

การพัฒนาโมเดลการเรียนรู้เชิงลึกโดยใช้โครงข่ายประสาทแบบคอนโวลูชันในการจดจำ
ประเภทของอาหาร

Development of Deep Learning Model using Convolutional Neural
Network for Food Recognition on Mobile Devices

นายธนาภุริพัศ ศรีใสว เลขประจำตัว 623040255-1

นายวรปรัชญ์ โลมาอินทร์ เลขประจำตัว 623040486-2

รายงานนี้เป็นรายงานโครงการของนักศึกษาชั้นปีที่ 4 ซึ่งเสนอเป็นส่วนหนึ่งในหลักสูตรวิศวกรรมบัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น

พ.ศ. 2565

สารบัญ

| | |
|---|-----------|
| บทที่ 1 บทนำ | 1 |
| 1.1 หลักการและเหตุผล..... | 1 |
| 1.2 วัตถุประสงค์ | 1 |
| 1.3 ขอบข่ายโครงการ | 1 |
| 1.4 ขั้นตอนและแผนการดำเนินโครงการ..... | 1 |
| 1.5 เครื่องมือที่ใช้ในโครงการ..... | 2 |
| 1.6 งบประมาณที่ใช้..... | 2 |
| 1.7 ผลที่คาดว่าจะได้รับ | 2 |
| บทที่ 2 ทฤษฎีที่เกี่ยวข้อง | 3 |
| 2.1 ทฤษฎีที่เกี่ยวข้อง | 3 |
| รูปที่ 1.1 Convolutional Neural Network Architecture | 3 |
| 2.2 งานวิจัยที่เกี่ยวข้อง..... | 5 |
| บทที่ 3 วิธีการดำเนินงาน | 7 |
| 3.1 ออกแบบวิธีการดำเนินงาน..... | 7 |
| 3.2 เครื่องมือและเทคโนโลยีที่ใช้งาน | 7 |
| 3.3 การเตรียมข้อมูล | 7 |
| 3.4 ออกแบบและพัฒนาโมเดลการเรียนรู้เชิงลึก | 8 |
| บทที่ 4 ผลการดำเนินงาน | 10 |
| 4.1 Label ข้อมูลและเตรียมไฟล์สำหรับการทำ segmentation..... | 10 |
| 4.2 แปลงไฟล์จากการ label เป็นภาพ segmentation | 10 |
| 4.3 การแปลง Object Detection โดยภาษา Python | 11 |
| 4.4 การแปลง Json file เป็น Pascal VOC format file | 11 |
| 4.5 การ train object detection model ด้วย TensorFlow Lite Model Maker | 12 |

| | |
|--|-----------|
| 4.6 ผลจาก train model..... | 13 |
| 4.7 การนำ model ลงใน mobile application..... | 15 |
| บทที่ 5 สรุปผลการดำเนินโครงการ | 16 |
| 5.1 สรุปผล..... | 16 |
| 5.2 ข้อเสนอแนะ | 16 |
| อ้างอิง | 17 |
| ภาคผนวก | 18 |

บทที่ 1 บทนำ

1.1 หลักการและเหตุผล

ในสังคมปัจจุบันโครงสร้างสังคมประชากรไทยได้มีการเปลี่ยนแปลงไปสู่การเป็นสังคมของผู้สูงอายุหรือผู้สูงวัย (aging society) ซึ่งส่งผลกระทบในด้านของ เศรษฐกิจ สังคม และด้านสุขภาพ ปัญหาของสังคมผู้สูงอายุคือ การมีพฤติกรรม สุขภาพที่พึงประสงค์ จึงต้องมีการส่งเสริมด้านสุขภาพ และโภชนาการ เพื่อลดค่าใช้จ่ายด้านบริการสุขภาพ ลดภาระด้านเศรษฐกิจของประเทศ สังคมและครอบครัว

โภชนาการจัดเป็นหนึ่งในเรื่องในด้านสุขภาพที่ส่งผลกระทบมากที่สุดต่อสุขภาพของผู้สูงอายุ ปัจจัยสำคัญของโภชนาการ คือ การบริโภคอาหาร ซึ่งการบริโภคอาหารที่ดี จะสามารถป้องกันและลดการเกิดโรคต่างๆในวัยของผู้สูงอายุ และทำให้อายุยืนยาวเพิ่มขึ้นได้ การมีพฤติกรรมของการบริโภคอาหารที่ถูกและเหมาะสมนั้นจะคงไว้ซึ่งสุขภาพร่างกายที่ดี ป้องกันการเจ็บป่วยที่มีโอกาสจะเกิดขึ้นจากการบริโภคไม่เหมาะสมกับวัยได้

ดังนั้นแล้ว ทางคณะผู้จัดทำจึงมีการนำเทคโนโลยีทางด้านคอมพิวเตอร์ แอปพลิเคชันและปัญญาประดิษฐ์ ซึ่งจัดเป็นเทคโนโลยีขั้นสูง จะเข้ามามีบทบาทในการส่งเสริมภาวะสุขภาพที่ดี และช่วยงานทางด้านการแพทย์ สาธารณสุข อย่างไรก็ตาม เทคโนโลยีเหล่านี้ยังจำเป็นต้องอาศัยการกรอกข้อมูลเมนูอาหาร ชนิดอาหาร และปริมาณอาหารที่บริโภค แล้วระบบจึงจะสามารถคำนวณออกมาเป็นคำแนะนำและข้อมูลต่างๆ เพื่อช่วยเหลือและลดทอนเวลา ของผู้สูงอายุ บุคคลทุกช่วงวัยที่มีชีวิตที่เร่งรีบ และอาจจะปริมาณของอาหารได้ไม่เหมาะสม

ด้วยเหตุที่กล่าวไปข้างต้น โครงการนี้จึงมีวัตถุประสงค์เพื่อจะออกแบบและพัฒนาโมเดลการเรียนรู้เชิงลึกโดยใช้โครงข่ายประสาทแบบคอนโวลูชัน ในการจดจำการประมวลผลและการประเมินปริมาณน้ำหนักรูปร่างของอาหาร จากภาพถ่ายอาหาร โดยโมเดลที่ได้จากโครงการนี้ จะนำไปติดตั้งเป็นแอปพลิเคชันบนอุปกรณ์สื่อสารเคลื่อนที่ เพื่อให้ผู้ใช้งานสามารถทำนายคุณค่าทางโภชนาการได้ผ่านการใช้กล้องของอุปกรณ์ ที่ช่วยให้เพิ่มความอำนวยความสะดวกแก่ผู้สูงอายุและบุคคลทุกช่วงวัยในปัจจุบัน เพื่อในการเลือกบริโภคอาหารที่เหมาะสมและมีคุณค่าเหมาะสมตามโภชนาการของช่วงวัย

1.2 วัตถุประสงค์

1.2.1 เพื่อออกแบบและพัฒนาโมเดลการเรียนรู้เชิงลึก (Deep Learning) โดยใช้โครงข่ายประสาทแบบคอนโวลูชัน (Convolutional neural network) ในการจดจำการประมวลผลและการประเมินปริมาณน้ำหนักรูปร่างของอาหารที่ถูกต้องและแม่นยำ

ตารางที่ 1.1 ขั้นตอนและแผนการดำเนินโครงการ

1.5 เครื่องมือที่ใช้ในโครงการ

1.5.1 คณิตศาสตร์

- a) การเรียนรู้ของเครื่อง (Machine learning)
- b) โครงข่ายประสาทแบบคอนโวลูชัน (Convolutional neural network)

1.5.2 ซอฟต์แวร์

- a) Python 3.7
- b) LabelMe
- c) TensorFlow Lite
- d) Google Colab

1.5.3 เครื่องมือเก็บข้อมูล

- a) โทรศัพท์มือถือระบบปฏิบัติการ Android/iOS

1.6 งบประมาณที่ใช้

1.7 ผลที่คาดว่าจะได้รับ

ได้โมเดลทางปัญญาประดิษฐ์และ mobile application สำหรับการประมวลผลข้อมูลชนิดและ
น้ำหนักอาหารเพื่อแปรผลคุณค่าทางโภชนาการ

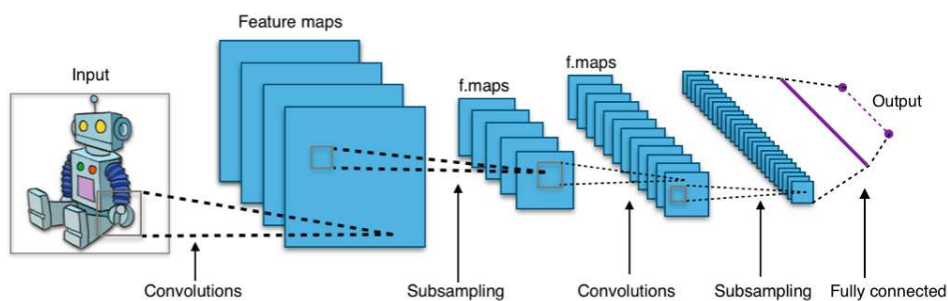
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 การเรียนรู้เชิงลึก (Deep learning)

การเรียนรู้ด้วยตัวเอง ของเครื่องจักร หรือ Machine learning จะมีการเรียนรู้ประเภทหนึ่ง คือ การเรียนรู้เชิงลึก ที่มีการคำนวณแบบไม่เชิงเส้น (nonlinear) หลายชั้น เป็นโครงข่ายประสาทเทียมขนาดใหญ่ ซึ่งได้รับความนิยมในการจดจำภาพ

โครงข่ายประสาทคอนโวลูชัน โครงข่ายประสาทคอนโวลูชัน ใช้ในการเรียนรู้ด้วยตัวเองของเครื่องจักร นำมาประยุกต์ใช้กันอย่างแพร่หลายในการตรวจจับและจดจำ มีสถาปัตยกรรมเช่นเดียวกับโครงข่ายประสาทเทียมหลายชั้น เช่นเดียวกับเปอร์เซ็ปตรอนหลายชั้น (Multi-Layer Perceptron) โครงข่ายประสาทคอนโวลูชัน ประกอบด้วย ชั้นอินพุตและชั้นเอาต์พุต รวมด้วยชั้นซ่อนหลายชั้น (Multiple Hidden Layer) การคำนวณในเลเยอร์ใช้หลักการคำนวณทางคณิตศาสตร์แบบสังวัตนาการ (Convolution) โครงสร้างของโครงข่ายประสาทคอนโวลูชัน(ราตรี คำโมง, 2565)



รูปที่ 1.1 Convolutional Neural Network Architecture

2.1.2 Volume Estimated

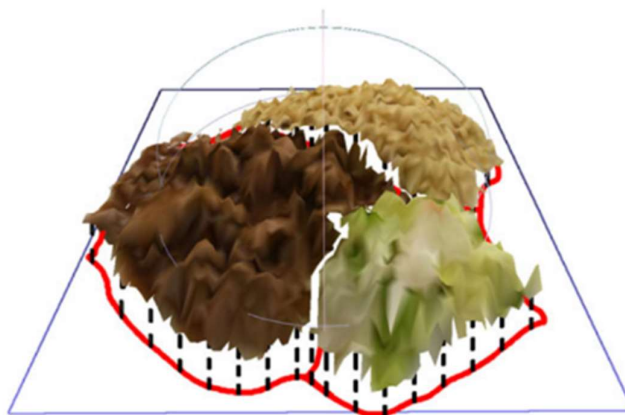
การทำนายปริมาตรเป็นวิธีที่จะนำไปช่วยในการค้นหาปริมาตรที่ใกล้เคียงของวัตถุ ซึ่งสอดคล้องกับในตัวโครงการนี้ ที่จะทำนายปริมาตรของอาหาร โดยสามารถทำได้หลากหลายวิธี

2.1.2.1 การทำนายปริมาตรโดยใช้ภาพ 2 มิติ การทำนายปริมาตรโดยวิธีนี้นั้นจะแบ่งขั้นตอนเป็น 3 ขั้นตอน ดังนี้

1) Food surface Extraction เป็นการแบ่งส่วนภาพที่มีและลบพื้นหลังออก จากนั้นรูปร่างของอาหารจะถูกสร้างขึ้นมาจากพิกัดภาพ 2 มิติ และพื้นผิวจะถูกแบ่งเป็นสามเหลี่ยมที่มีขนาดต่างกัน ที่จะแสดงให้เห็นพื้นผิวที่ต่างกัน

2) Dish Surface Extraction งานส่วนมากเป็นงานที่ทำมาจากวัสดุพลาสติก หรือ เซรามิก ซึ่งมีคุณสมบัติที่สะท้อนแสง ส่งผลทำให้ต้องนำออก เพราะยากต่อการนำมากะขนาด จึงนิยมใช้สิ่งของบางอย่างมาใช้ในการเปรียบเทียบแทน เช่น บัตเตอร์ครีม เหยี่ยว เป็นต้น ซึ่งมีขนาดที่แน่นอน และยังแบนราบไประนาบของพื้นผิว เหมาะที่จะนำมาเป็นตัวเปรียบเทียบขนาดปริมาตรของอาหาร

3) Volume Calculation เป็นการผนวกรวมขั้นตอนก่อนหน้านี้ เราจะได้ข้อมูลพิกัดของอาหารในพิกัดระนาบ และได้ขนาดเปรียบจากวัตถุที่เป็น บัตเตอร์ เหยี่ยว ฯลฯ ทำให้สามารถนำมาคำนวณเปรียบเทียบขนาดของอาหาร ที่เราจะสามารถประมาณค่าปริมาตรของอาหารได้



รูปที่ 1.2 Volume estimation: food surface

2.1.2.2 การทำนายปริมาตรอาหารโดยความลึกของกล้อง การใช้ Depth camera จะเป็นการยิงคลื่นแม่เหล็กไฟฟ้าอินฟราเรด ไปยังพื้นที่หรือวัตถุ แล้วสะท้อนกลับมาเป็นค่าระยะทาง

2.1.2.3 การทำนายปริมาตรอาหารโดยการปรับองศารับภาพ โดยการใช้กล้องถ่ายภาพวัตถุในองศาต่างๆ แล้วนำมาใช้เทียบกับรูปทรงเลขาคณิตที่เตรียมไว้ เพื่อเป็นการประเมินรูปทรงของอาหารในรูปแบบทรงรูปทรงเลขาคณิต และทำนายค่าปริมาตรจากรูปทรงนั้น

2.2 งานวิจัยที่เกี่ยวข้อง

งานวิจัยที่เกี่ยวข้องทางคณะผู้จัดทำโครงการได้ศึกษาค้นคว้าที่เป็นประโยชน์ของโครงการ

2.2.1 A Large-Scale Benchmark for Food Image Segmentation

2.2.1.1 ผู้จัดทำ Xiongwei Wu, Xin Fu, Ying Liu, Ee-Peng Lim, Steven C.H Hoi,

Qianru Sun

2.2.1.2 รายละเอียดงานวิจัย

Food image segmentation หรือ การแบ่งส่วนภาพอาหารเป็นงานที่สำคัญและขาดไม่ได้สำหรับการพัฒนาแอปพลิเคชันที่เกี่ยวข้องกับสุขภาพ เช่นการประมาณปริมาณแคลอรีของอาหารและสารอาหาร แบบจำลองการแบ่งส่วนภาพอาหารที่มีอยู่ทำงานได้ไม่เต็มประสิทธิภาพเนื่องจากเหตุผล 2 ประการ คือ ขาดชุดข้อมูลของภาพอาหารที่มีคุณภาพ หรือชุดข้อมูลมีขนาดเล็ก และ รูปแบบของอาหารที่ซับซ้อนทำให้ยากต่อการแยกส่วนและจดจำส่วนผสมในภาพอาหาร เช่น ส่วนผสมอาจทับซ้อนกันในรูปเดียวกัน และส่วนประกอบที่เหมือนกันอาจปรากฏซ้ำกันในรูปอาหารต่าง ๆ

ในงานวิจัยนี้ผู้วิจัยได้สร้างชุดข้อมูลภาพอาหารใหม่ FoodSeg103 (และส่วนขยาย FoodSeg154) ที่มีรูปภาพ 9,490 รูป เราใส่คำอธิบายประกอบภาพเหล่านี้มี 154 คลาสสำหรับส่วนผสมของอาหาร และแต่ละภาพมี label เฉลี่ย 6 ภาพละ 6 label และมาส์กแบบพิกเซล นอกจากนี้ ยังได้เสนอวิธีการ training แบบ multi-modality pre-training approach เรียกว่า ReLeM ที่จัดเตรียมแบบจำลองการแบ่งส่วนไว้อย่างชัดเจนในเรื่องอาหาร ในการทดลอง เราใช้วิธี segmentation สามแบบที่เป็นที่นิยม (เช่น Dilated Convolution based, Feature Pyramid based และ Vision Transformer based)

2.2.2 Food Image Analysis: Segmentation, Identification and Weight Estimation

2.2.2.1 ผู้จัดทำ Ye He, Chang Xu, Nitin Khanna, Carol J. Boushey, Edward J. Delp

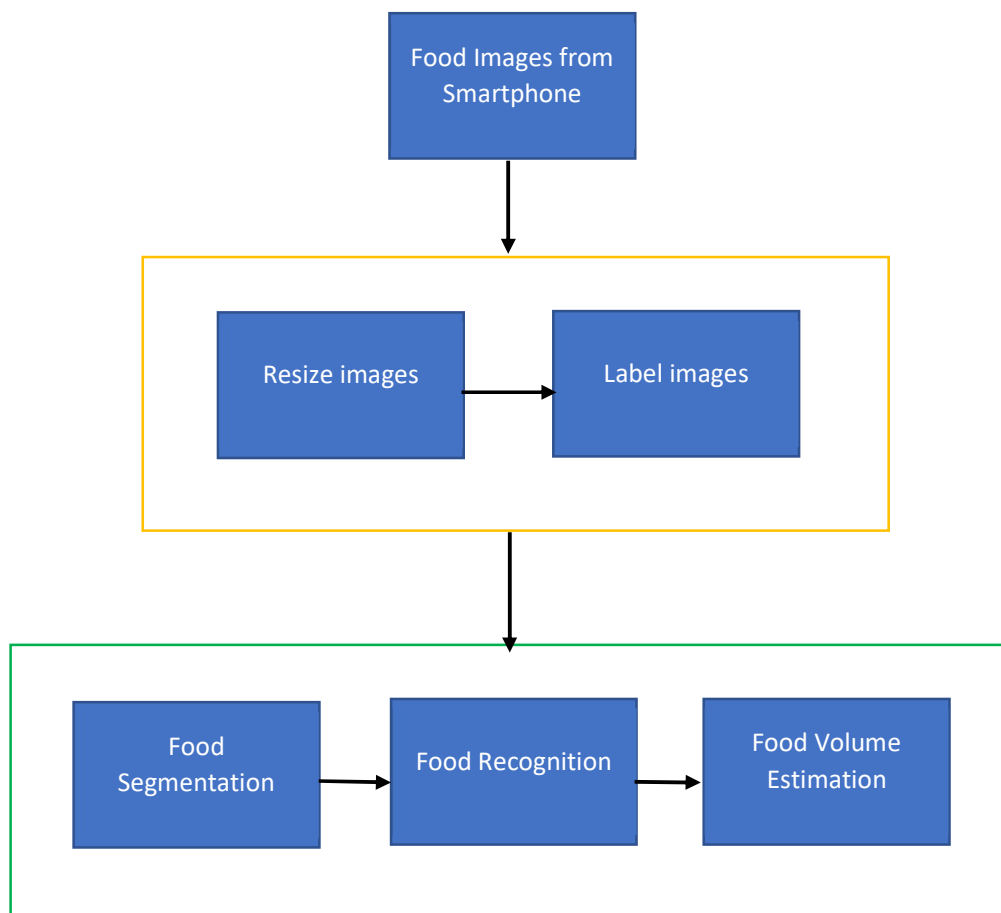
2.2.2.2 รายละเอียดงานวิจัย

ผู้วิจัยได้พัฒนาระบบการประเมินอาหารที่บันทึกการบริโภคอาหารในแต่ละวันผ่านการถ่ายภาพอาหารที่ถ่ายในมือถืออาหาร จากนั้นภาพอาหารจะถูกวิเคราะห์เพื่อแยกปริมาณสารอาหารใน

อาหาร ในบทความนี้ เราจะอธิบายเครื่องมือวิเคราะห์รูปภาพเพื่อกำหนดบริเวณที่มีอาหาร (การแบ่งส่วนรูปภาพ) ระบุประเภทอาหาร (การจำแนกคุณลักษณะ) และประมาณน้ำหนักของรายการอาหาร (การประเมินน้ำหนัก) การแบ่งส่วนภาพและการจำแนกประเภทสามารถช่วยในการปรับปรุงการแบ่งส่วนอาหารและการระบุความถูกต้อง จากนั้นเราประมาณการน้ำหนักของอาหารเพื่อตั้งปริมาณสารอาหารจากภาพเดียวโดยใช้แม่แบบรูปร่างสำหรับอาหารที่มีรูปร่างปกติและการประมาณน้ำหนักตามพื้นที่สำหรับอาหารที่มีรูปร่างผิดปกติ

บทที่ 3 วิธีการดำเนินงาน

3.1 ออกแบบวิธีการดำเนินงาน



รูปที่ 1.3 ขั้นตอนแผนการทำงาน

3.2 เครื่องมือและเทคโนโลยีที่ใช้งาน

เครื่องมือที่ทางคณะผู้จัดทำเลือกในการ labeling ของรูปภาพ เลือกใช้ LabelMe และ Roboflow เนื่องจากทั้งสองบริการมีการใช้งานที่ง่าย อีกทั้งหน้าตาการใช้งานเป็นแบบ GUI

3.3 การเตรียมข้อมูล

การเตรียมข้อมูลรูปภาพ ต้องการมีการ resize รูปภาพก่อนซึ่งการ resize รูปภาพ ใช้ 2 วิธีด้วยกัน

1. การ resize ด้วยภาษา python ในการปรับรูปภาพให้เป็นสี่เหลี่ยมจัตุรัสขนาด 640x640
2. การ resize ด้วย โปรแกรม Adobe Lightroom ปรับรูปภาพให้เป็น 480x640 จากนั้นใช้ code ภาษา python ในการปรับให้เป็นสี่เหลี่ยมจัตุรัส

3.4 ออกแบบและพัฒนาโมเดลการเรียนรู้เชิงลึก

ในส่วนของโมเดลการเรียนรู้เชิงลึกที่ใช้งานเป็น TensorFlow Lite ซึ่งตัวของ TensorFlow Lite เป็น library สำหรับการสร้าง machine learning models จาก Google และยังสามารถใช้กับภาษา Python

แพลตฟอร์มที่สามารถทำงานร่วมกับ TensorFlow Lite มีหลากหลายแพลตฟอร์ม เช่น คอมพิวเตอร์ ระบบปฏิบัติการ Windows, macOS, Linux คลาวด์เว็บเซอร์วิส โทรศัพท์มือถือทั้งสองระบบปฏิบัติการ Android และ iOS

โมเดลการเรียนรู้เชิงลึกของ TensorFlow มีหลากหลายตัว โดยในโครงการนี้ตัวโมเดลจะนำไปใช้งานในอุปกรณ์โทรศัพท์มือถือบนระบบปฏิบัติการ Android และ iOS ซึ่งตารางต่อไปนี้จะเป็นการเปรียบเทียบตัวโมเดลต่างๆบนทั้งสองระบบปฏิบัติการ

| OS | Type | Model name | Image size | Inference | Phone spec |
|---------|---------------------|--------------------------------------|--|-----------|--|
| Android | Object segmentation | Deep lab v3+, mobile net v2 backbone | 257 x 257 (converted from 513 x 513 using android app sample code for DeepLab) | 571 ms | ไม่ได้บอกรุ่นมือถือ Number of thread = 4 |
| | Object detection | COCO SSD MobileNet v1 | 300 x 300 | 29 ms | Pixel 4 |

| | | | | | |
|-----|---------------------|--------------------------------------|-----------|---------------------------------------|---|
| IOS | Object segmentation | MobileNet V2 (1.0, 224, float) | ไม่ระบุ | 9.22 ms(cpu) 6.21 ms(gpu) | iPhone 8+ (Apple A11, iOS 13.2.3) iPhone XS (Apple A12, iOS 13.2.3) iPhone 11 Pro (Apple A13, iOS 13.2.3) |
| | Object detection | COCO SSD MobileNet v1 | 300 x 300 | 11 ms(cpu 2 thread) 7.6 ms(gpu) | iPhone 8+ (Apple A11, iOS 13.2.3) iPhone XS (Apple A12, iOS 13.2.3) iPhone 11 Pro (Apple A13, iOS 13.2.3) |

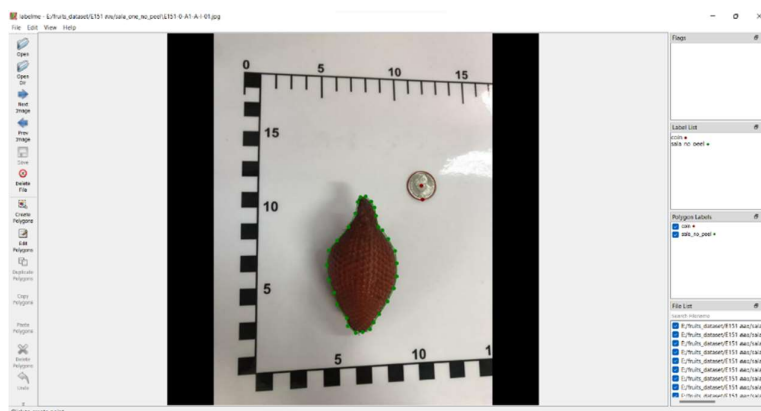
| Model architecture | Size(MB)* | Latency(ms)** | Average Precision*** |
|--------------------|-----------|---------------|----------------------|
| EfficientDet-Lite0 | 4.4 | 146 | 25.69% |
| EfficientDet-Lite1 | 5.8 | 259 | 30.55% |
| EfficientDet-Lite2 | 7.2 | 396 | 33.97% |
| EfficientDet-Lite3 | 11.4 | 716 | 37.70% |
| EfficientDet-Lite4 | 19.9 | 1886 | 41.96% |

ตารางที่ 1.2 ตารางเปรียบเทียบโมเดล

บทที่ 4 ผลการดำเนินงาน

4.1 Label ข้อมูลและเตรียมไฟล์สำหรับการทำ segmentation

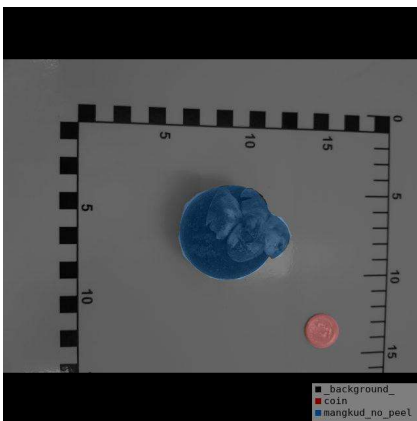
ทางผู้จัดทำโครงการหลังจากรวบรวมรูปภาพข้อมูลทั้งหมด ได้ทำการ labeling ด้วยโปรแกรม labelme โดยทำการ label รูปผลไม้แยกเป็น class ต่าง ๆ พร้อมกับเหรียญเพิ่มใช้ในการประมาณปริมาตรของอาหาร จากนั้นได้ทำการแปลงไฟล์ JSON ที่ได้จากการ label ข้อมูลมาเป็นไฟล์ Pascal Visual Object Classes(VOC) สำหรับการทำให้ image segmentation ต่อไป



รูปที่ 1.4 การ label ผลไม้และเหรียญด้วยโปรแกรม labelme

4.2 แปลงไฟล์จากการ label เป็นภาพ segmentation

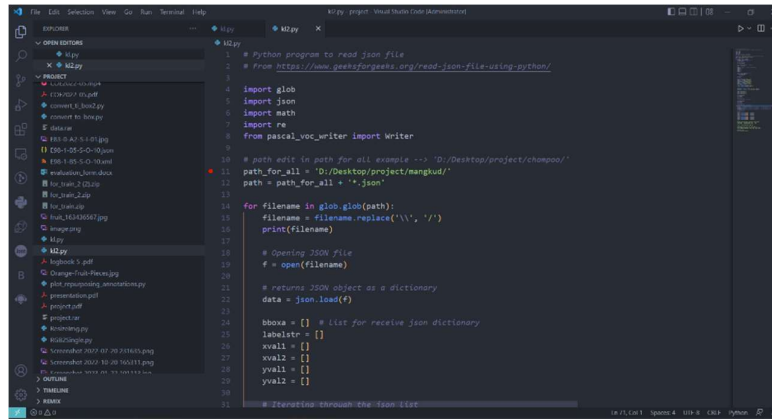
ไฟล์ JSON จากการ label นั้นจะยังไม่สามารถใช้ได้โดยตรง ทางผู้จัดทำโครงการจึงต้องแปลงเป็นภาพที่นำไปใช้ได้ โดยใช้ format Pascal Visual Object Classes(VOC) แปลงเป็นภาพ segmentation ที่จะสามารถนำไปเทรนต่อโมเดลได้



รูปที่ 1.5 ผลลัพธ์ image segmentation

4.3 การแปลง Object Detection โดยภาษา Python

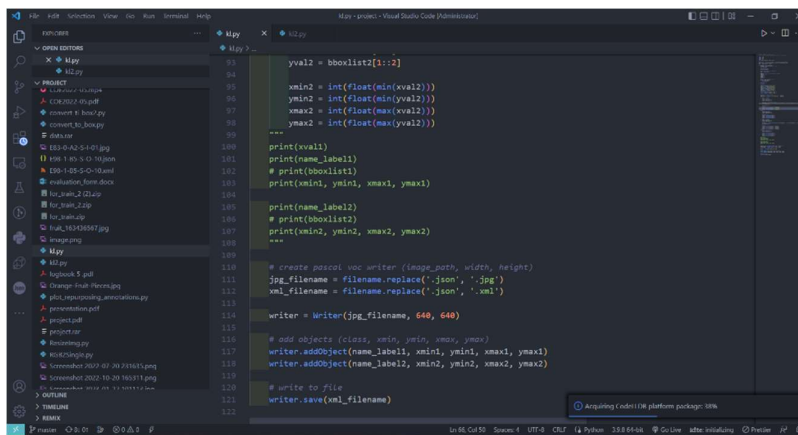
จากการ labeling แบบ Segmentation ก่อนจะไปทำการ train ต้องแปลงเป็น labeling แบบ bounding box เพื่อที่จะนำมาทำ Object Detection โดยใช้ภาษา Python ในการแปลง



รูปที่ 1.6 การแปลงภาพ Segmentation เป็น bounding box

4.4 การแปลง JSON file เป็น Pascal VOC format file

หลังจากได้ทำการแปลง labeling แบบ bounding box แล้ว ในไฟล์ Python เดียวกันได้ใช้ library เสริมในการแปลงไฟล์ JSON เป็น Pascal VOC คือ from pascal_voc_writer import writer และบันทึกไฟล์เป็น .xml



รูปที่ 1.7 การแปลง JSON เป็น Pascal VOC format

4.5 การ train object detection model ด้วย TensorFlow Lite Model Maker

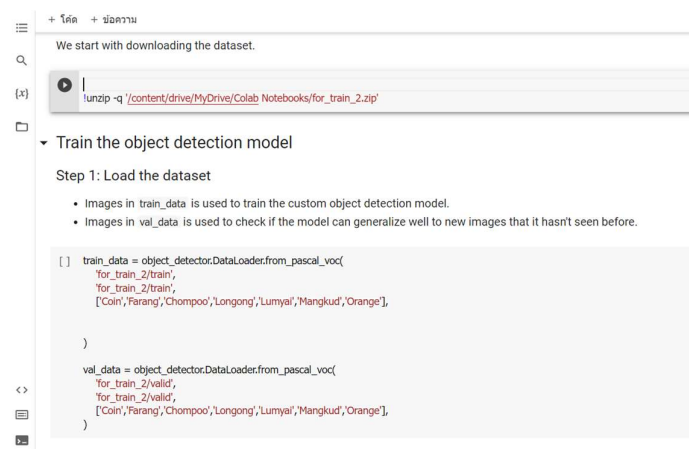
การ train model ในครั้งนี้ ทางผู้จัดทำได้เลือก Google Colab ซึ่ง Google Colab เป็นบริการ Python IDE บนคลาวด์ของ Google ที่สามารถเขียนและแก้ไขจากเบราว์เซอร์ และ model ที่ใช้ในการ train ทางผู้จัดทำได้เลือก model efficientdet_lite2 หลังจากได้ทดลองและวิเคราะห์ โดยจากอ้างอิงตารางนี้

| Model architecture | Size(MB)* | Latency(ms)** | Average Precision*** |
|--------------------|-----------|---------------|----------------------|
| EfficientDet-Lite0 | 4.4 | 146 | 25.69% |
| EfficientDet-Lite1 | 5.8 | 259 | 30.55% |
| EfficientDet-Lite2 | 7.2 | 396 | 33.97% |
| EfficientDet-Lite3 | 11.4 | 716 | 37.70% |
| EfficientDet-Lite4 | 19.9 | 1886 | 41.96% |

รูปที่ 1.8 ตารางเปรียบเทียบ

https://www.tensorflow.org/lite/models/modify/model_maker/object_detection

โดยการ train ใช้ library tfLite model maker เพื่อใช้ในการสร้าง model หลังจากการติดตั้ง library และ import library จากนั้นผู้จัดทำได้ upload dataset ที่ได้ทำการ export มาไว้ใน Google Drive และทำการ mount drive แล้วจึง load dataset ตามลำดับ ซึ่งคำสั่งจากภาพ format ของ dataset คือ Pascal VOC เป็น format เดียวกันที่ทางผู้จัดทำได้ทำเลือกไว้



```

+ โคลด + ข้อความ
We start with downloading the dataset.

!unzip -q '/content/drive/MyDrive/Colab Notebooks/for_train_2.zip'

Train the object detection model
Step 1: Load the dataset
• Images in train_data is used to train the custom object detection model.
• Images in val_data is used to check if the model can generalize well to new images that it hasn't seen before.

[ ] train_data = object_detector.DataLoader.from_pascal_voc(
    'for_train_2/train',
    'for_train_2/train',
    ['Coin', 'Farang', 'Chompoo', 'Longong', 'Lumyai', 'Mangkud', 'Orange'],
)

val_data = object_detector.DataLoader.from_pascal_voc(
    'for_train_2/valid',
    'for_train_2/valid',
    ['Coin', 'Farang', 'Chompoo', 'Longong', 'Lumyai', 'Mangkud', 'Orange'],
)

```

รูปที่ 1.8 ตัวอย่างคำสั่ง load dataset ใน Google Colab

ในส่วนของการ train ทางผู้จัดทำโครงการได้กำหนดค่า parameter ที่ใช้ในการ train ดังนี้
 model_spec = Efficientdet_lite2 (model ที่ใช้ในการฝึก) batch_size=8 (การแบ่งกลุ่มของ 1 ชุดข้อมูล

ในกรณีนี้มีรูปที่ใช้ในการฝึก 1200 รูป ขนาดของกลุ่มเป็น 8 ดังนั้นข้อมูลจะถูกแบ่งเป็น $1200/8 = 150$ กลุ่ม epochs=25 (เป็นจำนวนครั้งในการฝึกชุดข้อมูล ในกรณีนี้จะการฝึกทั้งหมด 25 ครั้ง)

```
model = object_detector.create(train_data, model_spec=spec, batch_size=8, train_whole_model=True, epochs=25, validation_data=val_data)

Epoch 1/25
615/615 [=====] - 284s 388ms/step - det_loss: 0.6459 - cls_loss: 0.4527 - box_loss: 0.0039 - reg_l2_loss: 0.0769 - loss: 0.7228 - learning_rate: 0.0090 - gradient_norm: 2.1014 - val_d
Epoch 2/25
615/615 [=====] - 235s 382ms/step - det_loss: 0.3769 - cls_loss: 0.2488 - box_loss: 0.0026 - reg_l2_loss: 0.0771 - loss: 0.4541 - learning_rate: 0.0099 - gradient_norm: 2.0074 - val_d
Epoch 3/25
615/615 [=====] - 235s 381ms/step - det_loss: 0.3467 - cls_loss: 0.2264 - box_loss: 0.0024 - reg_l2_loss: 0.0772 - loss: 0.4239 - learning_rate: 0.0097 - gradient_norm: 1.7185 - val_d
Epoch 4/25
615/615 [=====] - 234s 381ms/step - det_loss: 0.3227 - cls_loss: 0.2112 - box_loss: 0.0022 - reg_l2_loss: 0.0771 - loss: 0.3998 - learning_rate: 0.0095 - gradient_norm: 1.5603 - val_d
Epoch 5/25
615/615 [=====] - 241s 392ms/step - det_loss: 0.3133 - cls_loss: 0.2056 - box_loss: 0.0022 - reg_l2_loss: 0.0771 - loss: 0.3903 - learning_rate: 0.0092 - gradient_norm: 1.5094 - val_d
Epoch 6/25
615/615 [=====] - 236s 383ms/step - det_loss: 0.3048 - cls_loss: 0.2006 - box_loss: 0.0021 - reg_l2_loss: 0.0769 - loss: 0.3817 - learning_rate: 0.0088 - gradient_norm: 1.4428 - val_d
Epoch 7/25
615/615 [=====] - 236s 384ms/step - det_loss: 0.2882 - cls_loss: 0.1908 - box_loss: 0.0019 - reg_l2_loss: 0.0767 - loss: 0.3649 - learning_rate: 0.0078 - gradient_norm: 1.5170 - val_d
Epoch 8/25
615/615 [=====] - 235s 383ms/step - det_loss: 0.2991 - cls_loss: 0.1967 - box_loss: 0.0020 - reg_l2_loss: 0.0768 - loss: 0.3759 - learning_rate: 0.0083 - gradient_norm: 1.5319 - val_d
Epoch 9/25
615/615 [=====] - 236s 384ms/step - det_loss: 0.2882 - cls_loss: 0.1908 - box_loss: 0.0019 - reg_l2_loss: 0.0767 - loss: 0.3649 - learning_rate: 0.0078 - gradient_norm: 1.5170 - val_d
Epoch 10/25
615/615 [=====] - 235s 383ms/step - det_loss: 0.2841 - cls_loss: 0.1887 - box_loss: 0.0019 - reg_l2_loss: 0.0766 - loss: 0.3607 - learning_rate: 0.0072 - gradient_norm: 1.6137 - val_d
Epoch 11/25
615/615 [=====] - 235s 383ms/step - det_loss: 0.2762 - cls_loss: 0.1825 - box_loss: 0.0019 - reg_l2_loss: 0.0765 - loss: 0.3527 - learning_rate: 0.0066 - gradient_norm: 1.5850 - val_d
Epoch 12/25
615/615 [=====] - 236s 384ms/step - det_loss: 0.2648 - cls_loss: 0.1764 - box_loss: 0.0018 - reg_l2_loss: 0.0764 - loss: 0.3412 - learning_rate: 0.0060 - gradient_norm: 1.5388 - val_d
Epoch 13/25
615/615 [=====] - 235s 381ms/step - det_loss: 0.2594 - cls_loss: 0.1737 - box_loss: 0.0017 - reg_l2_loss: 0.0763 - loss: 0.3357 - learning_rate: 0.0053 - gradient_norm: 1.5740 - val_d
Epoch 14/25
615/615 [=====] - 235s 382ms/step - det_loss: 0.2514 - cls_loss: 0.1692 - box_loss: 0.0016 - reg_l2_loss: 0.0762 - loss: 0.3276 - learning_rate: 0.0047 - gradient_norm: 1.6351 - val_d
Epoch 15/25
615/615 [=====] - 236s 383ms/step - det_loss: 0.2455 - cls_loss: 0.1655 - box_loss: 0.0016 - reg_l2_loss: 0.0761 - loss: 0.3216 - learning_rate: 0.0040 - gradient_norm: 1.5766 - val_d
```

รูปที่ 1.9 ตัวอย่างคำสั่ง train model ใน Google Colab

4.6 ผลจาก train model

หลังจากที่ได้ทำการ train model แล้ว ก่อนนำไปใช้งานหรือทดสอบ ควรที่จะต้องมีการตรวจสอบความถูกต้องของโมเดล จากคำสั่งแล้วคือคำสั่ง evaluate ซึ่งใน dataset มีการแบ่งภาพไว้ใช้ในการตรวจสอบความถูกต้อง ได้ผลลัพธ์ดังภาพต่อไปนี้

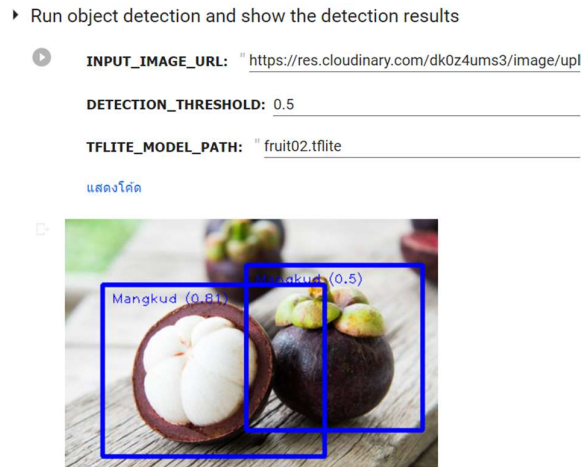
```
6/6 [=====] - 48s 4s/step

{'AP': 0.8904189,
 'AP50': 0.99619436,
 'AP75': 0.9921338,
 'APs': 0.7,
 'APm': 0.82784563,
 'AP1': 0.9223586,
 'ARmax1': 0.9120397,
 'ARmax10': 0.932022,
 'ARmax100': 0.93271387,
 'ARs': 0.7,
 'ARm': 0.87565666,
 'AR1': 0.9543674,
 'AP_coin': 0.77381325,
 'AP_farang': 0.93817616,
 'AP_chompoo': 0.9490674,
 'AP_longong_one_no_peel': 0.9163642,
 'AP_lamyai_one_peel': 0.86386836,
 'AP_mangkud': 0.8395978,
 'AP_orange_whole_no_peel': 0.95204496}
```

รูปที่ 1.10 ผลการตรวจสอบความถูกต้อง

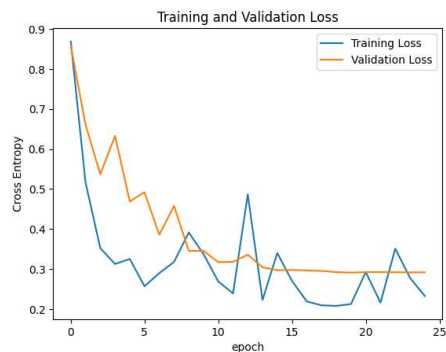
จากผลลัพธ์สามารถอธิบายได้ดังนี้ AP(Average Precision) คือ ความแม่นยำเฉลี่ย โดยความแม่นยำเฉลี่ยในภาพรวม ได้ร้อยละ 89 แล้วความแม่นยำเฉลี่ยแยกเป็นแต่ละ class เช่น ใน class coin ความแม่นยำเฉลี่ย

อยู่ที่ร้อยละ 77 เป็นต้น อีกทั้งยังมีคำสั่งใน Google Colab ที่สามารถทดสอบโดยใช้รูปภาพที่นำเข้ามาจากแหล่งภายนอกมาทดสอบได้ตัวอย่างเป็นไปดังภาพต่อไปนี้



รูปที่ 1.11 ผลการตรวจสอบความถูกต้องโดยใช้ภาพ

ต่อมาทำการพล็อตกราฟของ training loss และ validation loss จากการเทรน 25 รอบ (epochs) โดยแกนตั้งคือค่า loss ที่ได้จากการเทรนและแกนนอนคือจำนวนรอบที่ฝึก (epochs) โดยได้ผลดังนี้



กราฟที่ 1 ค่า training loss และ validation loss

โดยจะสังเกตได้ว่าค่า training loss นั้นจะลดลงอย่างมากในช่วงแรกและมีการเพิ่มขึ้นอย่างไม่คงที่ในช่วงหลัง ซึ่งอาจเกิดจากมีข้อมูลในการเทรนโมเดลไม่มากพอ ในส่วนของค่า validation loss นั้นจะลดลงอย่างมากในช่วงแรกเช่นเดียวกันแต่ในรอบการฝึกที่ 15 ขึ้นไปนั้นค่า validation loss จะคงที่ต่างจาก training loss

4.7 การนำ model ลงใน mobile application

ในด้านการพัฒนา mobile application ทางผู้จัดทำโครงได้เลือกใช้ภาษา Kotlin และพัฒนามบน Android Studio โดยเลือกใช้ code พื้นฐานของ TensorFlow และทำการออกแบบเพิ่มเติม แล้วจึงได้นำ model ที่ได้จากการ train ข้างต้นมาใช้กับตัว application นี้



รูปที่ 1.12 ภาพ interface ของ application

ในตัวของ application สามารถถ่ายรูปและตอบกลับเป็น ภาพที่ถ่ายไป ชื่อผลไม้ พร้อมกลับมีเปอร์เซ็นต์ที่ทำนายได้ เวลาที่ใช้ในการทำงานของ application มีความหน่วงประมาณ 1 วินาที



รูปที่ 1.13 ภาพ interface ของ application ในการใช้งานจริง

บทที่ 5 สรุปผลการดำเนินโครงการ

5.1 สรุปผล

โมเดลการเรียนรู้เชิงลึกโดยใช้โครงข่ายประสาทแบบคอนโวลูชันในการจดจำประเภทของอาหาร ที่ถูกพัฒนาบนสถาปัตยกรรมโครงข่ายของ EfficientDet ในโครงการนี้สามารถทำนายชนิดของผลไม้และตอบกลับมาได้อย่างถูกต้อง โดยผลไม้ที่ได้รับฝึกของโมเดลได้แก่ ส้ม ลองกอง ลำไย ฝรั่ง มังคุด และชมพู ซึ่งมีความแม่นยำร้อยละ 70 เนื่องจากข้อมูลมีความหลากหลายน้อยไปในบางจุด และมีความคล้ายคลึงกันในลักษณะทางกายภาพ เช่น รูปทรง สี เป็นต้น ในส่วนของการนำมาประยุกต์ใช้ใน mobile application สามารถใช้งานได้มีการตอบสนองที่เร็ว แต่ยังทำงานได้เพียงอุปกรณ์ android ยังไม่สามารถทำงานบนอุปกรณ์ iOS ได้

5.2 ข้อเสนอแนะ

5.2.1 เลือกโปรแกรม label ให้เหมาะสมกับโมเดลเพื่อป้องกันไม่ให้เกิดมีปัญหากับประเภทไฟล์ของ datasets สำหรับการเทรน

5.2.2 ตรวจสอบ bounding box ของหลักจาก label ก่อนการเทรนโมเดลว่าถูกต้องหรือไม่

5.2.3 ผลไม้บางชนิดมีความหลากหลายน้อย เช่น ส้มที่เทรนมีลักษณะเปลือกสีเขียว ในชีวิตมีทั้งสีเขียวและสีส้ม

อ้างอิง

1. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. December, 1989 .Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation. Vol. 1, No. 4, pp. 541-551.
2. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. November, 1998 .Gradient- Based Learning Applied to Document Recognition. in Proceedings of the IEEE. Vol. 86, No. 11, pp. 2278–2324.
3. M. Subhi, and S. H. Ali. 2018. A Deep Convolutional Neural Network for Food Detection and Recognition. IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES), Sarawak, Malaysia. pp. 284-287.
4. M. M. Rahman, S. M. Islam, R. Sassi, and M. Aktaruzzaman. November, 2019. Convolutional neural networks performance comparison for handwritten Bengali numerals recognition. SN Applied Sciences.
5. ราตรี คำโมง. มกราคม-มิถุนายน 2565. การตรวจจับมังคุดด้วยการเรียนรู้เชิงลึก Mangoteen Detection Using Deep Learning. วารสารเทคโนโลยีสารสนเทศ ปีที่ 18 ฉบับที่ 1.
6. Joachim Dehais, Student Member, IEEE, Marios Anthimopoulos, Member, IEEE, Sergey Shevchik, and Stavroula Mougiakakou, Member, IEEE. MAY 2017 . Two-View 3D Reconstruction for Food Volume Estimation. IEEE TRANSACTIONS ON MULTIMEDIA. VOL. 19, NO. 5.
7. Aqeel Anwar, “What is Average Precision in Object Detection & Localization Algorithms and how to calculate it? | by Aqeel Anwar | Towards Data Science” Towards Data Science

ภาคผนวก

ขั้นตอนการแปลงไฟล์ JSON เป็น Pascal VOC format ด้วยภาษา Python

```

1 # Python program to read json file
2 #
3   From https://www.geeksforgeeks.org/read-json-file
   -using-python/
4 import glob
5 import math
6 import json
7 import re
8 from pascal_voc_writer import Writer
9
10 #
11   path edit in path for all example --> 'D:/Desktop
   p/project/chompoo/'
12 path_for_all =
13   'D:/Desktop/project/for_train3/orange_whole/New fo
   lder/'
14 path = path_for_all + '*.json'
15
16 for filename in glob.glob(path):
17     filename = filename.replace('\\', '/')
18     print(filename)
19
20     # Opening JSON file
21     f = open(filename)
22
23     # returns JSON object as a dictionary
24     data = json.load(f)
25
26     bboxa = [] # List for receive json dictionary
27     labelstr = []
28     xval1 = []
29     xval2 = []
30     yval1 = []
31     yval2 = []
32
33     # Iterating through the json list
34     for i in data['shapes']:
35         bboxa.append(i)
36
37     # Closing file
38     f.close()
39
40     # Put bounding box from list in str
41     type1 = str(bboxa[0]['shape_type'])
42     bboxstr1 = str(bboxa[0]['points'])
43     name_label1 = str(bboxa[0]['label'])
44     type2 = str(bboxa[1]['shape_type'])
45     bboxstr2 = str(bboxa[1]['points'])
46     name_label2 = str(bboxa[1]['label'])
47
48     # Put label in list
49     labelstr.append(str(bboxa[0]['label']))
50     labelstr.append(str(bboxa[1]['label']))
51
52     # Spilt str to get x y of bbox
53     bboxlist1 = re.findall(r"[-+]?(\d*\.\d+)"
54     , bboxstr1)
55     bboxlist2 = re.findall(r"[-+]?(\d*\.\d+)"
56     , bboxstr2)
57
58     if type1 == 'circle':
59         #define xval yval
60         xval1 = bboxlist1[::2]
61         yval1 = bboxlist1[1::2]
62
63         #calculate distance point to point
64         distance_1 = int(math.sqrt(pow(abs(int(
65         float(xval1[1])) - int(float(xval1[0]))), 2) + pow
66         (abs(int(float(yval1[1])) - int(float(yval1[0]))),
67         , 2)))
68         print(distance_1)

```

```

1
2         #create new point
3         #Label1
4         xmax1 = int(float(xval1[0])) + distance_1
5         xmin1 = int(float(xval1[0])) - distance_1
6         ymax1 = int(float(yval1[0])) + distance_1
7         ymin1 = int(float(yval1[0])) - distance_1
8     else:
9         #define xval yval
10        xval1 = bboxlist1[:,2]
11        yval1 = bboxlist1[1::2]
12
13        # float to int
14        xmin1 = int(float(min(xval1)))
15        xmax1 = int(float(max(xval1)))
16        ymin1 = int(float(min(yval1)))
17        ymax1 = int(float(max(yval1)))
18
19    if type2 == 'circle':
20        xval2 = bboxlist2[:,2]
21        yval2 = bboxlist2[1::2]
22
23        distance_2 = int(math.sqrt(pow(abs(int(
24            float(xval2[1])) - int(float(xval2[0]))), 2) + pow
25            (abs(int(float(yval2[1])) - int(float(yval2[0]))),
26            , 2)))
27        print(distance_2)
28
29        #Label2
30        xmax2 = int(float(xval2[0])) + distance_2
31        xmin2 = int(float(xval2[0])) - distance_2
32        ymax2 = int(float(yval2[0])) + distance_2
33        ymin2 = int(float(yval2[0])) - distance_2
34    else:
35        xval2 = bboxlist2[:,2]
36        yval2 = bboxlist2[1::2]
37
38        xmin2 = int(float(min(xval2)))
39        ymin2 = int(float(min(yval2)))
40        xmax2 = int(float(max(xval2)))
41        ymax2 = int(float(max(yval2)))
42    """
43    print(xval1)
44    print(name_label1)
45    # print(bboxlist1)
46    print(xmin1, ymin1, xmax1, ymax1)
47
48    print(name_label2)
49    # print(bboxlist2)
50    print(xmin2, ymin2, xmax2, ymax2)
51    """
52
53    #
54    create pascal voc writer (image_path, width, height)
55
56    jpg_filename = filename.replace('.json',
57    '.jpg')
58    xml_filename = filename.replace('.json',
59    '.xml')
60
61    writer = Writer(jpg_filename, 640, 640)
62
63    # add objects (class, xmin, ymin, xmax, ymax)
64    writer.addObject(
65        name_label1, xmin1, ymin1, xmax1, ymax1)
66    writer.addObject(
67        name_label2, xmin2, ymin2, xmax2, ymax2)
68
69    # write to file
70    writer.save(xml_filename)

```

รูปที่ 1 การแปลง JSON เป็น Pascal VOC format