



โครงการ

การคัดแยกผลไม้ 3 ชนิด แอปเปิ้ล กล้วย และมะเขือเทศ

สมาชิก

63102105105 นายธนวัฒน์ จ้อยจิต

63102105112 นายอัศรพล พิกุลศรี

63102105140 นายชลสิทธิ์ สีสถาน

เสนอ

ผู้ช่วยศาสตราจารย์ กรรณิการ์ กมลรัตน์

โครงการนี้เป็นส่วนหนึ่งของรายวิชา
การประมวลผลภาพดิจิทัล (14123401)
ภาคเรียนที่ 1 ปีการศึกษา 2564

สาขาวิชาวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์และเทคโนโลยี
มหาวิทยาลัยราชภัฏสกลนคร

คำนำ

โครงการนี้มีวัตถุประสงค์เพื่อศึกษา และลงมือปฏิบัติ เพื่อให้สอดคล้องตามวัตถุประสงค์ของรายวิชาประมวลผลภาพดิจิทัล (14123401) คือ ผู้เรียนสามารถประยุกต์ใช้งานด้านการประมวลผลภาพ ในโครงการนี้ได้ดำเนินการศึกษาเกี่ยวกับการสร้างแบบจำลองการคัดแยกผลไม้ ซึ่งมี 3 ชนิด คือ แอปเปิ้ล กล้วย และมะเขือเทศ

ปัจจุบันการคัดแยกผลไม้ที่มีจำนวนมาก และหลากหลายรูปแบบโดยใช้แรงงานมนุษย์นั้นค่อนข้างใช้เวลาในการคัดแยก และมีประสิทธิภาพในการทำงานไม่ค่อยดีเนื่องจากแรงงานมนุษย์อาจมีความเหนื่อยล้า ทำให้ประสิทธิภาพในการคัดแยกผลไม้ทำได้ไม่เต็มที่ ซึ่งอาจจะเสียต้นทุนหรือทรัพยากรที่มี มากจนเกินไป โดยคณะผู้จัดทำได้เล็งเห็นถึงความสำคัญของปัญหานี้ จึงได้ดำเนินการทำโครงการเพื่อสร้างแบบจำลองที่จะสามารถคัดแยกผลไม้ได้โดยไม่ต้องใช้แรงงานของมนุษย์ ซึ่งใช้เทคโนโลยีการประมวลผลภาพ ร่วมกับโครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network) ซึ่งเป็นการเรียนรู้เชิงลึก (Deep Learning) โดยใช้แบบจำลองที่ชื่อว่า ResNet50 มาทำการถ่ายโอนการเรียนรู้ (Transfer Learning) เพื่อสร้างแบบจำลองคัดแยกผลไม้ทั้ง 3 ชนิด ได้อย่างมีประสิทธิภาพ

คณะผู้จัดทำ

สารบัญ

คำนำ.....	I
สารบัญ.....	II
สารบัญภาพ	III
บทที่ 1 บทนำ	1
1.1 หลักการและเหตุผล	1
1.2 วัตถุประสงค์ของโครงการ	1
1.3 ประโยชน์ที่คาดว่าจะได้รับ	1
บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง	2
2.1 ทฤษฎีที่เกี่ยวข้อง.....	2
2.1.1 การประมวลผลภาพ (Image Processing)	2
2.1.2 การเรียนรู้ของเครื่อง (Machine Learning).....	2
2.1.3 การวัดผลแบบจำลอง Machine Learning	2
2.1.4 การเรียนรู้เชิงลึก (Deep Learning).....	3
2.1.5 โครงข่ายประสาทเทียมแบบ 50 ชั้นลึก (ResNet50).....	3
2.2 งานวิจัยที่เกี่ยวข้อง	4
2.2.1 แบบจำลองการคัดแยกผลไม้แบบหนึ่งชนิดด้วยการเรียนรู้ของเครื่องเพื่อตรวจจับรูปแบบ เปิด โดย ศิริชัย โชติชาติมาลา.....	4
2.2.2 การประมวลผลภาพสำหรับการจำแนกคุณภาพมะม่วงพันธุ์โชคอนันต์โดยการจำลองการ มองเห็นของมนุษย์ด้วยวิธีการเรียนรู้เชิงลึก โดย นพรุจ พัฒนสาร และ ณัฐวุฒิ ศรีวิบูลย์	4
บทที่ 3 วิธีการดำเนินงาน.....	5
3.1 การเก็บรวบรวมข้อมูล.....	5
3.2 การออกแบบหน้าจอ	6
3.3 การเขียนโปรแกรมควบคุมการทำงาน	7
3.3.1 การ import library มีดังนี้.....	7
3.3.2 ขั้นตอนการนำเข้าภาพ (Get Images) เข้ามาใน Google Colaboratory	8

3.3.3 ขั้นตอนการแยกภาพเป็นชุดฝึกฝน (Training Data) และชุดตรวจสอบ (Validation data) มีดังนี้	9
3.3.3.1 ในส่วนการตั้งค่าตัวแปรมีดังนี้	9
3.3.3.2 ในส่วนการแบ่งชุดข้อมูลสำหรับการฝึกฝนแบบจำลองมีดังนี้	9
3.3.4 ขั้นตอนการฝึกฝนแบบจำลอง (Training Model)	11
3.3.4.1 ในส่วนการเตรียมแบบจำลองก่อนการฝึกฝนแบบจำลองมีดังนี้	11
3.3.4.2 ในส่วนการฝึกฝนแบบจำลองมีดังนี้	13
3.3.5 ขั้นตอนการทำนายและประเมินผล (Prediction and Evaluation Model)	13
3.3.5.1 ในส่วนการสรุปผลของการฝึกฝนจำลองมีดังนี้	13
3.3.5.2 ในส่วนการนำแบบจำลองมาทดสอบทำนาย (Prediction) มีดังนี้	14
3.3.5.3 ในส่วนการประเมินผล (Evaluation) ของแบบจำลองมีดังนี้	16
3.3.6 ขั้นตอนการการนำแบบจำลองไปปรับใช้ (Deploy Model)	17
บทที่ 4 ผลการดำเนินงาน	18
4.1 ผลการทำงานของโปรแกรม	18
4.1.1 ผลลัพธ์การฝึกฝนแบบจำลอง	18
4.1.2 ผลลัพธ์การทำนายของแบบจำลองจากชุดข้อมูลทดสอบ	19
4.1.3 ผลลัพธ์การประเมินผลแบบจำลองจากชุดข้อมูลทดสอบ	19
4.2 ผลลัพธ์ที่ได้	20
บทที่ 5 สรุปผลการดำเนินงาน	21
5.1 สรุปผล	21
5.2 ข้อเสนอแนะ	21
บรรณานุกรม	22

สารบัญภาพ

ภาพที่	หน้า
ภาพที่ 1 สมการ Precision และ Recall ที่มา Mr.P L.	2
ภาพที่ 2 สมการ F1 Score ที่มา Mr.P L.	3
ภาพที่ 3 ตัวอย่างภาพ แอปเปิ้ล กล้วย และมะเขือเทศจาก ชุดข้อมูลภาพผลไม้ 360 (Fruit360).....	5
ภาพที่ 4 ภาพแสดงโฟลเดอร์ของรูปภาพ.....	5
ภาพที่ 5 ภาพ Notebook Jupyter บนแพลตฟอร์ม Google Colaboratory	6
ภาพที่ 6 การนำเข้า library	7
ภาพที่ 7 คำสั่ง git clone.....	8
ภาพที่ 8 โฟลเดอร์รูปภาพหลังจากที่ใช้ คำสั่ง git clone.....	8
ภาพที่ 9 คำสั่งในการเช็คจำนวนไฟล์ในโฟลเดอร์ train	8
ภาพที่ 10 การตั้งค่าตัวแปรขนาดของรูปภาพและที่อยู่ของโฟลเดอร์ train	9
ภาพที่ 11 คำสั่งที่ใช้ในการแบ่งข้อมูลเป็นชุดฝึกฝน	9
ภาพที่ 12 คำสั่งในการแบ่งภาพเป็นชุดตรวจสอบในการฝึกฝนแบบจำลอง.....	10
ภาพที่ 13 คำสั่งในการตรวจสอบคลาสของรูปภาพที่ได้จากการแบ่งชุดข้อมูลแบบอัตโนมัติ	10
ภาพที่ 14 คำสั่งการดาวน์โหลดแบบจำลอง ResNet50	11
ภาพที่ 15 คำสั่งในการตรวจสอบคุณลักษณะของแบบจำลอง	11
ภาพที่ 16 ผลลัพธ์จากคำสั่ง summary	11
ภาพที่ 17 ส่วนของคำสั่งในการปรับเปลี่ยนแบบจำลอง.....	12
ภาพที่ 18 การเรียกใช้ เมธอด compile	12
ภาพที่ 19 ผลลัพธ์การคอมไพล์แบบจำลองหลังจากการปรับเปลี่ยน.....	13
ภาพที่ 20 คำสั่งในการฝึกฝนแบบจำลอง.....	13
ภาพที่ 21 ชุดคำสั่งการสร้างกราฟ เพื่อสรุปค่าความถูกต้อง (Accuracy)	13
ภาพที่ 22 ชุดคำสั่งการสร้างกราฟ เพื่อสรุปค่าความสูญเสีย (Loss)	14
ภาพที่ 23 คำสั่งในการเช็คจำนวนไฟล์ในโฟลเดอร์ test.....	14
ภาพที่ 24 ฟังก์ชันสำหรับเรียกใช้แบบจำลองในการทำนายภาพ	15
ภาพที่ 25 คำสั่งในการสุ่มภาพแอปเปิ้ลจำนวน 5 ภาพจากโฟลเดอร์ test.....	15
ภาพที่ 26 คำสั่งในการสุ่มภาพกล้วยจำนวน 5 ภาพจากโฟลเดอร์ test.....	16
ภาพที่ 27 คำสั่งในการสุ่มภาพมะเขือเทศจำนวน 5 ภาพจากโฟลเดอร์ test	16
ภาพที่ 28 คำสั่งในการอ่านภาพเข้าสู่ระบบเพื่อใช้ในการประเมินผลแบบจำลอง	16
ภาพที่ 29 คำสั่งในการกำหนดค่าจริงของภาพ เพื่อใช้ในการประเมินผลแบบจำลอง	17

ภาพที่ 30 คำสั่งการเรียกใช้ไลบรารี classification_report เพื่อประเมินผลแบบจำลอง.....	17
ภาพที่ 31 คำสั่งบันทึกแบบจำลองเพื่อนำไปใช้งาน	17
ภาพที่ 32 ผลลัพธ์การฝึกฝนแบบจำลอง	18
ภาพที่ 33 กราฟแสดงแนวโน้มของค่าความถูกต้อง (Accuracy) ของการฝึกฝนแบบจำลอง.....	18
ภาพที่ 34 กราฟแสดงแนวโน้มของค่าความสูญเสีย (Loss) ของการฝึกฝนแบบจำลอง	18
ภาพที่ 35 ผลลัพธ์การทำนายของแบบจำลองจากการสุ่มภาพจำนวน 5 ภาพ	19
ภาพที่ 36 ผลลัพธ์การประเมินผลแบบจำลองด้วยไลบรารี classification_report.....	19

บทที่ 1 บทนำ

1.1 หลักการและเหตุผล

ประเทศไทยเป็นประเทศเกษตรกรรม กว่าครึ่งหนึ่งของประชากรไทยประกอบอาชีพเกษตรกรรม ดังนั้นการพัฒนาทางการเกษตรจึงมีความสำคัญเป็นอย่างมาก เพราะจะส่งผลให้ประชากรส่วนใหญ่ของประเทศมีรายได้ และมีความเป็นอยู่ที่ดีขึ้น ซึ่งในภาคอุตสาหกรรมขนาด/ย่อม ในธุรกิจการผลิตผลไม้สด (เกษตรกร) และธุรกิจแปรรูปผลไม้ ที่มีธุรกิจที่เกี่ยวข้องเชื่อมโยงหลายภาคส่วน เช่น ธุรกิจด้านการคมนาคมขนส่ง หนึ่งตำบลหนึ่งผลิตภัณฑ์ (OTOP) เป็นต้น

ในปัจจุบันการคัดแยกผลไม้ที่มีจำนวนมาก และหลากหลายรูปแบบโดยใช้แรงงานมนุษย์นั้นค่อนข้างใช้เวลาในการคัดแยก และมีประสิทธิภาพในการทำงานไม่ค่อยดีเนื่องจากแรงงานมนุษย์อาจมีความเหนื่อยล้า ซึ่งทำให้ประสิทธิภาพในการคัดแยกผลไม้ทำได้ไม่เต็มที่ ทำให้อาจเสียต้นทุนหรือทรัพยากรที่มี มากจนเกินไป ซึ่งคณะผู้จัดทำได้เล็งเห็นถึงความสำคัญของปัญหานี้ จึงได้ดำเนินการจัดทำโครงการเพื่อสร้างแบบจำลองที่จะสามารถคัดแยกผลไม้ได้โดยไม่ต้องใช้แรงงานของมนุษย์ ซึ่งใช้เทคโนโลยีการประมวลผลภาพ ร่วมกับโครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network) ซึ่งเป็นการเรียนรู้เชิงลึก (Deep Learning) โดยใช้แบบจำลองที่ชื่อว่า ResNet50 มาทำการ การถ่ายโอนการเรียนรู้ (Transfer Learning) เพื่อสร้างแบบจำลองคัดแยกผลไม้ทั้ง 3 ชนิดได้อย่างมีประสิทธิภาพ

1.2 วัตถุประสงค์ของโครงการ

- 1.2.1 เพื่อพัฒนาแบบจำลองที่ช่วยในการคัดแยกผลไม้ทั้ง 3 ชนิดคือ แอปเปิ้ล กล้วยและมะเขือเทศได้อย่างมีประสิทธิภาพ
- 1.2.2 เพื่อทดสอบประสิทธิภาพของการคัดแยกผลไม้ทั้ง 3 ชนิด จากการถ่ายโอนการเรียนรู้ (Transfer Learning) จากแบบจำลองต้นแบบคือ ResNet50
- 1.2.3 เพื่อให้สอดคล้องตามวัตถุประสงค์ของรายวิชาประมวลผลภาพดิจิทัล คือ ผู้เรียนสามารถประยุกต์ใช้งานด้านการประมวลผลภาพ

1.3 ประโยชน์ที่คาดว่าจะได้รับ

- 1.7.1 สามารถคัดแยกผลไม้ทั้งสามชนิดได้อย่างมีประสิทธิภาพ และรวดเร็ว
- 1.7.2 แบบจำลองสามารถลดความผิดพลาดในการคัดแยกผลไม้จากแรงงานมนุษย์ลงได้
- 1.7.3 ช่วยลดต้นทุนและเวลาในการคัดแยกผลไม้ของเกษตรกรลงได้
- 1.7.4 แบบจำลองสามารถนำไปประยุกต์ใช้ในการคัดแยกผลไม้ชนิดอื่นๆ ได้

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 การประมวลผลภาพ (Image Processing)

การประมวลผลภาพ (Image Processing) คือเทคโนโลยีที่สามารถนำภาพมาประมวลผล เช่นการทำให้ภาพมีความคมชัดมากขึ้น หรือการแบ่งส่วนของวัตถุในภาพ เป็นต้น โดยการประมวลผลภาพสามารถนำไปใช้ประโยชน์ได้ในด้านต่างๆ เช่น ระบบตรวจจับใบหน้า ระบบตรวจจับลายนิ้วมือ เป็นต้น หลักการของการประมวลผลภาพมีขั้นตอนดังนี้ การได้มาของภาพ (Image Acquisition) ก่อนการประมวลผล (Preprocessing) การแยกส่วนภาพ (Segmentation) การแทนและการอธิบายข้อมูลภาพ (Representation and Description) การจดจำและการตีความ (Recognition and Interpretation) และทุกขั้นตอนต้องอาศัยฐานความรู้ (Knowledge Base)

2.1.2 การเรียนรู้ของเครื่อง (Machine Learning)

การเรียนรู้ของเครื่อง คืออัลกอริทึมที่มีพัฒนาการเรียนรู้ข้อมูล และทำนายข้อมูลได้ เป็นรูปแบบหนึ่งของการวิเคราะห์ข้อมูล ที่ดำเนินการวิเคราะห์ด้วยแบบจำลองแบบอัตโนมัติ ซึ่งเป็นส่วนหนึ่งของปัญญาประดิษฐ์ (Artificial Intelligence) ซึ่งอัลกอริทึมจะทำงานโดยอาศัยการสร้างแบบจำลองทางคณิตศาสตร์จากข้อมูลฝึกฝน (Training data) เพื่อนำมาทดสอบกับข้อมูลทดสอบ (Test data) ซึ่งแนวคิดของเทคโนโลยีนี้ตั้งอยู่บนรากฐานแนวคิดที่ว่า ระบบต่างๆ นั้นสามารถที่จะเรียนรู้และมีปฏิสัมพันธ์กับชุดข้อมูลต่างๆ รวมถึงสามารถระบุ และทราบรูปแบบต่างๆ ที่เกิดขึ้น และนำไปสู่การตัดสินใจได้เองโดยไม่จำเป็นต้องพึ่งพามนุษย์

2.1.3 การวัดผลแบบจำลอง Machine Learning

ผลการทำนายของแบบจำลองคือ $y_{\text{prediction}}$ ค่าจริง y_{actual} metrics คือค่าที่ได้จากการคำนวณเปรียบเทียบผล prediction vs. actual ว่าแบบจำลองทำนายได้ถูกต้องแค่ไหน Accuracy คือ metric ที่ใช้งานง่ายที่สุด ซึ่งใช้สำหรับบอกว่าแบบจำลองจะทำนายถูกทั้งหมดกี่เปอร์เซ็นต์

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

ภาพที่ 1 สมการ Precision และ Recall ที่มา Mr.P L.

เมื่อ tp คือ ข้อมูลที่ทำนายแล้วถูกต้องเมื่อเทียบกับเฉลย

fp คือ ข้อมูลที่อยู่ในเฉลยแต่ไม่มีในการทำนาย (ตรงข้ามกับ fn)

fn คือ ข้อมูลที่ทำนายแล้วไม่ถูกต้องเมื่อเทียบกับเฉลย

Precision คือ ค่าความแม่นยำ เกิดจากการนำ ค่า tp มาเทียบกับ fp

Recall คือ ค่าความถูกต้อง เกิดจากการนำค่า tp มาเทียบกับ fn

F1 Score คือ ค่าเฉลี่ยของ Precision และ Recall

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

ภาพที่ 2 สมการ F1 Score ที่มา Mr.P L.

2.1.4 การเรียนรู้เชิงลึก (Deep Learning)

การเรียนรู้เชิงลึกคือการเรียนรู้ของเครื่อง (Machine Learning) ซึ่งเป็นการฝึกฝนคอมพิวเตอร์ให้สามารถทำงานได้เหมือนมนุษย์ เช่นการจดจำใบหน้า การจดจำคำพูด แทนที่จะจัดระเบียบข้อมูลที่จะรันผ่านทางสมการที่กำหนดไว้ล่วงหน้า การเรียนรู้เชิงลึกจะกำหนดค่าพารามิเตอร์พื้นฐานเกี่ยวกับข้อมูลและฝึกให้คอมพิวเตอร์เรียนรู้ด้วยตัวเองโดยการจดจำรูปแบบโดยใช้การประมวลผลหลายชั้นเรียกว่า Hidden Layer โดยอยู่ในลักษณะคล้ายกับโครงข่ายประสาท (Neurons) ของสมองมนุษย์เรียกว่าโครงข่ายประสาทเทียม (Neural Network) ซึ่งคำว่า การเรียนรู้เชิงลึก (Deep Learning) นั้นมาจากการใช้โครงข่ายประสาทเทียมมากกว่า 2 ชั้น เพื่อให้เกิดการเรียนรู้และสร้างแบบจำลอง ดังนั้นแสดงว่าแต่ละชั้นของโครงข่ายประสาทเทียม (Neural Network) ยิ่งถูกใช้จำนวนมากในขั้นตอนการประมวลผล ยิ่งทำให้มีโครงสร้างการเรียนรู้ที่ลึกมากขึ้น

2.1.5 โครงข่ายประสาทเทียมแบบ 50 ชั้นลึก (ResNet50)

สถาปัตยกรรมโครงข่ายประสาทเทียมแบบ 50 ชั้นลึก (ResNet50) ซึ่ง ResNet นั้นย่อ มาจาก Residual Network เป็นโครงข่ายประสาทเทียมแบบใหม่ที่เปิดตัวครั้งแรกโดย Kaiming He, Xiangyu Zhang, Shaoqing Ren และ Jian Sun ในเอกสารวิจัยเกี่ยวกับวิสัยทัศน์คอมพิวเตอร์ (Computer Vision) 2015 บทความวิจัยเรื่อง “Deep Residual Learning for Image Recognition” ซึ่งสถาปัตยกรรมโครงข่ายประสาทเทียมแบบ 50 ชั้นลึก (ResNet50) นั้นเป็นอัลกอริทึมที่ผ่านการฝึกฝนจากชุดข้อมูลมากกว่าล้านภาพในชุดข้อมูล ImageNet มีชั้นโครงข่ายประสาทเทียม 50 ชั้น

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 แบบจำลองการคัดแยกผลไม้แบบหนึ่งชนิดด้วยการเรียนรู้ของเครื่องเพื่อตรวจจับรูปแบบเปิด โดย ศิริชัย โชติชาติมาลา

งานวิจัยกล่าวถึงการคัดแยกผลไม้แบบหนึ่งชนิดคือภาพแอปเปิ้ล ชุดข้อมูลภาพผลไม้ 360 องศา (Fruit-360) โดยใช้เทคนิคการตรวจสิ่งใหม่ (Novelty Detection) และใช้แบบจำลองจำแนกข้อมูลด้วยเครื่องเวกเตอร์ค้ำยันประเภทเดียว (One-Class Support Vector Machine) และแบบจำลองแบบป่าแยก (Isolation Forest) ซึ่งสามารถทำนายภาพแอปเปิ้ลได้ค่าความถูกต้อง (Accuracy) ของแบบจำลองจำแนกข้อมูลด้วยเครื่องเวกเตอร์ค้ำยันประเภทเดียว (One-Class Support Vector Machine) และแบบจำลองป่าแยก (Isolation Forest) ที่ค่าความถูกต้องร้อยละ 91 เท่ากัน ซึ่งงานวิจัยได้มีข้อเสนอแนะให้ประยุกต์ใช้กับอัลกอริทึมการเรียนรู้เชิงลึก (Deep Learning) แบบจำลองจำแนกข้อมูลด้วยโครงข่ายประสาทเทียม

2.2.2 การประมวลผลภาพสำหรับการจำแนกคุณภาพมะม่วงพันธุ์โชคอนันต์โดยการจำลองการมองเห็นของมนุษย์ด้วยวิธีการเรียนรู้เชิงลึก โดย นพรุจ พัฒนสาร และ ณัฐวุฒิ ศรีวิบูลย์

งานวิจัยกล่าวถึงการจำแนกคุณภาพมะม่วงพันธุ์โชคอนันต์โดยการจำลองการมองเห็น โดยชุดข้อมูลภาพนั้นได้มาจากการเก็บรวบรวมข้อมูลภาพถ่ายมะม่วงจากเจ้าของสวนมะม่วงและเก็บข้อมูลการจำแนกคุณภาพมะม่วงพันธุ์โชคอนันต์โดยได้กำหนดคุณภาพออกเป็น 4 ระดับ คุณภาพระดับเกรด A, คุณภาพระดับเกรด B, คุณภาพระดับเกรด C และคุณภาพระดับเกรด D ซึ่งใช้อัลกอริทึมการเรียนรู้เชิงลึก (Deep Learning) แบบจำลองการจำแนกข้อมูลด้วยโครงข่ายประสาทเทียม (Convolutional Neural Network) ได้ค่าความแม่นยำสูงสุดคือร้อยละ 99.79

2.2.3 การตรวจหารอยขีดของแอปเปิ้ลโดยเทคนิควิเคราะห์ภาพถ่ายคลื่นรังสีความร้อนอินฟราเรด โดย ณัฐภัทร พิตรปรีชา

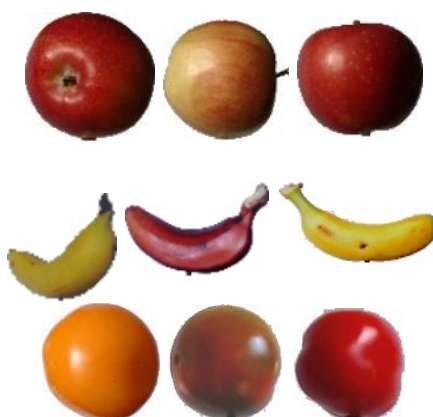
งานวิจัยกล่าวถึงการวิเคราะห์ภาพถ่ายคลื่นรังสีความร้อนแบบอินฟราเรดเพื่อตรวจหารอยขีดของแอปเปิ้ลเป็นวิธีการนำเสนอเพื่อตรวจหารอยขีดที่เกิดขึ้นใหม่ ซึ่งเทคนิคที่ใช้ในการทำงานของระบบตรวจสอบรอยขีดมีโครงสร้างหลักคือ ResNet50 เป็นการเรียนรู้เชิงลึก (Deep Learning) ใช้ในการเรียนรู้ข้อมูลภาพด้วยหลักการถ่ายโอนการเรียนรู้ (Transfer Learning) โดยประยุกต์ร่วมกับ YOLO ซึ่งเป็นระบบตรวจหาวัตถุในภาพที่ประมวลผลได้ตามเวลาจริง และใช้การวัดผลแบบ Confusion matrix และ F score ที่เป็นมาตรฐานชี้วัดสำหรับระบบประมวลผลภาพ โดยผลการทดสอบระบบตรวจสอบรอยขีดห้าครั้งพบว่า ผลลัพธ์การตรวจหารอยขีดของแอปเปิ้ลจากภาพถ่ายรังสีความร้อนแบบอินฟราเรดมีค่า Confusion matrix และ F score ที่ค่อนข้างดี ความถูกต้องของการระบุรอยขีดในตาราง Confusion matrix สูงกว่าร้อยละ 90 ระบุข้อมูลการเกิดรอยขีดทั้งตำแหน่งซ้ายขวาของแอปเปิ้ลได้ถูกต้องทั้งหมด

บทที่ 3 วิธีการดำเนินงาน

3.1 การเก็บรวบรวมข้อมูล

ในการแยกผลไม้ทั้ง 3 ชนิดในโครงการนี้ ทางคณะผู้จัดทำได้ตัดสินใจใช้ชุดข้อมูลภาพจากแพลตฟอร์ม Kaggle โดยเป็นชุดข้อมูลภาพผลไม้ 360 องศา (Fruit-360) ภายใต้ลิขสิทธิ์ Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0) โดยได้รับลิขสิทธิ์จาก © 2007-2020 Mihai Oltean, Horea Muresan ประกอบด้วยภาพผักและผลไม้จำแนกได้ 131 ชนิดจำนวน 90,483 ภาพ เป็นภาพถ่ายผลไม้ขณะที่กำลังหมุนภาพและมีพื้นหลังเป็นสีขาว ภาพมีขนาด 100 x 100 พิกเซล ซึ่งในโครงการนี้จะนำภาพ ทั้ง 3 ชนิดคือ แอปเปิ้ล กล้วยและมะเขือเทศ โดยจะนำภาพมาเพียงบางส่วน

โดยทำการโหลดข้อมูลภาพจากเว็บไซต์ Kaggle แล้วทำการแยกเป็น 3 ชนิด (class) คือ แอปเปิ้ล กล้วย และมะเขือเทศ โดยแบ่งเป็นส่วนสำหรับฝึกฝนโมเดลหรือชุดเทรน (Training Set) จำนวน 4,290 ภาพ ซึ่งแบ่งเป็น 3 ประเภท ประเภทละ 1,430 ภาพเท่ากัน และชุดสำหรับทดสอบ (Test Set) จำนวน 1,200 ภาพ แบ่งเป็น 3 ประเภท ประเภทละ 400 ภาพเท่ากัน



ภาพที่ 3 ตัวอย่างภาพ แอปเปิ้ล กล้วย และมะเขือเทศจาก ชุดข้อมูลภาพผลไม้ 360 (Fruit360)

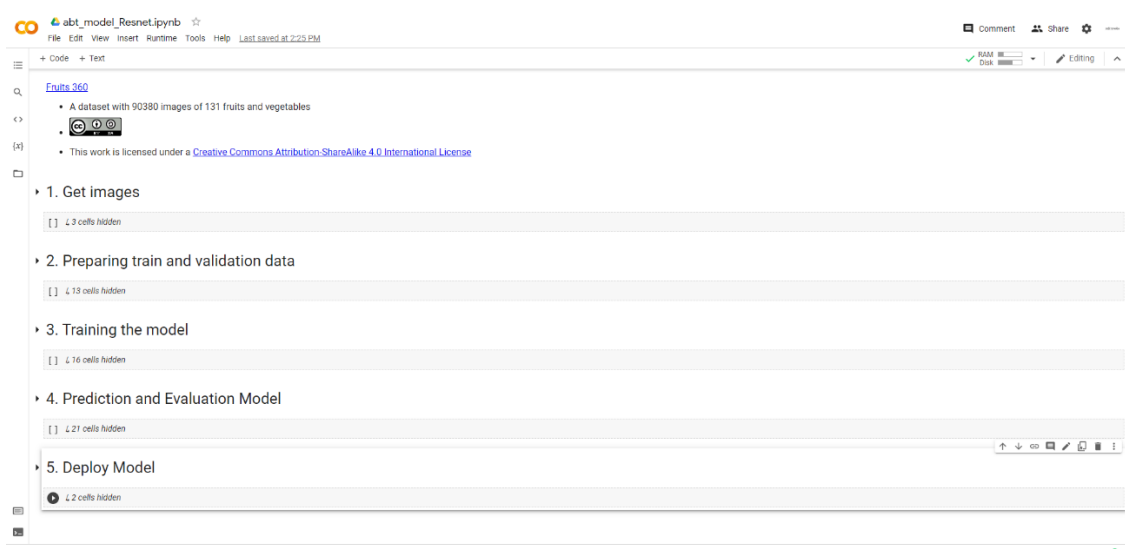
ซึ่งหลังแบ่งชุดข้อมูลเสร็จแล้ว คณะผู้จัดทำได้ทำการอัปโหลดขึ้นไปไว้บน GitHub Repository เพื่อให้ง่ายต่อการนำรูปภาพเข้าสู่ Jupyter Notebook บน Google Colaboratory

.git	11/11/2564 14:15	File folder	
test	9/11/2564 19:50	File folder	
train	9/11/2564 19:50	File folder	
LICENSE.md	9/11/2564 19:50	Markdown Source ...	21 KB
MIT License.md	9/11/2564 19:50	Markdown Source ...	2 KB
README.md	9/11/2564 19:50	Markdown Source ...	1 KB

ภาพที่ 4 ภาพแสดงไฟล์เดอร์ของรูปภาพ

3.2 การออกแบบหน้าจอ

ในโครงการนี้ เป็นการพัฒนาแบบจำลองการคัดแยกผลไม้ทั้ง 3 ชนิด ได้ดำเนินการตามกระบวนการการสร้างแบบจำลอง แบ่งเป็น 5 ขั้นตอน ซึ่งมีขั้นตอนดังนี้ คือ การได้มาซึ่งรูปภาพ (Get Images) กระบวนการแยกภาพเป็นชุดฝึกฝน (Training Data) และชุดตรวจสอบ (Validation data) การฝึกฝนแบบจำลอง (Training Model) การทำนายและประเมินผล (Prediction and Evaluation Model) การนำแบบจำลองไปปรับใช้ (Deploy Model) โดยการทำโครงการนี้จะใช้ Google Colaboratory ซึ่งคือ Jupyter notebook ดัดแปลงที่รันอยู่บนคลาวด์ และไม่จำเป็นต้องติดตั้งโปรแกรมใด ๆ ก่อนใช้งาน โดยสามารถใช้งานได้เพียงแค่มียูเอชไอ Google Drive เพื่อใช้ในการจัดเก็บตัวโค้ด โดยมีภาษา Python เป็นภาษาหลักที่ใช้ในการเขียนและรันงานบนเว็บ Colab ใช้ในการเขียนโปรแกรมเพื่อทำการเตรียมภาพต่างๆ ไปจนถึงการฝึกฝนแบบจำลอง และทดสอบแบบจำลอง



ภาพที่ 5 ภาพ Notebook Jupyter บนแพลตฟอร์ม Google Colaboratory

3.3 การเขียนโปรแกรมควบคุมการทำงาน

3.3.1 การ import library มีดังนี้

```

1 # Preparing train and validation data - section
2 import tensorflow as tf
3 # Trainig The model - section
4 from tensorflow.keras.applications import ResNet50
5 from keras import Model
6 from keras.layers import Dense
7 # Trainig Graph - section
8 import matplotlib.pyplot as plt
9 # classification-report - section
10 import numpy as np
11 from imageio import imread
12 from google.colab.patches import cv2_imshow
13 from skimage.transform import resize
14 from tensorflow.keras.applications.resnet50 import preprocess_input
15 from sklearn.metrics import classification_report
16 from random import randint

```

ภาพที่ 6 การนำเข้า library

ในส่วนแรกนี้เป็นการนำเข้า library ที่จำเป็นในการสร้างแบบจำลอง ในบรรทัดที่ 2 import tensorflow as tf เป็นส่วนสำหรับนำเข้าไลบรารี tensorflow และตั้งชื่อเล่นว่า tf เพื่อให้เรียกใช้งานได้ง่าย ในส่วนนี้นำเข้าเพื่อนำมาใช้ในการแยกส่วนของข้อมูลภาพออกเป็นสองชุด คือ ชุดสำหรับฝึกฝน (Training set) และชุดสำหรับตรวจสอบ (Validation set) ในขณะฝึกฝนแบบจำลอง

บรรทัดที่ 4 – 6 เป็นการนำเข้าไลบรารีตระกูล keras ทำการนำเข้าแบบจำลอง ResNet50 library Model และ Dense เพื่อนำมาใช้ในการปรับเปลี่ยนแบบจำลองเพื่อให้เหมาะสมกับงาน

บรรทัดที่ 8 เป็นการนำเข้าไลบรารี matplotlib.pyplot ตั้งชื่อเล่นว่า plt ใช้สำหรับการสร้างกราฟ เพื่อแสดงค่าความถูกต้องและค่าความสูญเสียหรือผิดพลาด ของแบบจำลองหลังการฝึกฝน (Train)

บรรทัดที่ 10 เป็นการนำเข้าไลบรารี numpy ตั้งชื่อเล่นว่า np ใช้ในการจัดการข้อมูลประเภทอาร์เรย์ที่ได้จากการอ่านภาพเข้าสู่ตัวแปร อีกทั้งยังสามารถจัดการข้อมูลประเภทคณิตศาสตร์ได้อย่างดี

บรรทัดที่ 11 เป็นการนำเข้าไลบรารี imageio โดยนำเข้า imread เพื่อนำมาใช้ในการอ่านภาพ

บรรทัดที่ 12 เป็นการนำเข้าไลบรารี cv2_imshow นำเข้ามาเพื่อใช้ในการแสดงผลภาพที่ถูกอ่านเข้ามาแล้วจากการใช้ไลบรารี imread ซึ่งใช้ในแพลตฟอร์ม Google Colaboratory

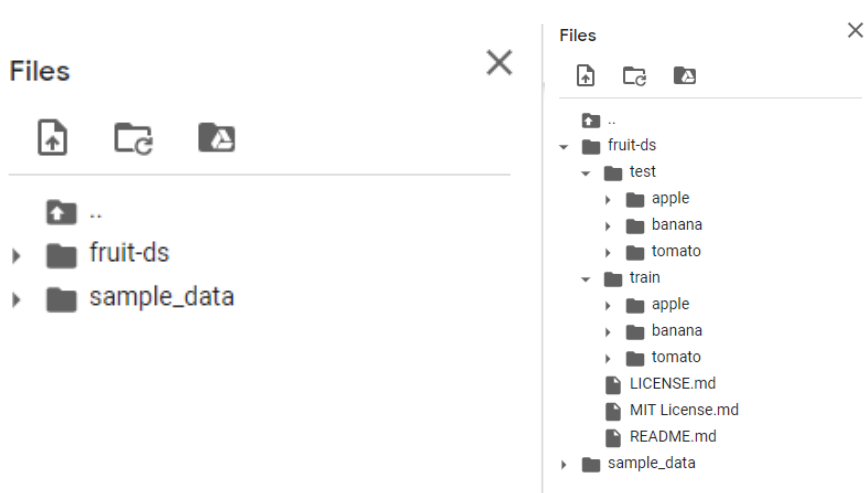
บรรทัดที่ 13 - 4 เป็นการนำเข้าไลบรารี resize จาก sklearn และ preprocess_input จาก tensorflow ที่ใช้ในการเตรียมรูปภาพในส่วนก่อนการประมวลผล (preprocessing) โดย resize ใช้สำหรับการปรับขนาดรูปภาพเพื่อให้เหมาะสมกับแบบจำลองและ preprocess_input ใช้ในการแปลงรูปภาพ จาก RGB เป็น BGR เพื่อเตรียมภาพสำหรับใช้เป็นพารามิเตอร์สำหรับการทำนายของแบบจำลอง

3.3.2 ขั้นตอนการนำเข้าภาพ (Get Images) เข้ามาใน Google Colaboratory

```
1 !git clone https://github.com/lacakp/fruit-ds.git
```

```
Cloning into 'fruit-ds'...
remote: Enumerating objects: 17153, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 17153 (delta 1), reused 6 (delta 1), pack-reused 17133
Receiving objects: 100% (17153/17153), 376.56 MiB | 33.47 MiB/s, done.
Resolving deltas: 100% (50/50), done.
```

ภาพที่ 7 คำสั่ง git clone



ภาพที่ 8 โฟลเดอร์รูปภาพหลังจากที่ใช้ คำสั่ง git clone

เมื่อทำการโคลนเสร็จสิ้นจะได้ โฟลเดอร์ fruit-ds และมีโฟลเดอร์ย่อยๆ โดยแบ่งเป็นชุด train และ test ดังภาพ

ใช้คำสั่ง git clone ในการโคลนรูปภาพจาก GitHub Repository เป็นภาพที่แยกออกจาก Fruit 360 เพื่อให้ง่ายต่อการดำเนินการสร้างแบบจำลอง

```
[143] 1 print("All train image in 3 class apple, banana, tomato")
2 !ls /content/fruit-ds/train/*/* | wc -l
3
4 print("\n-----")
5
6 print("Apple image count")
7 !ls /content/fruit-ds/train/apple/* | wc -l
8
9 print("Banana Image Count")
10 !ls /content/fruit-ds/train/banana/* | wc -l
11
12 print("Tomato Image Count")
13 !ls /content/fruit-ds/train/tomato/* | wc -l
```

ภาพที่ 9 คำสั่งในการเช็คจำนวนไฟล์ในโฟลเดอร์ train

หลังจากทำการโคลนรูปภาพมาแล้ว ได้ทำการเขียนสคริปต์ของระบบปฏิบัติการลินุกซ์ ซึ่ง Google Colaboratory มีพื้นฐานอยู่บนลินุกซ์อยู่แล้ว เพื่อทำการเช็คข้อมูลรูปภาพที่มีในโฟลเดอร์ train ซึ่งเป็นชุดภาพสำหรับการฝึกฝนแบบจำลอง

3.3.3 ขั้นตอนการแยกภาพเป็นชุดฝึกฝน (Training Data) และชุดตรวจสอบ (Validation data) มีดังนี้

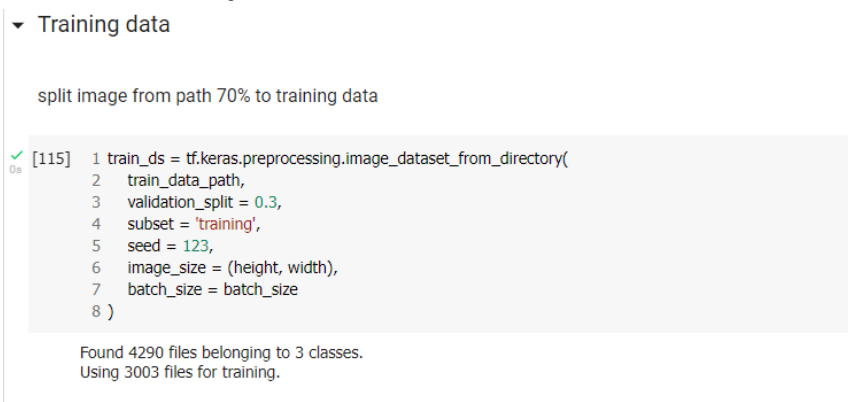
3.3.3.1 ในส่วนการตั้งค่าตัวแปรมีดังนี้

```
1 train_data_path = '/content/fruit-ds/train'
2 height = 224
3 width = 224
4 batch_size = 32
```

ภาพที่ 10 การตั้งค่าตัวแปรขนาดของรูปภาพและที่อยู่ของโฟลเดอร์ train

ในบรรทัดที่ 1 เป็นการประกาศตัวแปรสำหรับ เก็บค่าที่อยู่ของโฟลเดอร์ train ที่เป็นโฟลเดอร์ชุดข้อมูลฝึกฝน (training data) ในบรรทัดที่ 2 – 3 เป็นการตั้งตัวแปรเพื่อเก็บขนาดของรูปภาพซึ่งมีขนาด 224x224 และในบรรทัดที่ 4 เป็นตัวแปรเก็บขนาดของ batch_size โดยมีค่า 32 ซึ่งก็คือจำนวนรายการข้อมูลที่จะให้ ตัวเพิ่มประสิทธิภาพ (Optimizer) คำนวณในหนึ่งครั้ง เช่น ข้อมูลมี 3,040 รายการ ถ้ากำหนด Batch size เป็น 32 แปลว่า ตัวเพิ่มประสิทธิภาพ (Optimizer) จะต้องทำงาน 95 ครั้ง จึงจะครบทั้ง 3,040 รายการ และถ้ากำหนด Epoch เป็น 10 ก็หมายถึงการทำงาน 95 ครั้ง 10 รอบ โดย Epoch นั้นจะกล่าวในส่วนของการฝึกฝนแบบจำลองในส่วนถัดไป

3.3.3.2 ในส่วนการแบ่งชุดข้อมูลสำหรับการฝึกฝนแบบจำลองมีดังนี้



ภาพที่ 11 คำสั่งที่ใช้ในการแบ่งข้อมูลเป็นชุดฝึกฝน

ส่วนแรกสำหรับการแบ่งชุดข้อมูลคือ ในส่วนของการแบ่งชุดข้อมูลที่เป็นชุดฝึกฝน (Training data) โดยทำการเรียกใช้ไลบรารี tensorflow ที่นำเข้ามาแล้ว เพื่อเรียกใช้ keras.preprocessing.image_dataset_from_directory เพื่อทำการแบ่งชุดข้อมูลเป็นชุดฝึกฝนแบบอัตโนมัติ จากที่อยู่ของโฟลเดอร์มีค่าพารามิเตอร์ดังนี้

- Train_data_path คือตัวแปรที่เป็นที่อยู่ของโฟลเดอร์ train
- Validation_split 0.3 คือแบ่งอัตราส่วน 70% เป็นภาพฝึกฝนและ 30% เป็นภาพตรวจสอบ
- subset = 'training' หมายถึงเป็นส่วนของภาพชุดฝึกฝน (Training data)

- seed = 123 หมายถึงพารามิเตอร์สำหรับการสุ่มทางเลือกสำหรับการสับเปลี่ยนและการแปลงข้อมูลภาพ
- image_size = (height, width) เป็นการกำหนดขนาดภาพตามที่ได้ตั้งค่าตัวแปรไว้แล้ว
- batch_size เป็นจำนวนรายการข้อมูลที่จะให้ ตัวเพิ่มประสิทธิภาพคำนวณในหนึ่งครั้ง

▼ Validation data

```
[116] 1 val_ds = tf.keras.preprocessing.image_dataset_from_directory(
      2     train_data_path,
      3     validation_split = 0.3,
      4     subset = 'validation',
      5     seed = 123,
      6     image_size = (height, width),
      7     batch_size = batch_size
      8 )
```

Found 4290 files belonging to 3 classes.
Using 1287 files for validation.

ภาพที่ 12 คำสั่งในการแบ่งภาพเป็นชุดตรวจสอบในการฝึกฝนแบบจำลอง

เป็นการทำงานคล้ายกันกับส่วนของ การแบ่งชุดข้อมูลฝึกฝน แต่ในส่วนนี้เป็นการแบ่งชุดข้อมูลสำหรับตรวจสอบ โดยค่าในส่วนของ subset จะเปลี่ยนเป็น 'validation' แต่ในส่วนอื่นๆจะยังคงเดิม ซึ่งแบ่งเป็น 30% จากรูปภาพทั้งหมด 4290 ภาพ ซึ่งใช้ 1,287 ภาพสำหรับชุดตรวจสอบ ที่ใช้สำหรับการฝึกฝนแบบจำลอง ซึ่งจะกล่าวในส่วนถัดไป

▼ Get Class

```
[117] 1 classes = train_ds.class_names
      2 print(classes)
      3 print(train_ds)
```

['apple', 'banana', 'tomato']
<BatchDataset shapes: ((None, 224, 224, 3), (None,)), types: (tf.float32, tf.int32)>

ภาพที่ 13 คำสั่งในการตรวจสอบคลาสของรูปภาพที่ได้จากการแบ่งชุดข้อมูลแบบอัตโนมัติ

เป็นการตรวจสอบค่าของป้าย (Labels) แต่ละชนิดของรูปภาพที่ได้จากการใช้ไลบรารี keras สำหรับแบ่งชุดข้อมูลแบบอัตโนมัติจากโฟลเดอร์ train ซึ่งมี 3 ชนิด (class) ผลลัพธ์ตามภาพ

3.3.4 ขั้นตอนการฝึกฝนแบบจำลอง (Training Model)

3.3.4.1 ในส่วนการเตรียมแบบจำลองก่อนการฝึกฝนแบบจำลองมีดังนี้

```
1 model = ResNet50(weights='imagenet')

Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_kernels.h5
102973440/102967424 [=====] - 1s 0us/step
102981632/102967424 [=====] - 1s 0us/step
```

ภาพที่ 14 คำสั่งการดาวน์โหลดแบบจำลอง ResNet50

ดาวน์โหลดแบบจำลอง ResNet50 และกำหนดน้ำหนัก (Weight) เป็น Imagenet แทนไว้ในตัวแปร model

```
✓ [120] 1 model.summary()
```

ภาพที่ 15 คำสั่งในการตรวจสอบคุณลักษณะของแบบจำลอง

เป็นการรันเพื่อดูรายละเอียดของแบบจำลองว่าผ่านการฝึกฝนมาอย่างไรบ้างและมีกี่เซลล์ประสาท

```
1 model.summary()

conv5_block2_1_conv (Conv2D) (None, 7, 7, 512) 1049088 ['conv5_block2_1_conv[0][0]']
conv5_block2_1_bn (BatchNormal (None, 7, 7, 512) 2048 ['conv5_block2_1_conv[0][0]']
activation)
conv5_block2_1_relu (Activation (None, 7, 7, 512) 0 ['conv5_block2_1_bn[0][0]']
n)
conv5_block2_2_conv (Conv2D) (None, 7, 7, 512) 2329808 ['conv5_block2_1_relu[0][0]']
conv5_block2_2_bn (BatchNormal (None, 7, 7, 512) 2048 ['conv5_block2_2_conv[0][0]']
activation)
conv5_block2_2_relu (Activation (None, 7, 7, 512) 0 ['conv5_block2_2_bn[0][0]']
n)
conv5_block2_3_conv (Conv2D) (None, 7, 7, 2048) 1050624 ['conv5_block2_2_relu[0][0]']
conv5_block2_3_bn (BatchNormal (None, 7, 7, 2048) 8192 ['conv5_block2_3_conv[0][0]']
activation)
conv5_block2_add (Add) (None, 7, 7, 2048) 0 ['conv5_block2_3_conv[0][0]',
conv5_block2_1_bn[0][0]]
conv5_block2_out (Activation (None, 7, 7, 2048) 0 ['conv5_block2_add[0][0]']
n)
conv5_block3_1_conv (Conv2D) (None, 7, 7, 512) 1049088 ['conv5_block2_out[0][0]']
conv5_block3_1_bn (BatchNormal (None, 7, 7, 512) 2048 ['conv5_block3_1_conv[0][0]']
activation)
conv5_block3_1_relu (Activation (None, 7, 7, 512) 0 ['conv5_block3_1_bn[0][0]']
n)
conv5_block3_2_conv (Conv2D) (None, 7, 7, 512) 2329808 ['conv5_block3_1_relu[0][0]']
conv5_block3_2_bn (BatchNormal (None, 7, 7, 512) 2048 ['conv5_block3_2_conv[0][0]']
activation)
conv5_block3_2_relu (Activation (None, 7, 7, 512) 0 ['conv5_block3_2_bn[0][0]']
n)
conv5_block3_3_conv (Conv2D) (None, 7, 7, 2048) 1050624 ['conv5_block3_2_relu[0][0]']
conv5_block3_3_bn (BatchNormal (None, 7, 7, 2048) 8192 ['conv5_block3_3_conv[0][0]']
activation)
conv5_block3_add (Add) (None, 7, 7, 2048) 0 ['conv5_block3_3_conv[0][0]',
conv5_block3_1_bn[0][0]]
conv5_block3_out (Activation (None, 7, 7, 2048) 0 ['conv5_block3_add[0][0]']
n)
avg_pool (GlobalAveragePooling (None, 2048) 0 ['conv5_block3_out[0][0]']
2D)
predictions (Dense) (None, 1000) 2049000 ['avg_pool[0][0]']

=====
Total params: 25,636,712
Trainable params: 25,583,592
Non-trainable params: 53,120

=====
predictions (Dense) (None, 1000) 2049000 ['avg_pool[0][0]']

=====
Total params: 25,636,712
Trainable params: 25,583,592
Non-trainable params: 53,120
=====
```

ภาพที่ 16 ผลลัพธ์จากคำสั่ง summary

แบบจำลอง ResNet50 เดิมถูกฝึกฝนมาด้วยข้อมูลจำนวน 1,000 ชนิดหรือ 1,000 เซลล์ประสาทนั่นเอง หลังจากนั้นจะทำการปรับเปลี่ยนแบบจำลองนี้เพื่อให้เหมาะสมกับงานของโครงงานนี้

▼ Custom model

```
✓ [122] 1 abt_output = Dense(3, activation='softmax')
        2 abt_output = abt_output(model.layers[-2].output)
        3
        4 abt_input = model.input
        5 abt_model = Model(inputs=abt_input, outputs=abt_output)
        6
        7 for layer in abt_model.layers[:-1]:
        8     layer.trainable = False
```

ภาพที่ 17 ส่วนของคำสั่งในการปรับเปลี่ยนแบบจำลอง

ส่วนนี้จะเป็นการปรับเปลี่ยนแบบจำลอง

ในบรรทัดที่ 1 แทนตัวแปร abt_output โดยใช้ ไลบรารี Dense ที่นำเข้ามาทำการสร้างเซลล์ประสาทไว้ 3 เซลล์ประสาท

ในบรรทัดที่ 2 ทำการนำเซลล์ประสาทในบรรทัดที่ 1 ไปแทนที่ เซลล์ประสาทเดิมของแบบจำลอง ResNet50

ในบรรทัดที่ 4 - 5 ทำการ ใช้ไลบรารี Model ที่นำเข้ามาทำการสร้างแบบจำลองใหม่ขึ้นมา โดยมีพารามิเตอร์ 2 ตัวคือ inputs ให้แทนค่าอินพุตจากตัวแบบจำลองเดิม ในบรรทัดที่ 4 และ เอาต์พุตที่สร้างขึ้นใหม่ แทนค่าเป็นตัวแปร abt_model

ในบรรทัดที่ 7-8 เป็นการปิดการฝึกฝนในชั้นอื่นๆ ของแบบจำลองเดิม เพื่อให้การฝึกฝนแบบจำลองใหม่มีความรวดเร็วและวัดผลได้ง่าย

▼ Compile model

```
✓ [123] 1 abt_model.compile(
        2     loss='sparse_categorical_crossentropy',
        3     optimizer='adam',
        4     metrics=['accuracy']
        5 )
```

ภาพที่ 18 การเรียกใช้ เมธอด compile

ทำการคอมไพล์แบบจำลองใหม่ที่ทำเตรียมไว้จากขั้นตอนก่อนหน้านี้ โดยเรียกใช้เมธอด compile มีพารามิเตอร์ 3 ตัวได้แก่ loss optimizer และ metrics accuracy

โดย loss คือฟังก์ชันการสูญเสีย ซึ่งเป็นอัลกอริทึมที่กำหนดว่าการคาดการณ์หรือการทำนายของโมเดลผิดพลาดเพียงใด เป้าหมายของการฝึกฝนแบบจำลอง (Training) คือการลดความสูญเสีย (loss) จะใช้ sparse categorical crossentropy ซึ่งถือว่า ถ้าค่าป้าย (Label) เป็น 0 หมายความว่า เซลล์ประสาทแรกในชั้นเอาต์พุต ทำงานมากที่สุด เช่นกันหากการทำงานที่ค่าป้าย (Label) 1 หมายความว่าเซลล์ประสาทที่สองทำงานมากที่สุด และป้าย (Label) ที่ 2 แสดงว่าเซลล์ประสาทที่สามทำงานมากที่สุดนั่นเอง

ตัวเพิ่มประสิทธิภาพ (Optimizer) จะระบุพารามิเตอร์ที่เป็นอัลกอริทึมที่จะใช้ในการปรับน้ำหนักในแบบจำลอง (Model) ซึ่งในโครงงานนี้จะใช้ Adam

Accuracy คือความแม่นยำของแบบจำลองว่า ทำนายได้แม่นยำเพียงใด (ในส่วนนี้เป็นการแสดงผลภาพรวม ซึ่งไม่ได้ส่งผลใดๆ ในการฝึกฝนแบบจำลอง)

```
dense_1 (Dense)          (None, 3)          6147      ['avg_pool[0][0]']
```

```
=====
Total params: 23,593,859
Trainable params: 6,147
Non-trainable params: 23,587,712
```

ภาพที่ 19 ผลลัพธ์การคอมไพล์แบบจำลองหลังจากการปรับเปลี่ยน

หลังจากที่ทำการคอมไพล์ด้วยเมธอด compile แล้วทำการ เรียกใช้ model.summary() อีกครั้งจะได้ผลลัพธ์ตามภาพคือ เซลล์ประสาทของแบบจำลองถูกแทนที่ด้วย เซลล์ประสาทใหม่ 3 เซลล์

3.3.4.2 ในส่วนการฝึกฝนแบบจำลองมีดังนี้

```
[125] 1 ABTFruit = abt_model.fit(train_ds,
2           validation_data = val_ds,
3           epochs=4
4           )
```

ภาพที่ 20 คำสั่งในการฝึกฝนแบบจำลอง

ทำการเรียกใช้ เมธอด fit เพื่อทำการฝึกฝนแบบจำลองซึ่งระหว่างการฝึก โมเดลจะถูกปรับเพื่อลดค่า loss ให้เหลือน้อยที่สุดโดยมี พารามิเตอร์ดังนี้คือ

- train_ds คือชุดข้อมูลสำหรับฝึกฝนแบบจำลอง
- validation_data คือชุดข้อมูลที่ใช้สำหรับตรวจสอบแบบจำลองขณะทำการฝึกฝน
- epochs epochs คือ การระบุจำนวนครั้งที่ทำการฝึกฝนแบบจำลอง ในโครงงานนี้ได้ระบุ 4 epoch หมายถึงทำการเรียกใช้เมธอด fit() 4 ครั้งติดต่อกันด้วยชุดข้อมูลเดียวกัน

3.3.5 ขั้นตอนการทำนายและประเมินผล (Prediction and Evaluation Model)

3.3.5.1 ในส่วนการสรุปผลของการฝึกฝนจำลองมีดังนี้

```
1 # Model Accuracy
2 plt.plot(ABTFruit.history['accuracy'])
3 plt.plot(ABTFruit.history['val_accuracy'])
4 plt.axis(ymin=0.8, ymax=1)
5 plt.grid()
6 plt.title('Model Accuracy')
7 plt.ylabel('Accuracy')
8 plt.xlabel('Epochs')
9 plt.legend(['train', 'validation'])
10 plt.show()
```

ภาพที่ 21 ชุดคำสั่งการสร้างกราฟ เพื่อสรุปค่าความถูกต้อง (Accuracy)

ในบรรทัดที่ 2 – 3 เป็นการดึงค่า accuracy และ val_accuracy จากแบบจำลองด้วยเมธอด history เพื่อทำการพล็อตกราฟ และในบรรทัดที่ 4 - 5 เป็นการกำหนดช่วงของค่าแกน y และใช้เส้นกริดของกราฟ ในบรรทัดที่ 6-8 เป็นการตั้งชื่อหัวข้อของกราฟเป็น Model Accuracy และกำหนดชื่อป้าย (Label) ในแกน x และแกน y ในบรรทัดที่ 9 เป็นการแสดงคำอธิบายเส้นของกราฟ และบรรทัดสุดท้าย บรรทัดที่ 10 คือการแสดงกราฟ

```
[128] 1 # Model Loss
2 plt.plot(ABTFruit.history['loss'])
3 plt.plot(ABTFruit.history['val_loss'])
4 plt.grid()
5 plt.title('Model Loss')
6 plt.ylabel('Loss')
7 plt.xlabel('Epochs')
8 plt.legend(['train', 'validation'])
9 plt.show()
```

ภาพที่ 22 ชุดคำสั่งการสร้างกราฟ เพื่อสรุปค่าความสูญเสีย (Loss)

ในส่วนนี้เป็นการสร้างกราฟ เพื่อสรุปค่าความสูญเสีย (Loss) ของแบบจำลองโดยในบรรทัดที่ 2 – 3 เป็นการดึงค่า loss และ val_loss จากแบบจำลองด้วยเมธอด history เพื่อทำการพล็อตกราฟ และในบรรทัดที่ 4 เป็นการใช้เส้นกริดของกราฟ ในบรรทัดที่ 5-7 เป็นการตั้งชื่อหัวข้อของกราฟเป็น Model Loss และกำหนดชื่อป้าย (Label) ในแกน x และแกน y ในบรรทัดที่ 8 เป็นการแสดงคำอธิบายเส้นของกราฟ และบรรทัดสุดท้าย บรรทัดที่ 10 คือการแสดงกราฟ

3.3.5.2 ในส่วนการนำแบบจำลองมาทดสอบทำนาย (Prediction) มีดังนี้

```
▼ Predictions

[133] 1 test_data_path = '/content/fruit-ds/test'
2 batch_size=32

[158] 1 print("All test image in 3 class apple, banana, tomato")
2 !ls /content/fruit-ds/test/* | wc -l
3
4 print("\n-----")
5
6 print("Apple image count")
7 !ls /content/fruit-ds/test/apple/* | wc -l
8
9
10 print("Banana Image Count")
11 !ls /content/fruit-ds/test/banana/* | wc -l
12
13
14 print("Tomato Image Count")
15 !ls /content/fruit-ds/test/tomato/* | wc -l

All test image in 3 class apple, banana, tomato
1200

-----
Apple image count
400
Banana Image Count
400
Tomato Image Count
400
```

ภาพที่ 23 คำสั่งในการเช็คจำนวนไฟล์ในโฟลเดอร์ test

```

1 def abt_predict(image):
2     """
3     ฟังก์ชันสำหรับการทำนายภาพผลไม้ แอปเปิ้ล กล้วย และมะเขือเทศ
4     Prediction Function
5     args: image (<class 'numpy.ndarray'>)
6     """
7     resized = cv2.resize(image, (224, 224))
8     img_array = tf.keras.preprocessing.image.img_to_array(resized)
9     img_array = tf.expand_dims(img_array, 0)
10    pred = abt_model.predict(img_array)
11    score = pred[0]
12
13    result = np.argmax(score)
14    if result == 0:
15        print('prediction: this is apple')
16    if result == 1:
17        print('prediction: this is banana')
18    if result == 2:
19        print("prediction: this is tomato")
20
21    print("prediction score: Apple {:.2f} %".format(score[0]*100))
22    print("prediction score: Banana {:.2f} %".format(score[1]*100))
23    print("prediction score: Tomato {:.2f} %".format(score[2]*100))
24    print("-----")
25

```

ภาพที่ 24 ฟังก์ชันสำหรับเรียกใช้แบบจำลองในการทำนายภาพ

เป็นฟังก์ชันสำหรับการเรียกใช้แบบจำลองเพื่อการทำนายรูปภาพ โดยมีชื่อฟังก์ชันว่า `abt_predict` รับอาร์กิวเมนต์ 1 ตัวคือ `image` ซึ่งเป็นรูปภาพที่ถูกอ่านมาแล้ว ในบรรทัดที่ 2-5 เป็น `document` ของฟังก์ชัน ในบรรทัดที่ 7 – 9 เป็นการเตรียมภาพที่รับเข้ามาเพื่อใช้เป็นพารามิเตอร์ในการทำนายของแบบจำลองในบรรทัดที่ 10 ในบรรทัดที่ 11 เป็นการดึงข้อมูลประเภท `array` ที่ซ่อนกันอยู่จากผลลัพธ์ที่ได้จากการทำนายของแบบจำลอง ในบรรทัดที่ 13 เป็นการหาค่าสูงสุดที่แบบจำลองทำนายได้ หมายถึงเซลล์ประสาทที่เอาต์พุตทำงานมากที่สุดนั่นเอง และในบรรทัดที่ 14 – 24 เป็นการแสดงผลลัพธ์ของการทำนาย และแสดงผลเป็นค่าเปอร์เซ็นต์ ที่ได้จากการทำนายของแบบจำลองในแต่ละคลาส

```

1 # Apple
2 for i in range(5):
3     im = cv2.imread(f'{test_data_path}/apple/apple ({randint(1,400)}).jpg')
4     print(f"apple figure({i+1})")
5     cv2.imshow(cv2.resize(im, (30, 30)))
6     abt_predict(im)
7

```

ภาพที่ 25 คำสั่งในการสุ่มภาพแอปเปิ้ลจำนวน 5 ภาพจากโฟลเดอร์ `test`

เป็นการสุ่มภาพแอปเปิ้ล และทำการอ่านภาพจากโฟลเดอร์ `test` 5 ครั้ง และทำการเรียกใช้ฟังก์ชัน `abt_predict` โดยมีพารามิเตอร์เป็นภาพที่อ่านมาในรอบนั้นๆ เพื่อการทำนาย และแสดงผลลัพธ์เป็นเปอร์เซ็นต์ความเป็นไปได้ในแต่ละคลาส

```

1 # Banana
2 for i in range(5):
3     im = cv2.imread(f'{test_data_path}/banana/banana ({randint(1,400)}).jpg')
4     print(f"banana figure({i+1})")
5     cv2.imshow(cv2.resize(im, (30, 30)))
6     abt_predict(im)

```

ภาพที่ 26 คำสั่งในการสุ่มภาพกล้วยจำนวน 5 ภาพจากโฟลเดอร์ test

เป็นการสุ่มภาพกล้วย และทำการอ่านภาพจากโฟลเดอร์ test 5 ครั้ง และทำการเรียกใช้ฟังก์ชัน abt_predict โดยมีพารามิเตอร์เป็นภาพที่อ่านมาในรอบนั้นๆ เพื่อทำการทำนาย และแสดงผลเป็นเปอร์เซ็นต์ความเป็นไปได้ในแต่ละคลาส

```

1 # Tomato
2 for i in range(5):
3     im = cv2.imread(f'{test_data_path}/tomato/tomato ({randint(1,400)}).jpg')
4     print(f"tomato figure({i+1})")
5     cv2.imshow(cv2.resize(im, (30, 30)))
6     abt_predict(im)

```

ภาพที่ 27 คำสั่งในการสุ่มภาพมะเขือเทศจำนวน 5 ภาพจากโฟลเดอร์ test

เป็นการสุ่มภาพมะเขือเทศ และทำการอ่านภาพจากโฟลเดอร์ test 5 ครั้ง และทำการเรียกใช้ฟังก์ชัน abt_predict โดยมีพารามิเตอร์เป็นภาพที่อ่านมาในรอบนั้นๆ เพื่อทำการทำนาย และแสดงผลเป็นเปอร์เซ็นต์ความเป็นไปได้ในแต่ละคลาส

3.3.5.3 ในส่วนการประเมินผล (Evaluation) ของแบบจำลองมีดังนี้

```

✓ [136] 1 data = np.empty((1200, 224, 224, 3))

Apples Images from test folder 400 image

✓ [137] 1 for i in range(400):
2     im = imread('/content/fruit-ds/test/apple/apple ({i}).jpg'.format(i+1))
3     im = preprocess_input(im)
4     im = resize(im, output_shape=(224, 224))
5     data[i] = im

Bananas Images from test folder 400 image

✓ [138] 1 for i in range(400):
2     im = imread('/content/fruit-ds/test/banana/banana ({i}).jpg'.format(i+1))
3     im = preprocess_input(im)
4     im = resize(im, output_shape=(224, 224))
5     data[i + 400] = im

tomatoes Images from test folder 400 image

✓ [139] 1 for i in range(400):
2     im = imread('/content/fruit-ds/test/tomato/tomato ({i}).jpg'.format(i+1))
3     im = preprocess_input(im)
4     im = resize(im, output_shape=(224, 224))
5     data[i + 800] = im

```

ภาพที่ 28 คำสั่งในการอ่านภาพเข้าสู่ระบบเพื่อใช้ในการประเมินผลแบบจำลอง

ทำการประกาศตัวแปร data เก็บค่าอาร์เรย์ว่างความจุสำหรับเก็บภาพ จำนวน 1200 ช่อง ขนาดภาพ 224x224 อ่านภาพจากโฟลเดอร์ test ในแต่ละชนิดจำนวน 400 ภาพ 3 ชนิดทั้งหมด 1200 ภาพ เก็บไว้ในตัวแปร data

```

1 # Generating labels
2 labels = np.empty(1200, dtype=int)
3 labels[:400] = 0
4 labels[400:800] = 1
5 labels[800:] = 2
6
7 classes = ['apple', 'banana', 'tomato']

```

ภาพที่ 29 คำสั่งในการกำหนดค่าจริงของภาพ เพื่อใช้ในการประเมินผลแบบจำลอง

ในช่วงตำแหน่งที่ 0:400 ในตำแหน่งที่ 0 คือภาพแอปเปิ้ล 400:800 ในตำแหน่งที่ 1 ภาพของกล้วย และช่วง 800:1200 ในตำแหน่งที่ 2 เป็นภาพมะเขือเทศ

```

1 y_pred = abt_model.predict(data, batch_size=32, verbose=1)
2 y_pred_bool = np.argmax(y_pred, axis=1)
3
4 print(classification_report(labels, y_pred_bool, target_names=classes))

```

ภาพที่ 30 คำสั่งการเรียกใช้ไลบรารี `classification_report` เพื่อประเมินผลแบบจำลอง

ในบรรทัดที่ 1 `y_pred` เรียกใช้แบบจำลองเพื่อทำนายด้วยชุดข้อมูลทดสอบ (Test data) โดยมีพารามิเตอร์เป็นตัวแปร `data` ซึ่งคือภาพทั้ง 3 ชนิด 1,200 ภาพ ให้ `batch_size = 32`, `verbose=1` หมายถึงจะแสดงแถบความคืบหน้า ในบรรทัดที่ 2 คือการแปลงผลที่ได้จากการทำนายเป็นตัวเลขจำนวนเต็มที่เป็นค่า label 0 1 และ 2 ในบรรทัดที่ 3 ทำการใช้ `classification_report` ทำการวัดผล โดยมีพารามิเตอร์เป็น `labels` ซึ่งหมายถึงชุดข้อมูลป้ายที่เป็นผลลัพธ์ และเปรียบเทียบกับ `y_pred_bool` ที่เป็นผลที่ได้จากการทำนายภาพจากชุดข้อมูลทดสอบ (Test data)

3.3.6 ขั้นตอนการการนำแบบจำลองไปปรับใช้ (Deploy Model)

```

1 abt_model.save('abt_resnetmodel', save_format='h5')

```

ภาพที่ 31 คำสั่งบันทึกแบบจำลองเพื่อนำไปใช้งาน

เป็นการบันทึกแบบจำลองโดยตั้งชื่อว่า `abt_resnetmodel` ซึ่งเป็นไฟล์ `keras model`

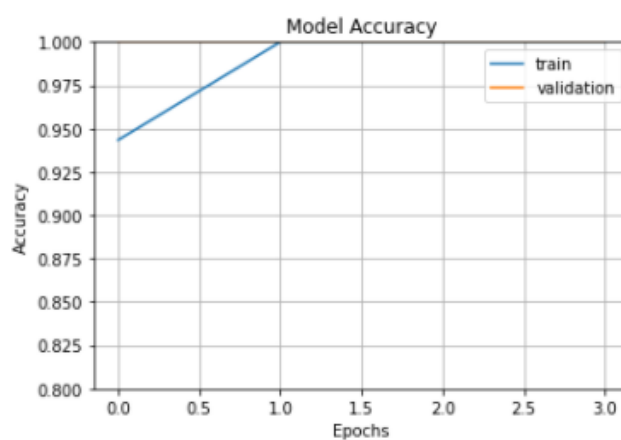
บทที่ 4 ผลการดำเนินงาน

4.1 ผลการทำงานของโปรแกรม

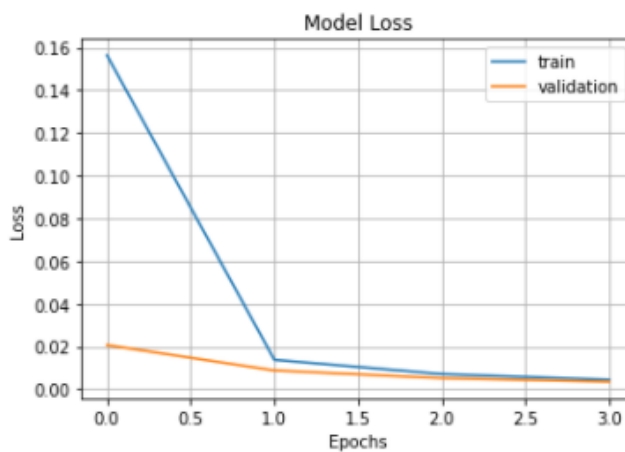
4.1.1 ผลลัพธ์การฝึกฝนแบบจำลอง

```
Epoch 1/4
94/94 [=====] - 37s 346ms/step - loss: 0.1564 - accuracy: 0.9434 - val_loss: 0.0207 - val_accuracy: 1.0000
Epoch 2/4
94/94 [=====] - 30s 312ms/step - loss: 0.0138 - accuracy: 1.0000 - val_loss: 0.0088 - val_accuracy: 1.0000
Epoch 3/4
94/94 [=====] - 30s 308ms/step - loss: 0.0071 - accuracy: 1.0000 - val_loss: 0.0052 - val_accuracy: 1.0000
Epoch 4/4
94/94 [=====] - 29s 308ms/step - loss: 0.0045 - accuracy: 1.0000 - val_loss: 0.0035 - val_accuracy: 1.0000
```

ภาพที่ 32 ผลลัพธ์การฝึกฝนแบบจำลอง


















ภาพที่ 33 กราฟแสดงแนวโน้มของค่าความถูกต้อง (Accuracy) ของการฝึกฝนแบบจำลอง



ภาพที่ 34 กราฟแสดงแนวโน้มของค่าความสูญเสีย (Loss) ของการฝึกฝนแบบจำลอง

4.1.2 ผลลัพธ์การทำนายของแบบจำลองจากชุดข้อมูลทดสอบ

 <p>prediction: this is apple prediction score: Apple 99.90 % prediction score: Banana 0.04 % prediction score: Tomato 0.06 %</p>	 <p>prediction: this is banana prediction score: Apple 0.00 % prediction score: Banana 100.00 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is tomato prediction score: Apple 5.11 % prediction score: Banana 0.39 % prediction score: Tomato 94.50 %</p>
 <p>prediction: this is apple prediction score: Apple 99.64 % prediction score: Banana 0.34 % prediction score: Tomato 0.02 %</p>	 <p>prediction: this is banana prediction score: Apple 0.03 % prediction score: Banana 99.97 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is tomato prediction score: Apple 3.10 % prediction score: Banana 14.42 % prediction score: Tomato 82.48 %</p>
 <p>prediction: this is apple prediction score: Apple 99.91 % prediction score: Banana 0.09 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is banana prediction score: Apple 0.01 % prediction score: Banana 99.99 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is tomato prediction score: Apple 0.63 % prediction score: Banana 0.78 % prediction score: Tomato 98.60 %</p>
 <p>prediction: this is apple prediction score: Apple 100.00 % prediction score: Banana 0.00 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is banana prediction score: Apple 0.00 % prediction score: Banana 100.00 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is tomato prediction score: Apple 0.37 % prediction score: Banana 0.77 % prediction score: Tomato 98.86 %</p>
 <p>prediction: this is apple prediction score: Apple 99.88 % prediction score: Banana 0.06 % prediction score: Tomato 0.06 %</p>	 <p>prediction: this is banana prediction score: Apple 0.01 % prediction score: Banana 99.99 % prediction score: Tomato 0.00 %</p>	 <p>prediction: this is tomato prediction score: Apple 0.13 % prediction score: Banana 0.25 % prediction score: Tomato 99.62 %</p>

ภาพที่ 35 ผลลัพธ์การทำนายของแบบจำลองจากการสุ่มภาพจำนวน 5 ภาพ

ภาพผลลัพธ์ที่ได้จากการสุ่มภาพ จากชุดข้อมูลทดสอบ (Test data) จำนวน 5 ภาพ ทั้ง 3 ชนิด แอปเปิ้ล กล้วย และมะเขือเทศ ตามลำดับ ซึ่งได้ผลลัพธ์ดังภาพ

4.1.3 ผลลัพธ์การประเมินผลแบบจำลองจากชุดข้อมูลทดสอบ

38/38 [=====] - 7s 196ms/step				
	precision	recall	f1-score	support
apple	0.97	1.00	0.99	400
banana	0.95	1.00	0.97	400
tomato	1.00	0.92	0.96	400
accuracy			0.97	1200
macro avg	0.97	0.97	0.97	1200
weighted avg	0.97	0.97	0.97	1200

ภาพที่ 36 ผลลัพธ์การประเมินผลแบบจำลองด้วยไลบรารี classification_report

4.2 ผลลัพธ์ที่ได้

จากผลการทดลองพบว่า การฝึกฝนแบบจำลองจำนวน 4 ครั้ง แบบจำลองมีค่าความถูกต้อง (Accuracy) ร้อยละ 98.58 หลังจากการฝึกฝนแบบจำลอง ได้ทำการทดสอบแบบจำลองโดยให้แบบจำลองจำแนกภาพผลไม้ทั้ง 3 ชนิด คือ ภาพแอปเปิ้ล ภาพกล้วย และภาพมะเขือเทศ ด้วยแบบจำลองที่ผ่านการถ่ายโอนการเรียนรู้ (Transfer Learning) จากแบบจำลอง การเรียนรู้เชิงลึก (Deep Learning) ประเภทโครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolution Neural Network) สถาปัตยกรรมโครงข่ายประสาทเทียมแบบ 50 ชั้นลึก (ResNet50) ที่ได้จากชุดข้อมูลทดสอบ และทำการสุ่ม 5 ภาพ ซึ่งผลลัพธ์ที่ได้จากการทำนาย ดังนี้ ภาพแอปเปิ้ล ได้ค่าความถูกต้องร้อยละ 99.86 ภาพกล้วย ได้ค่าความถูกต้องร้อยละ 99.99 % และภาพมะเขือเทศ ได้ค่าความถูกต้องร้อยละ 94.81

จากการวัดผลแบบจำลองด้วยไลบรารี classification_report จาก sklearn.metrics ด้วยชุดข้อมูลทดสอบ (Test data) จำนวน 1,200 ภาพทั้งหมด 3 ชนิด แบ่งเป็น ภาพแอปเปิ้ลจำนวน 400 ภาพ ภาพกล้วยจำนวน 400 ภาพ และภาพมะเขือเทศจำนวน 400 ภาพ ผลการประเมินผลสรุปว่าแบบจำลองมีค่าความถูกต้องอยู่ที่ร้อยละ 97 โดยสามารถจำแนกภาพแอปเปิ้ลมีค่าความแม่นยำ (precision) ที่ร้อยละ 97 ค่าความถูกต้อง (recall) ที่ร้อยละ 100 ค่าเฉลี่ยของค่าความแม่นยำ และค่าความถูกต้อง (F1 Score) ที่ร้อยละ 99 ภาพกล้วยมีค่าความแม่นยำ (precision) ที่ร้อยละ 95 ค่าความถูกต้อง (recall) ที่ร้อยละ 100 ค่าเฉลี่ยของค่าความแม่นยำ และค่าความถูกต้อง (F1 Score) ที่ร้อยละ 97 และภาพมะเขือเทศ มีค่าความแม่นยำ (precision) ที่ร้อยละ 100 ค่าความถูกต้อง (recall) ที่ร้อยละ 92 ค่าเฉลี่ยของค่าความแม่นยำ และค่าความถูกต้อง (F1 Score) ที่ร้อยละ 96

บทที่ 5 สรุปผลการดำเนินงาน

5.1 สรุปผล

โครงการนี้ได้ถูกจัดทำขึ้นมาเนื่องจากการคัดแยกผลไม้ที่มีจำนวนมาก และหลากหลายรูปแบบโดยใช้แรงงานมนุษย์ นั้นค่อนข้างใช้เวลาในการคัดแยก และมีประสิทธิภาพในการทำงานไม่ค่อยดีเนื่องจากแรงงานมนุษย์อาจมีความเหนื่อยล้า ซึ่งทำให้ประสิทธิภาพในการคัดแยกผลไม้ทำได้ไม่ดีนัก ซึ่งอาจทำให้เสียต้นทุนหรือทรัพยากรที่มี มากจนเกินไป

จากปัญหาที่กล่าวมาทั้งหมดคณะผู้จัดทำได้ดำเนินการจัดทำโครงการเพื่อสร้างแบบจำลองที่จะสามารถคัดแยกผลไม้ได้โดย ไม่ต้องใช้แรงงานของมนุษย์ ซึ่งใช้เทคโนโลยีการประมวลผลภาพ ร่วมกับโครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network) ซึ่งเป็นการเรียนรู้เชิงลึก (Deep Learning) โดยใช้แบบจำลองที่ชื่อว่า ResNet50 มาทำการ การถ่ายโอนการเรียนรู้ (Transfer Learning) เพื่อสร้างแบบจำลองคัดแยกผลไม้ทั้ง 3 ชนิดคือแอปเปิ้ล กล้วย และมะเขือเทศ โดยได้ดำเนินการสร้างแบบจำลองและทำการประเมินผลแบบจำลองแล้ว ได้ผลลัพธ์ของแบบจำลองที่ผ่านการทดสอบด้วยชุดข้อมูลทดสอบ (Test data) จำนวน 1,200 ภาพ ซึ่งแบบจำลองสามารถจำแนกผลไม้ทั้ง 3 ชนิดได้อย่างแม่นยำ ซึ่งได้ค่าความถูกต้อง (Accuracy) อยู่ที่ร้อยละ 97

5.2 ข้อเสนอแนะ

5.2.1 ในโครงการนี้ได้ใช้ชุดข้อมูลภาพ ผลไม้ 360 (Fruit360) จาก Kaggle ซึ่งมีพื้นหลังเป็นสีขาวซึ่งการจำแนกภาพผลไม้ในทางปฏิบัติแล้ว ภาพอาจมีความหลากหลาย ผลไม้ที่ต้องการจำแนกอาจอยู่รวมกับผลไม้ชนิดอื่นๆ โครงการหรืองานวิจัยในอนาคตอาจมีการนำรูปภาพผลไม้ที่มีฉากหลังอื่นๆ อยู่ด้วยเพื่อนำมาเป็นชุดข้อมูลสำหรับสร้างแบบจำลองให้มีประสิทธิภาพมากยิ่งขึ้น ซึ่งในโครงการนี้ได้เลือกใช้ แอปเปิ้ล กล้วย และมะเขือเทศ มาทดสอบซึ่งเป็นเพียงตัวอย่างในการจำแนก

5.2.2 นำชุดข้อมูลภาพ ผลไม้ 360 (Fruit360) จาก Kaggle หรือชุดข้อมูลภาพที่ได้จากการเตรียมด้วยตนเองหรือแหล่งใดๆ ก็ตามมาประยุกต์ใช้กับอัลกอริทึมการเรียนรู้เชิงลึก (Deep Learning) ประเภทอื่นเช่น แบบจำลองจำแนกข้อมูลด้วยสถาปัตยกรรมโครงข่ายประสาทเทียมตระกูล Inception หรือสถาปัตยกรรมโครงข่ายประสาทเทียม VGG16 และอัลกอริทึมอื่นๆ

บรรณานุกรม

- ณัฐภัทร พิตรปรีชา. (2562). การตรวจหารอยซ้ำของแอปเปิ้ลโดยเทคนิควิเคราะห์ภาพถ่ายคลื่นรังสีความร้อนแบบอินฟราเรด (วิทยานิพนธ์ปริญญาโทมหาบัณฑิต). สถาบันเทคโนโลยีไทย-ญี่ปุ่น สืบค้นจาก
<http://library.tni.ac.th/thesis/upload/files/ThesisISNey/Nattapat%20Pitpreecha%20Thesis%20MIT%202019.pdf>
- นพรุจ พัฒนสาร และ ณัฐวุฒิ ศรีวิบูลย์. (2563). การประมวลผลภาพสำหรับการจำแนกคุณภาพมะม่วงพันธุ์โชคอนันต์โดยการจำลองการมองเห็นของมนุษย์ด้วยวิธีการเรียนรู้เชิงลึก. กาฬสินธุ์. มหาวิทยาลัยกาฬสินธุ์ สืบค้นจาก
<https://ph02.tci-thaijo.org/index.php/JIST/article/view/234447>
- ศิริชัย โชติชาติมาลา. (2563). แบบจำลองการคัดแยกผลไม้แบบหนึ่งชนิดด้วยการเรียนรู้ของเครื่องเพื่อตรวจจับรูปแอปเปิ้ล (วิทยานิพนธ์ปริญญาโทมหาบัณฑิต). มหาวิทยาลัยศรีนครินทรวิโรฒ. สืบค้นจาก
<http://ir-ithesis.swu.ac.th/dspace/handle/123456789/1085>
- DataRockie. (2562). อธิบาย 10 Metrics พื้นฐานสำหรับวัดผลโมเดล Machine Learning. สืบค้น 8 พฤศจิกายน 2564. จาก
<https://datarockie.com/blog/top-ten-machine-learning-metrics/>
- Gaudenz Boesch. (2564). Deep Residual Learning for Image Recognition, ResNet50. สืบค้น 7 พฤศจิกายน 2564. จาก
https://viso.ai/deep-learning/resnet-residual-neural-network/#elementor-toc__heading-anchor-12
- Mr.P L. (2561). Evaluate Model นั้นสำคัญอย่างไร ? : Machine Learning 101. สืบค้น 8 พฤศจิกายน 2564. จาก
<https://medium.com/mmp-li/evaluate-model-precision-recall-f1-score-machine-learning-101-89dbbada0c96>
- Pan Park. (2561). สาย Machine Learning ไม่ควรพลาดกับ Google Colab. สืบค้น 8 พฤศจิกายน 2564. จาก
<https://medium.com/@gnothaigamero/%E0%B8%AA%E0%B8%B2%E0%B8%A%E0%B9%84%E0%B8%A1%E0%B9%88%E0%B8%84%E0%B8%A7%E0%B8%A3%E0%B8%9E%E0%B8%A5%E0%B8%B2%E0%B8%94%E0%B8%81%E0%B8%B1%E0%B8%9A-google-colab-a9f826cbf156>

SAS Institute Inc. (2564). การเรียนรู้เชิงลึก. สืบค้น 7 พฤศจิกายน 2564. จาก
https://www.sas.com/th_th/insights/analytics/deep-learning.html