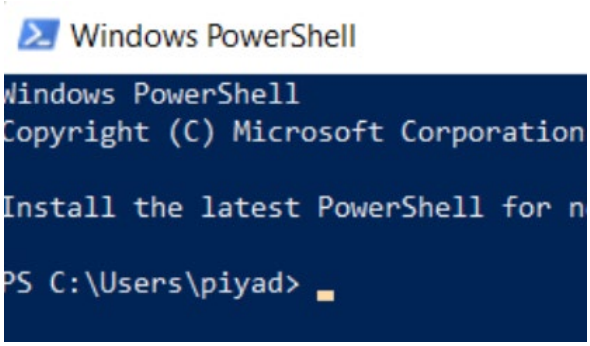


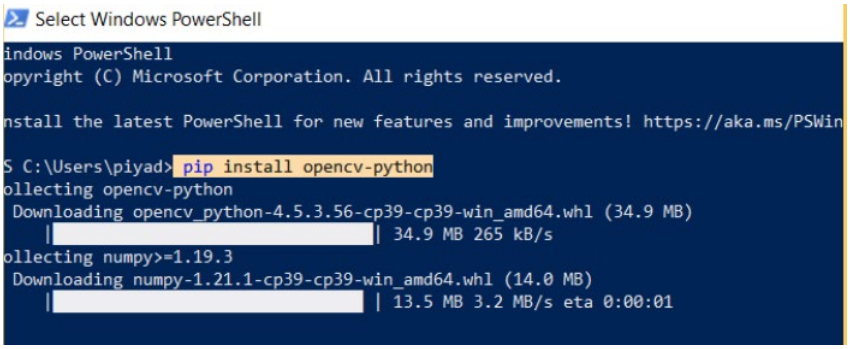
Opencv

1. ทำการติดตั้งตัว Opencv



ทำการเปิด powershell เพิ่มขึ้นมาอีก 1 หน้าต่าง





NUMPY

1. ติดตั้ง NUMPYโดยใช้ 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
```

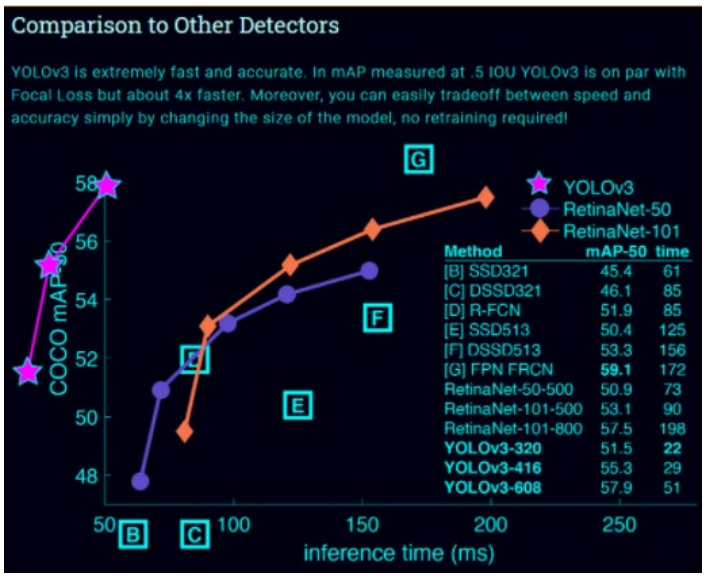
PIP

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

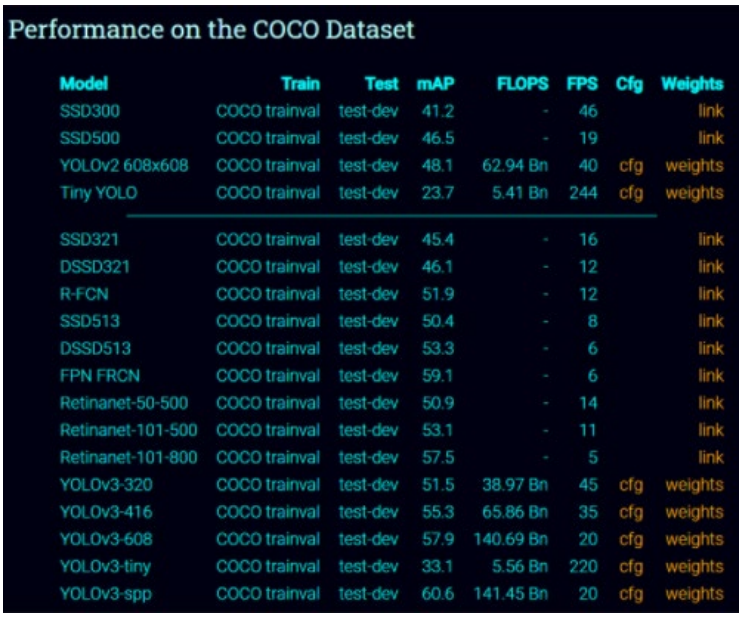
```
pip install numpy
```

YOLO

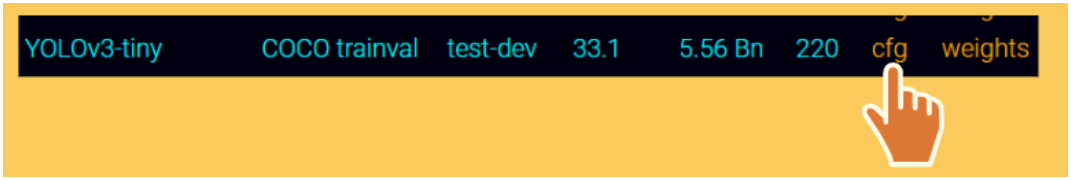
1. พิจารณาว่าจะใช้ YOLO ตัวไหนในการทำงาน (ในที่นี้ YOLO-tiny)



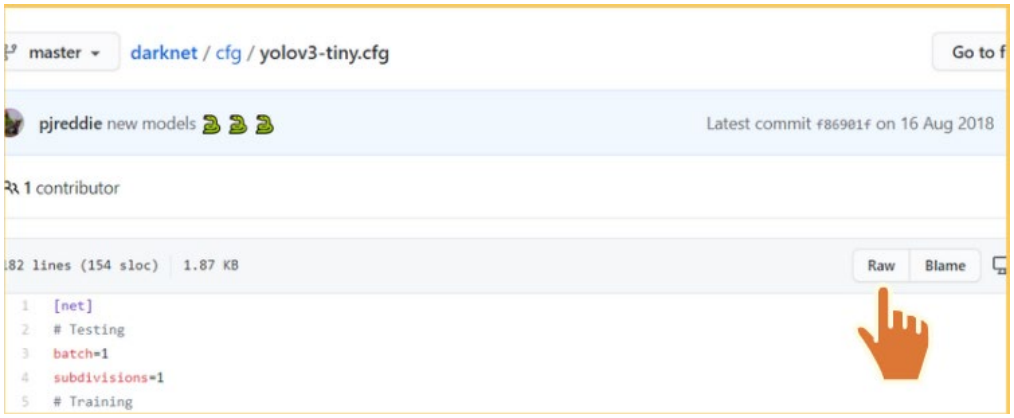
เลือกตัวที่เราต้องการใช้งานจากเว็บ



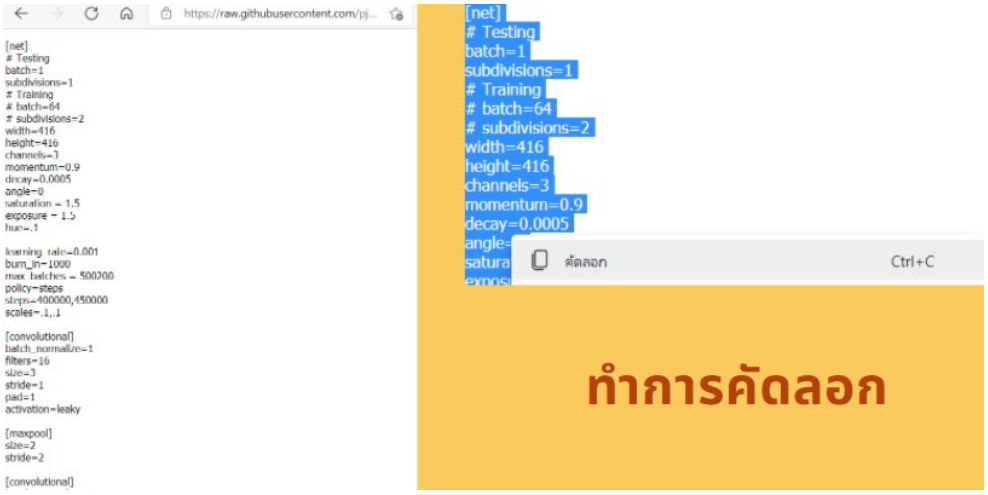
3. เลือก download ไฟล์ cfg



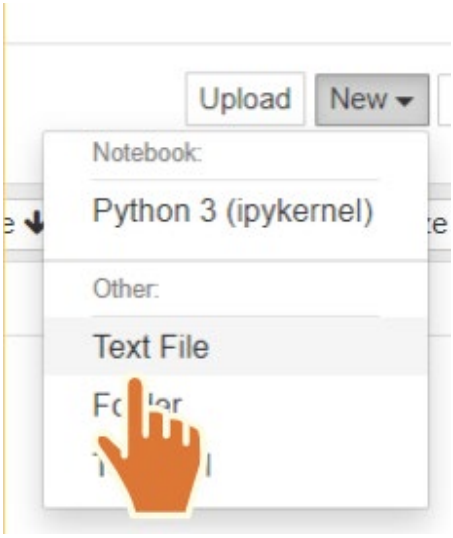
4. เลือกที่ RAW



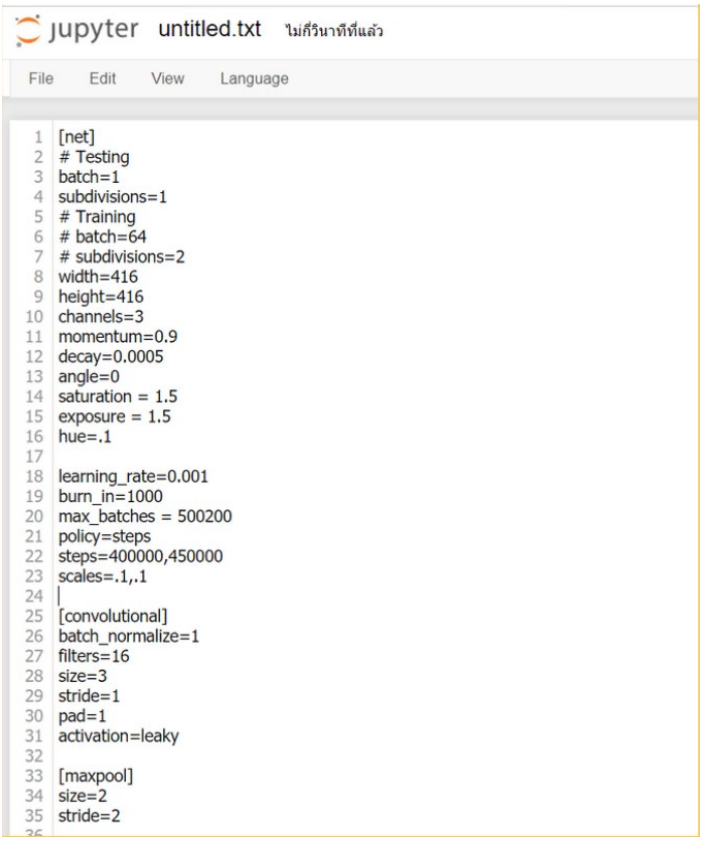
คัดลอกข้อความภายในทั้งหมด



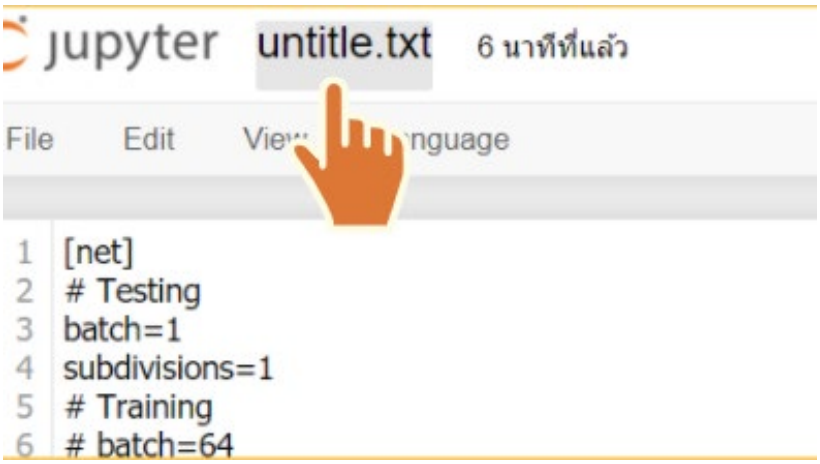
6. กลับมาที่ jupyter book แล้วทำการสร้าง text file



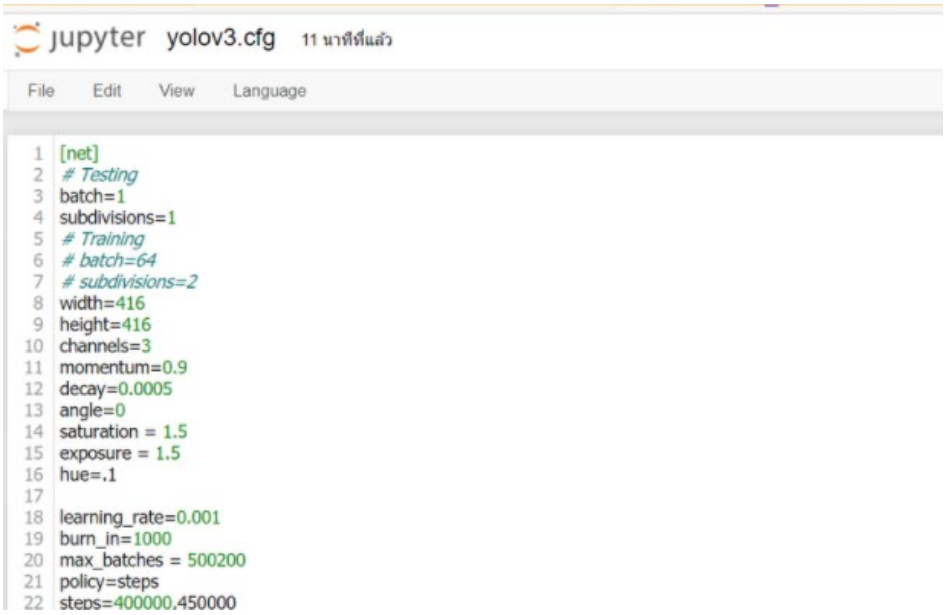
7. นำมาวางไว้ที่ text file ที่สร้างเอาไว้



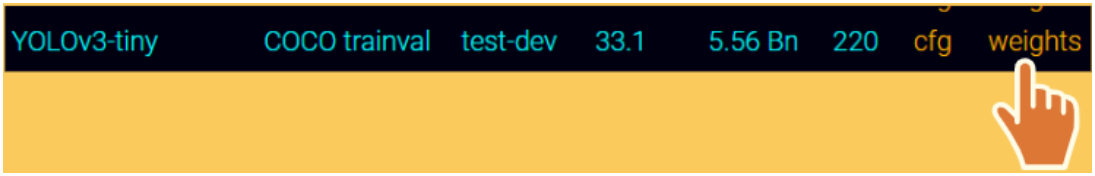
8. แก้ไขชื่อไพล์เป็น yolov3.cfg



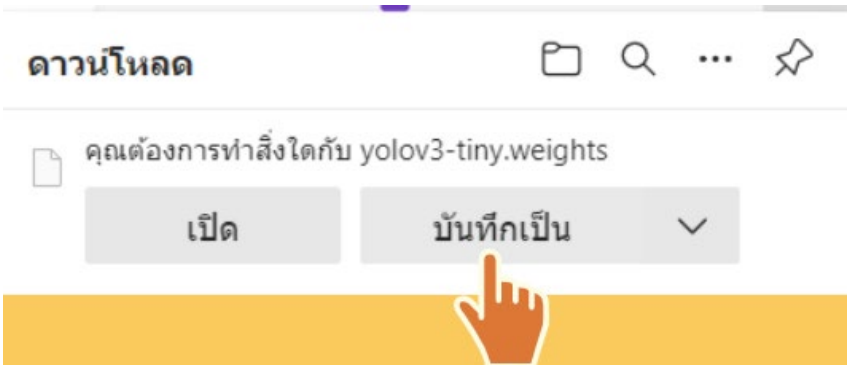
9. หน้าตาของไพล์ที่จะได้ออกมา



download ไพล์ weights



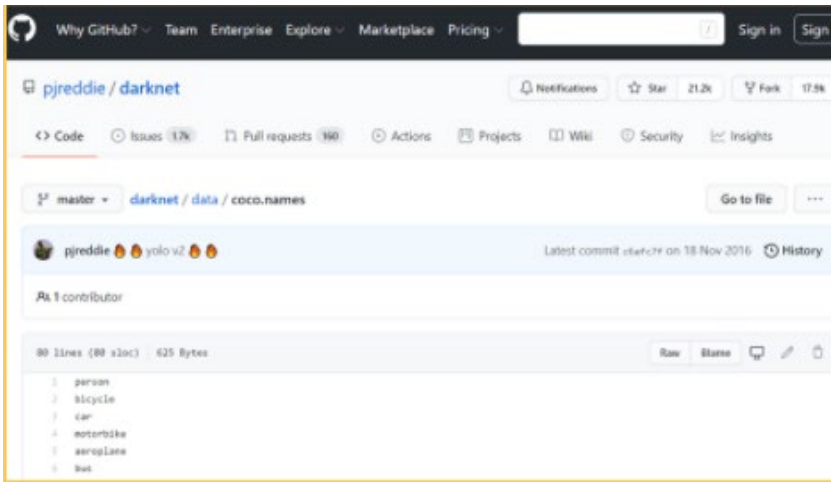
11. กด บันทึกเป็น หรือ save as



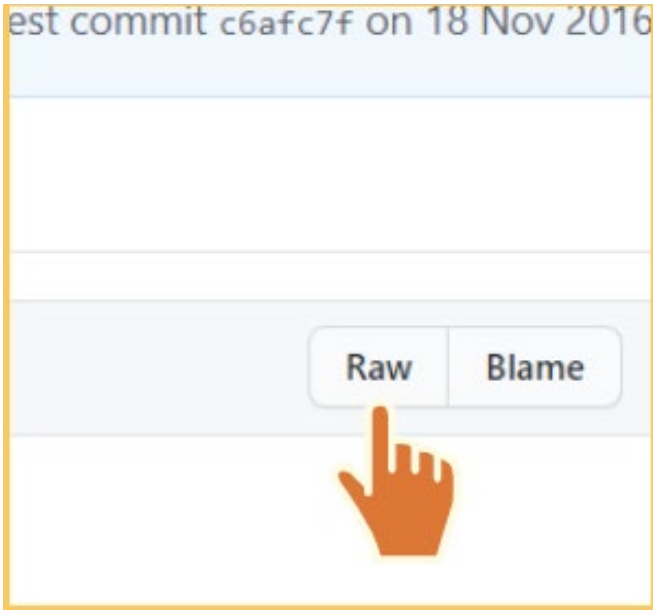
12. ไพล์ที่จะต้องมีในตอนนี้

COCO

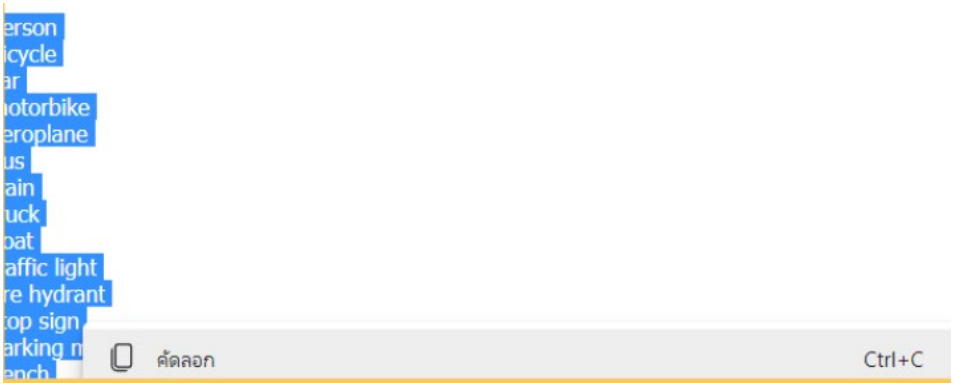
1. สามารถเข้าไปได้ที่ https://github.com/pjreddie/darknet/blob/master/data/coco.names



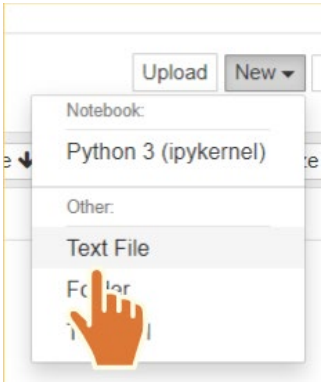
download มาด้วยการกดที่ปุ่ม RAW

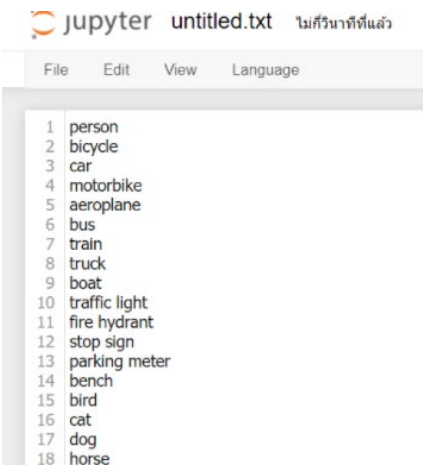


3. คัดลอกจาก notepad

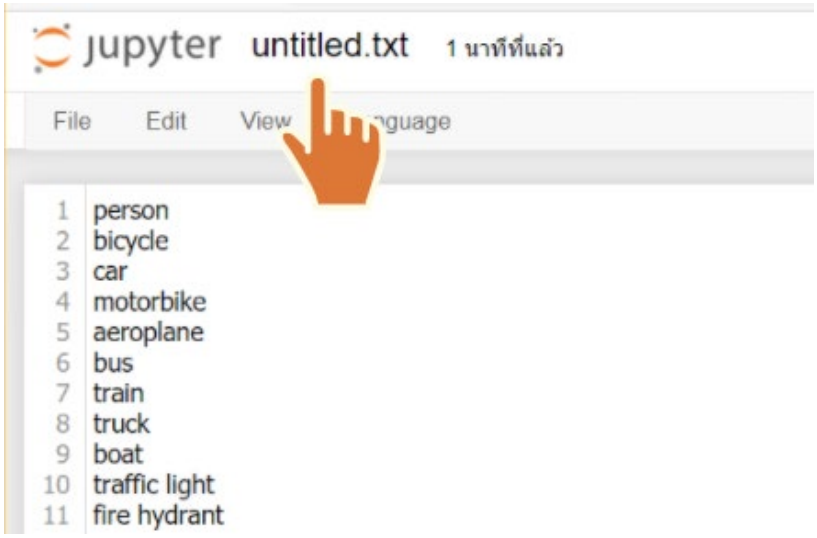


4. สร้างไพล์ text file ขึ้นมา

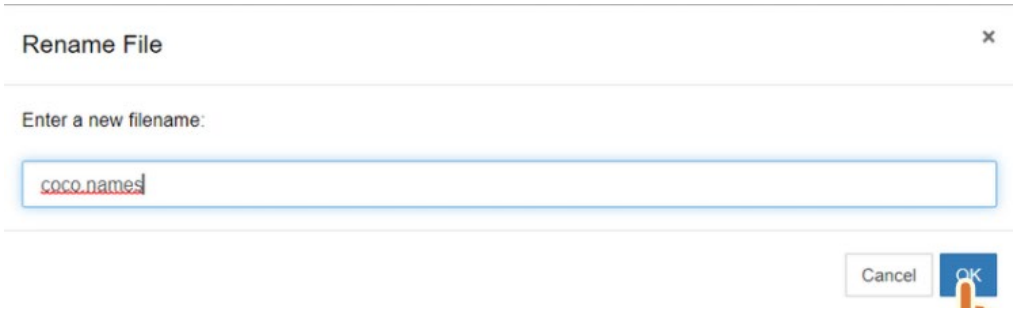




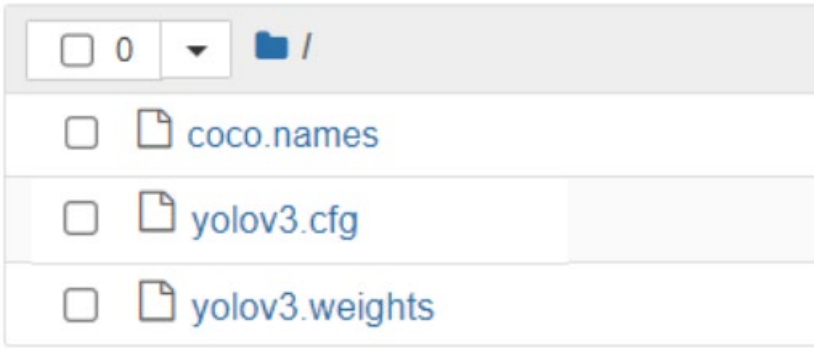
6. เปลี่ยนชื่อ ไพล์ เป็น coco.names



7. คลิ๊ก OK

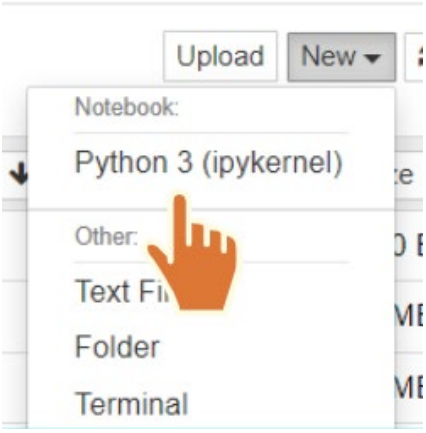


ไพล์ที่เราต้องมีในตอนนี้

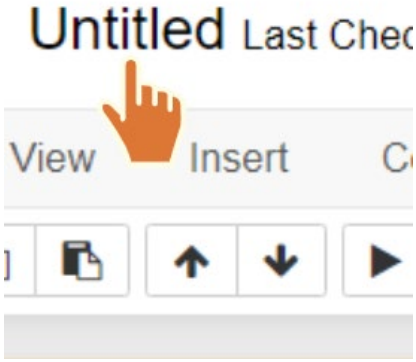


9. ไฟล์ภาพที่ได้ทำการเตรียมไว้ให้ผู้รวมอบรมสามารถนำรูปอื่นมาใช้งานได้

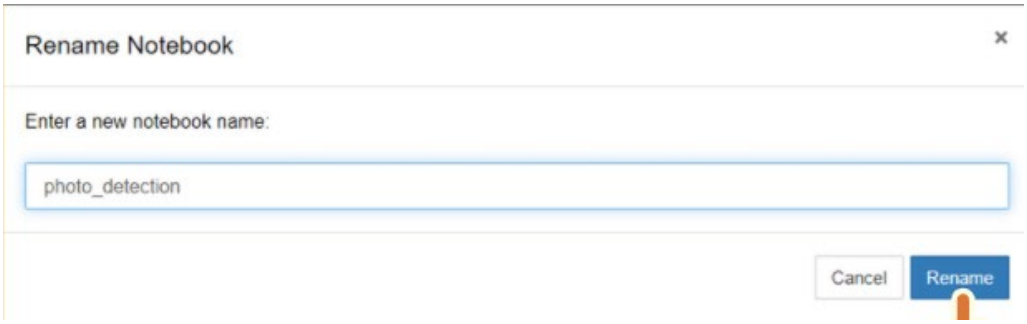
10. ไฟล์ที่จะมีอยู่



12. กดที่ชื่อของไฟล์เพื่อเปลี่ยนชื่อ

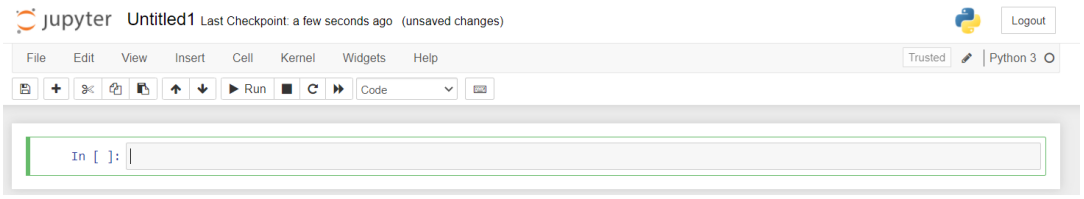


13. หลังจากตั้งชื่อแล้วให้ทำการ กดที่ rename



เริ่มการเขียนโค้ด

1. ไปที่หน้าต่างไฟล์ Python ที่สร้างขึ้นมา



2. นำเข้า Library ต่าง ๆ

```
import numpy as np


import time
```

3. ทำการสร้างตัวแปลเพื่อเก็บค่าที่จะถูกนำเข้ามา

```
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 โดยให้เรียงเป็นบรรทัดเอาไว้

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

5. คำสั่งให้สร้างค่า seed ที่ถูกสุ่มขึ้นมาจากนั้นนำไปสุ่มสีที่แตกต่างกัน สำหรับนำไปแสดงเป็นกรอบรอบชิ้นวัตถุที่ถูก Detection

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

6. คำสั่งให้สร้างตัวแปร net ขึ้นมาเพื่อนำค่า CONFIG และ WEIGHT ส่งไปยัง เซิร์ฟเวอร์ของ Darknet เพื่อนำมาคำนวณในโปรแกรมต่อไป

```
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

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

9. สร้างตัวแปร blob นำไปลดขนาดรูปภาพที่นำเข้ามาทดสอบให้เหลือ 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(ln)
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. คำสั่งวนซ้ำแต่ละเอาต์พุตของเลเยอร์

```
for output in layerOutputs:
```

13. คำสั่งวนซ้ำการตรวจจับแต่ละครั้ง

```
for detection in output:
```

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

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

15. หากค่า confidence ที่คาดการณ์ว่าตรวจจับ มีค่ามากกว่าเกณฑ์ความน่าจะเป็นวัตถุในภาพ

```
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. คำสั่งวนรอบซ้ำดัชนีที่เราเก็บไว้ใน 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])
```

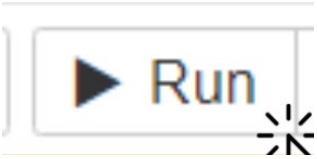
23. สร้างกล่องสีที่เป็น output ออกมาที่ภาพ

```
            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

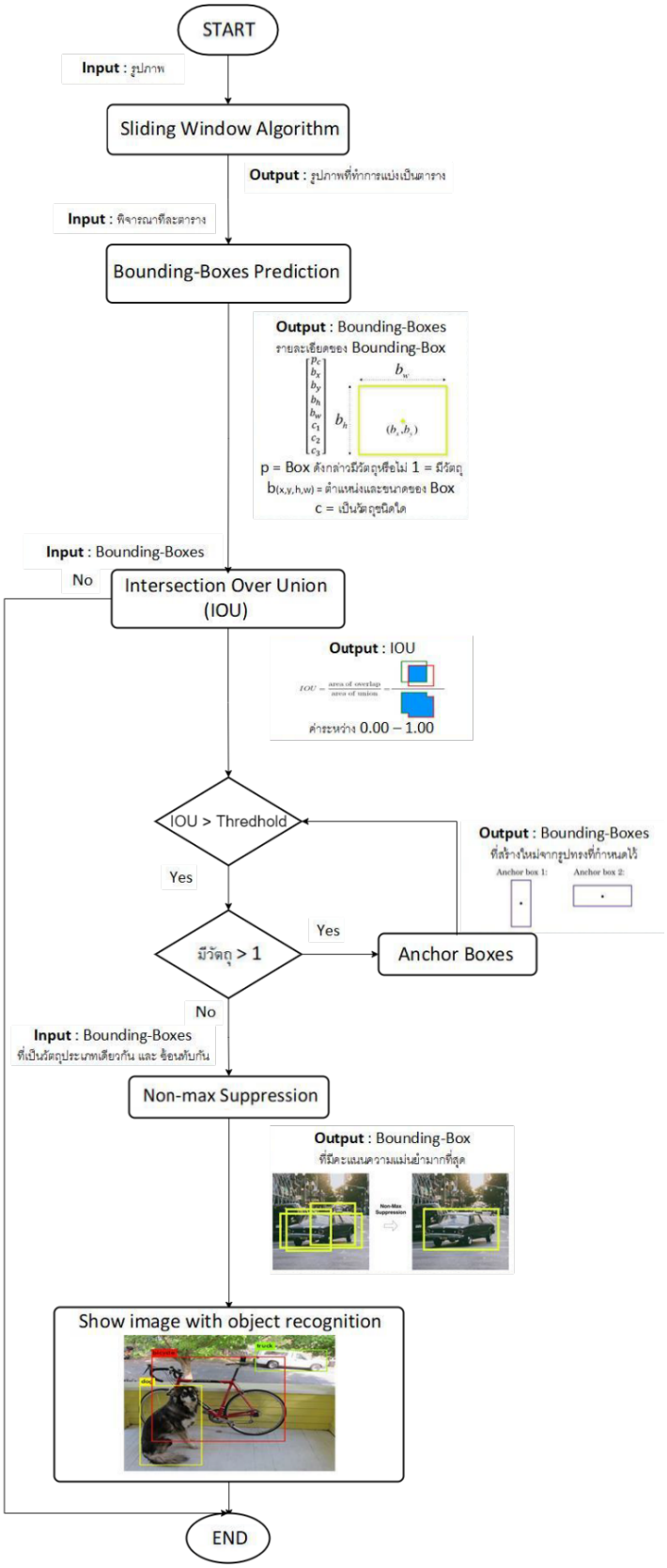


Figure 36 กระบวนการประมวลผล YOLO

3. Optical Character Recognition

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

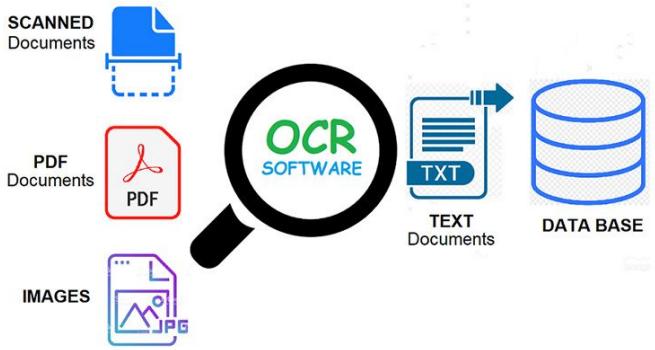
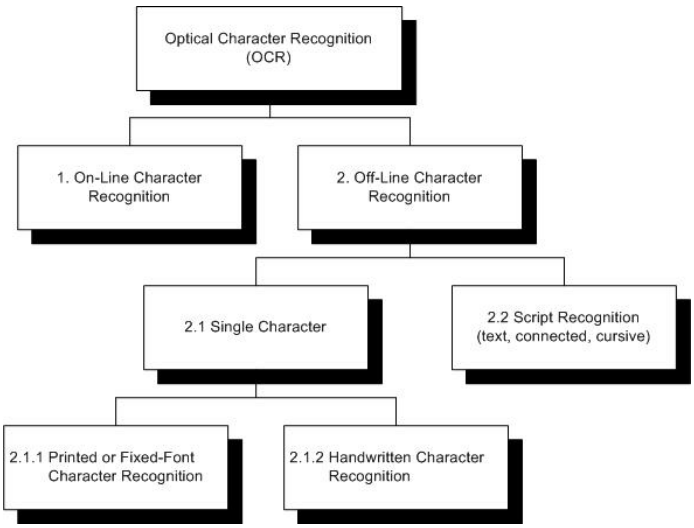


Figure 37 เทคนิคการรู้จำตัวอักษรด้วยแสง

ประเภทของเทคนิคการรู้จำตัวอักษรด้วยแสง



ประเภทของเทคนิคการรู้จำตัวอักษรด้วยแสง

การรู้จำตัวอักษรแบบออนไลน์ (On-line Character Recognition)

เป็นการประมวลตัวอักษรด้วยปากกาอิเล็กทรอนิกส์ที่นิยมใช้ภายในมือถือ หรือเว็บไซต์ต่าง ๆ การประมวลผลจะใช้การลากจากจุดหนึ่งไปยังจุดหนึ่ง ตามลักษณะของการเขียนตัวอักษร ซึ่งเป็นการประมวลผลที่ซับซ้อนมากกว่าการรู้จำแบบออฟไลน์ เนื่องจากเป็นการวิเคราะห์ข้อมูลแบบทิศทางการเขียน การลากเส้น และความเร็วของการลากเส้น ทำให้การประมวลผลในส่วนนี้มีความยาก

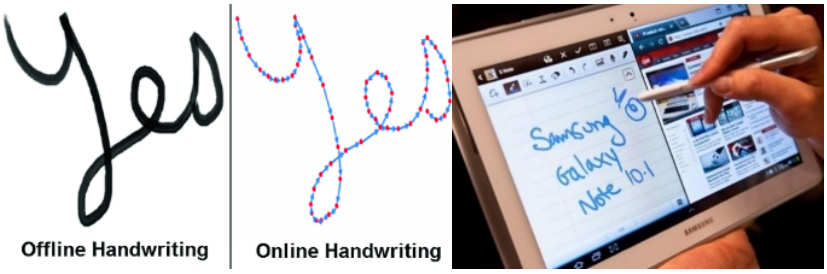


Figure 39 การรู้จำตัวอักษรแบบออนไลน์ (On-line Character Recognition)

การเรียนรู้ตัวอักษรแบบออฟไลน์ (Off-line Character Recognition)

การรู้จำตัวอักษรแบบออฟไลน์ เป็นการประมวลผลตัวอักษรด้วยเครื่องสแกน แบบพิมพ์หรือเทคนิคการจดจำ ซึ่งเป็นการประมวลผลตัวอักษร ได้ทั้งแบบเดี่ยว แบบติดกันเป็นกลุ่มคำของตัวอักษร โดยสามารถจำแนกการประมวลผลตามลักษณะของตัวอักษรดังนี้

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

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

A B C D E F G H I J K

L M N O P Q R S ... Z

Figure 40 ตัวพิมพ์แบบฟอนต์เฉพาะ

2. การรู้จำลายมือเขียนแบบตัวโดด (Isolated Handprint Character Recognition) การประมวลผลในส่วนนี้เป็นการรู้จำตัวอักษรด้วยลายมือเขียน มีระยะห่างจากกัน ไม่ติดหรือทับซ้อนกับตัวอักษรตัวอื่น ๆ โดยเขียนทีละตัวแยกจากกันชัดเจน เช่น เบอร์มือถือ ซึ่งเป็นตัวเลขจากลายมือเขียน เป็นต้น การประมวลผลที่มีความยากขึ้นอีกระดับจากตัวอักษรแบบโดด เนื่องจากลายมือของผู้เขียนตัวอักษรไม่เหมือนกัน มีความหลากหลายที่แตกต่างกันทำให้มีความผิดพลาดในการประมวลผลเนื่องจากโครงสร้างของตัวอักษรในการประมวลผลไม่เป็นที่แน่นอน

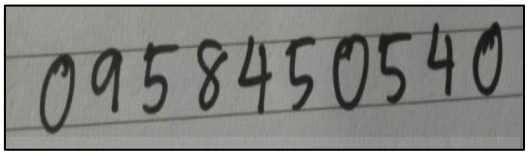
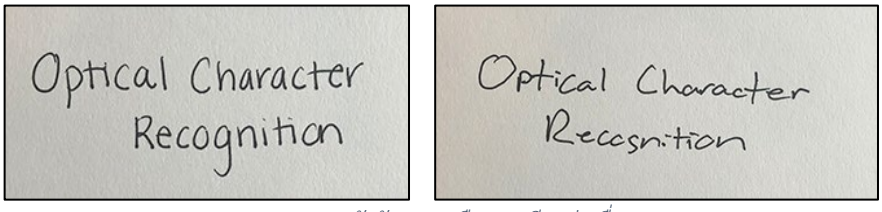


Figure 41 ลายมือเขียนแบบตัวโดด

ตัวอักษรลายมือแบบเขียนต่อเนื่อง (Script recognition) การประมวลผลในส่วนนี้เป็นการรู้จำที่ยากที่สุดเนื่องจากลักษณะการเขียนด้วยลายมือที่ไม่มีขอบเขตในการเขียน ทั้งยังเป็นการประมวลผลโดยตัวอักษรเขียนได้อย่างต่อเนื่อง เป็นตัวอักษรที่มีเส้นลากเชื่อมติดกัน ตัวติดกัน และมีโครงสร้างของตัวอักษรที่แตกต่างกัน ไม่กำหนดฟอนต์ของตัวอักษร มีความแตกต่างของลายมือ



ตัวอักษรลายมือแบบเขียนต่อเนื่อง

ขั้นตอนเทคนิคการรู้จำตัวอักษรด้วยแสง (Optical Character Recognition)

การประมวลผลหรือขั้นตอนในการจดจำตัวอักษรมีการประมวลผลหลักอยู่ 3 ขั้นตอน ดังต่อไปนี้

Pre-Processing)

การประมวลผลขั้นต้นเป็นการประมวลผลขั้นแรกของเทคนิคโอซีอาร์ (OCR) เป็นการเข้าถึงตัวอักษรและการปรับปรุงภาพ โดยทำการตรวจสอบภาพที่ก่อนนำไปรู้จำ สามารถแยกย่อยขั้นตอนการแปลงภาพได้หลายขั้นตอน ขึ้นอยู่กับการประมวลผลภาพซึ่งเรียกกระบวนการประมวลผลขั้นต้นว่า Pre-Processing เช่น

การลดสัญญาณรบกวน (Noise Filtering) เป็นการกรองข้อมูลภาพที่มีสิ่งแทรกซ้อนที่ไม่ต้องการโดยมีจุดประสงค์เพื่อลดความผิดพลาด สิ่งรบกวน ภายในภาพเช่น รอยขูดขีด หรือการเติมเต็มความคมชัดของภาพเพื่อให้สามารถเพิ่มประสิทธิผลของการรู้จำมากยิ่งขึ้น

Noise คือ จุดรบกวนในภาพที่เกิดจาก การปรับเพิ่มค่าความไวแสง เมื่อเราเพิ่มค่าความไวแสงของกล้องดิจิตอลก็เป็นการสั่งให้เพิ่มกระแสไฟฟ้าบนตัวเซนเซอร์ให้สูงขึ้น ก็เพื่อเพิ่มความไวต่อแสงให้มากขึ้น สิ่งที่ตามมาคือสัญญาณรบกวนที่เพิ่มสูงขึ้นตามไปด้วย ซึ่งกรณีที่ค่าค่าความไวแสงต่ำ มักไม่ค่อยจะปรากฏให้เห็น ซึ่ง Noise จะแสดงให้เห็นเป็นจุดรบกวนเม็ดเล็ก ๆ สีต่าง ๆ ในเงามืด หรือรายละเอียดต่าง ๆ ของภาพ

โดยยิ่งเราถ่ายภาพซึ่งใช้ ค่าเพิ่มค่าความไวแสง สูง ๆ เท่าใด ยิ่งจะปรากฏ Noise มากขึ้นเท่านั้น แน่นอนส่งผลโดยตรงกับคุณภาพของภาพถ่ายเราได้

ปัญหา Noise หรือสัญญาณรบกวนนั้นถือเป็นเรื่องปกติที่มากับกล้องดิจิตอลอยู่แล้ว เพราะมันเกิดจากสัญญาณรบกวนจากกระแสไฟจากกล้องอยู่แล้ว โดยเฉพาะอย่างยิ่งการใช้ค่าความไวแสงสูง ๆ ในยุคปัจจุบันนี้ กล้องจะมีวงจรลดสัญญาณรบกวนอยู่แล้ว และยังมีฟังก์ชันลดสัญญาณรบกวนให้ผู้ใช้กล้องสามารถปรับเพิ่มเติมได้อีก ทำให้กล้องรุ่นใหม่ ๆ สามารถถ่ายภาพได้โดยใช้ค่า ISO ที่สูง ๆ ได้สบาย ๆ โดยไม่ปรากฏ Noise ออกมามาก

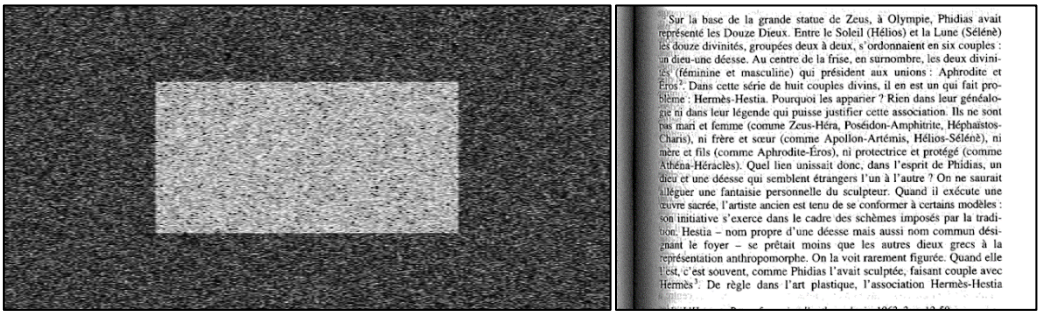

Sur la base de la grande statue de Zeus, à Olympie, Phidias avait représenté les Douze Dieux. Entre le Soleil (Hélios) et la Lune (Séléné) les douze divinités, groupées deux à deux, s'ordonnaient en six couples : un dieu-une déesse. Au centre de la frise, en surnombre, les deux divinités (féminine et masculine) qui président aux unions : Aphrodite et Éros[2]. Dans cette série de huit couples divins, il en est un qui fait problème : Hermès-Hestia. Pourquoi les apparier ? Rien dans leur généalogie ni dans leur légende qui puisse justifier cette association. Ils ne sont pas mari et femme (comme Zeus-Héra, Poséidon-Amphitrite, Héphaïstos-Charis), ni frère et sœur (comme Apollon-Artémis, Hélios-Séléné), ni mère et fils (comme Aphrodite-Éros), ni protectrice et protégé (comme Athéna-Héraclès). Quel lien unissait donc, dans l'esprit de Phidias, un dieu et une déesse qui semblent étrangers l'un à l'autre ? On ne saurait alléguer une fantaisie personnelle du sculpteur. Quand il exécute une œuvre sacrée, l'artiste ancien est tenu de se conformer à certains modèles : son initiative s'exerce dans le cadre des schèmes imposés par la tradition. Hestia – nom propre d'une déesse mais aussi nom commun désignant le foyer – se prêtait moins que les autres dieux grecs à la représentation anthropomorphe. On la voit rarement figurée. Quand elle l'est, c'est souvent, comme Phidias l'avait sculptée, faisant couple avec Hermès[3]. De règle dans l'art plastique, l'association Hermès-Hestia

Figure 43 ภาพที่เกิดปัญหาสัญญาณรบกวน

ฟังก์ชัน cv2.threshold() ใช้คัดกรองภาพโดยพิจารณาความสว่าง เพื่อที่จะคัดเอาหรือปรับค่าบางส่วน

Threshold มี 5 แบบ

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

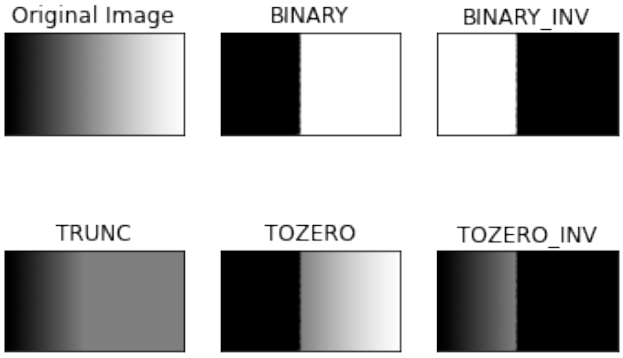


Figure 44 ฟังก์ชัน Threshold 5 แบบ

ตัวอย่างการลดสัญญาณรบกวน

Original

Global Thresholding

Otsu_s_Thresholding

Gaussian

ตัวอย่างการลดสัญญาณรบกวน

Original

Global Thresholding

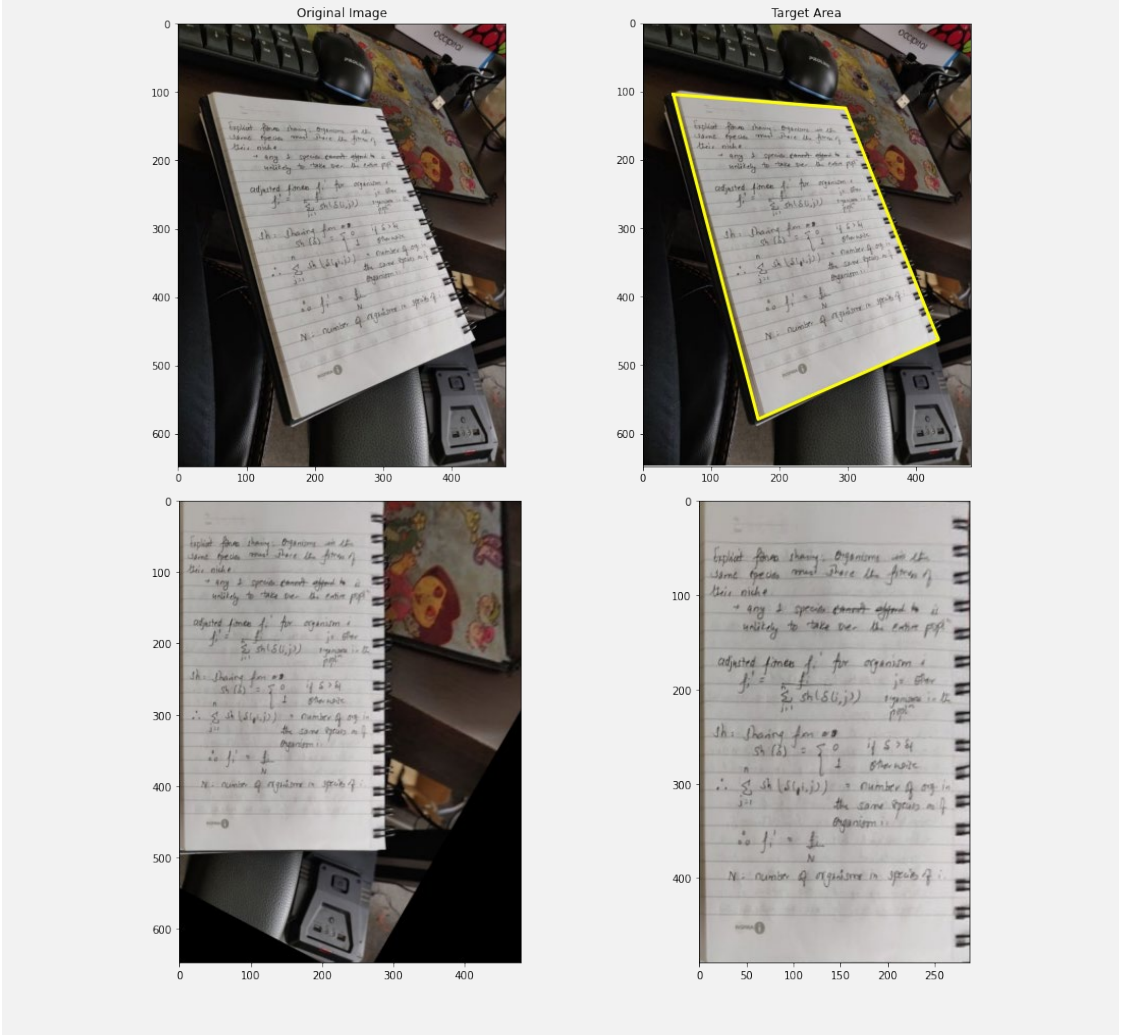
Otsu_s_Thresholding

Gaussian

Skew correction ตรวจหาค่าความเอียงและการหมุนกลับของภาพเอกสาร

ตัวอย่างการการหมุนกลับของภาพ

ตัวอย่างการการหมุนกลับของภาพ



การรู้จำ (Recognition)

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

วิธีทางสถิติ (Statistical Approach) เป็นวิธีการที่ประยุกต์ใช้หลักการทางสถิติ โดยการคาดการความน่าจะเป็นของตัวเลขรวมถึงการใช้ฟังก์ชันการแจกแจงความน่าจะเป็นในการระบุตัวอักษร โดยใช้ภาพอินพุตที่ผ่านกระบวนการปรับปรุงภาพ สกัดลักษณะเด่นของตัวอักษร และประมวลผลตัวอักษรโดยใช้ความน่าจะเป็นว่าเป็นตัวอักษรตัวใด ๆ แล้วนำผลลัพธ์ในการรู้จำมาหาค่าความน่าจะเป็นของตัวอักษรที่ใกล้เคียงมากที่สุดแล้วแสดงผลลัพธ์เป็นตัวอักษรนั้นๆ

วิธีการวิเคราะห์ทางโครงสร้าง (Structural Analysis) คือ การวิเคราะห์ โครงสร้างภาพตัวอักษร โดยใช้โครงสร้างพื้นฐานของภาพตัวอักษร สกัดลักษณะเด่นของตัวอักษร โครงสร้างของตัวอักษ เช่น เส้นตรง บน ล่าง วงกลม วงรี เส้นตัด จุดเชื่อม เป็นต้น ในขั้นตอนการรู้จำจะใช้การวิเคราะห์ลักษณะของตัวอักษร เช่น ฟอร์มอล

แกรมมาแมชชีน (formal grammar machine) โครงสร้างกราฟ หรือโครงสร้างต้นไม้เป็นต้น เพื่อเป็นอัลกอริทึมในการตัดสิน เทคนิคการวิเคราะห์มักจะใช้กับความหลากหลายของตัวอักษร เทคนิคนี้ขึ้นอยู่กับการสร้างรูปแบบโครงสร้างและการวิเคราะห์โครงสร้าง

โครงข่ายประสาทเทียม (Neural Network) เป็นเทคนิคใหม่ที่ได้รับการพัฒนาเป็นอย่างมาก เนื่องจากประสิทธิผลความแม่นยำสูง ถูกนำไปใช้ในงานหลาย ๆ ด้าน โครงข่ายประสาทเทียมเป็นเทคนิคที่เลียนแบบการทำงานของสมองมนุษย์ มีโครงข่ายความจำย่อยจำนวนมากในการแบ่งโครงสร้างเป็นฐานข้อมูลความรู้ เป็นการแบ่งเลเยอร์หรือรูปแบบในการประมวลผลเป็นชั้น ๆ เพื่อค้นหาเลเยอร์ที่ถูกต้องของข้อมูลนั้น ๆ

วิธีการรู้จำด้วยการเปรียบเทียบคู่รูปแบบ (Template Matching) เป็นวิธีการรู้จำตัวอักษรแบบแรกเริ่มที่นำมาประยุกต์ใช้กับการรู้จำด้วยอักษร โดยมีหลักการพื้นฐาน คือ การสร้างแม่แบบหรือที่เรียกว่า template เพื่อนำมาอ่านและเปรียบเทียบความคล้ายคลึงกันของตัวอักษร โดยกำหนดตำแหน่งของพิกเซลตัวอักษรแล้วเปรียบเทียบหาความแตกต่างระหว่างตัวอักษรแต่ละตัวจากนั้นก็ระบุว่าเป็นรหัสด้วยอักษรอะไร โดยใช้ค่าต่างระดับในการตัดสินใจ วิธีนี้มีข้อเสียที่ความเปลี่ยนแปลงต่อข้อมูลที่แตกต่างกัน ขนาด ความเอียงของตัวอักษรซึ่งในการเปรียบเทียบแม่แบบมี 2 ลักษณะ คือ การเปรียบเทียบโครงสร้างของภาพ และเปรียบเทียบน้ำหนักของภาพ

ขบวนการประมวลผลขั้นสุดท้าย (Post-Processing)

การประมวลผลขั้นตอนสุดท้าย คือ การตรวจสอบและแก้ไขผลจากการรู้จำรูปตัวอักษรที่ผ่านกระบวนการรู้จำมีผลเป็นรหัสตัวอักษร ซึ่งอาจจะเป็นตัวอักษรที่ถูกต้องหรือผิดพลาดก็เป็นได้ ดังนั้นขั้นตอนสุดท้ายเป็นการตรวจสอบความถูกต้องของการรู้จำ โดยการแสดงผลการรู้จำเป็นตัวอักษรที่สามารถแก้ไขได้ ส่วนนี้มักเป็นอัลกอริทึมในการตรวจสอบความถูกต้อง เช่น ไวยากรณ์ภาษา เป็นต้น

เครื่องมือ OCR Open source

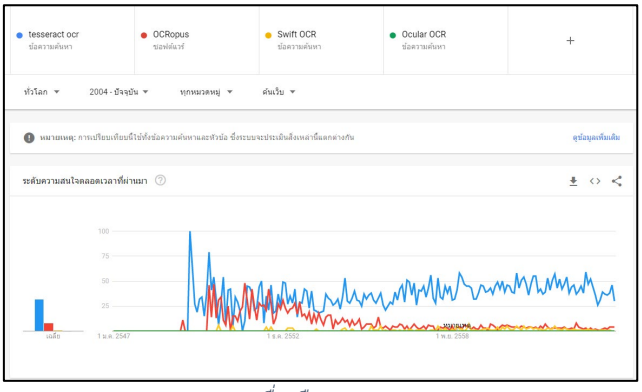


Figure 45 เครื่องมือ OCR Open source

Tesseract เป็นเอ็นจิ้นการรู้จำอักขระด้วยแสงสำหรับระบบปฏิบัติการต่าง ๆ เป็นซอฟต์แวร์เสรีปล่อยใต้สัญญาอนุญาต Apache เดิมทีพัฒนาโดย Hewlett-Packard เป็นซอฟต์แวร์ที่เป็นกรรมสิทธิ์ในทศวรรษ 1980 มันถูกปล่อยออกมาเป็นโอเพนซอร์สในปี 2005 และการพัฒนาได้รับการสนับสนุนจาก Google ตั้งแต่ปี 2006 Tesseract ได้รับการพิจารณาให้เป็นหนึ่งในเอ็นจิ้น OCR โอเพนซอร์สที่แม่นยำที่สุดที่มีอยู่

OCRopus ได้รับการออกแบบมาโดยเฉพาะเพื่อใช้ในโครงการแปลงหนังสือเป็นดิจิทัลจำนวนมาก เช่น Google Books, Internet Archive หรือห้องสมุด รองรับภาษาและแบบอักษรจำนวนมาก อย่างไรก็ตามสามารถใช้กับแอปพลิเคชันเดสก์ท็อปและสำนักงาน และแอปพลิเคชันสำหรับผู้พิการทางสายตา

Tesseract OCR

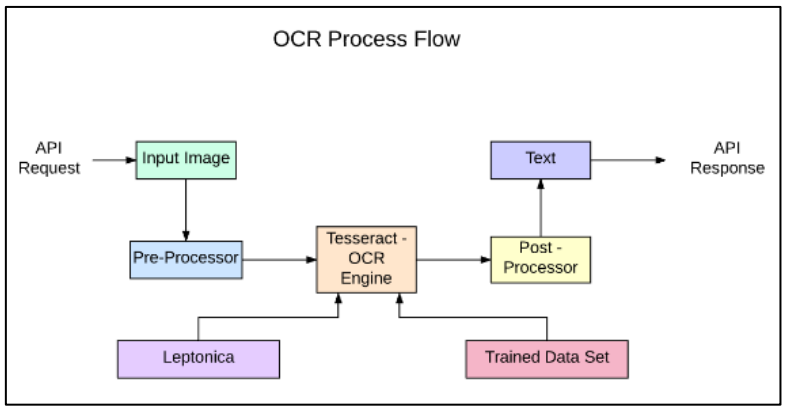


Figure 46 กระบวนการทำงานของ Tesseract OCR

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

โครงสร้างการประมวลผลปกติ (เวอร์ชั่นก่อนหน้า)

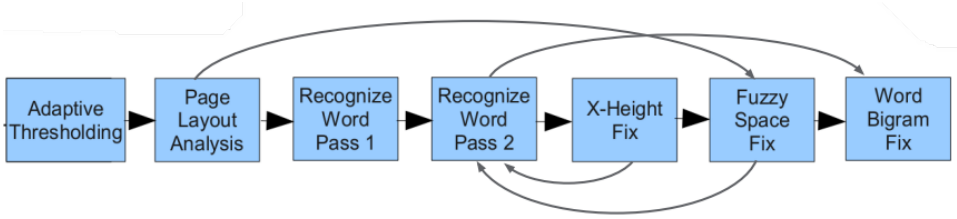


Figure 47 โครงสร้างการประมวลผลปกติ

โครงสร้างการประมวลผลเสริมด้วยเทคนิค LSTM (เวอร์ชั่น 4)

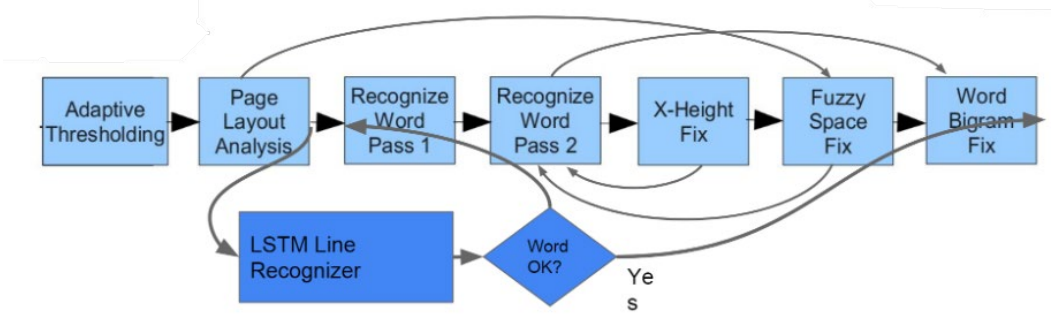


Figure 48 โครงสร้างการประมวลผลเสริมด้วยเทคนิค LSTM

ขั้นตอนการรู้จำของ Tesseract

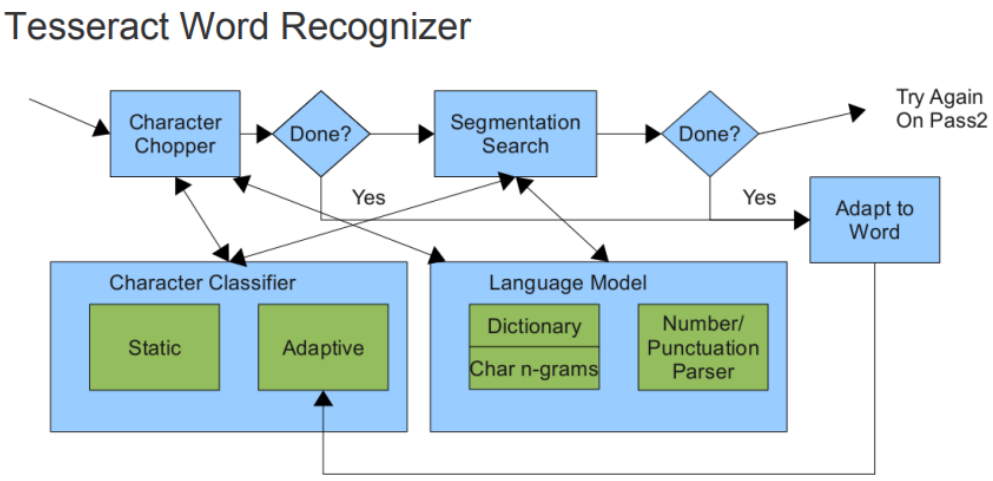


Figure 49 ขั้นตอนการรู้จำของ Tesseract

การติดตั้ง Tesseract

Install Tesseract For Window

Windows

Installer for Windows for Tesseract 3.05, Tesseract 4 and development version 5.00 Alpha are available from Tesseract at UB Mannheim. These include the training tools. Both 32-bit and 64-bit installers are available.

An installer for the **OLD version 3.02** is available for Windows from our download page. This includes the English training data. If you want to use another language, download the appropriate training data, unpack it using 7-zip, and copy the .traineddata file into the 'tessdata' directory, probably `C:\Program Files\Tesseract-OCR\tessdata`.

To access tesseract-OCR from any location you may have to add the directory where the tesseract-OCR binaries are located to the Path variables, probably `C:\Program Files\Tesseract-OCR`.

Experts can also get binaries build with Visual Studio from the build artifacts of the Appveyor Continuous Integration.

1. เข้าสู่หน้าเว็บ Tesseract at UB Mannheim เลือกเวอร์ชันตามระบบปฏิบัติการของท่าน

Tesseract at UB Mannheim

The Mannheim University Library (UB Mannheim) uses Tesseract to perform OCR (optical character recognition) of historical German newspapers (Allgemeine Preußische Staatszeitung, Deutscher Reichsanzeiger). The latest results with OCR from more than 360,000 scans are available online.

Tesseract installer for Windows

Normally we run Tesseract on Debian GNU Linux, but there was also the need for a Windows version. That's why we have built a Tesseract installer for Windows.

WARNING: Tesseract should be either installed in the directory which is suggested during the installation or in a new directory. The uninstaller removes the whole installation directory. If you installed Tesseract in an existing directory, that directory will be removed with all its subdirectories and files.

The latest installers can be downloaded here:

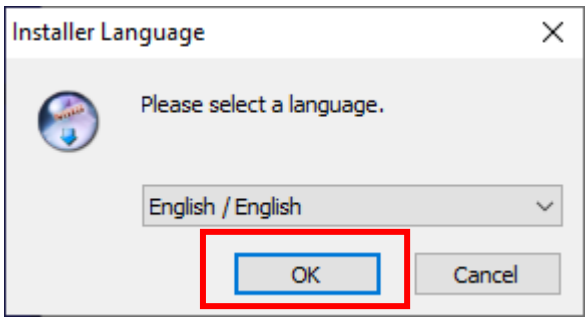
- tesseract-ocr-w32-setup-v5.0.0-alpha.20210811.exe (32 bit) and
- tesseract-ocr-w64-setup-v5.0.0-alpha.20210811.exe (64 bit) resp.

We don't provide an installer for Tesseract 4.1.0 because we think that the latest version 5.0.0-alpha is better for most Windows users in many aspects (functionality, speed, stability). Version 4.1 is only needed for people who develop software based on the Tesseract API and who need 100 % API compatibility with version 4.0.

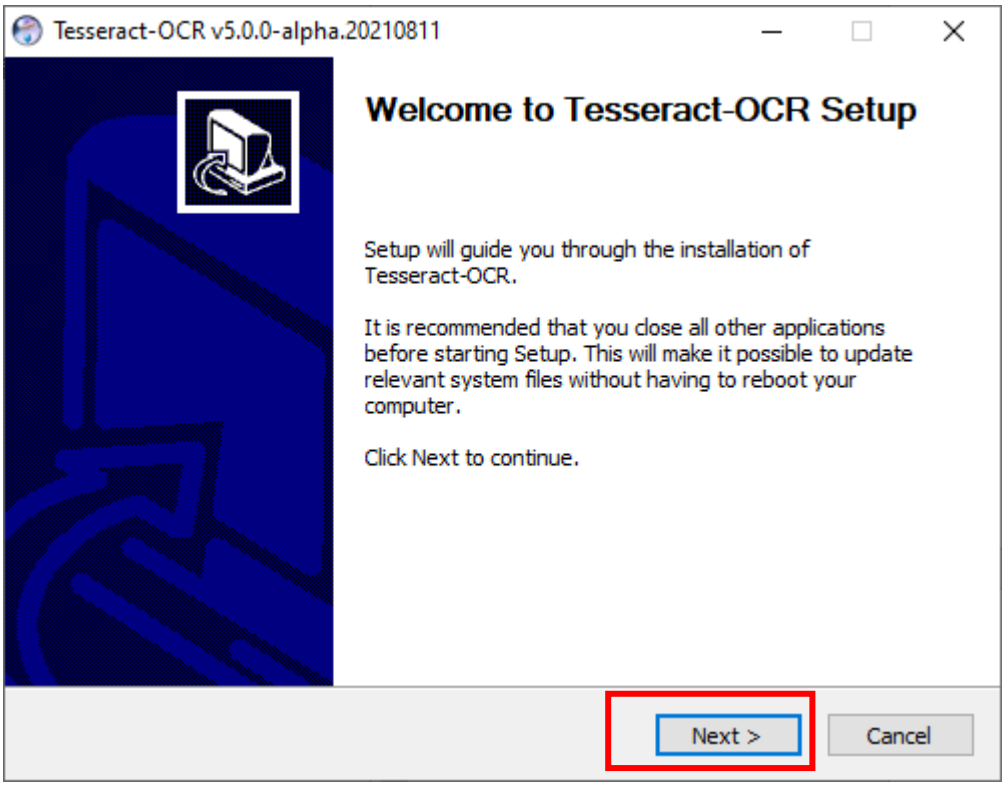
There are also older versions available.

In addition, we also provide documentation which was generated by Doxygen.

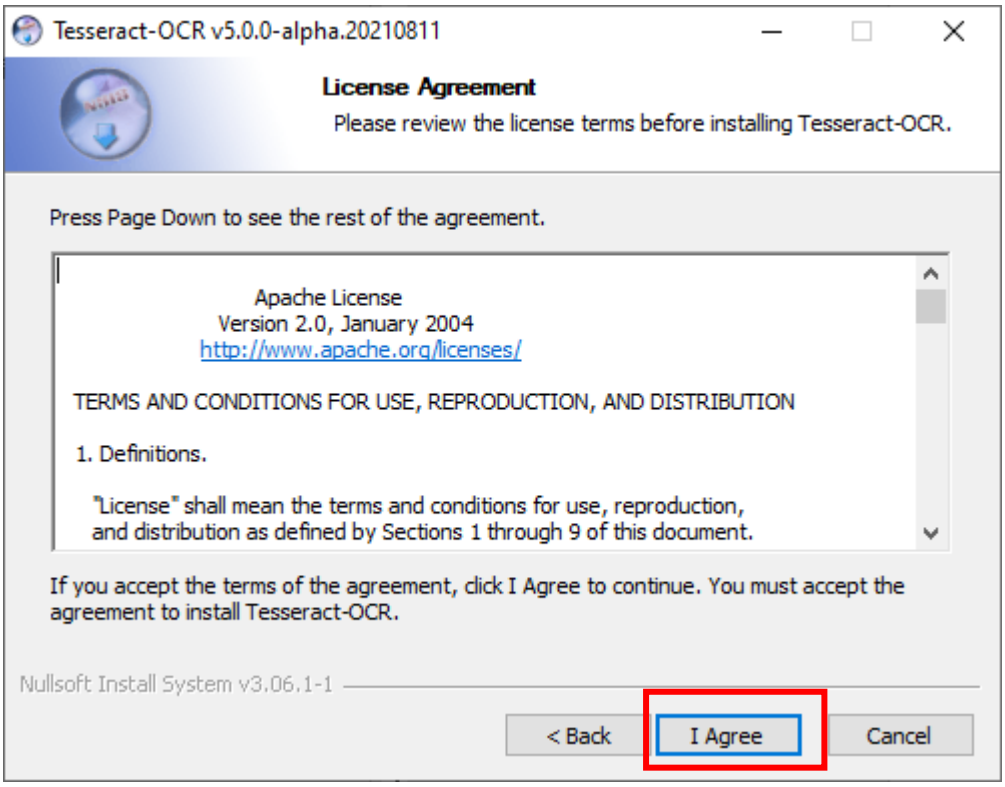
2. เลือกภาษาในการอธิบายการติดตั้งโดยค่าเริ่มต้น เป็นภาษาอังกฤษ กด OK



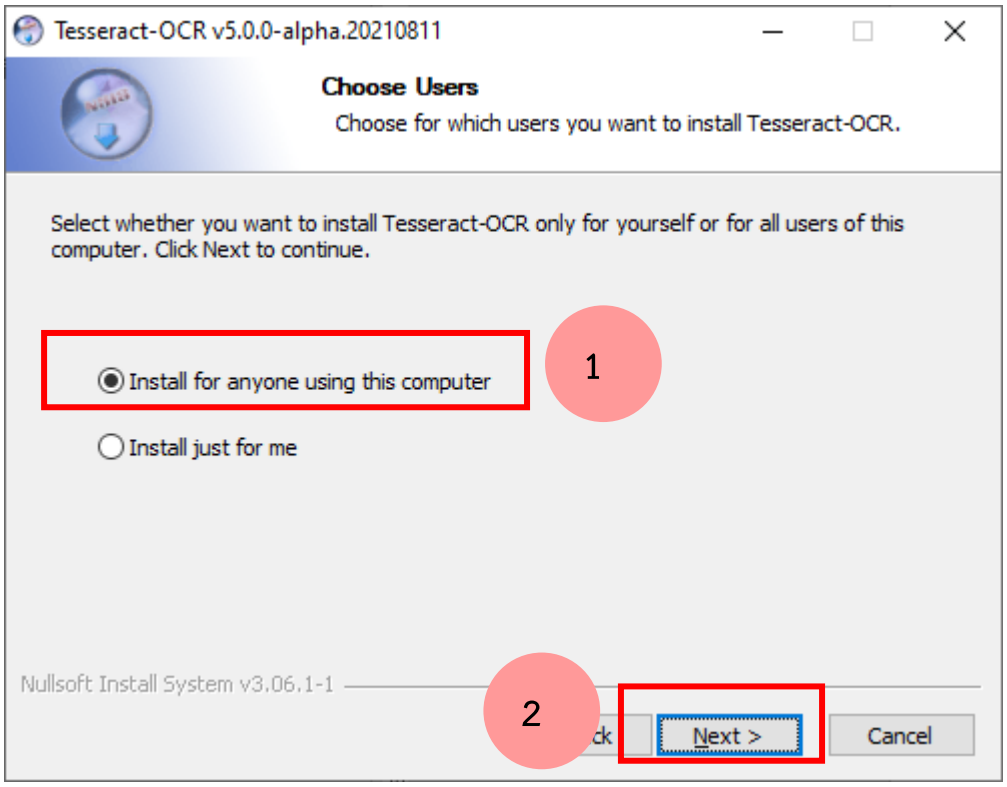
3. กด Next >



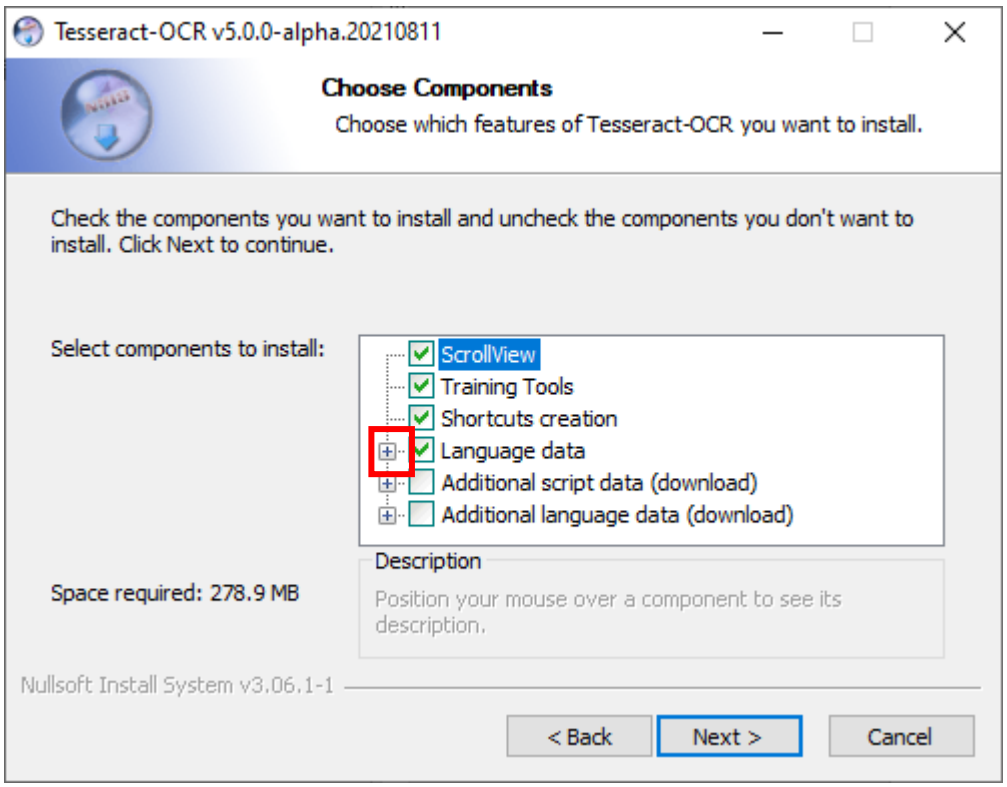
4. กด I Agree เพื่อยอมรับข้อตกลง



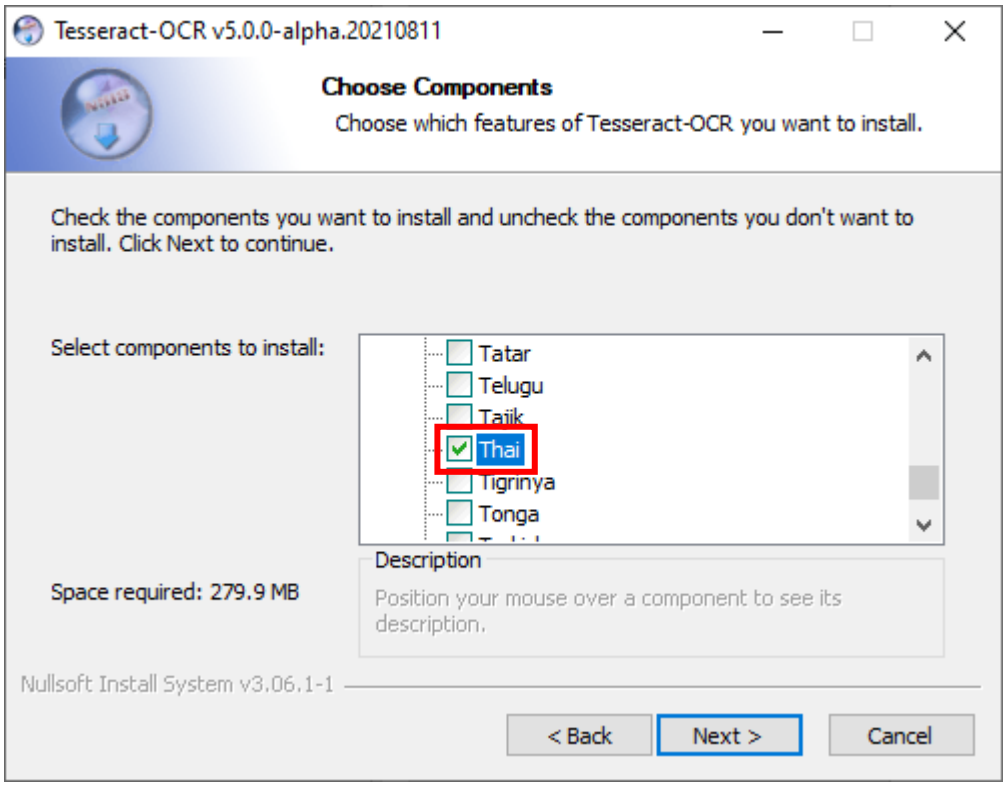
5. เลือกผู้ใช้งานเป็น Install for anyone using this computer* จำเป็น



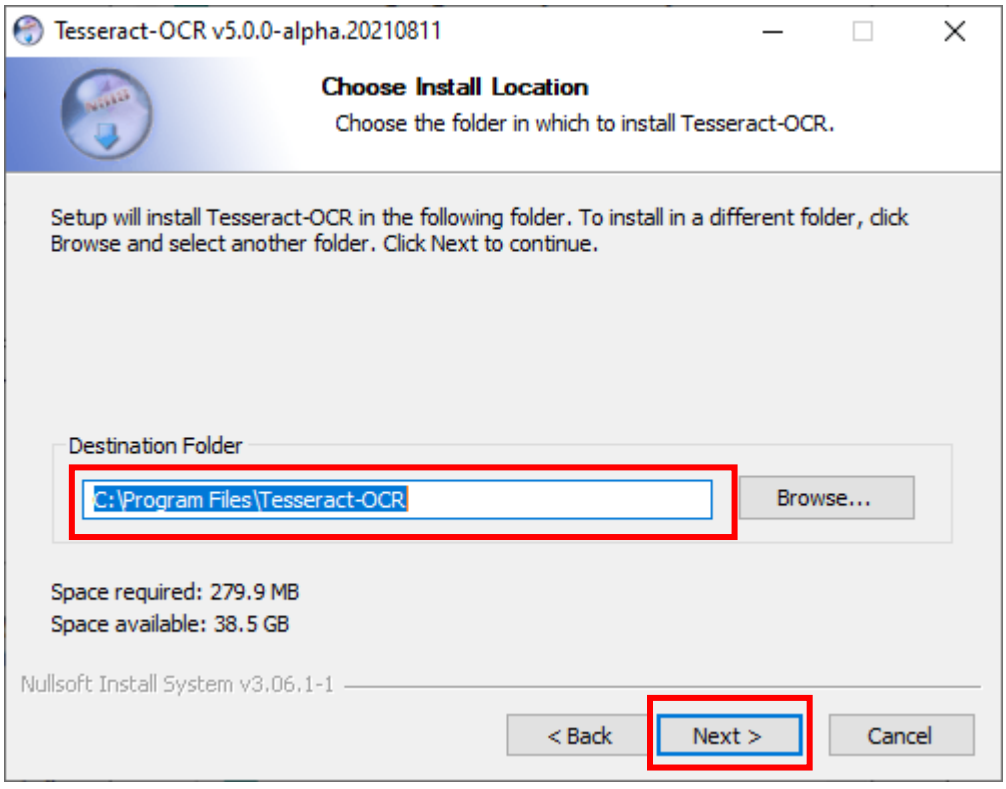
6. กดปุ่ม + ที่ Additional language data (download) เพื่อเลือกภาษาที่จะใช้งานเพิ่มเติม (ภาษาไทย)



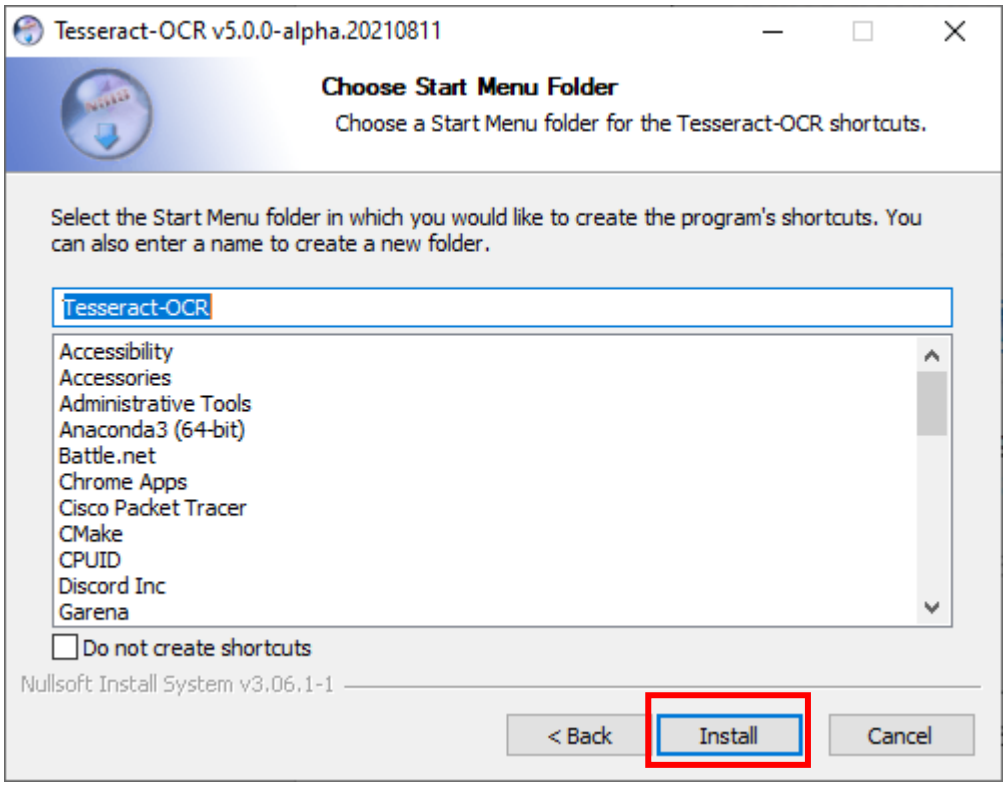
7. เลือกภาษาไทย หลังจากนั้น กด Next >



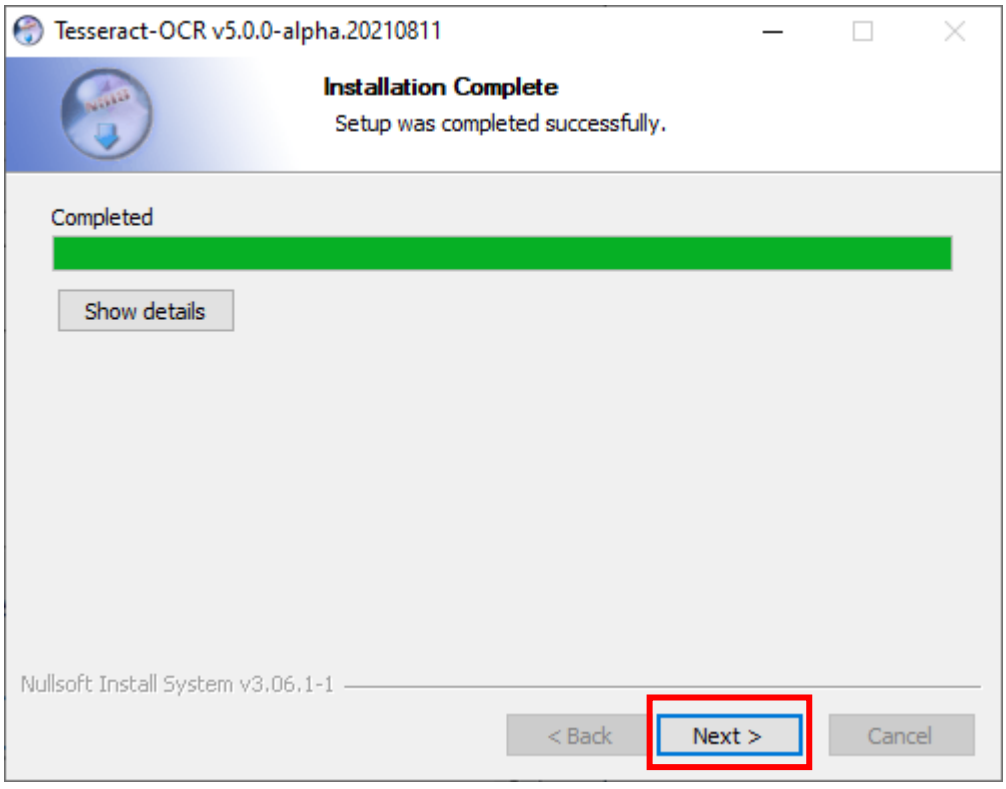
8. เลือกโฟลเดอร์ในการติดตั้ง โดยค่าเริ่มต้นเป็น C:\Program Files\Tesseract-OCR



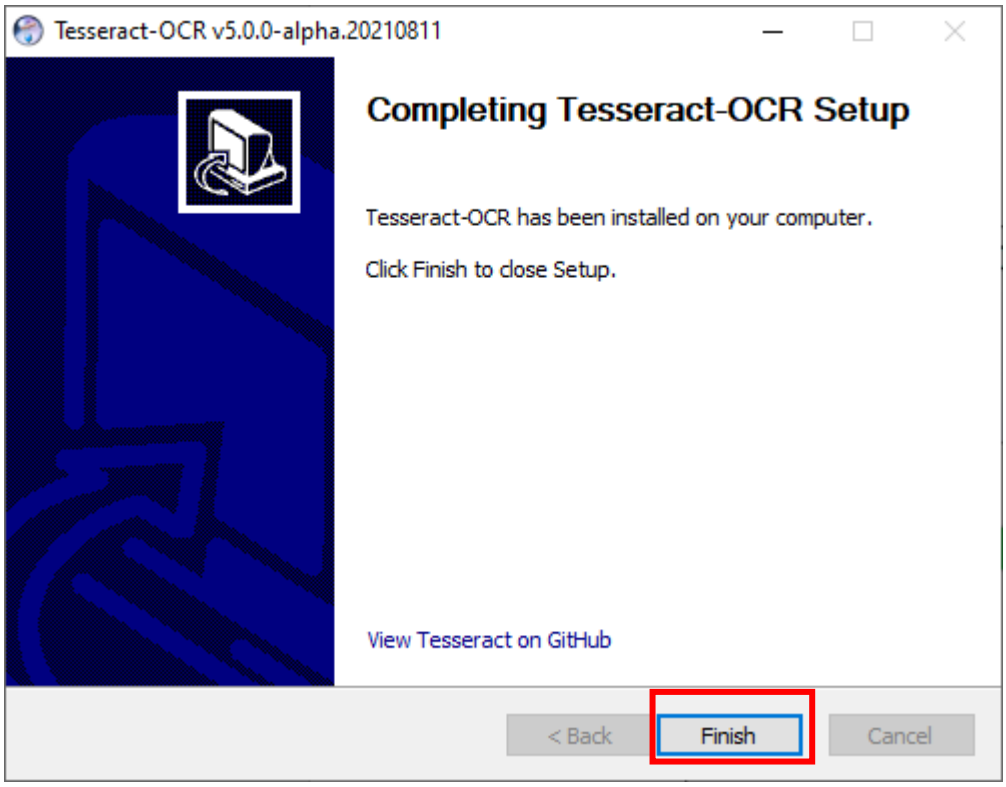
9. กด Install



10. กด Next >

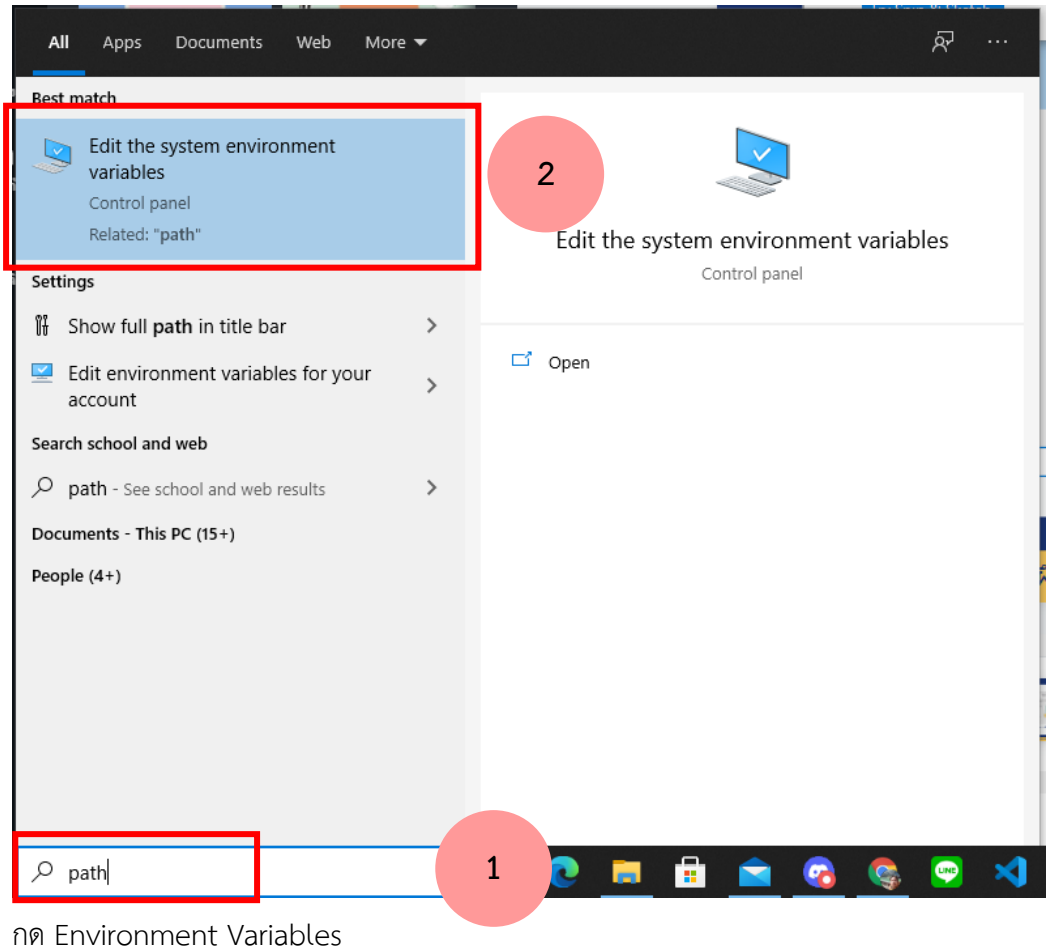


11. Finish

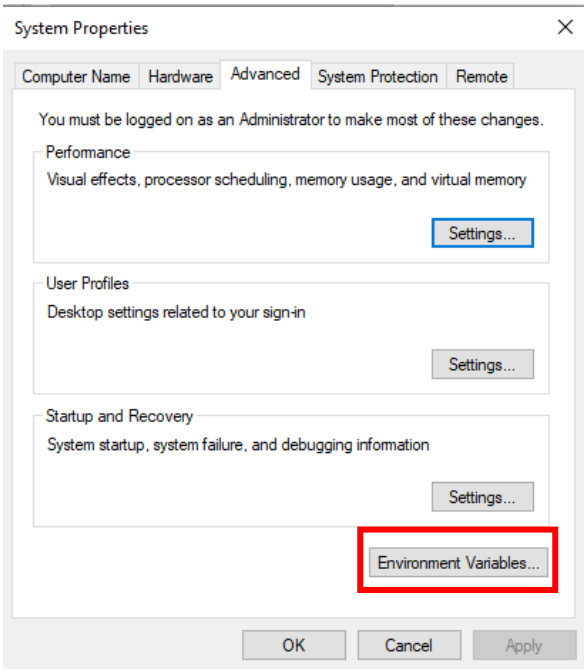

Tesseract-OCR v5.0.0-alpha.20210811
Completing Tesseract-OCR Setup
Tesseract-OCR has been installed on your computer.
Click Finish to close Setup.
View Tesseract on GitHub
< Back
Finish
Cancel

การตั้งค่า *Path* ไฟล์ เพื่อเรียกใช้งาน

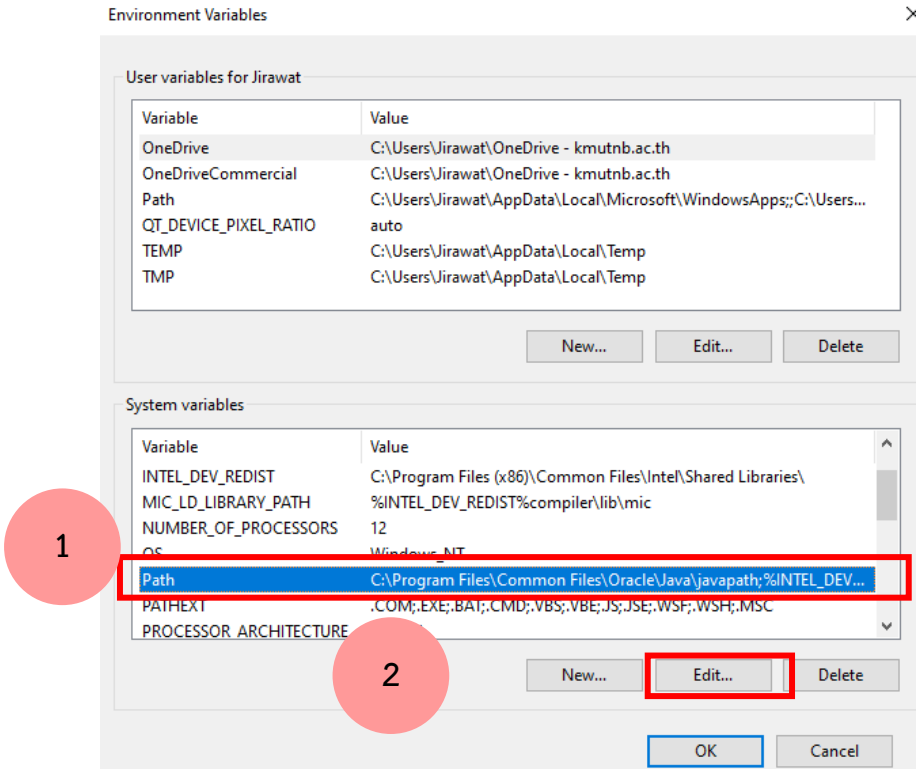
1. ค้นหา path ในช่อง window search



2. กด Environment Variables

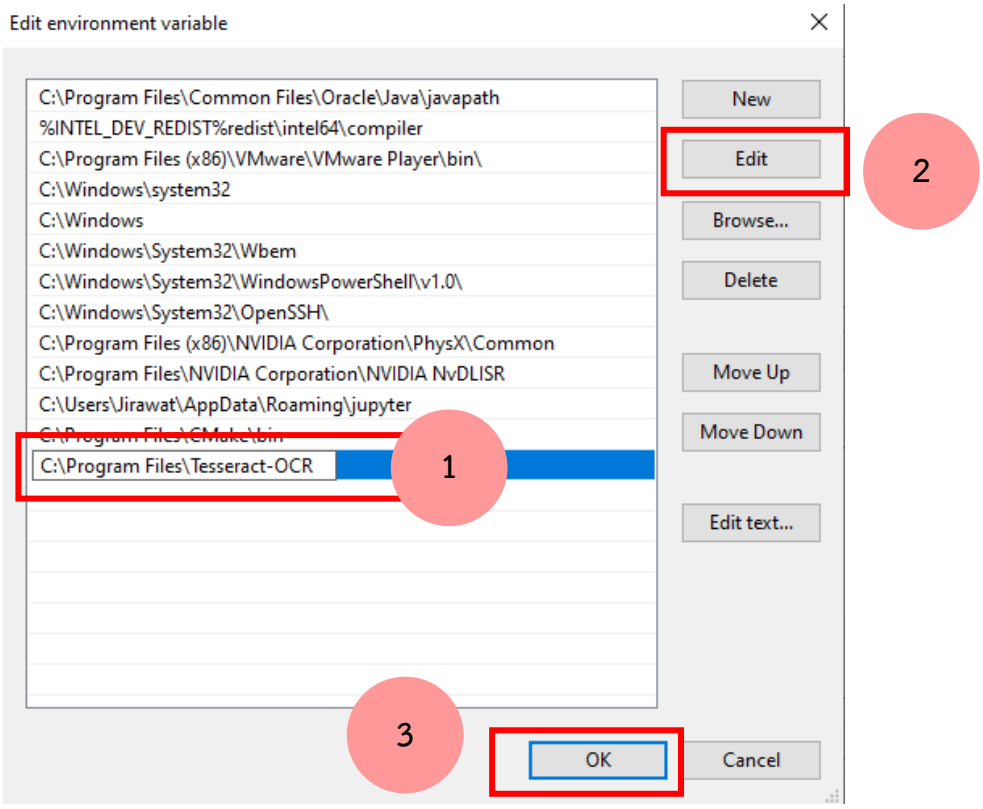


3. เลือก Variable Path > Edit

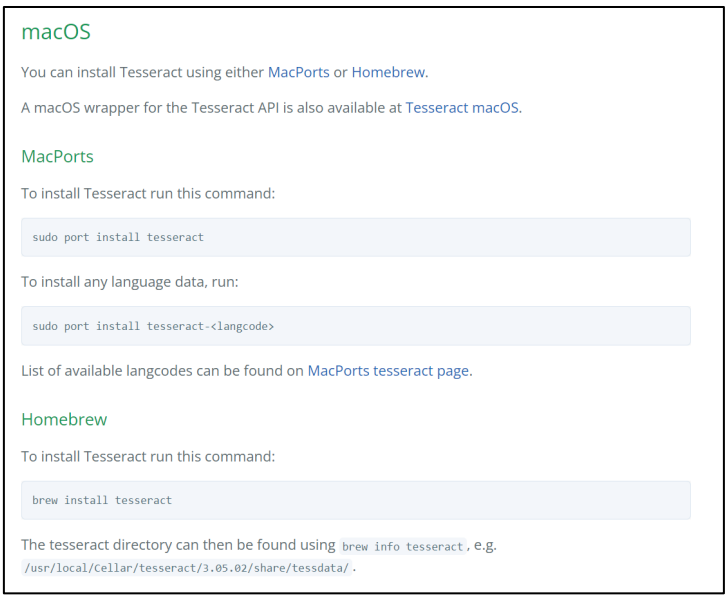


4. กดปุ่ม New

หลังจากนั้นใส่ Path C:\Program Files\Tesseract-OCR *ใส่ Path ที่เราติดตั้ง Tesseract* > กดปุ่ม OK



Install Tesseract For MAC



macOS

You can install Tesseract using either MacPorts or Homebrew.

A macOS wrapper for the Tesseract API is also available at Tesseract macOS.

MacPorts

To install Tesseract run this command:

```
sudo port install tesseract
```

To install any language data, run:

```
sudo port install tesseract-<langcode>
```

List of available langcodes can be found on MacPorts tesseract page.

Homebrew

To install Tesseract run this command:

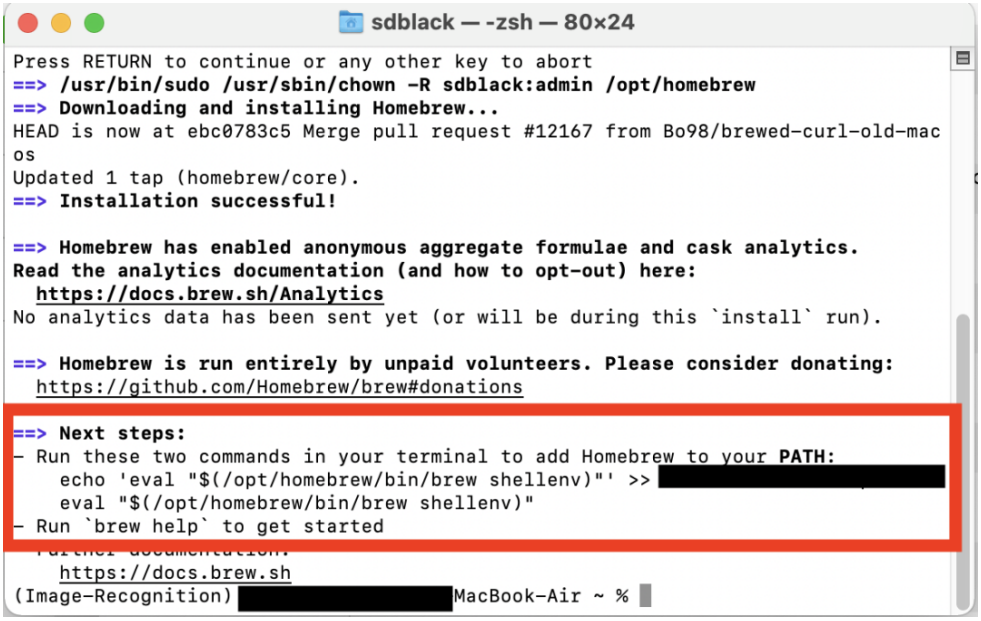
```
brew install tesseract
```

The tesseract directory can then be found using `brew info tesseract`, e.g. `/usr/local/Cellar/tesseract/3.05.02/share/tessdata/`.

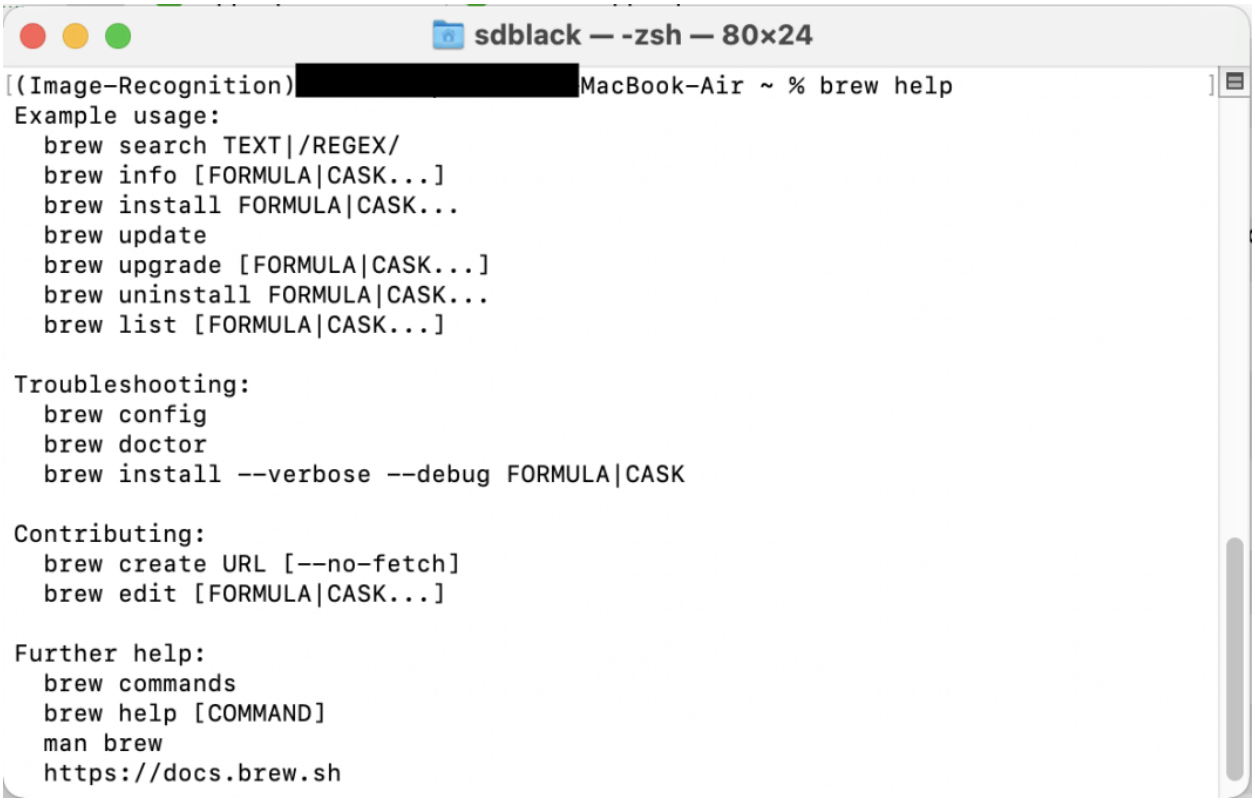
1. ติดตั้ง /bin/bash -c "$(curl -fsSL

   https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

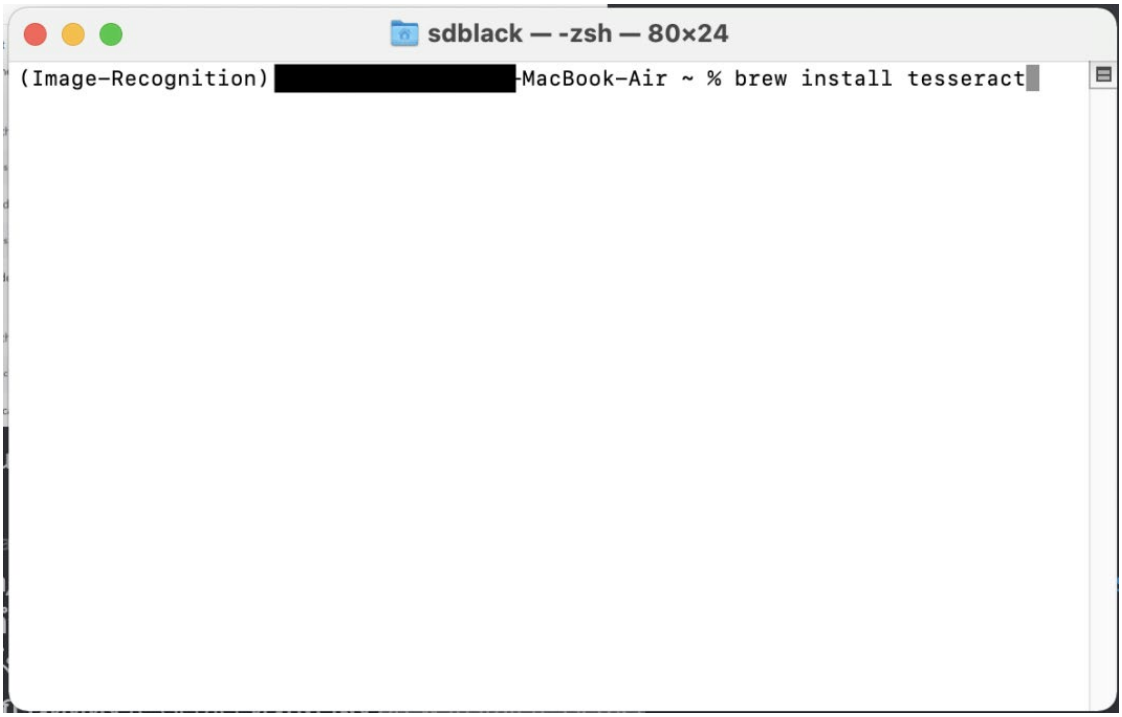
1.1. แล้วทำตามคำแนะนำที่ขึ้นตาม terminal ที่ Next steps:



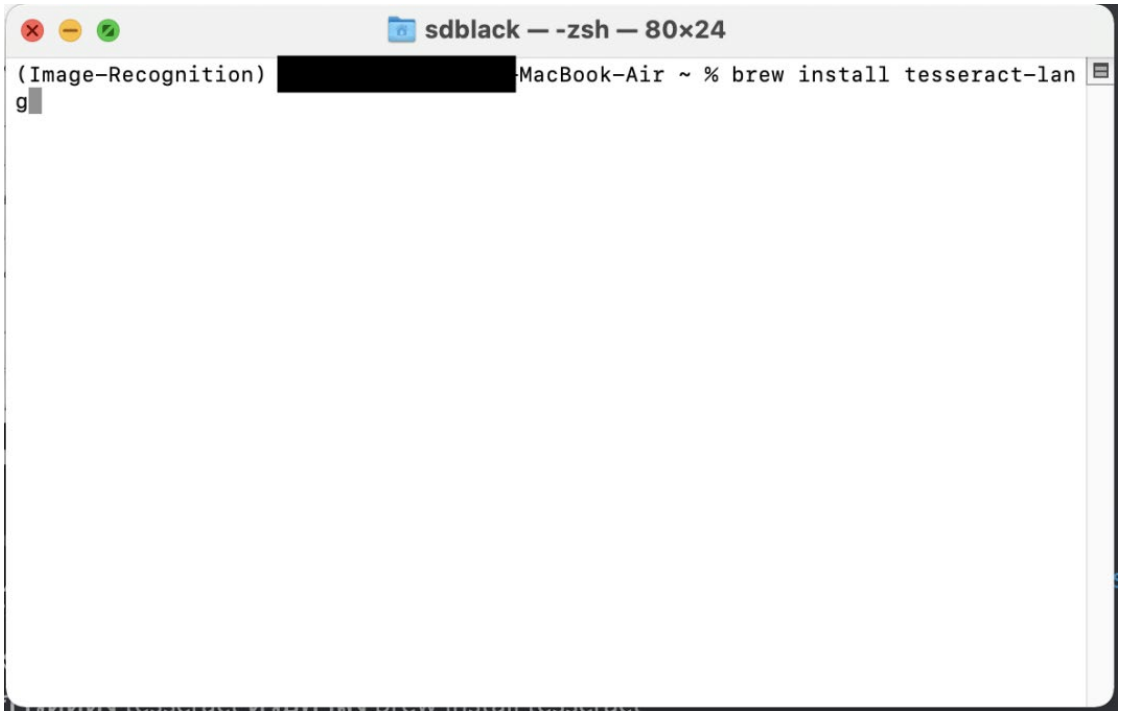
2. ทดลองใช้งาน brew โดยพิมพ์คำสั่ง $brew help



3. จากนั้นตำการติดตั้ง tesseract ด้วยคำสั่ง brew install tesseract



4. กรณีเพิ่มภาษาไทยใช้คำสั่ง brew install tesseract-lang เท่านี้ก็เรียบร้อยแล้วครับ



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
ละตัวที่ตรวจพบ
print(imgbox) # แสดงข้อมูลอักขระที่ตรวจพบ
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(' ') # แยกสตริงโดยใช้ ' '
    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) # เ
ป็นการวาดอักขระลงในรูป 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(): # ตรวจเช็คว่ากล้องพร้อมใช้งานหรือไม่
    print("Cannot open webcam") # ถ้าไม่จะแสดงข้อความ Cannot open video

while cap.isOpened(): # ถ้ากล้องยังใช้งานอยู่ก็ให้วนลูปต่อไป
    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 คือนำค่าที่ขึ้นบรรทัดใหม่มาเก็บไว้ในรูปแบบ List และวนอ่าน
ค่าซ้ำมาเก็บไว้ในตัวแปร 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 ในการสร้างกรอบสี่เหลี่ยม
        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 # ถ้าใช่ก็จบการวนลูปของ while
cap.release() # ทิ้ง Task ที่กำลังทำงานอยู่ออกไปให้หมด
cv2.destroyAllWindows() # เป็นฟังก์ชันที่ใช้ในการปิดหน้าจอการทำงาน
```

4. Face Recognition

ระบบการรู้จำใบหน้า หรือ ระบบการจดจำใบหน้า (อังกฤษ: Face Recognition system) คือ ระบบการตรวจหาใบหน้าของมนุษย์และปรับภาพใบหน้าโดยอัตโนมัติ กรอบจะปรากฏขึ้นบนใบหน้าที่ถูกตรวจจับ และโฟกัส สี และค่าการวัดแสงจะถูกปรับโดยอัตโนมัติ นอกจากนั้นเมื่อบันทึกด้วยคุณภาพแบบ HD เทคโนโลยีการบีบอัดจะจัดสรรความจุของข้อมูลให้ลดลง แต่ได้ข้อมูลที่เป็นประโยชน์มากขึ้นเพื่อปรับคุณภาพของภาพ ข้อมูลที่ได้จะถูกนำไปเปรียบเทียบกับข้อมูลตัวอย่างที่เก็บบันทึกไว้ อาจจะทั้งใบหน้า หรือเพียงบางส่วน ขึ้นกับชนิดของวิธีแยกเอกลักษณ์ใบหน้า ระบบการรู้จำใบหน้าเป็นส่วนหนึ่งของ เทคโนโลยีปัญญาประดิษฐ์ในส่วนเนื้อหาของเรื่อง การรับรู้ของเครื่อง (Machine perception)

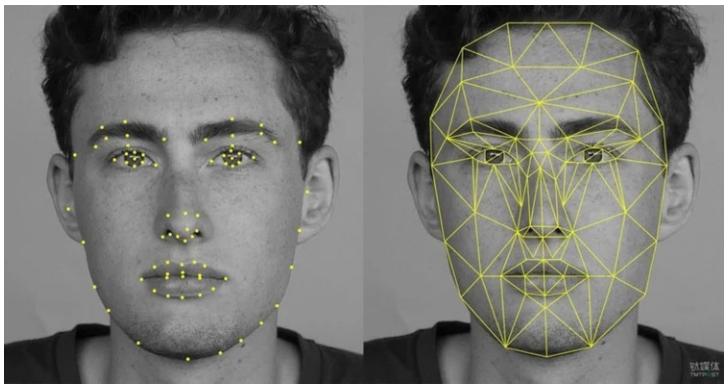


Figure 50 เทคโนโลยีการจดจำใบหน้า

ประวัติของระบบจดจำใบหน้า

จุดกำเนิดของเทคโนโลยี Face Recognition ปี 1960s สามารถจำแนกภาพ ใบหน้าแบบ Manua ผ่านการใช้งาน RAND Tablet สามารถใช้ในการบันทึกและจำแนกอัตลักษณ์ต่าง ๆ บนใบหน้า อาทิ ตา หู จมูก ปาก และไรผม ปี 1970s พัฒนาความแม่นยำให้มีความแม่นยำมากขึ้น โดยระบุอัตลักษณ์เจาะจง 21 จุด อาทิ สีผม ความหนาของริมฝีปาก แต่ระบบยังต้องใช้การคำนวณวัดและระบุตำแหน่งจากผู้ใช้ เช่นเดิม 1980s / 90s ใช้การวิเคราะห์ องค์ประกอบหลักกับใช้เทคนิคพีชคณิตเชิงเส้นทั่วไป เพื่อแก้ไขจุดบกพร่องเดิมที่ต้องคำนวณวัดค่าและระบุตำแหน่งแบบ Manua 1990s / 2000s โปรแกรม FERET สร้างฐานข้อมูลภาพใบหน้า ในชุดทดสอบมีภาพนิ่ง 2,413 ภาพ คิดเป็น 856 คน การทำภาพทดสอบสำหรับการจดจำใบหน้าได้สร้างแรงบันดาลใจให้เกิดนวัตกรรมและส่งผลให้เทคโนโลยีการจดจำใบหน้ามีประสิทธิภาพมากขึ้น 2000s มีคนกังวลเรื่องสิทธิส่วนบุคคล และความปลอดภัย รัฐบาลสหรัฐฯ จึงบังคับใช้กฎหมายสำหรับข้อมูลที่ใช้ในการปรับใช้เทคโนโลยีการจดจำใบหน้า 2006 อัลกอริทึมการจดจำใบหน้าล่าสุดที่มีอยู่ มีการใช้ภาพใบหน้าความละเอียดสูงการสแกนใบหน้า 3 มิติและภาพม่านตาในการทดสอบ ผลการวิจัยระบุว่าอัลกอริทึมใหม่มีความแม่นยำมากกว่าอัลกอริทึมการจดจำใบหน้าที่ผ่านมา 2010-ปัจจุบัน เริ่มใช้ฟังก์ชันการจดจำใบหน้าที่ช่วยระบุผู้คนที่ใบหน้าอาจมีรูปภาพที่ผู้ใช้ร่วมกับการจดจำใบหน้าให้เข้ากับชีวิตประจำวันของเรา

การตรวจหาใบหน้า

การตรวจหาตำแหน่งของใบหน้า เป็นจุดเริ่มต้นของระบบการรู้จำใบหน้า การตรวจหาใบหน้า เป็นการตรวจหาใบหน้าทั่ว ๆ ไป ยังไม่สามารถแยกแยะได้ว่า หน้านั้นเป็นหน้าของใคร วิธีที่นิยมคือ วิธีของวีโอลา โจนส์ พัฒนาขึ้นเมื่อปี พ.ศ. 2544 โดยพอล วีโอลา และไมเคิล โจนส์ อัลกอริธึมของวีโอลา โจนส์ นี้ถูกบรรจุไว้ใน OpenCV ในชื่อ cvHaarDetectObjects() การตรวจหาใช้วิธีตรวจหาตำแหน่งของ จมูก และ ตาทั้งสองข้าง ในรูปของเมตริก แนวตั้งและแนวนอน

วิธีการทำงานของ Face Recognition

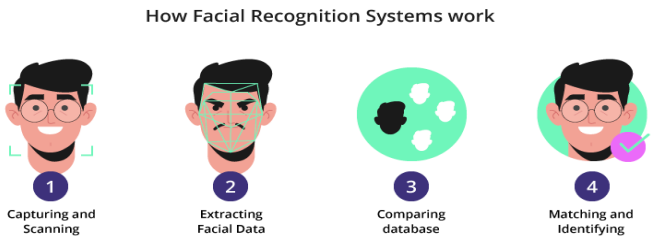


Figure 51 Face Recognition system

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 จะเป็นขั้นตอนหลังจากที่มีการส่งข้อมูลเข้าไปเปรียบเทียบกับข้อมูลใน data base ที่มีอยู่ เพื่อให้สามารถระบุได้ว่าบุคคลผู้นั้นเป็นใคร

Modern Face Recognition with Deep Learning

สำหรับการรู้จำใบหน้าด้วยไลบารี face_recognition จะมีกระบวนการในการทำงานดังนี้

Histogram of Oriented Gradients (HOG)

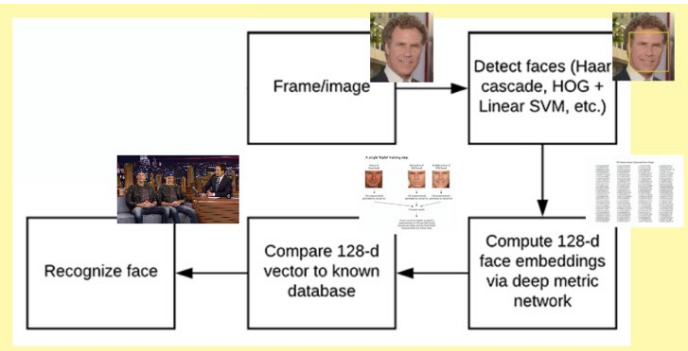


Figure 52 กระบวนการ Face Recognition

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 53 การนำเข้ารูปภาพเพื่อประมวลผลสำหรับ Histogram of Oriented Gradients (HOG)

ขั้นที่ 2 การค้นหาใบหน้าเพื่อการตัดส่วนที่ไม่จำเป็นในการรู้จำใบหน้าเช่น พื้นหลัง วัตถุอื่นๆที่ไม่ใช่ใบหน้า โดย HOG จะทำการการคำนวณพิกเซล โดยการคูณเมทริกของภาพ ผลลัพธ์ที่ได้จากการคำนวณจะเป็นจำนวนการไล่ระดับสีที่ชี้ในแต่ละทิศทางหลัก (จำนวนจุดขึ้น ชี้ขึ้นขวา ชี้ไปทางขวา ฯลฯ) จากนั้นจะแทนที่สี่เหลี่ยมจัตุรัสนั้นในภาพด้วยทิศทางลูกศรที่แข็งแกร่งที่สุด ผลลัพธ์ที่ได้คือการเปลี่ยนภาพต้นฉบับให้กลายเป็นภาพจำลองที่เรียบง่าย ซึ่งรวบรวมโครงสร้างพื้นฐานของใบหน้า โดยผลลัพธ์ที่ได้แสดงทิศทางและขอบของรูปภาพ และนำผลที่ได้มาเปรียบเทียบกับโมเดลที่ฝึกจากใบหน้าอื่นจำนวนมากเพื่อค้นหาตำแหน่งของใบหน้า โดยการค้นหาส่วนที่คล้ายกัน ดังภาพที่ 54

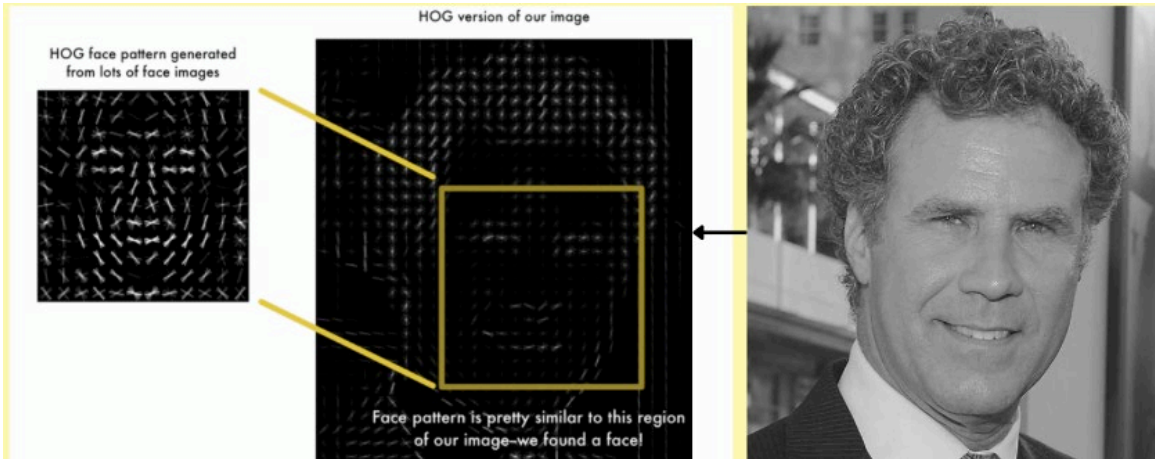


Figure 54 การค้นหาใบหน้า Histogram of Oriented Gradients (HOG)

Figure 55 ผลลัพธ์จากการค้นหาตำแหน่งใบหน้าด้วย Histogram of Oriented Gradients (HOG)

ขั้นที่ 3 การแปลงค่า 128 มิติจากใบหน้าโดยขั้นตอนนี้ เริ่มจากการแปลงรูปภาพเป็นค่าที่มีสกัดคุณลักษณะ (Feature Extractions) จากใบหน้า 128 มิติ โดยโมเดล Dlib จะทำการกำหนดจุดแลนด์มาร์คของใบหน้าที่ได้จากการค้นหาตำแหน่งใบหน้าที่ โดยมีอยู่การกำหนดจุดแลนด์มาร์ค 2 แบบ คือ กำหนดจุดใบหน้า 5 จุด และ 68 จุด ดังภาพที่ 56 ผลลัพธ์ที่ได้ คือ ตำแหน่งของตาและปากของใบหน้า เพื่อทำการหมุด ปรับขนาด และตัดภาพ ให้ดวงตาและปากอยู่ตรงกลางของภาพมากที่สุด โดยพยายามรักษาไม่ให้ภาพบิดเบี้ยวน้อยที่สุด โดยใช้การแปลงภาพพื้นฐานเท่านั้น เช่น การหมุนและสเกลที่รักษาเส้นคู่ขนาน (called affine transformations) หลังจากนั้นจะเข้าสู่กระบวนการแปลงใบหน้าที่ได้เป็นค่า 128 มิติดังภาพที่ 57 โดยโครงข่ายประสาทเทียมแบบคอนโวลูชั่น (Convolutional Neural Network : CNN)

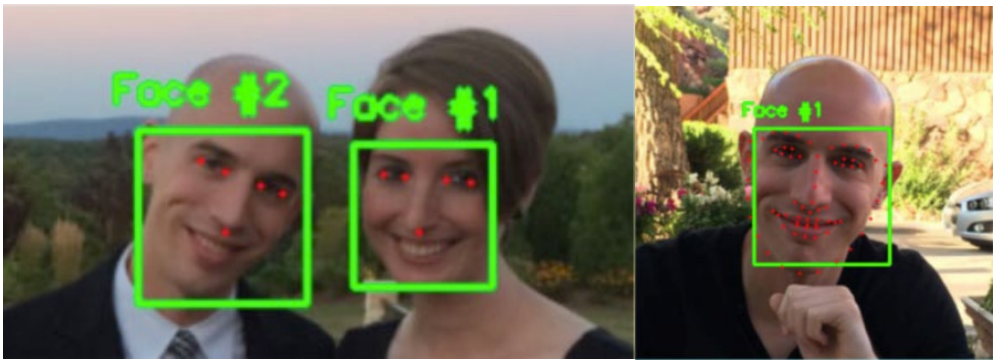


Figure 56 การกำหนดจุดสำคัญบนใบหน้า (face landmarks) การกำหนดจุดแบบ 5 จุด (ซ้าย) การกำหนดจุดแบบ 68 จุด (ขวา)

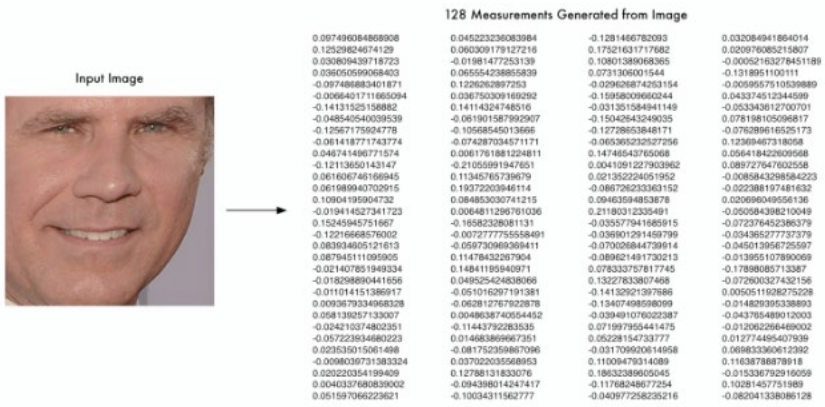


Figure 57 ค่าคุณลักษณะใบหน้าที่ได้ 128 มิติ

ขั้นที่ 4 การเปรียบเทียบค่าที่ได้จากการแปลงค่ากับฐานข้อมูล โดยจะนำค่าที่ได้ 128 มิติจากใบหน้ามาเปรียบเทียบโดยใช้กระบวนการ Triplet loss function โดยกระบวนการนี้มีจุดเด่นคือ ลดระยะเวลาในการคำนวณและลดปัญหาในการสร้างคลาสจากใบหน้ากรณีมีใบหน้าเพิ่ม 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 ระว่าง anchor และ positive (เป็นบุคคลเดียวกัน) และเพิ่ม distance ระหว่าง anchor และ negative (เป็นคนละคนกัน) ดังภาพที่ 59 จะได้ผลลัพธ์ของการรู้จำใบหน้ากรณีที่ค่าใกล้กันมีโอกาสที่จะเป็นบุคคลเดียวกัน โดยการกำหนดจากค่า TOLERANCE หรือค่าที่เรากำหนดขึ้น

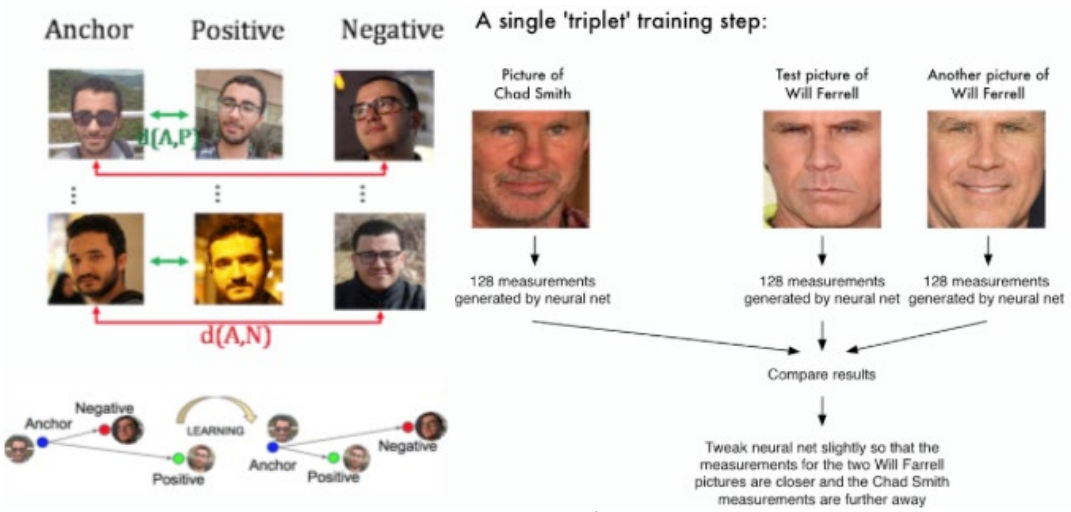


Figure 58 กระบวนการ Triplet loss เพื่อการรู้จำใบหน้า

ขั้นที่ 5 การนำผลลัพธ์มาแสดงผลเป็นระบบรู้จำใบหน้า

Figure 59 ผลลัพธ์ที่ได้การรู้จำใบหน้า

Convolutional Neural Network (CNN)

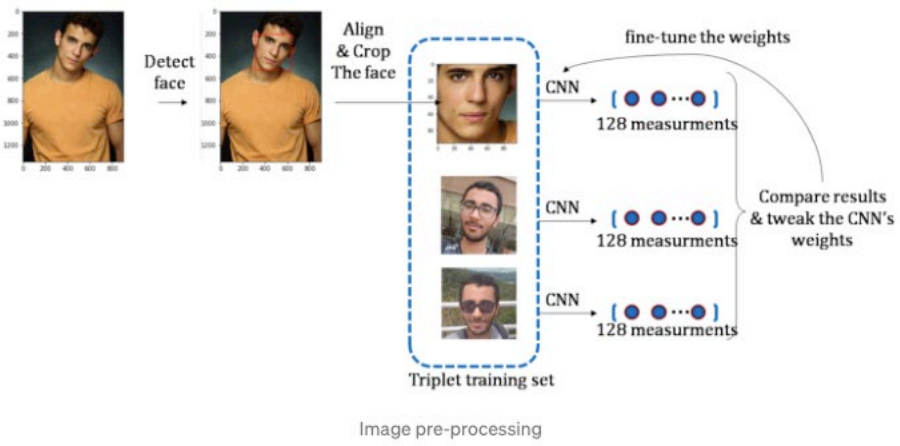


Figure 60 Face Recognition from Convolutional Neural Network (CNN)

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

ประโยชน์ของ Face Recognition

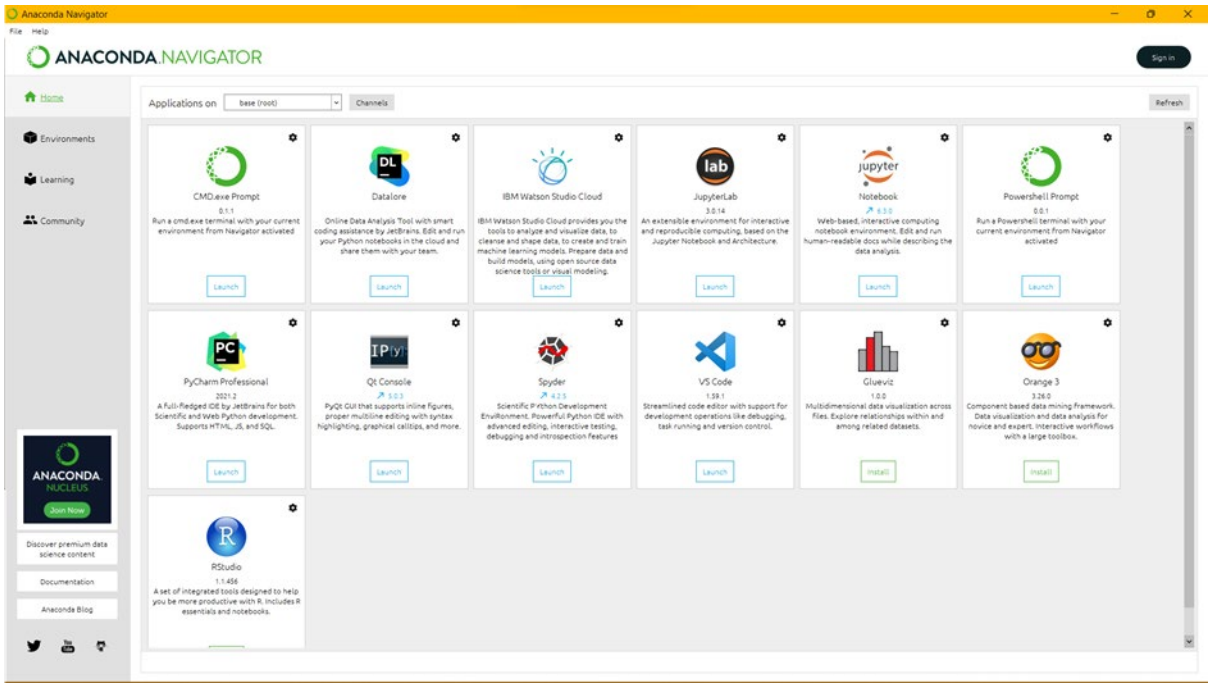
เห็นได้ชัดว่าฟีเจอร์นี้จะเน้นไปที่การจดจำใบหน้าของบุคคลที่ปรากฏบนกล้อง CCTV ดังนั้น จึงถูกนำมาใช้อย่างแพร่หลายได้กับระบบรักษาความปลอดภัย อุปกรณ์สแกนใบหน้าควบคุมการเปิดปิดประตู หรือระบบลงเวลาพนักงาน นอกจากนั้น ระบบสแกนใบหน้าตามอาคาร สำนักงานต่าง ๆ ที่เราคุ้นชินกัน ก็เกิดจากการปรับใช้เทคโนโลยีนี้ ซึ่งเป็นตัวอย่างที่เห็นภาพได้ง่ายมาก เพราะเมื่อมีบุคคลที่มีข้อมูลอยู่ในระบบหรือได้ผ่านการเก็บข้อมูลเรียบร้อยแล้ว เข้ามาสแกนใบหน้า ระบบจะส่งภาพที่ตรวจจับได้มาเปรียบเทียบกับฐานข้อมูลเพื่อระบุว่าใบหน้าที่ตรวจจับได้ตรงกับบุคคลใด เพื่อคัดกรองคนเข้าอาคาร ระบบจดจำใบหน้ายังสามารถปรับใช้ตามความต้องการของแต่ละธุรกิจ แต่ละอุตสาหกรรม และยังสามารถปรับใช้ภายในครัวเรือนได้ด้วย

การติดตั้ง Face Recognition

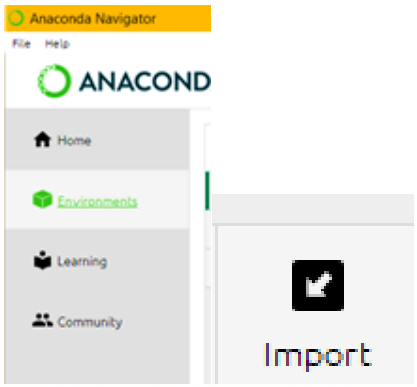
1.ให้ทำการดาวน์โหลดไฟล์ face_recognition.yml สำหรับการติดตั้ง ตามลิงค์ด้านล่าง

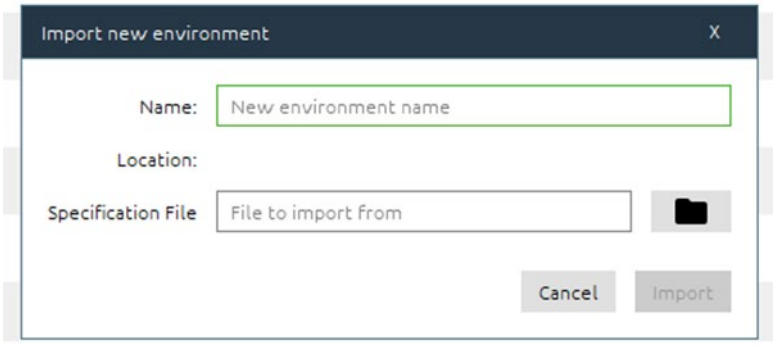
shorturl.asia/SqExp

2. เปิดโปรแกรม Anaconda

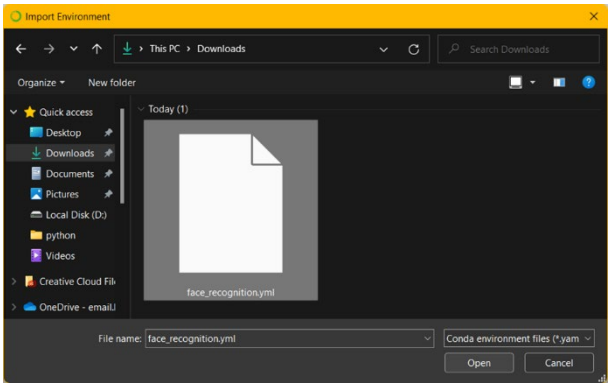


3.คลิกเลือก Environments > เลือก Import จะปรากฏหน้าต่างสำหรับเลือกไฟล์ที่ดาวน์โหลดไว้

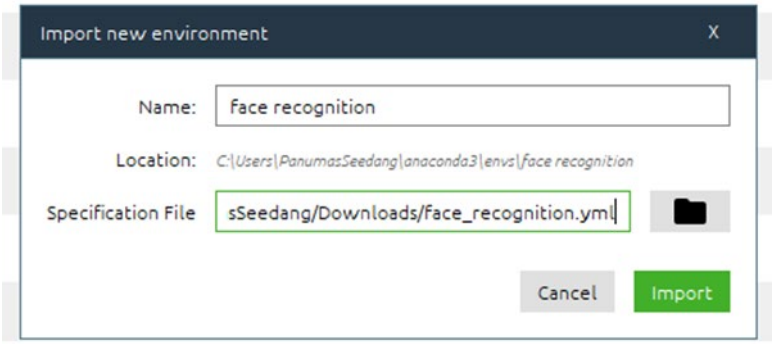




4. คลิกเลือกไฟล์ face_recognition.yml ที่ทำการดาวน์โหลดมา



5.คลิกปุ่ม Import



Work shop 1 face_rec_img

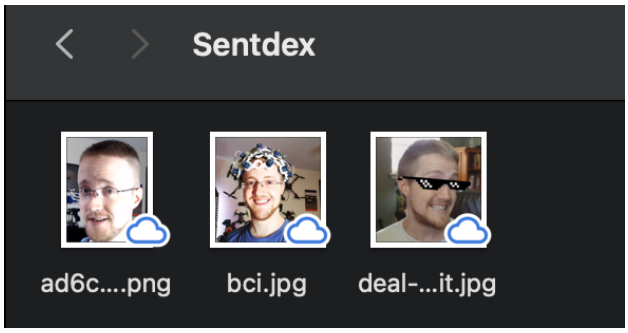
ให้ทำการดาวน์โหลดไฟล์ ตามลิงค์ด้านล่าง

shorturl.asia/SqExp

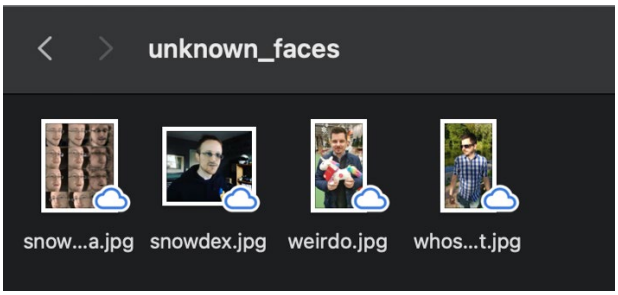
หลังจากนั้นให้ทำการย้ายตำแหน่งไฟล์ไปที่โปรเจคในตำแหน่งที่ต้องการทำโดยจะมีไฟล์ทั้งหมด 3 ส่วนคือ

1. known_faces ภายในจะประกอบด้วยโฟนเดอร์ที่เป็นชื่อของบุคคลที่ต้องการสร้างฐานข้อมูล โดยภายในแต่ละโฟนเดอร์ควรมีภาพอย่างน้อยสามภาพเพื่อความแม่นยำ เช่น

   1.1. Sentdex



2. unknown_faces ภายในจะประกอบด้วยภาพของข้อมูลที่เราต้องการตรวจสอบหรือรู้จำใบหน้า



3. 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)
#เปรียบเทียบใบหน้าที่ได้จากฐานข้อมูลกับภาพที่ต้องการตรวจสอบ
        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}')

#การใส่กรอบให้กับภาพที่จะแสดง
                        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.destroyWindow(filename)
```

Work shop 2 face_rec_camara

ให้ทำการปรับเปลี่ยนข้อมูลจากการดึงข้อมูลของไฟล์ unknown_faces เป็นการใช้กล้องในการเปรียบเทียบแทนโดยเปลี่ยนตามตำแหน่งดังต่อไปนี้

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 เป็นการใช้กล้องในการเปรียบเทียบแทนโดยเปลี่ยนตามตำแหน่งดังต่อไปนี้

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
```

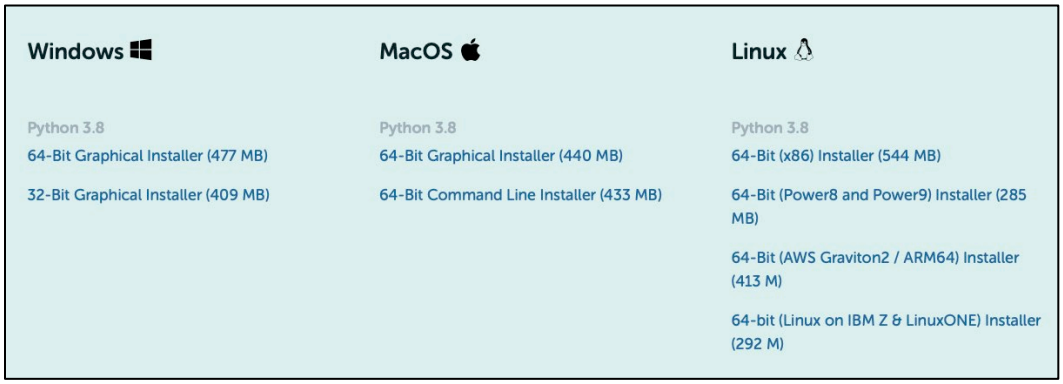
#การแสดงผลรูปภาพ

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

ภาคผนวก
คู่มือการติดตั้ง

คู่มือวิธีการติดตั้ง Anaconda

1. ทำการดาวน์โหลดโปรแกรม Anaconda จากลิ้งด้านล่าง (ภาพที่ 1)https://www.anaconda.com/products/individual#Downloads โดยเลือกดาวน์โหลดตามรุ่นระบบปฏิบัติการที่ตนเองใช้อยู่



ภาพที่ 1 หน้าเว็บดาวน์โหลด Anaconda

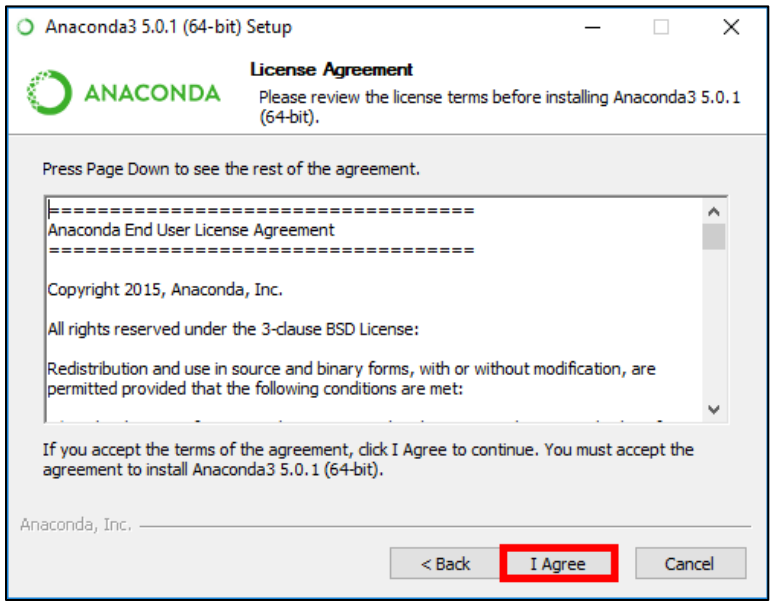
ตัวอย่างการติดตั้งในระบบ windows

1. ทำการดาวน์โหลดโปรแกรม 64-Bit Graphical Installer (477MB

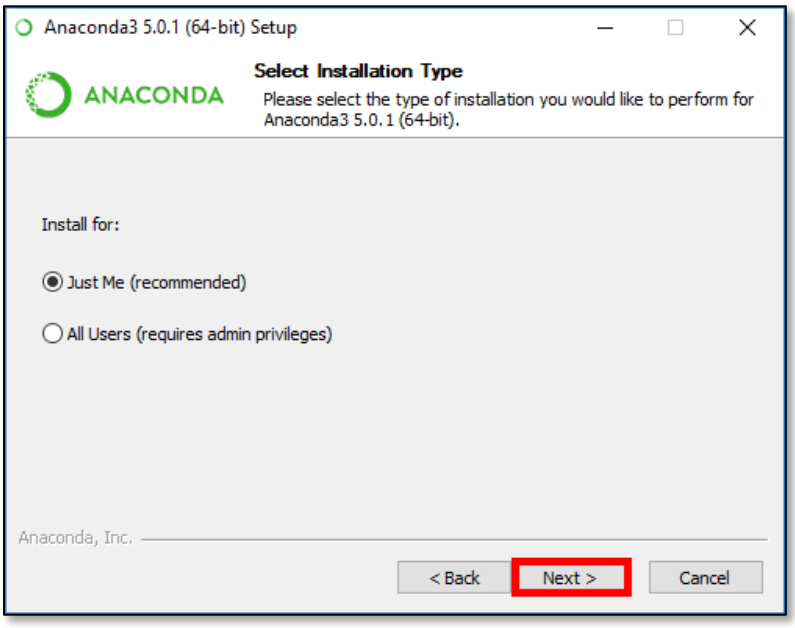
2. ดับเบิลคลิกไฟล์ที่ดาวน์โหลดมาเพื่อติดตั้ง Anaconda

3. คลิกที่ปุ่ม Next

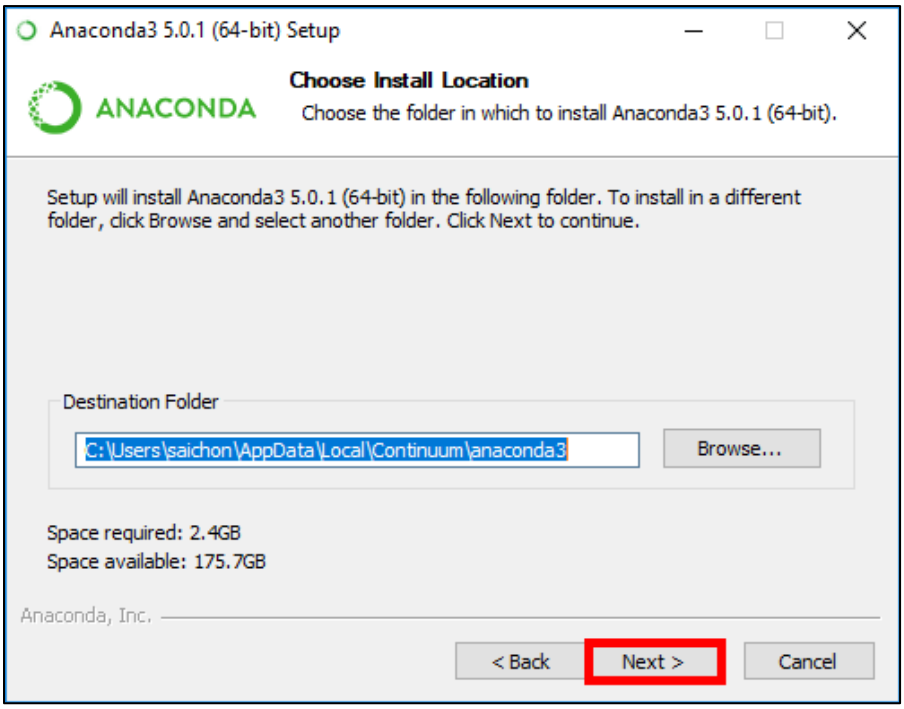
4. คลิกที่ปุ่ม I Agree



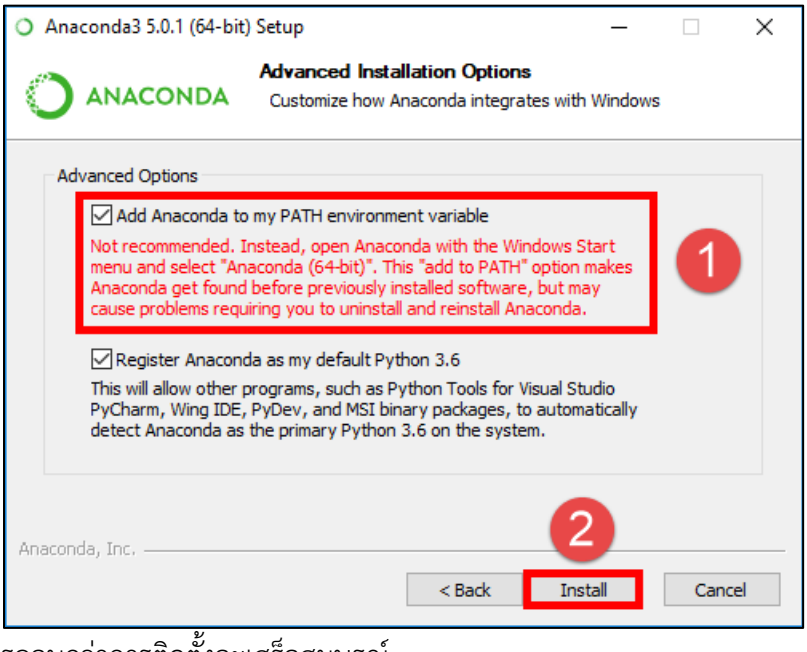
5. เลือก Just Me (recommended)แล้วคลิกปุ่ม Next



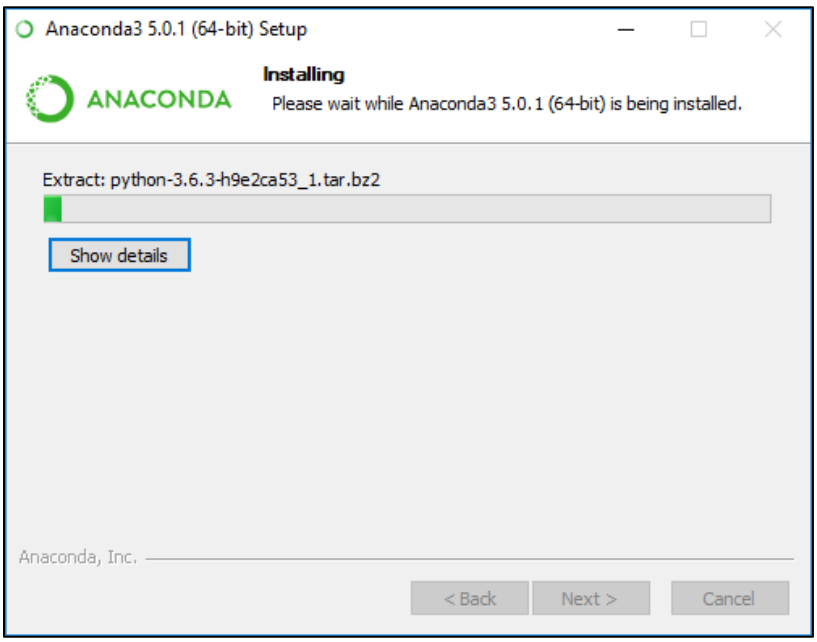
6. เลือกที่ติดตั้งตามต้องการและคลิกที่ปุ่ม Next



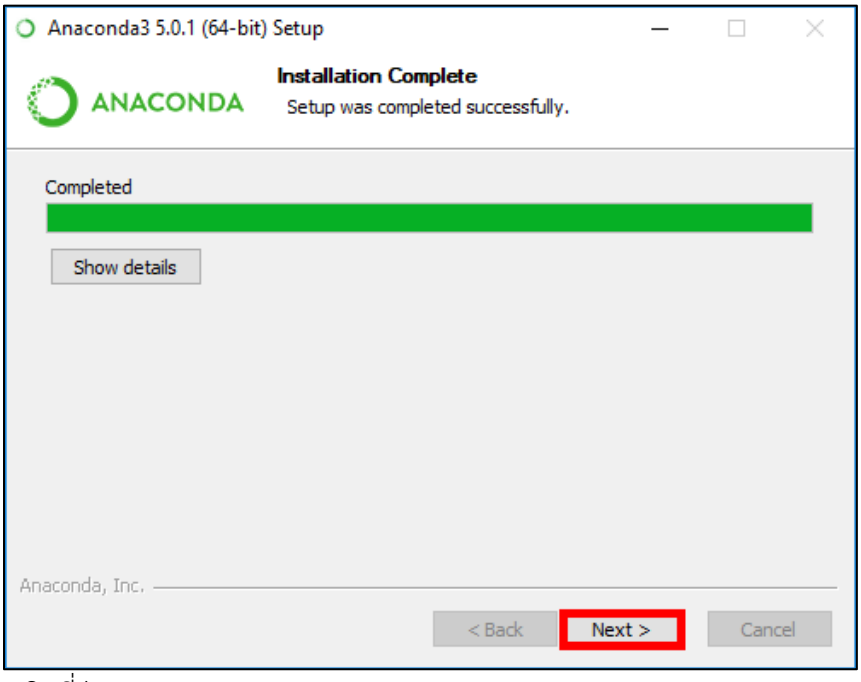
7. เลือก Add Anaconda to my PATH environment variable แล้วคลิกปุ่ม Install



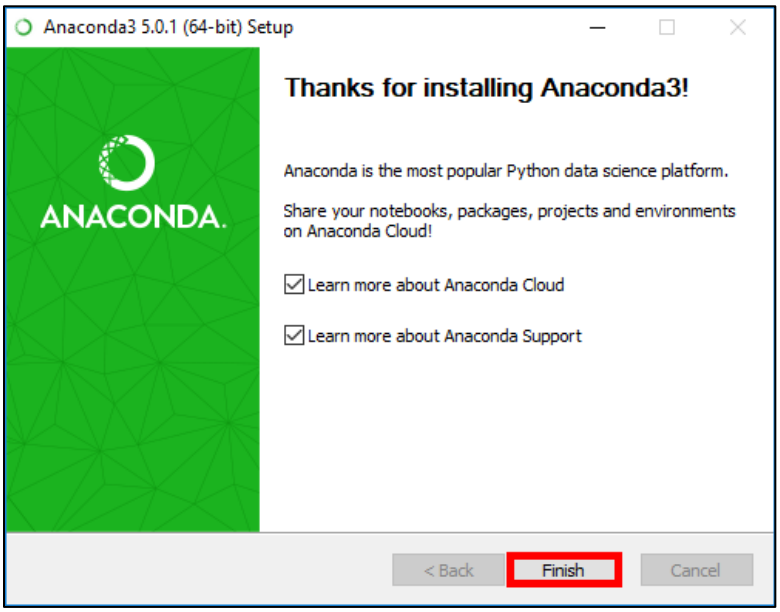
8. รอจนกว่าการติดตั้งจะเสร็จสมบูรณ์



9. เมื่อติดตั้งเสร็จสมบูรณ์แล้วให้คลิกปุ่ม Next

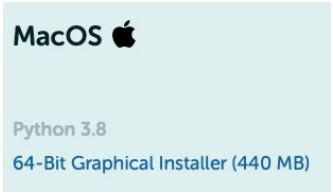


10. .คลิกที่ปุ่ม Finish

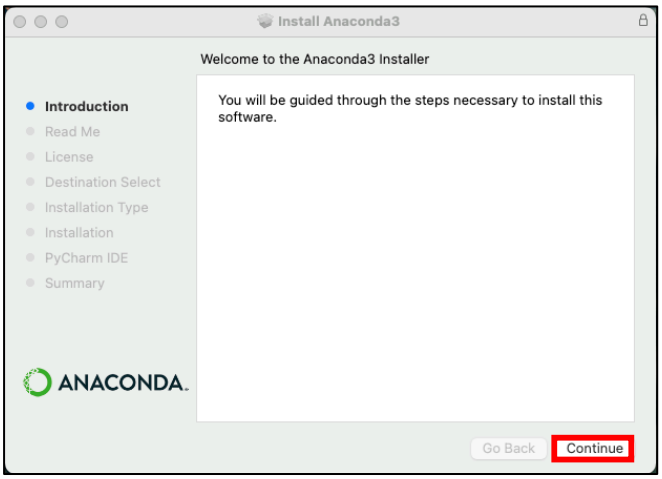


ตัวอย่างการติดตั้งในระบบ mac os

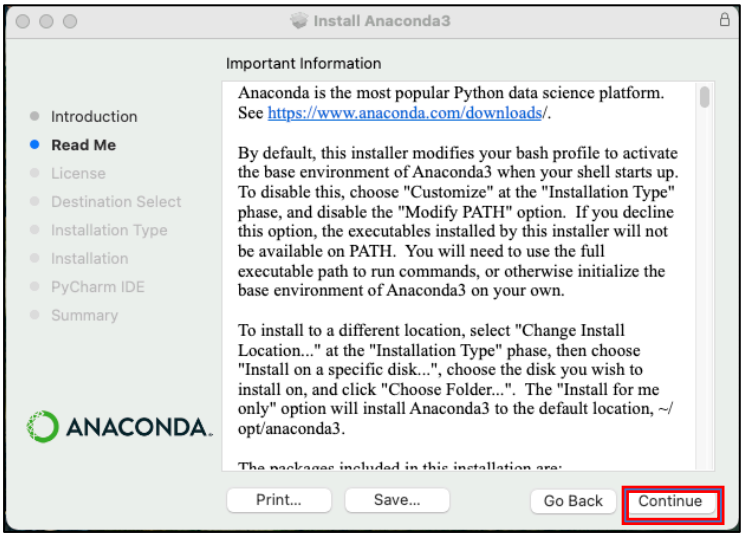
1. ทำการดาวน์โหลดโปรแกรม 64-Bit Graphical Installer (440MB)



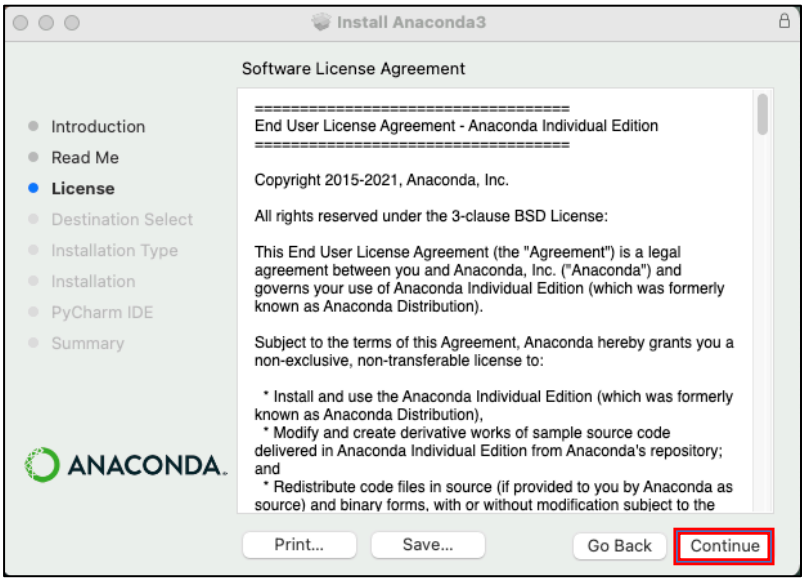
2. เปิดตัวติดตั้งขึ้นมา กดเลือก Continue



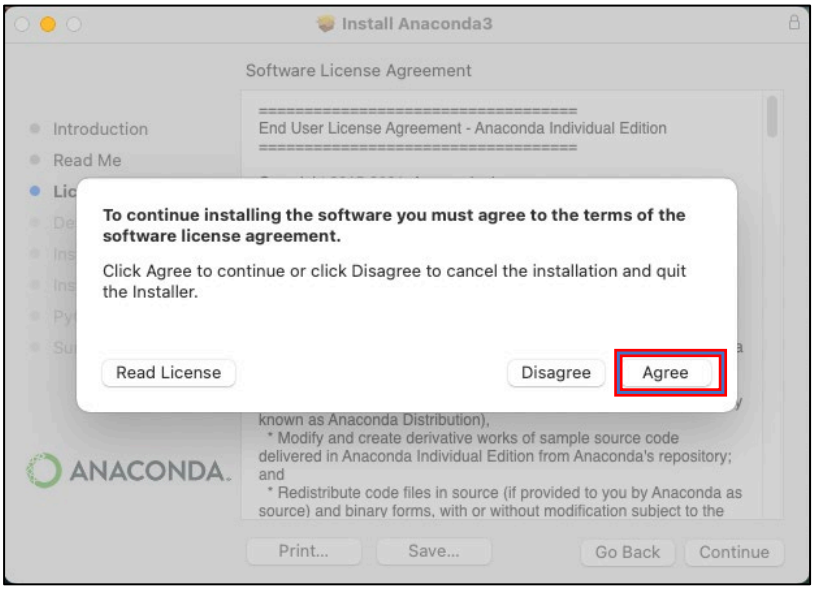
3. หน้าต่างถัดมากด Continue



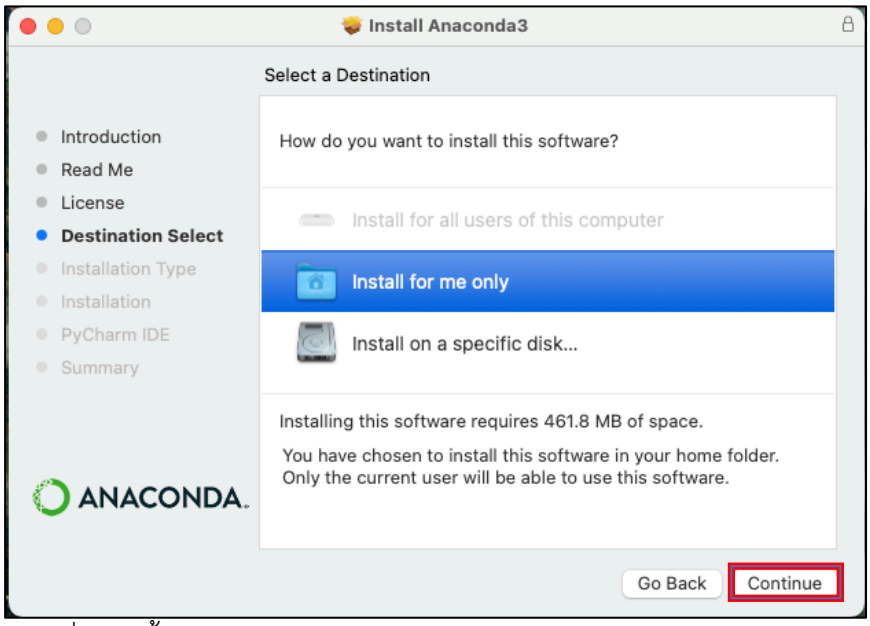
4. .หน้าต่างถัดมากด Continue



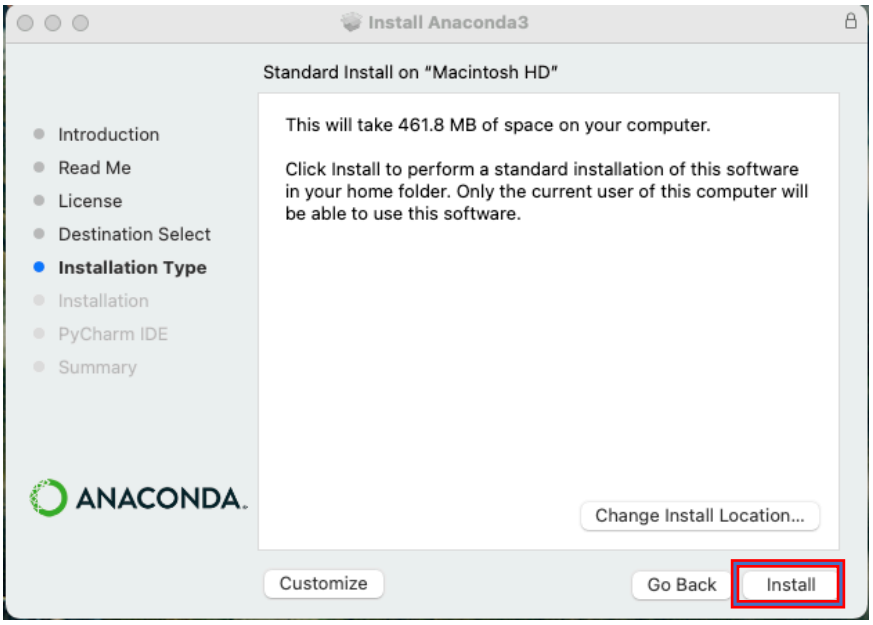
5. เลือก Agree



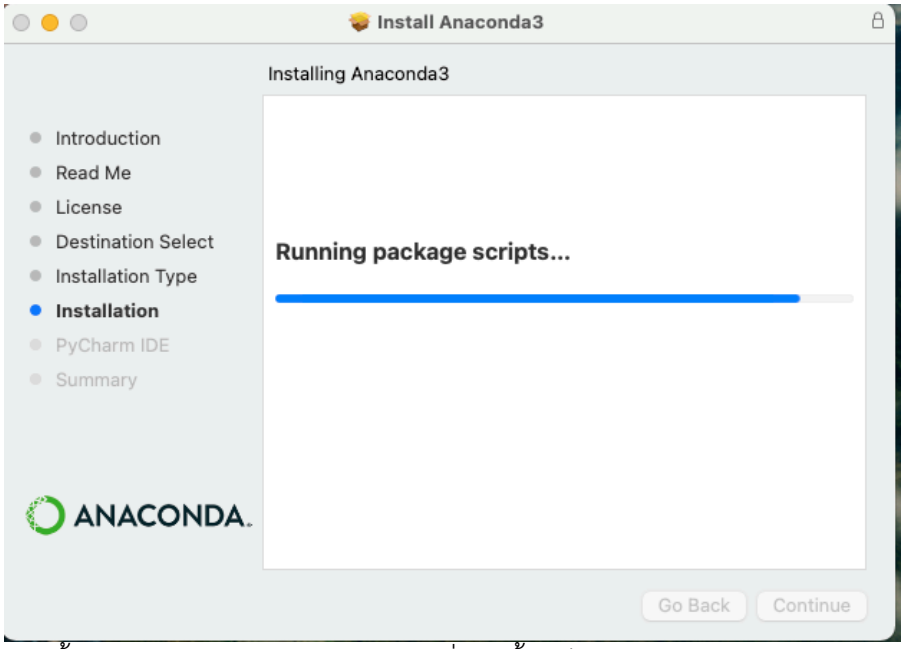
6. เลือกดิสก์ที่จะติดตั้ง จากนั้นกด Continue



7. เลือกตำแหน่งที่จะติดตั้ง กด Install

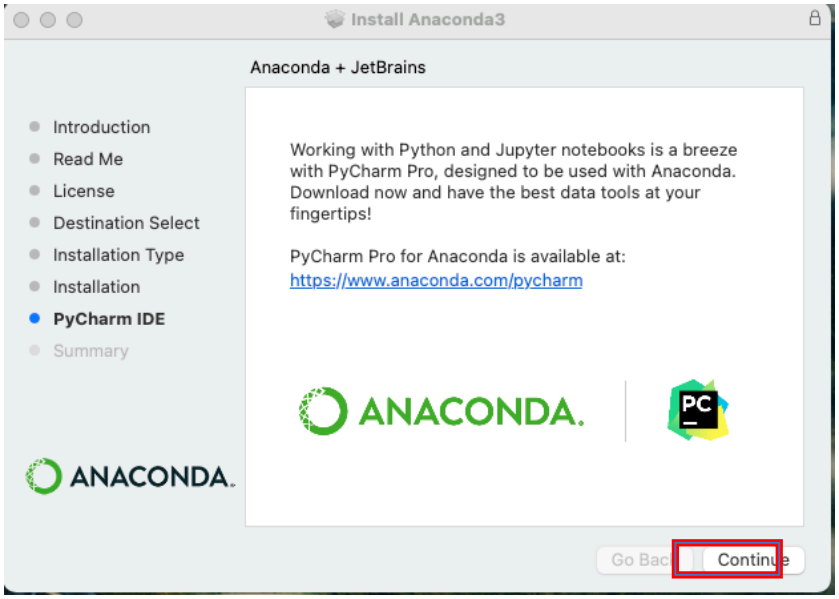


8. โปรแกรมทำการติดตั้ง

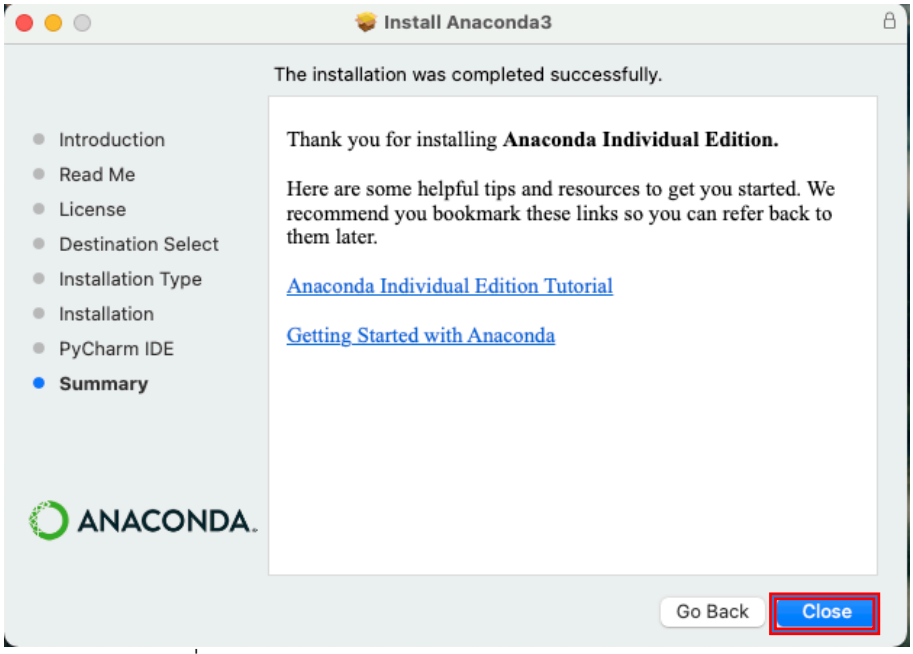


9. หากขึ้นขอ Access อะไรให้กดยอมรับ เมื่อติดตั้งเสร็จกด Continue

10. หน้าต่างถัดมากด Continue

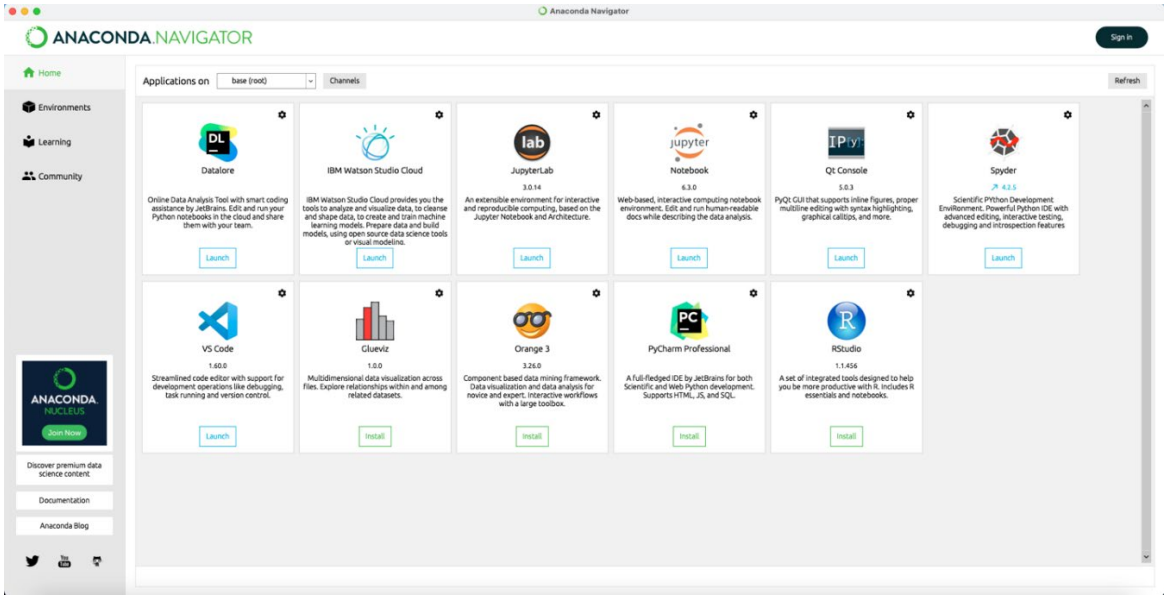


11. ติดตั้งเสร็จสิ้นกด Close



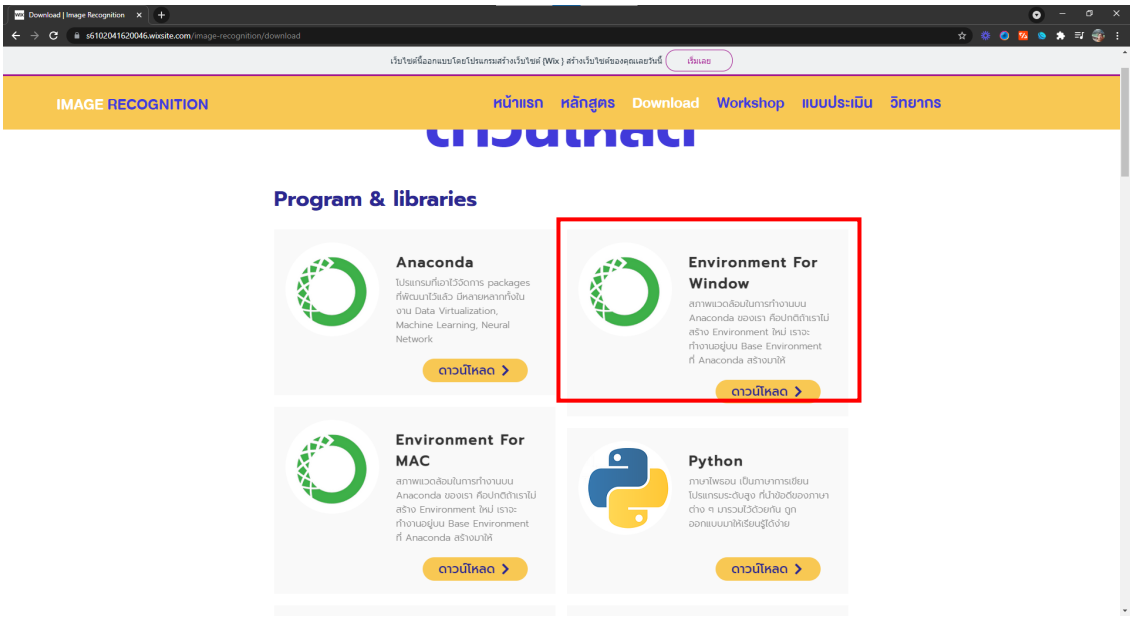
12. กดเข้าโปรแกรมที่ Lanunchpad ไอค่อนดังภาพ

13. จะได้โปรแกรมดังนี้ เสร็จสิ้นการติดตั้ง

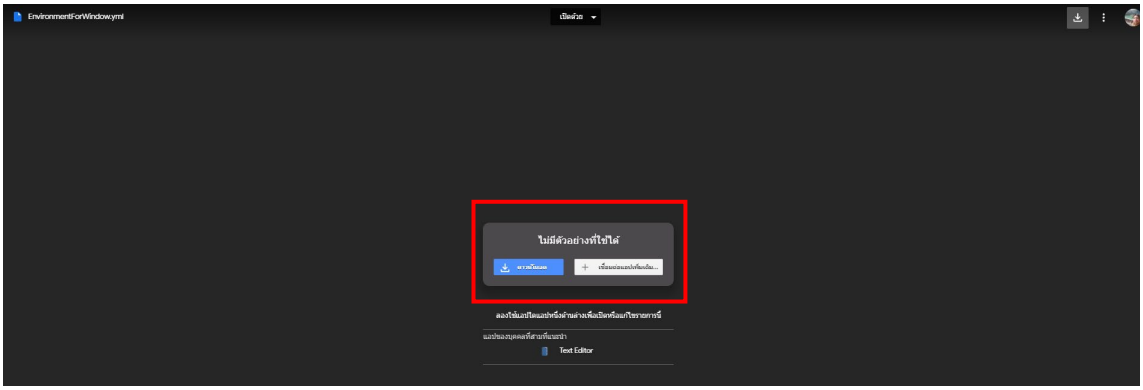


Import environment For Window

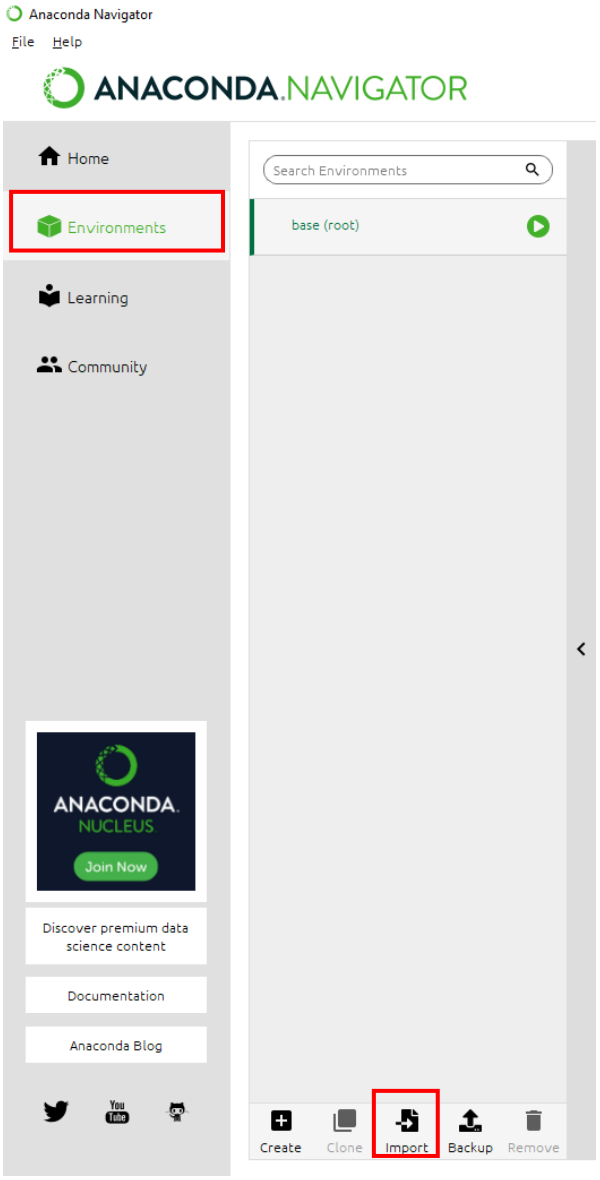
1. เข้าไปที่ https://s6102041620046.wixsite.com/image-recognition/download จากนั้นทำการดาวน์โหลดไฟล์ Environment For Window



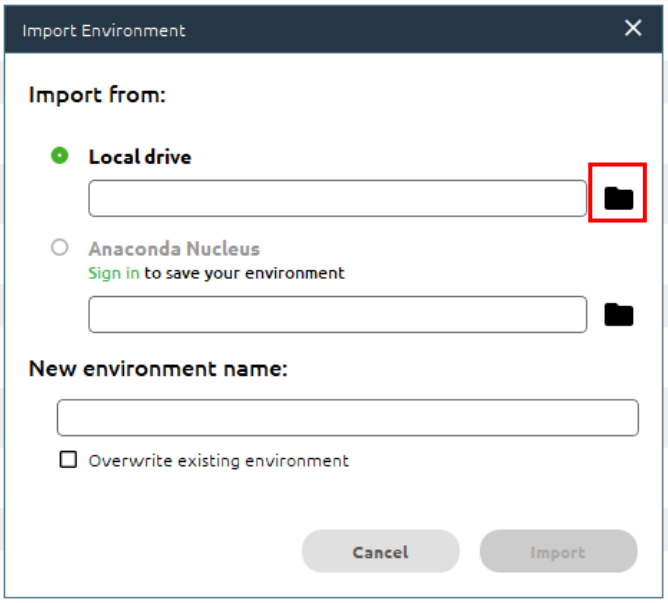
2. กดดาวน์โหลด



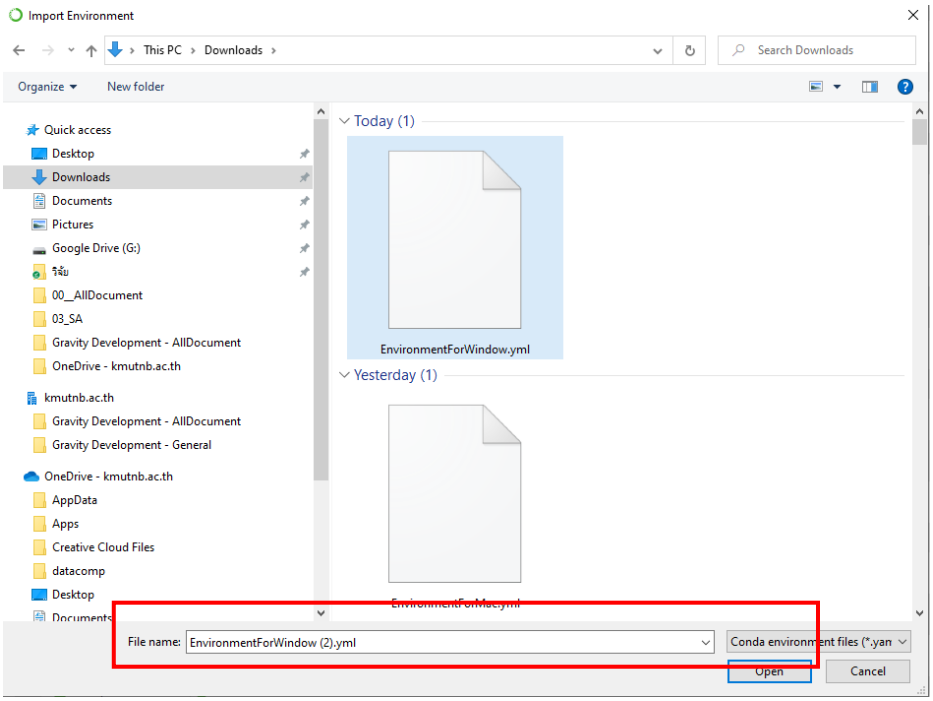
3. กดแท็บ Environments หลังจากนั้นให้ กด Import



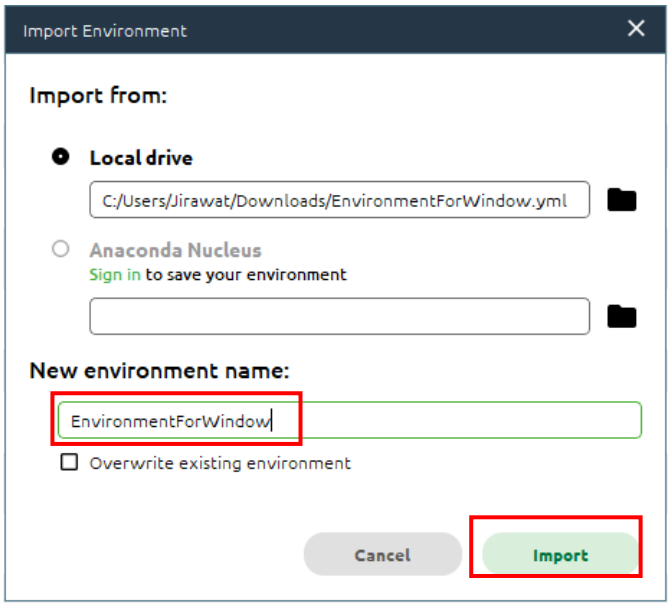
4. กดสัญลักษณ์โฟลเดอร์



5. เลือกไฟล์ environment



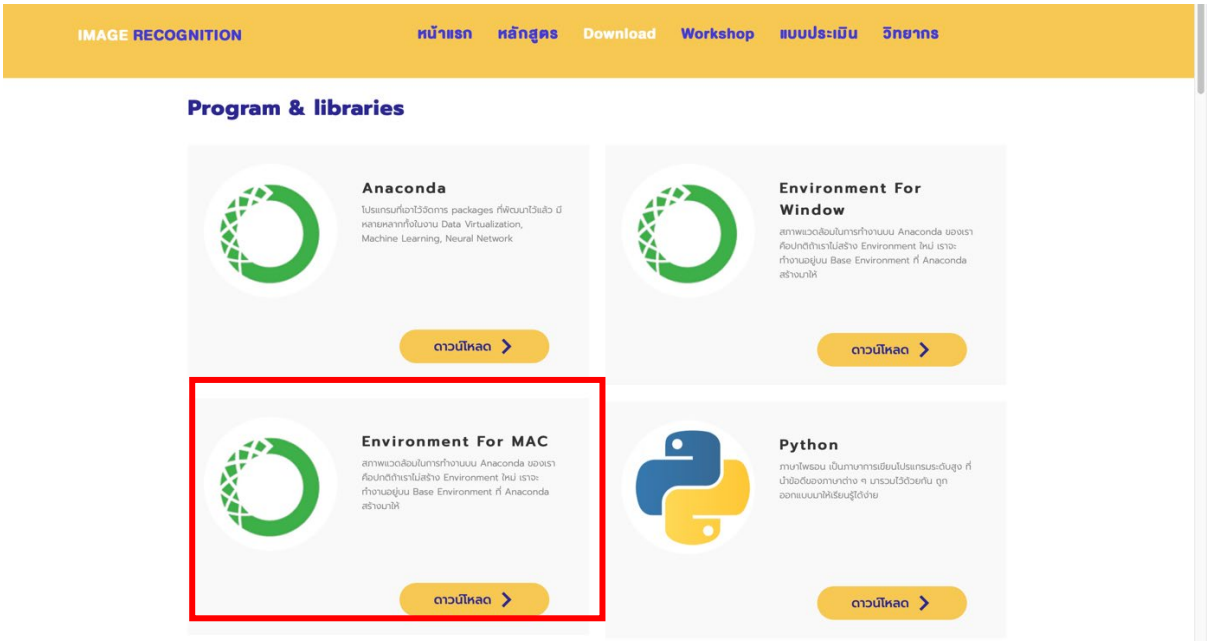
6. ตั้งชื่อ Environment for Windows



7. รอจนกว่า Import จนเสร็จ

Import environment For mac

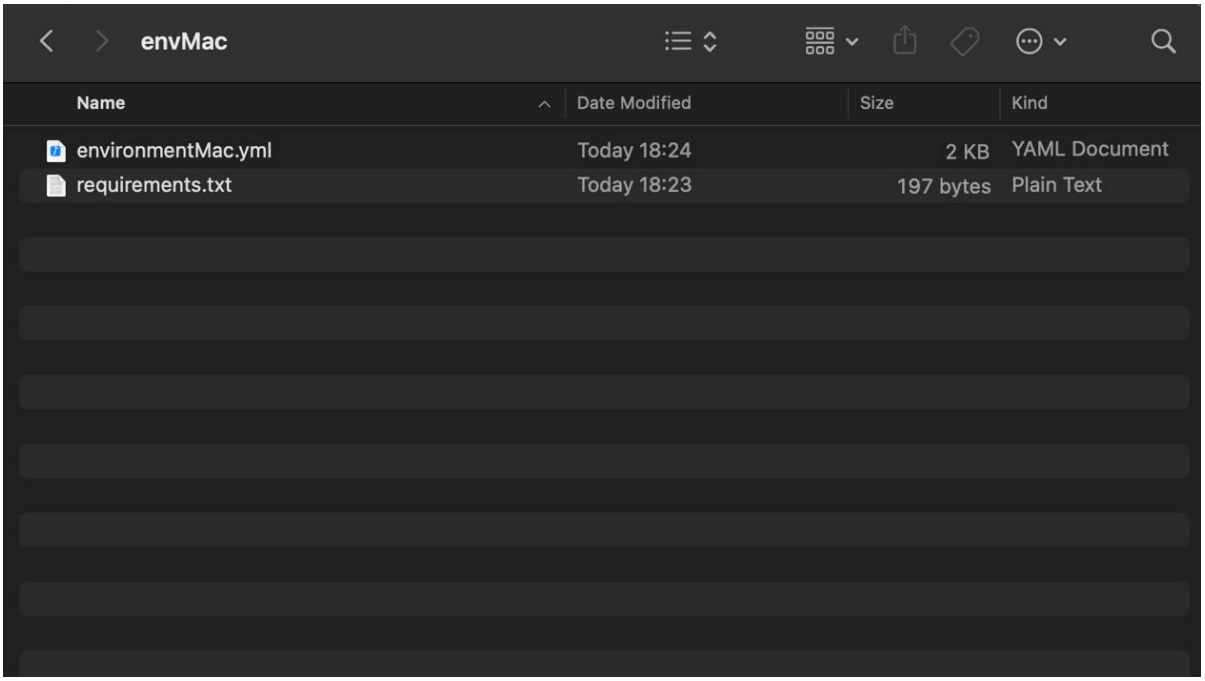
1. เข้าไปที่ https://s6102041620046.wixsite.com/image-recognition/download จากนั้นทำการดาวน์โหลดไฟล์ Environment For MAC



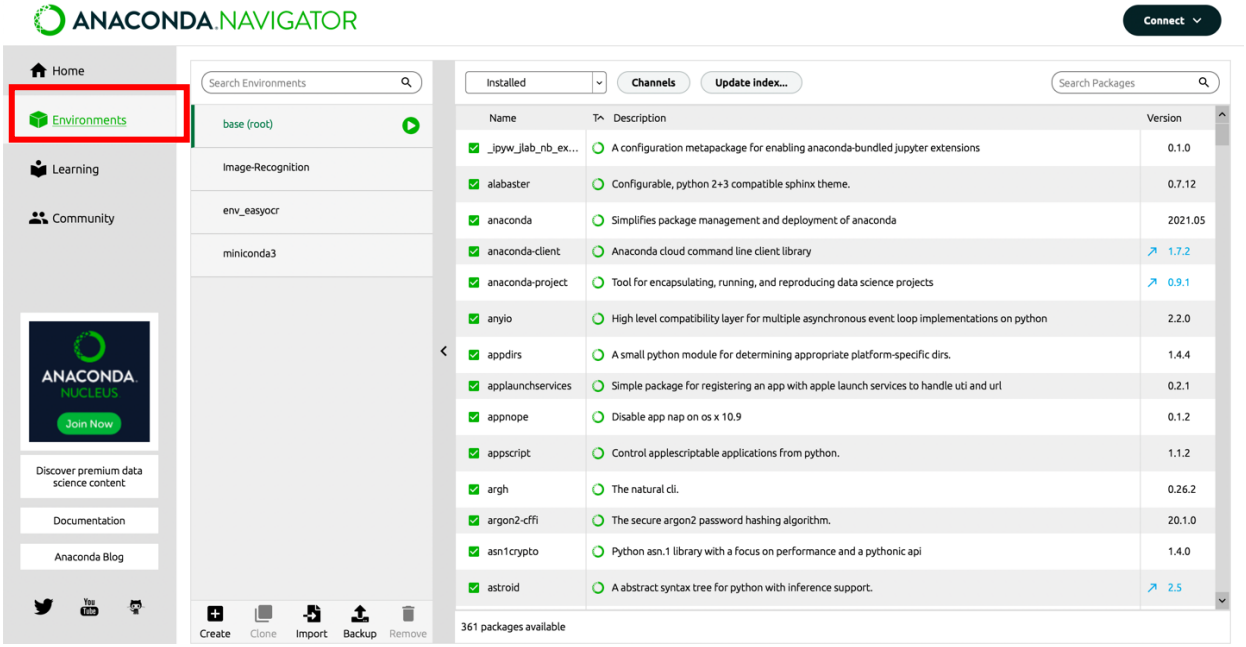
2. แตกไฟล์ที่ได้จากการดาวน์โหลดเอาไปไว้ในตำแหน่งที่เราต้องการสร้างงานโดยจะมีสองส่วนดังนี้

   2.1. ไฟล์ environmentMac.yml

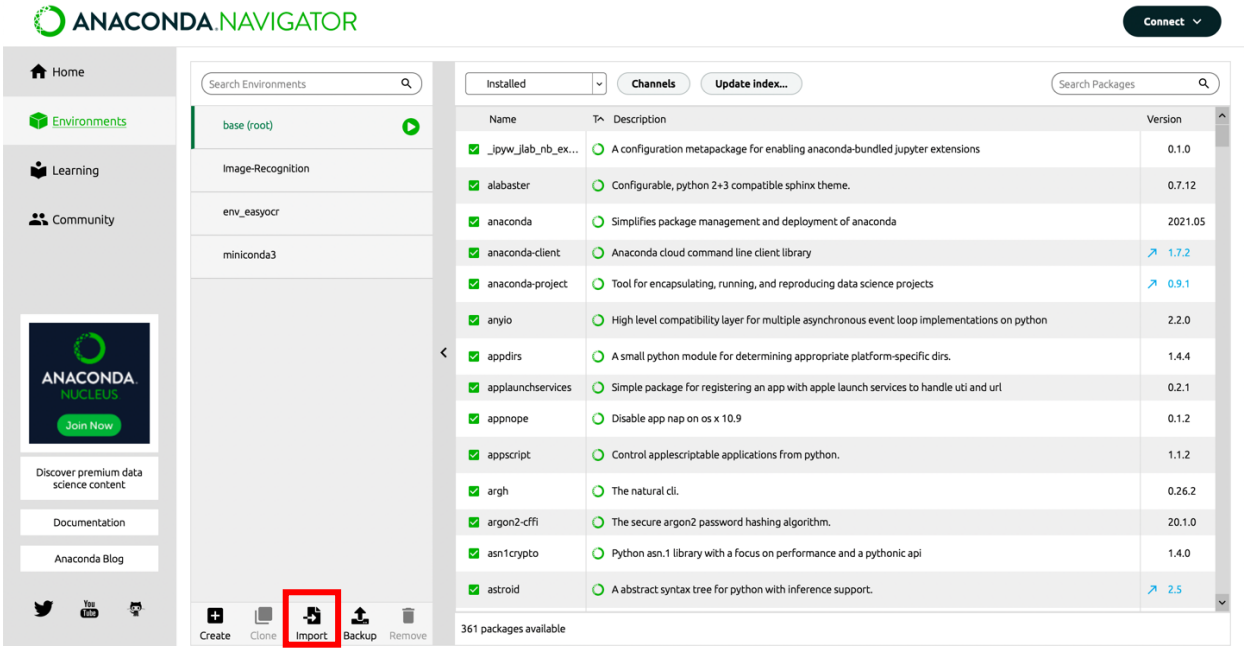
   2.2. ไฟล์ requirements.text



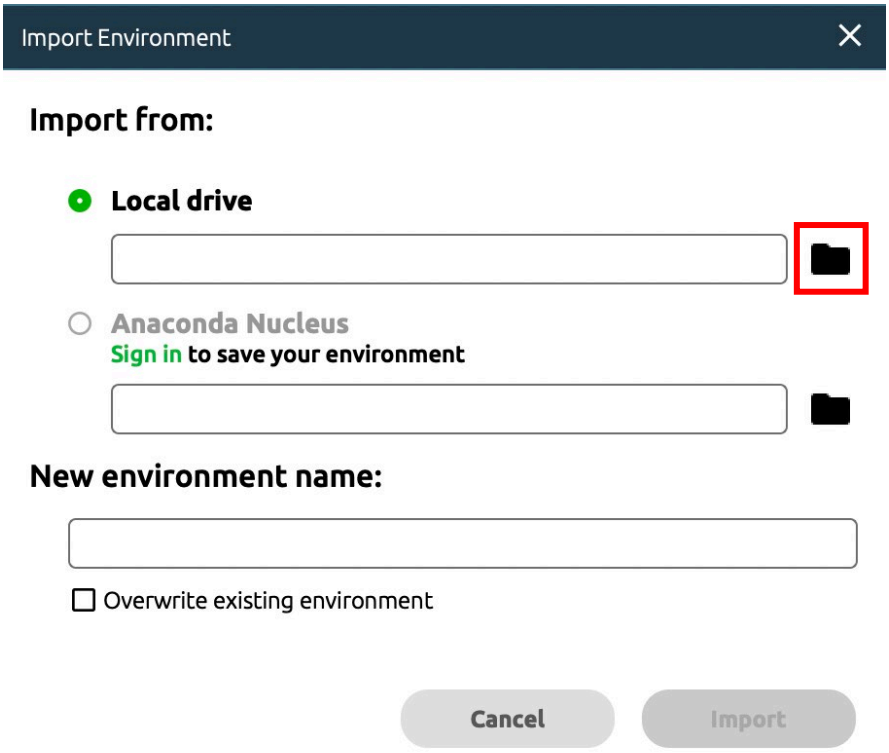
3. ทำการเปิดโปรแกรม Anaconda Navigator แล้วเลือกแถบ Environments



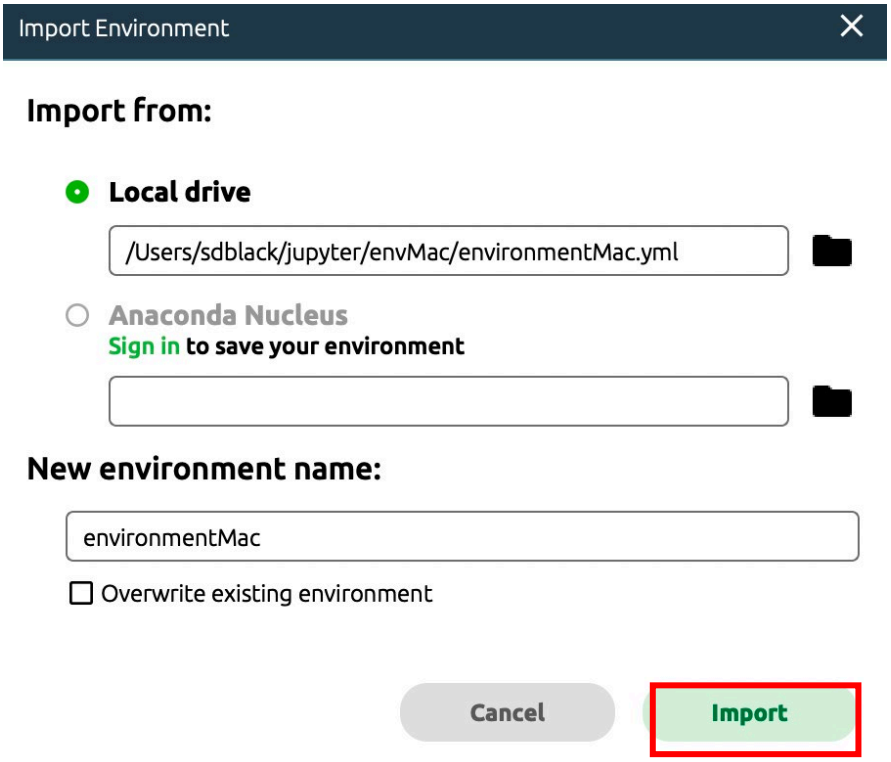
4. เลือก Import



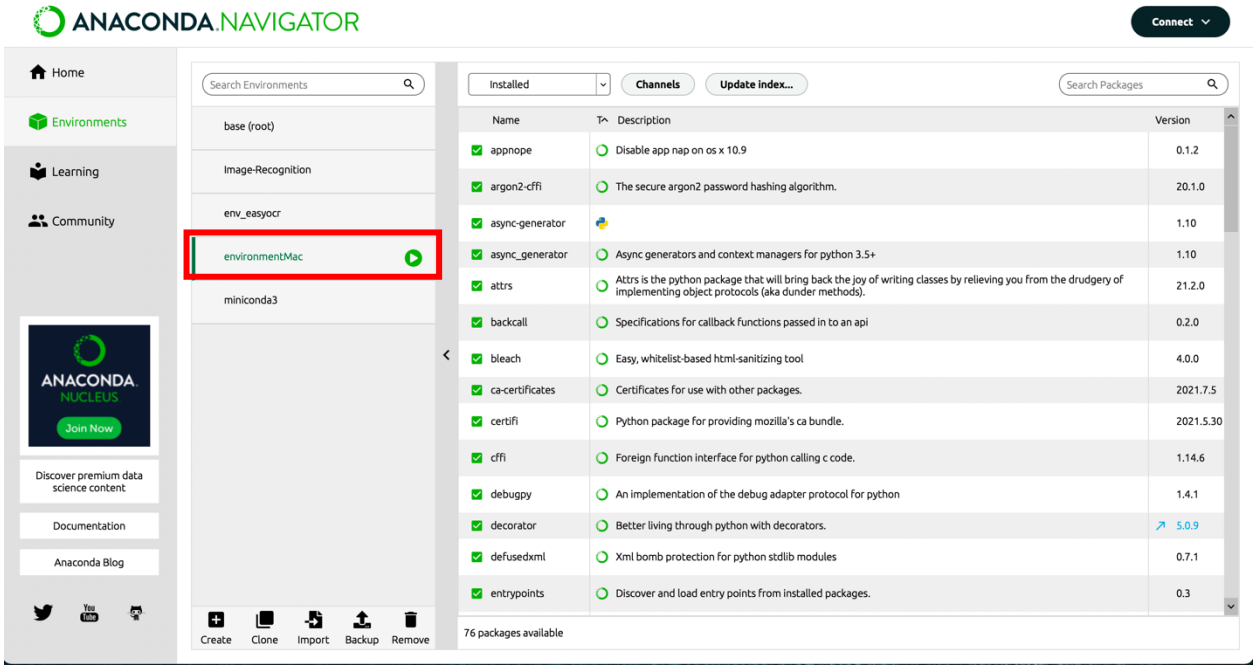
5. จะปรากฏหน้าต่างดังภาพ กดเลือกไฟล์ environmentMac.yml



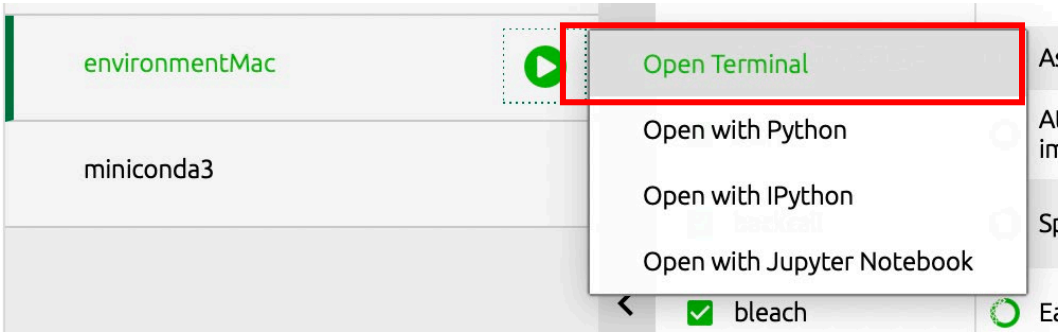
6. ทำการตั้งชื่อ environment และกด Import



7. จะปรากฏ environment ที่ทำการนำเข้ามา

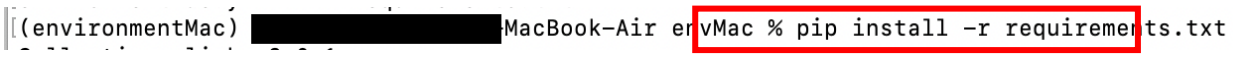


8. กดที่รูป  จากนั้นเลือก open Terminal

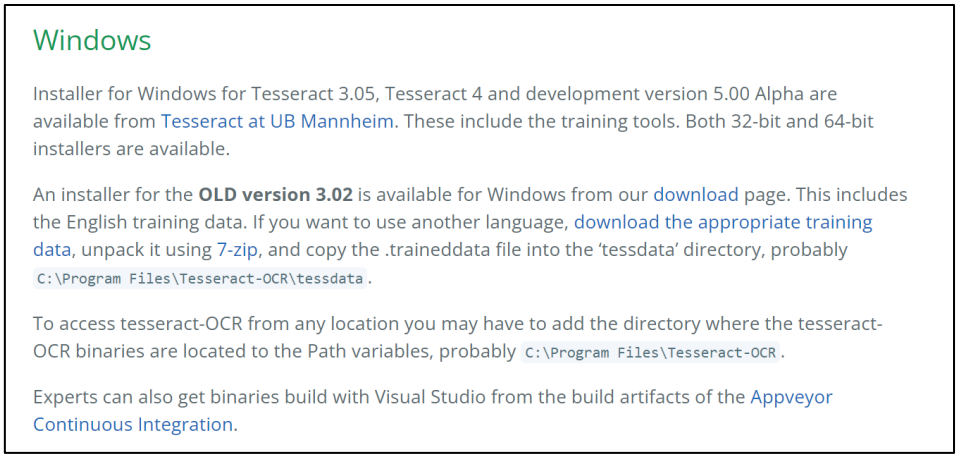


9. เข้าไปที่ตำแหน่งที่บันทึกไฟล์ requirements.text (ด้วยคำสั่งทาง Terminal เช่น cd Dowloads) จากนั้นใช้คำสั่ง

pip install -r requirements.txt



Install Tesseract For Window



Windows

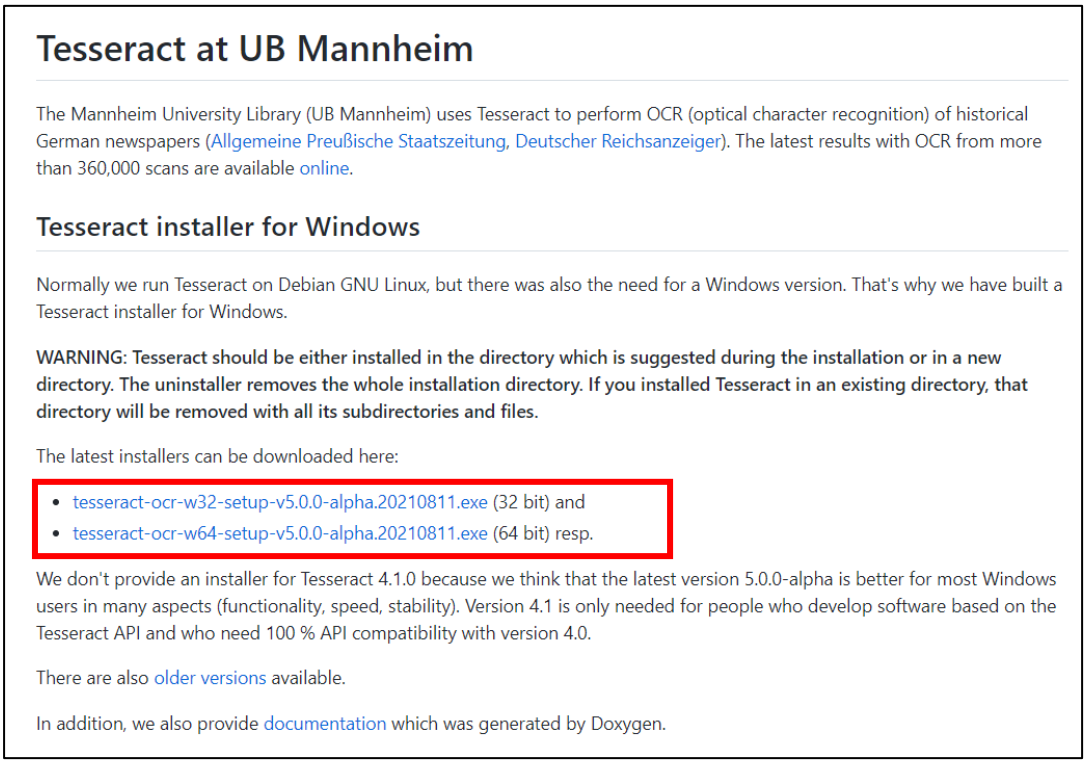
Installer for Windows for Tesseract 3.05, Tesseract 4 and development version 5.00 Alpha are available from Tesseract at UB Mannheim. These include the training tools. Both 32-bit and 64-bit installers are available.

An installer for the **OLD version 3.02** is available for Windows from our download page. This includes the English training data. If you want to use another language, download the appropriate training data, unpack it using 7-zip, and copy the .traineddata file into the 'tessdata' directory, probably `C:\Program Files\Tesseract-OCR\tessdata`.

To access tesseract-OCR from any location you may have to add the directory where the tesseract-OCR binaries are located to the Path variables, probably `C:\Program Files\Tesseract-OCR`.

Experts can also get binaries build with Visual Studio from the build artifacts of the Appveyor Continuous Integration.

1. เข้าสู่หน้าเว็บ Tesseract at UB Mannheim เลือกเวอร์ชันตามระบบปฏิบัติการของท่าน



Tesseract at UB Mannheim

The Mannheim University Library (UB Mannheim) uses Tesseract to perform OCR (optical character recognition) of historical German newspapers (Allgemeine Preußische Staatszeitung, Deutscher Reichsanzeiger). The latest results with OCR from more than 360,000 scans are available online.

Tesseract installer for Windows

Normally we run Tesseract on Debian GNU Linux, but there was also the need for a Windows version. That's why we have built a Tesseract installer for Windows.

WARNING: Tesseract should be either installed in the directory which is suggested during the installation or in a new directory. The uninstaller removes the whole installation directory. If you installed Tesseract in an existing directory, that directory will be removed with all its subdirectories and files.

The latest installers can be downloaded here:

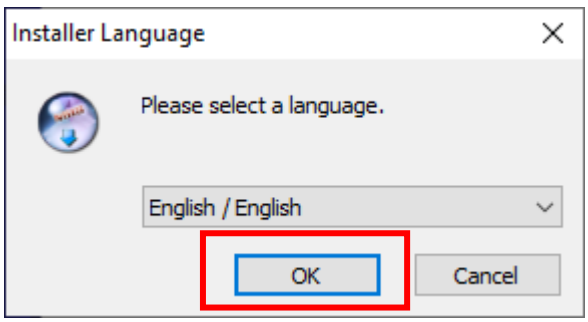
- tesseract-ocr-w32-setup-v5.0.0-alpha.20210811.exe (32 bit) and
- tesseract-ocr-w64-setup-v5.0.0-alpha.20210811.exe (64 bit) resp.

We don't provide an installer for Tesseract 4.1.0 because we think that the latest version 5.0.0-alpha is better for most Windows users in many aspects (functionality, speed, stability). Version 4.1 is only needed for people who develop software based on the Tesseract API and who need 100 % API compatibility with version 4.0.

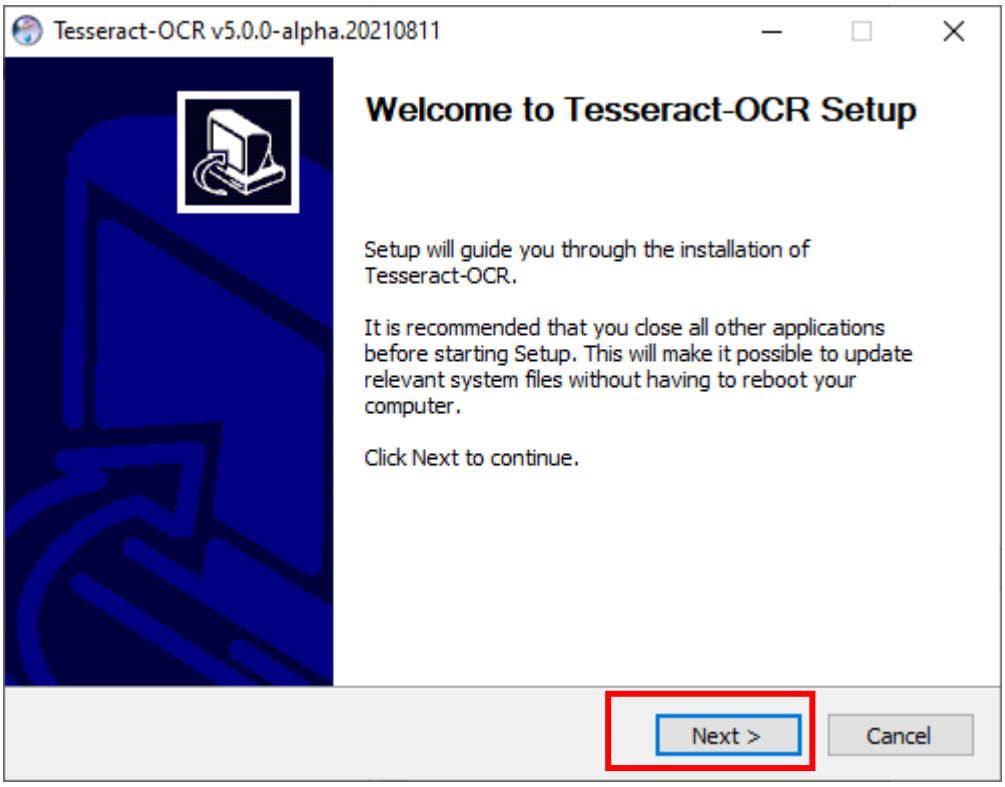
There are also older versions available.

In addition, we also provide documentation which was generated by Doxygen.

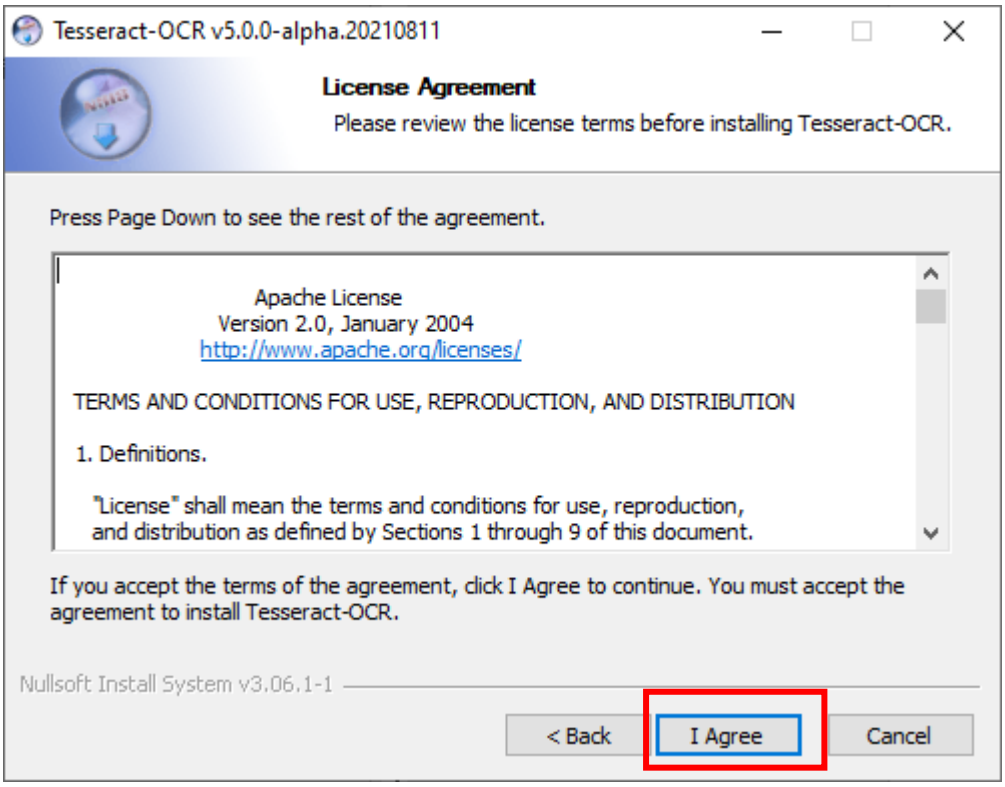
2. เลือกภาษาในการอธิบายการติดตั้งโดยค่าเริ่มต้น เป็นภาษาอังกฤษ กด OK



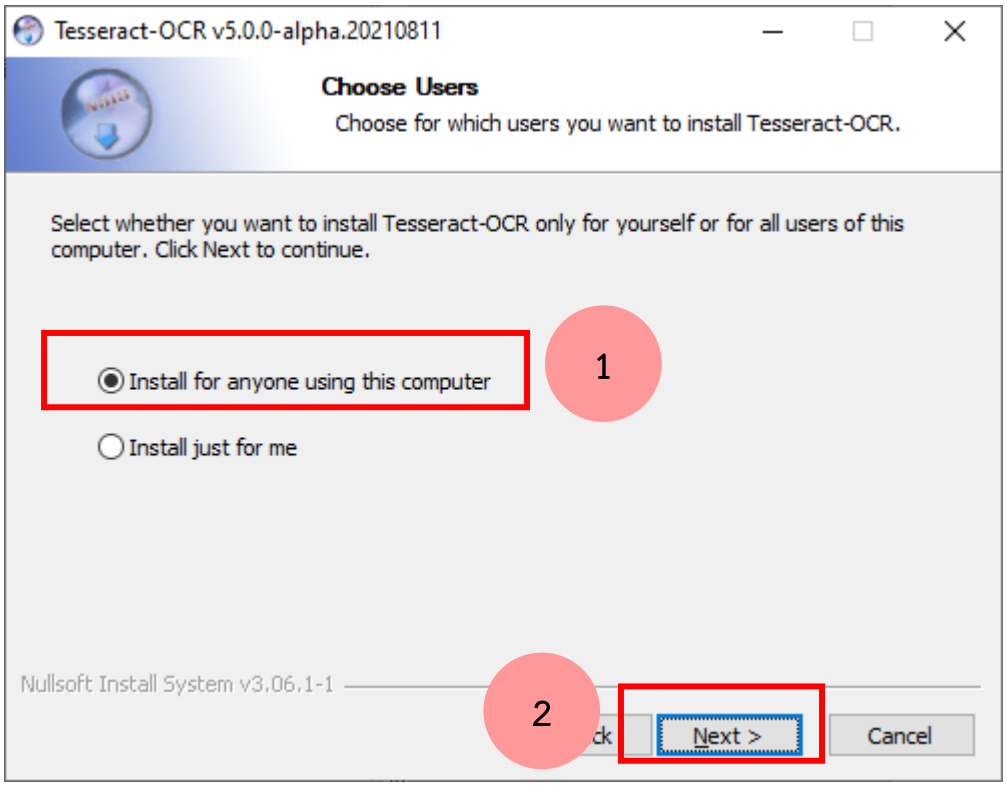
3. กด Next >



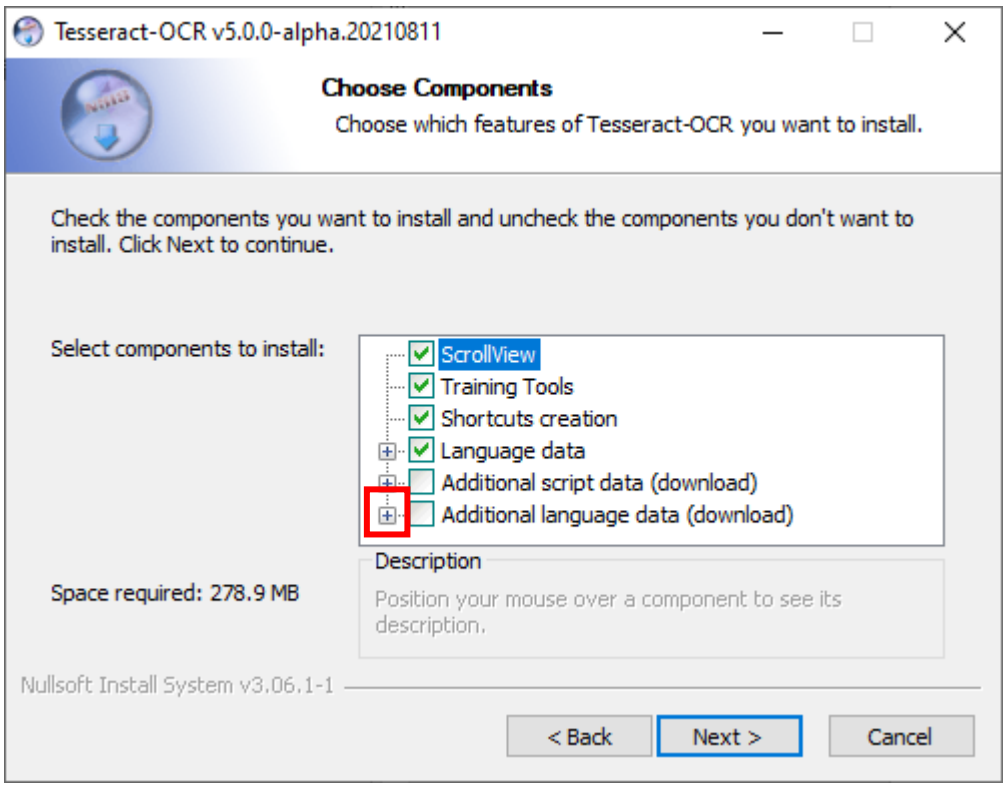
4. กด I Agree เพื่อยอมรับข้อตกลง



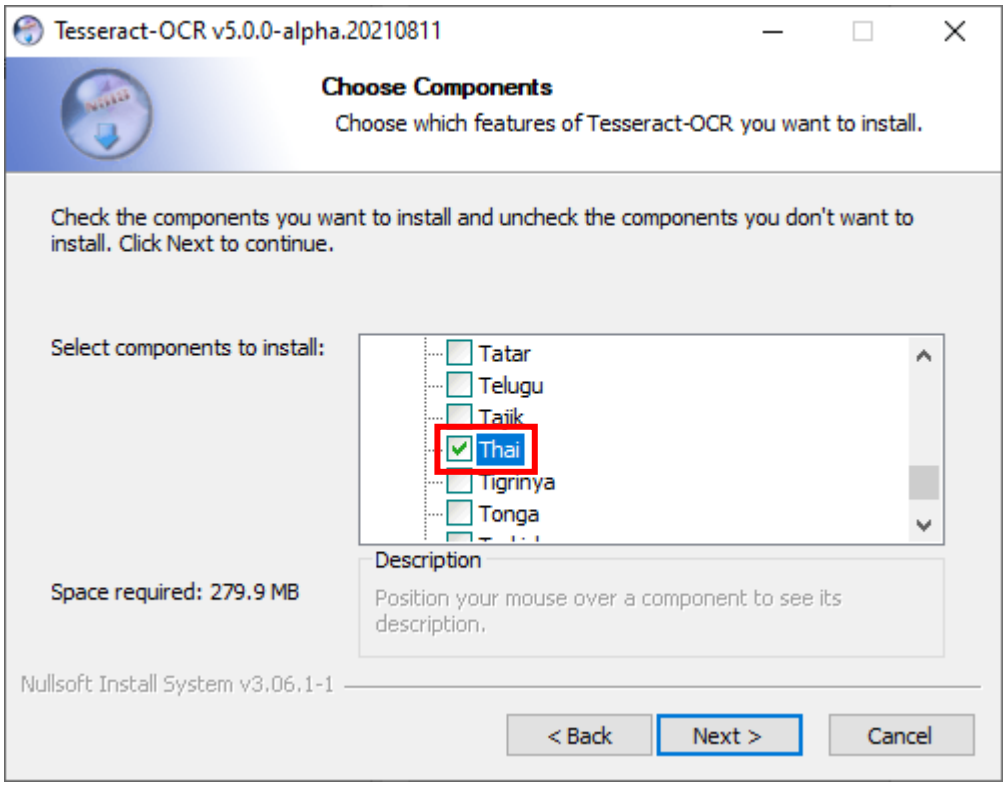
5. เลือกผู้ใช้งานเป็น Install for anyone using this computer* จำเป็น



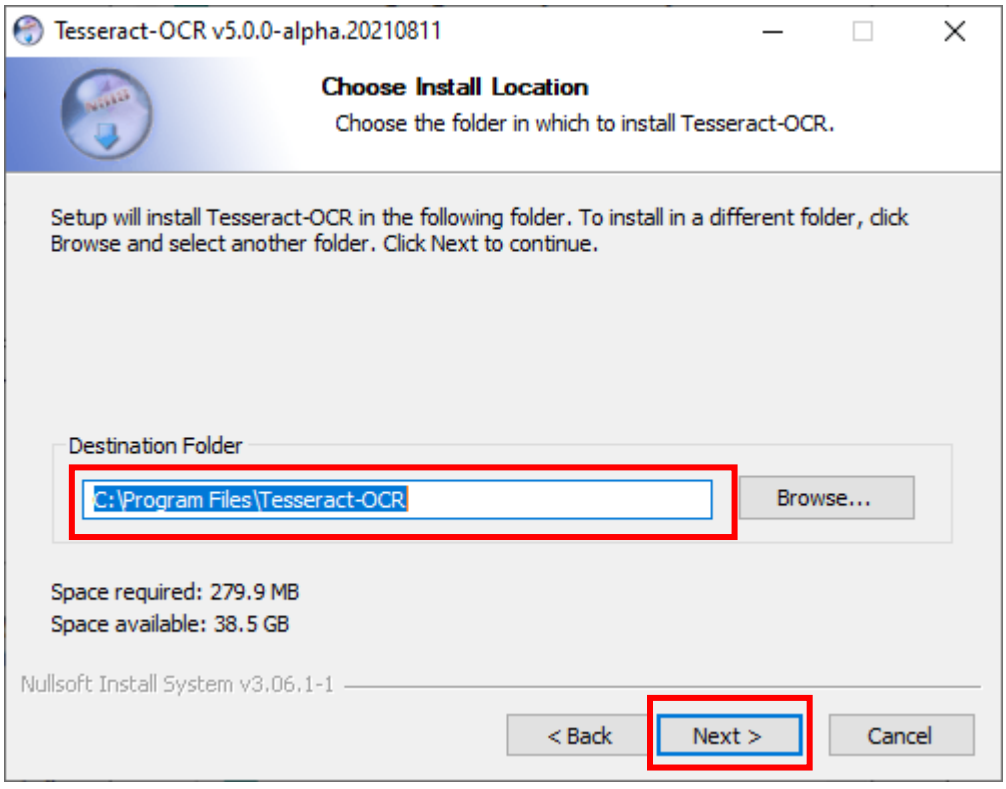
6. กดปุ่ม + ที่ Additional language data (download) เพื่อเลือกภาษาที่จะใช้งานเพิ่มเติม (ภาษาไทย)



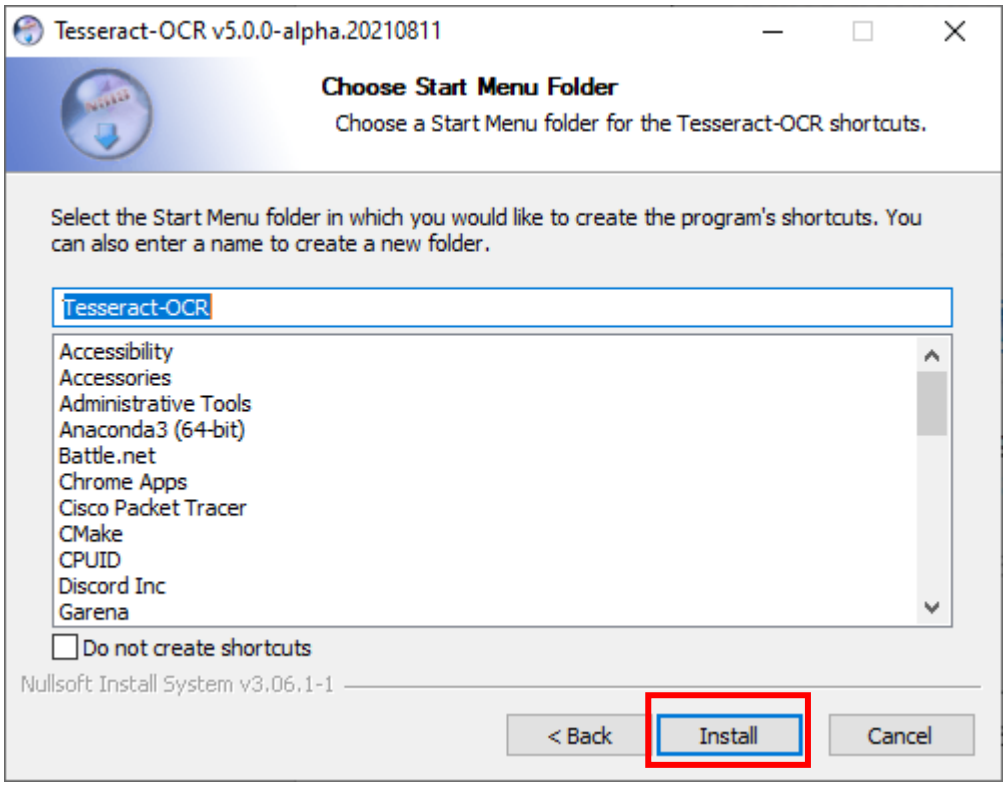
7. เลือกภาษาไทย หลังจากนั้น กด Next >



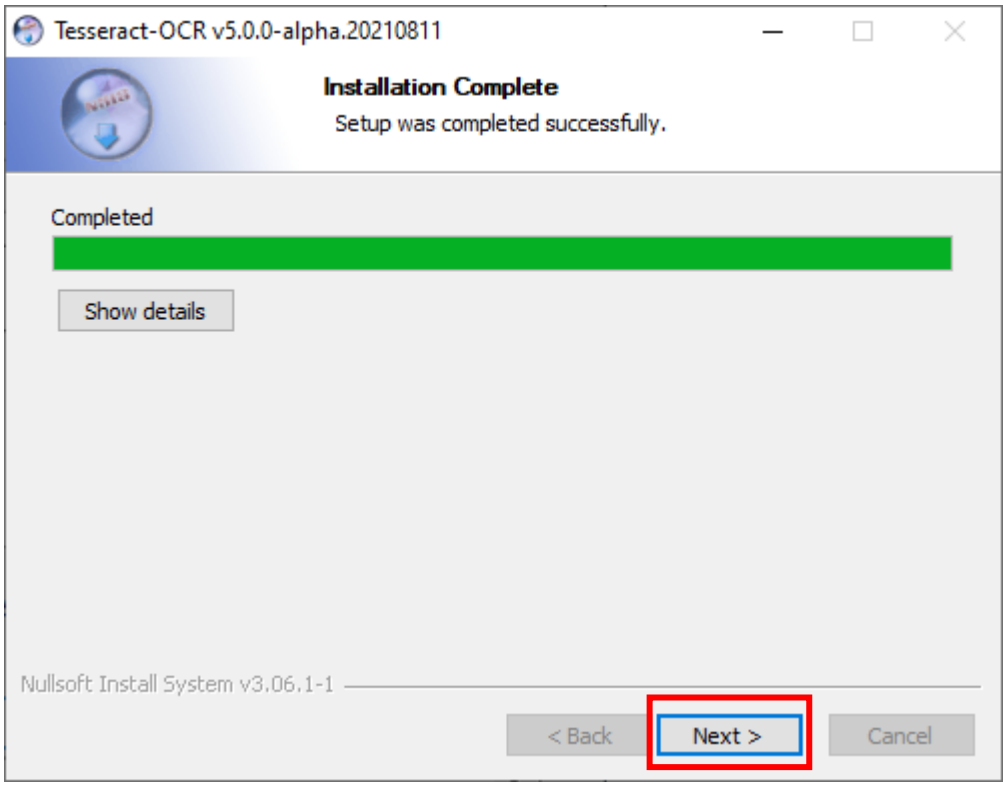
8. เลือกโฟลเดอร์ในการติดตั้ง โดยค่าเริ่มต้นเป็น C:\Program Files\Tesseract-OCR



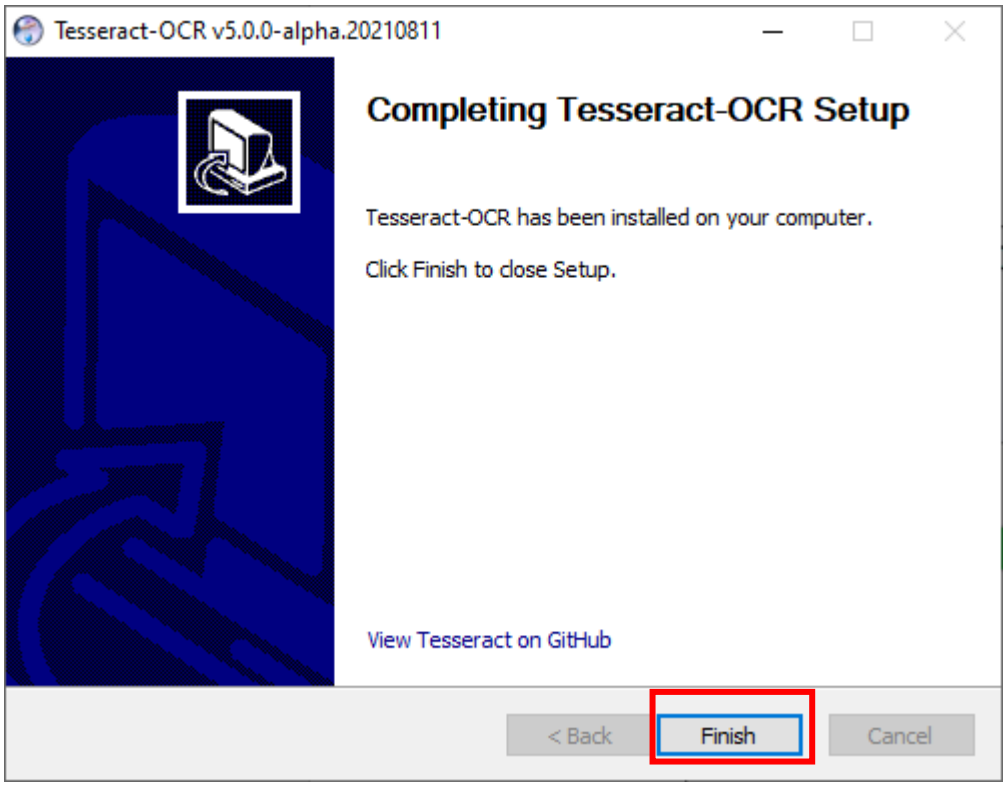
9. กด Install



10. กด Next >

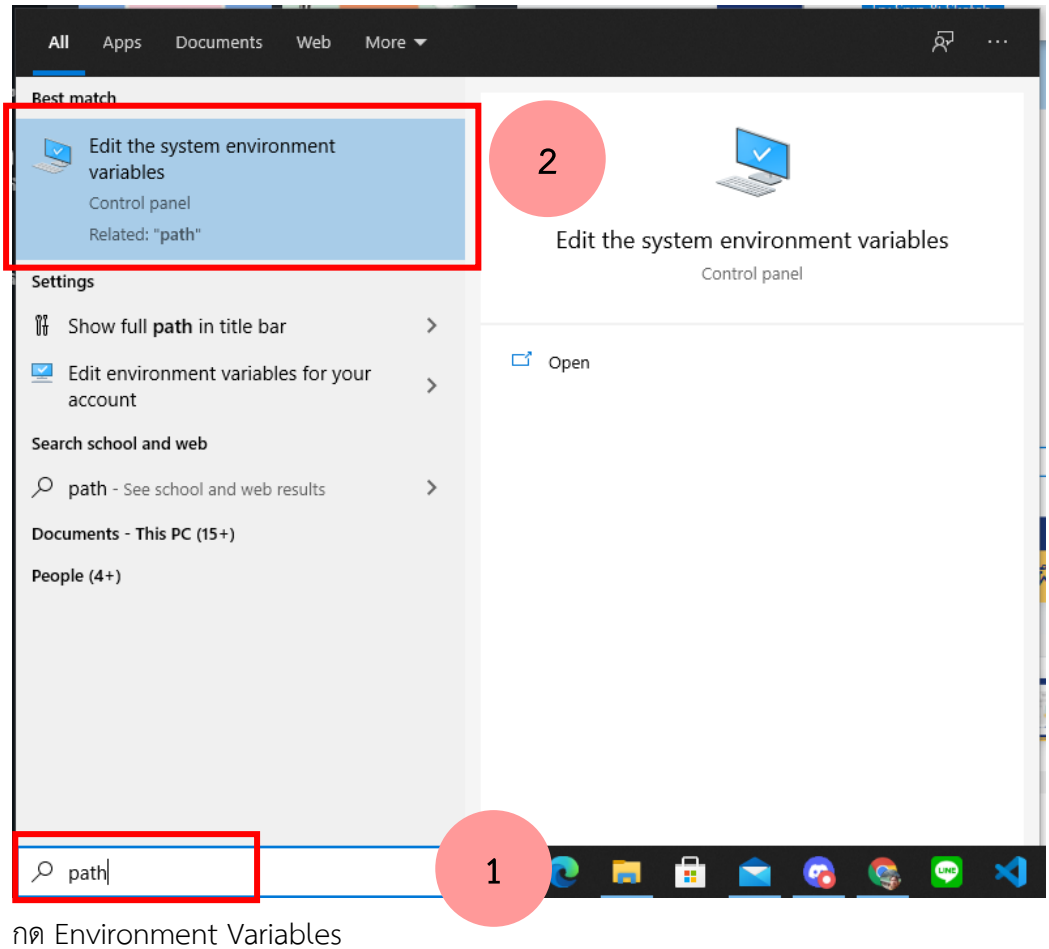


11. Finish

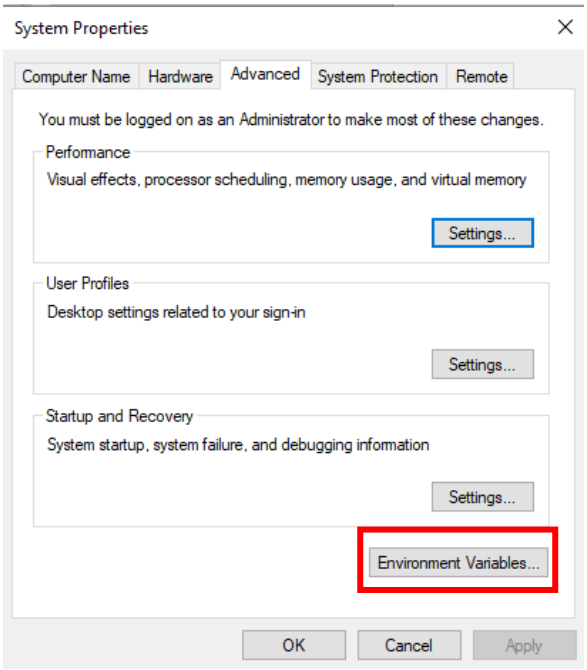

Tesseract-OCR v5.0.0-alpha.20210811
Completing Tesseract-OCR Setup
Tesseract-OCR has been installed on your computer.
Click Finish to close Setup.
View Tesseract on GitHub
< Back
Finish
Cancel

การตั้งค่า Path ไฟล์ เพื่อเรียกใช้งาน

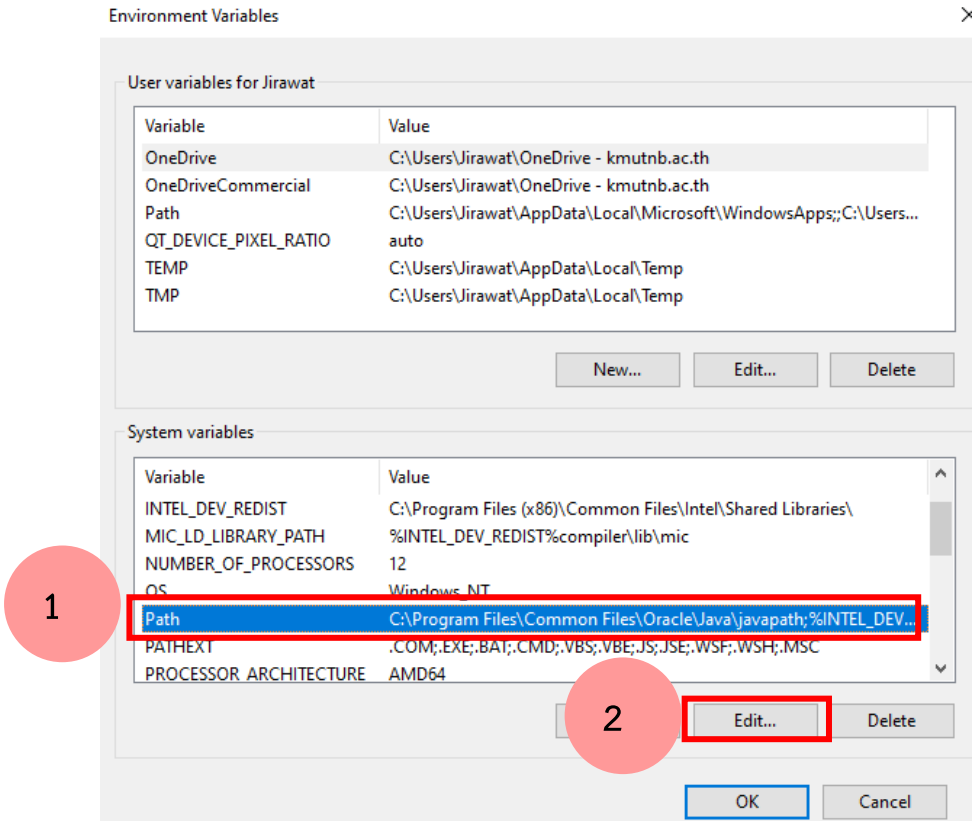
1. ค้นหา path ในช่อง window search



2. กด Environment Variables



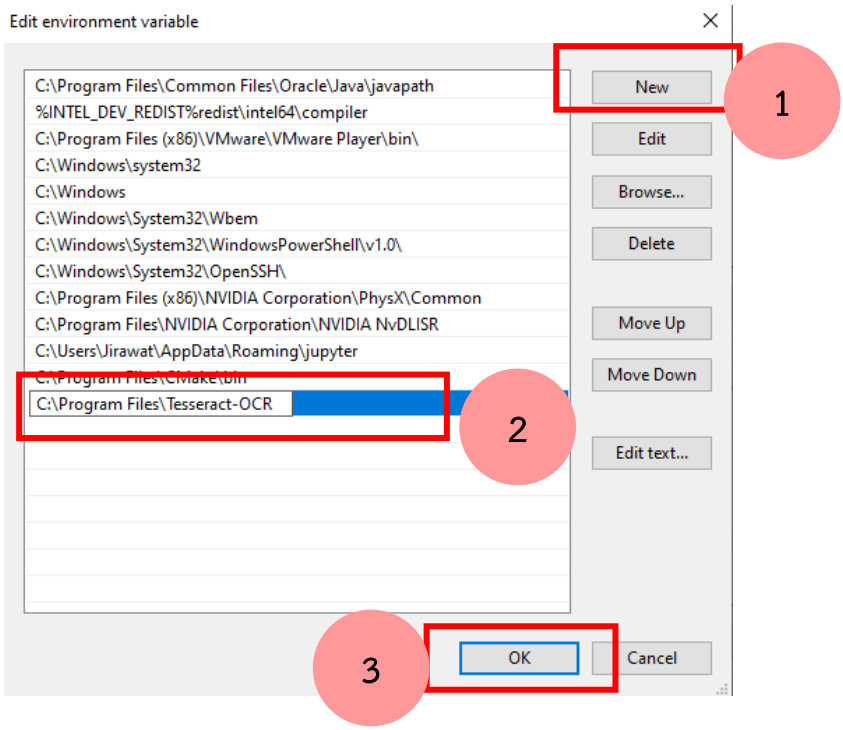
3. เลือก Variable Path > Edit



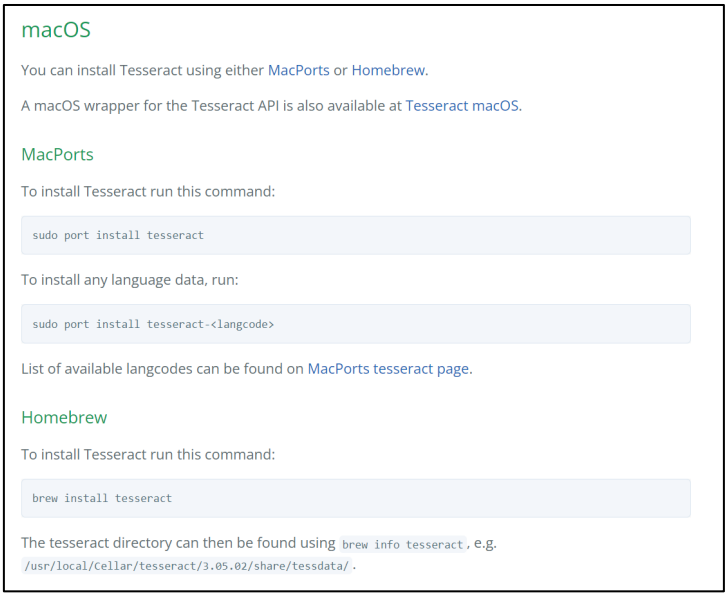
4. กดปุ่ม New

หลังจากนั้นใส่ Path C:\Program Files\Tesseract-OCR *ใส่ Path ที่เราติดตั้ง Tesseract >

กดปุ่ม OK



Install Tesseract For MAC



macOS

You can install Tesseract using either MacPorts or Homebrew.

A macOS wrapper for the Tesseract API is also available at Tesseract macOS.

MacPorts

To install Tesseract run this command:

```
sudo port install tesseract
```

To install any language data, run:

```
sudo port install tesseract-<langcode>
```

List of available langcodes can be found on MacPorts tesseract page.

Homebrew

To install Tesseract run this command:

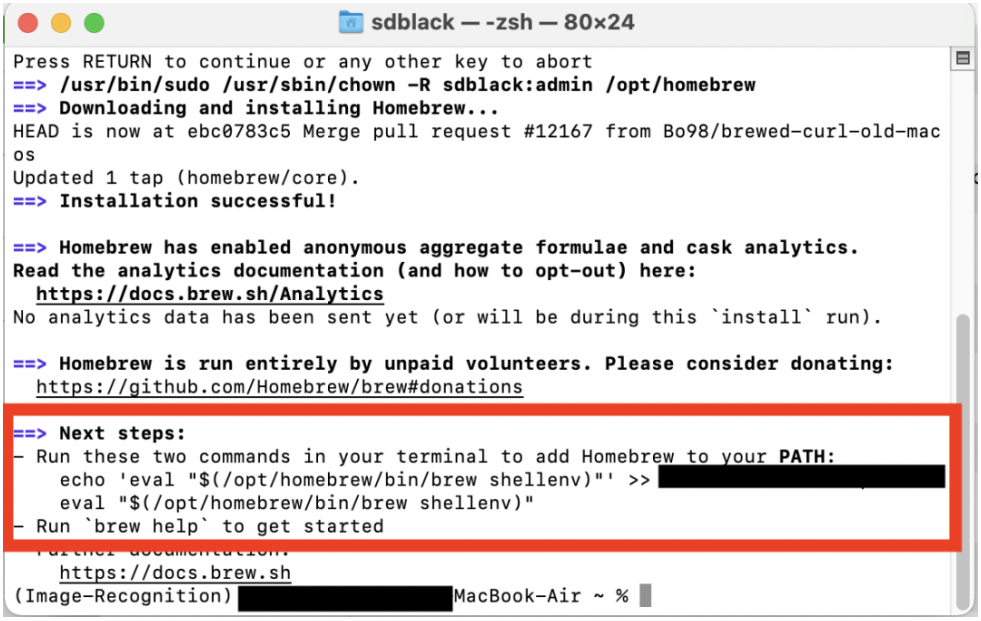
```
brew install tesseract
```

The tesseract directory can then be found using `brew info tesseract`, e.g. `/usr/local/Cellar/tesseract/3.05.02/share/tessdata/`.

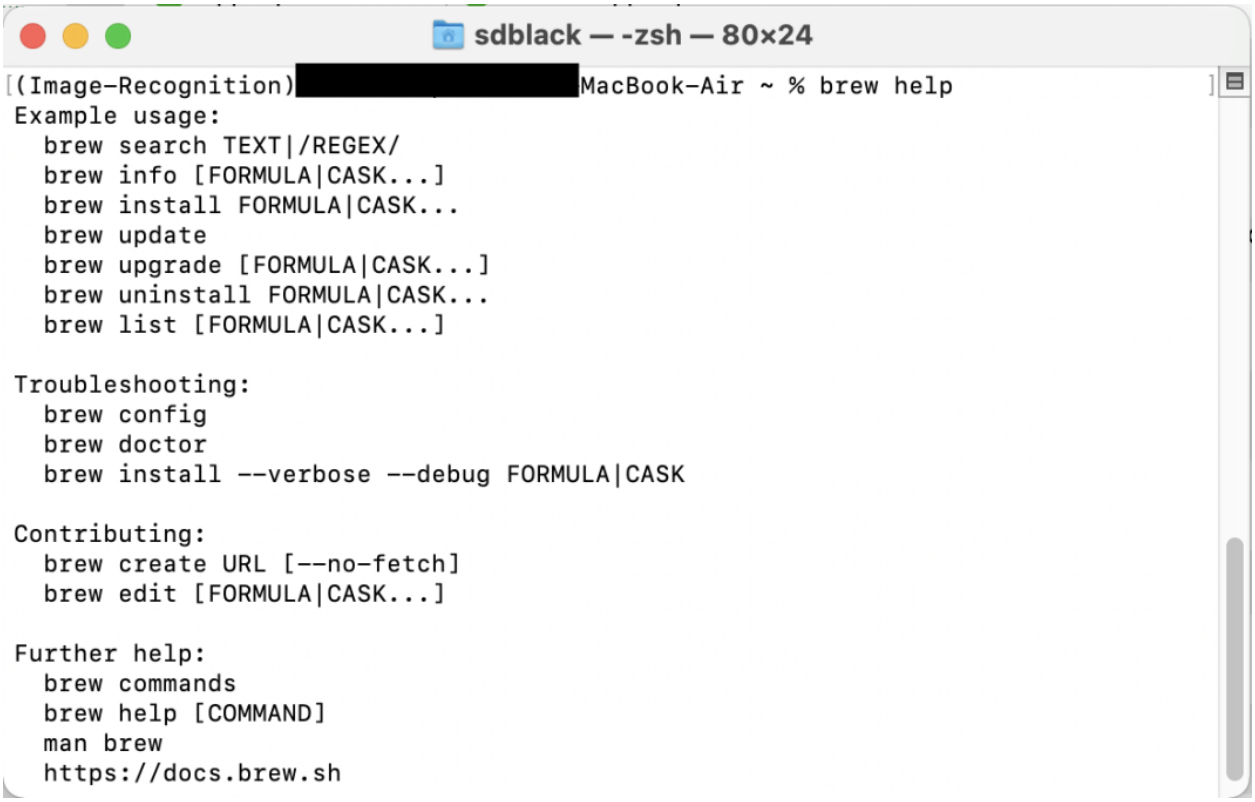
1. ติดตั้ง /bin/bash -c "$(curl -fsSL

   https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

1.1. แล้วทำตามคำแนะนำที่ขึ้นตาม terminal ที่ Next steps:



2. ทดลองใช้งาน brew โดยพิมพ์คำสั่ง $brew help



3. จากนั้นตำการติดตั้ง tesseract ด้วยคำสั่ง brew install tesseract

4. กรณีเพิ่มภาษาไทยใช้คำสั่ง brew install tesseract-lang เท่านี้ก็เรียบร้อยแล้วครับ