

โครงการเทคโนโลยีสารสนเทศเพื่อธุรกิจ

Senior Project in Information Technology for Business

ภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี

จุฬาลงกรณ์มหาวิทยาลัย

เรื่อง

66B13

การพัฒนาตัวแบบสำหรับการระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลอง
จานอาหารด้วยเทคนิคการเรียนรู้เชิงลึก

A Model Development for Identifying Macronutrient Ratio for Thai Food Images Based on the
Principle of Plate Model with Deep Learning Techniques

โดย
ชูลั旁กอร์
บิสเนส สคูล

อาจารย์ที่ปรึกษา

รศ.ดร.วรสิทธิ์ ชูชัยวัฒนา

กรรมการ

รศ.ดร.จันทร์เจ้า มงคลนภานิ

ผศ.ดร.ภูริพันธ์ รุจิขจร

ปีการศึกษา 2566

บทคัดย่อ

โครงการเทคโนโลยีสารสนเทศเพื่อธุรกิจนี้มีวัตถุประสงค์คือ เพื่อพัฒนาตัวแบบ (Model) การแบ่งส่วนความหมายของรูปภาพ (Semantic Segmentation) ที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทย ได้แก่ ข้าว กะเพรา, ข้าวคลุกกะปิ, ข้าวมันไก่, ก๋วยเตี๋ยวเส้นบางหมี่เหลืองแห้ง, และผัดไทย โดยยึดตามหลักการของแบบจำลองงานอาหาร เนื่องจากการระบุอัตราส่วนของงานอาหารที่มีความหลากหลายและซับซ้อนด้วยมหุษย์เป็นเรื่องที่ท้าทายและไม่แม่นยำ ดังนั้นการนำตัวแบบเพื่อมาเป็นเครื่องมือในการตัดสินใจในการระบุอัตราส่วนอาหารจะช่วยลดบัญหาดังกล่าว

ในการศึกษาครั้งนี้ผู้ศึกษาได้เริ่มจากการศึกษาองค์ความรู้พื้นฐานต่าง ๆ ที่เกี่ยวข้องกับการเรียนรู้เชิงลึก (Deep Learning) และการแบ่งส่วนความหมายของรูปภาพ และเครื่องมือในการพัฒนาตัวแบบ ได้แก่ ภาษา自然语言 และ ไอลบรารีต่าง ๆ จากนั้นจึงเริ่มกระบวนการสร้างป้ายกำกับ (Label) ระดับพิกเซล, การวิเคราะห์ข้อมูล, การพัฒนาตัวแบบ, ประเมินผลตัวแบบ, และเลือกตัวแบบที่ดีที่สุดนำไปใช้ในการระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทย

หลังจากการศึกษาพบว่าตัวแบบในการระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยนั้นมีประสิทธิภาพที่สามารถระบุอัตราส่วนและพื้นที่ของโภชนาการในจากอาหารได้ดี แต่ยังคงมีข้อจำกัด ได้แก่ งานอาหารและพื้นหลังต้องมีสีโทนขาว ตัวแบบถึงจะระบุอัตราส่วนได้ดี เนื่องจากตัวแบบถูกฝึกฝนมาจากข้อมูลชุดเดียวที่มีลักษณะดังกล่าว

CHULALONGKORN
BUSINESS SCHOOL

FLAGSHIP FOR LIFE

กิตติกรรมประกาศ

โครงการโนโนโลยีสารสนเทศเพื่อธุรกิจฉบับนี้สามารถประสบความสำเร็จได้ เนื่องด้วยความกรุณาของผู้อยู่เบื้องหลังผลงานท่าน ผู้ศึกษาจึงขอขอบคุณทุกท่านที่ได้ให้ความช่วยเหลือตลอดการทำโครงการครั้งนี้

ขอขอบพระคุณ รศ.ดร.วรสิทธิ์ ชูชัยวัฒนา อาจารย์ที่ปรึกษาโครงการเป็นอย่างสูง ที่จัดหาหัวข้อโครงการที่ท้าทาย อีกทั้งยังให้คำปรึกษา ข้อคิดเห็น และช่วยตรวจสอบความบกพร่อง ตลอดการทำโครงการนี้

ขอขอบพระคุณอาจารย์ทุกท่านในภาควิชาสถิติ ที่ถ่ายทอดความรู้ให้ผู้ศึกษาตลอดเวลา 4 ปีที่ศึกษา โดยผู้ศึกษาได้นำความรู้และแนวคิดต่าง ๆ ที่ได้รับมา มาประยุกต์ใช้ในการทำโครงการครั้งนี้

ขอขอบพระคุณอาจารย์และนิสิตคณะสหเวชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ที่ได้ให้ความช่วยเหลือต่าง ๆ ทั้งการจัดหาข้อมูล การให้คำแนะนำ ตลอดการทำโครงการครั้งนี้

ขอขอบพระคุณบิดา มารดา ที่คอยให้กำลังและให้การสนับสนุนตลอดระยะเวลาที่ได้จัดทำโครงการครั้ง ทำให้ผู้ศึกษามีทั้งแรงกายและแรงใจในการทำโครงการครั้งนี้ให้ประสบความสำเร็จได้

CHULALONGKORN
BUSINESS SCHOOL

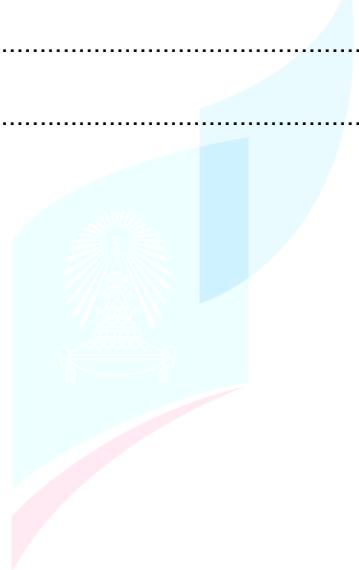
FLAGSHIP FOR LIFE

สารบัญ

บทคัดย่อ	2
กิตติกรรมประกาศ	3
บทที่ 1 บทนำ	1
1.1) ความสำคัญและที่มาของโครงการ	1
1.2) วัตถุประสงค์โครงการ	2
1.3) ขอบเขตการศึกษา	2
1.4) วิธีการศึกษา	2
1.5) ประโยชน์ที่คาดว่าจะได้รับ	3
บทที่ 2 แนวคิดหรือหลักการพื้นฐาน	4
2.1) แบบจำลองจานอาหาร (PLATE MODEL)	4
2.2) การเรียนรู้ของเครื่อง (MACHINE LEARNING)	4
2.3) การเรียนรู้เชิงลึก (DEEP LEARNING)	6
2.4) โครงข่ายประสาทเทียมแบบคอนโวโลชัน (CONVOLUTION NEURAL NETWORK: CNN)	8
2.5) วิธีการจัดการตัวแบบที่เข้ากันได้ดีมากเกินไปกับข้อมูลชุดฝึก (OVERFITTING)	11
2.5.1) การเพิ่มความหลากหลายของข้อมูล (Data Augmentation)	11
2.5.2) Dropout	12
2.5.3) การหยุดการเรียนรู้ก่อนกำหนด (Early Stopping)	12
2.6) การแบ่งส่วนความหมายของรูปภาพ (SEMANTIC SEGMENTATION)	12
2.7) การแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวโลชัน อย่างสมบูรณ์ (FCN-BASED SEMANTIC SEGMENTATION)	15
2.7.1) ลักษณะของข้อมูลที่ใช้พัฒนาตัวแบบการแบ่งส่วนความหมายของรูปภาพ	15
2.7.2) ลักษณะของสถาปัตยกรรมการแบ่งส่วนความหมายของรูปภาพ	16
2.7.3) พังก์ชันการสูญเสีย (Loss Function)	18
2.7.4) การประเมินผลตัวแบบการแบ่งส่วนความหมายของรูปภาพ	19
2.8) สถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวโลชัน	20

2.8.1) VGG16	20
2.8.2) U-Net.....	22
2.9) เครื่องมือที่เกี่ยวข้องสำหรับการพัฒนา	23
2.9.1) แมตแล็บ (Matrix Laboratory: MATLAB).....	24
2.9.2) ไฟร่อน 3 (Python3).....	24
2.9.3) Django 4.....	25
2.10) การทบทวนวรรณกรรม (LITERATURE REVIEW)	25
บทที่ 3 ชุดข้อมูลและการพัฒนาตัวแบบการเรียนรู้เชิงลึก	26
3.1) ลักษณะข้อมูล.....	26
3.2) การสร้างหน้ากากของรูปภาพ และการคำนวณอัตราส่วนโภชนาการ	27
3.3) การเพิ่มความหลากหลายของข้อมูล	31
3.3.1) การย้ายตำแหน่งพิกเซล	31
3.3.2) การปรับค่าพิกเซล	33
3.4) การนำเข้าข้อมูล	38
3.5) การประมวลผลข้อมูลก่อนพัฒนาตัวแบบ (DATA PREPROCESSING)	42
3.6) การพัฒนาตัวแบบ	44
3.7) การประเมินผลตัวแบบ (EVALUATION)	51
3.7.1) ค่า Mean Intersection over Union และ Loss ของข้อมูลชุดทดสอบ	52
3.7.2) กราฟเส้น (Line Graph).....	52
3.7.3) รูปหน้ากากที่ทำนายโดยตัวแบบ	53
บทที่ 4 การประเมินผล และการนำตัวแบบไปใช้งาน	55
4.1) การพัฒนาครั้งที่ 1	55
4.1.1) กะเพราไก่ราดข้าว	55
4.1.2) ข้าวคลุกกะปิ.....	57
4.1.3) ข้าวมันไก่	60
4.1.4) ก๋วยเตี๋ยวbalanceหมีเหลืองแห้ง	62
4.1.5) ผัดไทย	65
4.2) การพัฒนาครั้งที่ 2	67

4.2.1) ข้าวคลุกกะปิ.....	68
4.2.2) ข้าวมันไก่	69
4.2.3) ก๋วยเตี๋ยวบางหมี่เหลืองแห้ง	70
4.3) การนำตัวแบบไปใช้งาน.....	72
บทที่ 5 บทสรุป.....	83
5.1) สรุปผลการศึกษา.....	83
5.2) ข้อจำกัด และปัญหาที่เกิดขึ้น	84
5.3) ข้อเสนอแนะ	85
บรรณานุกรม	87



CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

สารบัญรูปภาพ

รูปที่ 2.1.1 การแบ่งปริมาณอัตราส่วนสารอาหารตามแบบจำลองอาหาร.....	4
รูปที่ 2.3.1 การเรียนรู้เชิงลึกเป็นส่วนหนึ่งของการเรียนรู้ของเครื่อง	6
รูปที่ 2.3.2 ลักษณะของโครงข่ายประสาทเทียม	7
รูปที่ 2.3.3 กระบวนการทำงานของโครงข่ายประสาทเทียม	7
รูปที่ 2.4.1 รูปแบบสถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน	8
รูปที่ 2.4.2 การคุณระหว่างตัวกรองและรูปภาพ	9
รูปที่ 2.4.3 การเลื่อนตัวกรองกับรูปภาพนำเข้า โดยกำหนดค่า STRIDE เท่ากับ 2	10
รูปที่ 2.4.4 ผลลัพธ์จากการเพิ่มหน่วยของคุณลักษณะเพื่อคงขนาดคุณลักษณะ	10
รูปที่ 2.4.5 ผลลัพธ์จากการทำพูลลิ่งด้วยค่าสูงสุด	11
รูปที่ 2.5.1.1 รูปภาพตั้งต้น (ซ้าย) และผลลัพธ์จากการทำการเพิ่มความหลากหลายของข้อมูล	12
รูปที่ 2.6.1 การแบ่งส่วนความหมายของรูปภาพกับการระบุตัวถุของ yan พาหะขับเคลื่อนอัตโนมัติ	13
รูปที่ 2.6.2 ข้อมูลรูปภาพและหน้ากาก	14
รูปที่ 2.7.1.1 หน้าจอของเครื่องมือแมตแล็บ (MATRIX LABORATORY: MATLAB).....	16
รูปที่ 2.7.2.1 สถาปัตยกรรมการแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวลูชันอย่างสมบูรณ์	16
รูปที่ 2.7.2.2 ผลลัพธ์จากการใช้วิธีการเพิ่มความละเอียดสองมิติ	18
รูปที่ 2.7.4.1 รูปแบบสมการของ INTERSECTION OVER UNION	19
รูปที่ 2.8.1.1 สถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน VGG16	20
รูปที่ 2.8.2.1 สถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน U-NET	22
รูปที่ 3.1.1 รูปภาพเมนูข้าวมันไก่ และอัตราส่วนที่วัดโดยการประมาณโดยวงกลมอัตราส่วน	26
รูปที่ 3.2.1 หน้าจอของแมตแล็บ	27

รูปที่ 3.2.2 แทบเครื่องมือของแมตแล็บ	28
รูปที่ 3.2.3 หน้าจอแอปพลิเคชัน IMAGE LABELER	28
รูปที่ 3.2.5 คำสั่งที่ใช้ในการแสดงรูปภาพอาหารและหน้ากากของรูปภาพนั้น	29
รูปที่ 3.2.6 ผลลัพธ์ของคำสั่งในรูปที่ 3.2.5 โดยสีนำเงินหมายถึงผัก สีเหลืองหมายถึงการโน้มีไฮเดรต และสีเขียวหมายถึงโปรตีน	29
รูปที่ 3.2.7 พังก์ชันคำสั่งที่ใช้ในการคำนวณอัตราส่วนโภชนาการได้	30
รูปที่ 3.2.8 ผลลัพธ์ของคำสั่งในรูปที่ 3.2.7 ใน การคำนวณอัตราส่วนโภชนาการของรูปภาพในรูปที่ 3.2.6 .30	
รูปที่ 3.3.1.1 ผลลัพธ์การย้ายตำแหน่งพิกเซลของรูปภาพ แต่ไม่ย้ายตำแหน่งพิกเซลหน้ากาก	31
รูปที่ 3.3.1.2 พังก์ชันคำสั่งที่ใช้ในการหมุนและการสะท้อน	32
รูปที่ 3.3.1.3 ผลลัพธ์จากการหมุนภาพ.....	32
รูปที่ 3.3.1.4 รูปภาพและหน้ากากก่อนการสะท้อน.....	33
รูปที่ 3.3.1.5 ผลลัพธ์จากการสะท้อนภาพจากรูปภาพและหน้ากากในรูปที่ 3.3.1.4	33
รูปที่ 3.3.2.1 พังก์ชันคำสั่งที่ใช้ในการปรับอุณหภูมิ	34
รูปที่ 3.3.2.2 พังก์ชันคำสั่งที่ใช้ในการปรับคอนทราสท์, ความสว่าง, ความคมชัด, และความอิ่มตัว	34
รูปที่ 3.3.2.3 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการปรับอุณหภูมิ (ขวา) ใหม่โทนเย็นมากขึ้น	35
รูปที่ 3.3.2.4 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการเพิ่มคอนทราสท์และความสว่าง (ขวา).	35
รูปที่ 3.3.2.5 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการเพิ่มความคมชัด (ขวา)	36
รูปที่ 3.3.2.6 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการเพิ่มความอิ่มตัว (ขวา).....	37
รูปที่ 3.3.1 พังก์ชันคำสั่งที่ใช้ในการเพิ่มความหลากหลายของข้อมูล	37
รูปที่ 3.4.1 โครงสร้างโฟลเดอร์เก็บรูปภาพ	39
รูปที่ 3.4.2 โครงสร้างโฟลเดอร์ย่อย	39
รูปที่ 3.4.3 รูปภาพในโฟลเดอร์ย่อย	39
รูปที่ 3.4.4 พังก์ชันคำสั่งที่ใช้ในการนำเข้ารูปภาพและหน้ากาก	40

รูปที่ 3.4.5 พังก์ชันคำสั่งที่ใช้ในการลดขนาดและปรับให้เป็นมาตรฐาน.....	40
รูปที่ 3.4.6 ผลลัพธ์จากการเรียกใช้พังก์ชันนำเข้าข้อมูลจากโฟลเดอร์ 3 (ข้ามมันไก่)	41
รูปที่ 3.4.7 ค่าพิกเซลที่ถูกจัดเก็บในอาเรย์.....	42
รูปที่ 3.5.1 ค่าพิกเซลของหน้ากากที่บ่งบอกถึงเป้าหมายการทำนายทั้งหมดที่เป็นไปได้	43
รูปที่ 3.5.2 ผลลัพธ์จากการทำ ONE-HOT ENCODING	43
รูปที่ 3.5.3 พังก์ชันคำสั่งการทำ ONE-HOT ENCODING	44
รูปที่ 3.6.1 สถาปัตยกรรมโครงข่าย U-NET ที่มี VGG16 เป็นกระดูกสันหลัง (ส่วนการเข้ารหัส)	44
รูปที่ 3.6.2 สถาปัตยกรรมโครงข่าย U-NET ที่มี VGG16 เป็นกระดูกสันหลัง (ส่วนการถอดรหัส และ convolutional โกลุชันแบบ 1X1).....	45
รูปที่ 3.6.8 การกำหนดค่าพารามิเตอร์ที่จำเป็นในการ COMPILE ตัวแบบ	48
รูปที่ 3.6.9 พังก์ชันคำสั่งที่ใช้ในการคำนวณค่าน้ำหนัก	49
รูปที่ 3.6.10 ผลลัพธ์ของพังก์ชันการคำนวณน้ำหนัก	49
รูปที่ 3.6.11 พังก์ชันการ CALLBACK	50
รูปที่ 3.6.12 พังก์ชันคำสั่งในการเริ่มฝึกฝนตัวแบบ.....	50
รูปที่ 3.6.13 การแสดงผลการฝึกหัดรอบของการเรียนรู้	51
รูปที่ 3.6.14 คำสั่งในการบันทึกตัวแบบ.....	51
รูปที่ 3.7.1.1 คำสั่งในการประเมินผลตัวแบบด้วยข้อมูลชุดทดสอบ	52
รูปที่ 3.7.1.2 ผลลัพธ์ของคำสั่งประเมินผลตัวแบบด้วยข้อมูลชุดทดสอบ	52
รูปที่ 3.7.2.1 คำสั่งที่ใช้ในการแสดงกราฟเส้น	53
รูปที่ 3.7.2.2 ตัวอย่างกราฟเส้นแสดงค่า LOSS และ MIOU ของข้อมูลชุดฝึกฝนและตรวจสอบ	53
รูปที่ 3.7.3.1 คำสั่งใช้ในการนำข้อมูลชุดทดสอบไปทำนาย และแสดงหน้ากากที่ตัวแบบทำนาย	54
รูปที่ 3.7.3.2 รูปภาพต้น (ซ้าย), หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา).....	54

รูปที่ 4.1.3.4 เปรียบเทียบระหว่างรูปภาพ (ช้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)	62
รูปที่ 4.1.4.1 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูก๋วยเตี๋ยวbalanceมีเหลืองแห้ง (ตัวแบบที่ 1)	63
รูปที่ 4.1.4.2 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูก๋วยเตี๋ยวbalanceมีเหลืองแห้ง (ตัวแบบที่ 2)	63
รูปที่ 4.1.4.3 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูก๋วยเตี๋ยวbalanceมีเหลืองแห้ง (ตัวแบบที่ 3)	64
รูปที่ 4.1.4.4 เปรียบเทียบระหว่างรูปภาพ (ช้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)	64
รูปที่ 4.1.5.1 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูผัดไทย (ตัวแบบที่ 1)	65
รูปที่ 4.1.5.2 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูผัดไทย (ตัวแบบที่ 2)	66
รูปที่ 4.1.5.3 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูผัดไทย (ตัวแบบที่ 3)	66
รูปที่ 4.1.5.4 เปรียบเทียบระหว่างรูปภาพ (ช้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)	67
รูปที่ 4.2.3.1 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูข้าวคลุกกะปิ	68
รูปที่ 4.2.1.1 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดตรวจทางของเมนูข้าวคลุกกะปิ	69

รูปที่ 4.2.2.1 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดทดสอบของเมนูข้าวมันไก่	69
รูปที่ 4.2.3.2 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)	70
รูปที่ 4.2.3.1 กราฟแสดงค่า LOSS เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) และกราฟแสดงค่า MIOU เทียบกับจำนวนรอบที่ฝึกฝน (EPOCH) ของข้อมูลชุดฝึกและชุดทดสอบของเมนูก๋วยเตี๋ยวจะหมี่เหลืองแห้ง ..	71
รูปที่ 4.2.3.2 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)	71
รูปที่ 4.3.1 โฟลเดอร์เว็บแอปพลิเคชัน	72
รูปที่ 4.3.2 โครงสร้างโฟลเดอร์เว็บแอปพลิเคชัน	72
รูปที่ 4.3.3 การเพิ่มแอปพลิเคชันใหม่ที่ SETTINGS.PY	73
รูปที่ 4.3.5 การสร้างหน้าแรกที่ VIEWS.PY	73
รูปที่ 4.3.6 ไฟล์ HYPERTEXT MARKUP LANGUAGE (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าแรก	73
รูปที่ 4.3.7 หน้าจอหน้าแรกของเว็บแอปพลิเคชัน.....	74
รูปที่ 4.3.8 ไฟล์ FORMS.PY	74
รูปที่ 4.3.9 การสร้างหน้าทำนายอัตราส่วนโภชนาการของข้าวมันไก่ที่ VIEWS.PY	75
รูปที่ 4.3.10 คำสั่งที่ใช้ในการนำเข้าตัวแบบ	75
รูปที่ 4.3.10 พังก์ชันคำสั่งที่ใช้ในการเตรียมรูปภาพและทำนาย	75
รูปที่ 4.3.11 พังก์ชันคำสั่งที่ใช้ในการคำนวณอัตราส่วนโภชนาการได้	76
รูปที่ 4.3.12 พังก์ชันคำสั่งที่ใช้ในการสร้างคำอธิบายประกอบอัตราส่วน	77
รูปที่ 4.3.13 ไฟล์ HYPERTEXT MARKUP LANGUAGE (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าทำนายอัตราส่วนโภชนาการอาหารของเมนูข้าวมันไก่	77
รูปที่ 4.3.14 ไฟล์ HYPERTEXT MARKUP LANGUAGE (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าทำนายอัตราส่วนโภชนาการอาหารของเมนูข้าวมันไก่ (ต่อ)	78

รูปที่ 4.3.15 “ไฟล์ HYPERTEXT MARKUP LANGUAGE (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้งานหน้า	
ทำนายอัตราส่วนโภชนาการอาหารของเมนูข้าวมันไก่ (ต่อ).....	78
รูปที่ 3.4.16 หน้าจอหน้าทำนายอัตราส่วนโภชนาการอาหาร	79
รูปที่ 3.4.17 หน้าจอหน้าทำนายอัตราส่วนโภชนาการอาหาร (ต่อ).....	79
รูปที่ 3.4.18 หน้าจอหน้าทำนายอัตราส่วนโภชนาการอาหาร (ต่อ).....	80
รูปที่ 3.4.19 หน้าจอ หน้าจอหน้าทำนายอัตราส่วนโภชนาการอาหาร (ต่อ)	80
รูปที่ 4.3.20 การสร้างหน้าคู่มือการใช้งานที่ VIEWS.PY	80
รูปที่ 4.3.21 “ไฟล์ HYPERTEXT MARKUP LANGUAGE (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้งานหน้า	
คู่มือการใช้งาน.....	81
รูปที่ 4.3.22 หน้าจอหน้าคู่มือการใช้งาน	81
รูปที่ 4.3.23 หน้าจอหน้าคู่มือการใช้งาน (ต่อ)	82
รูปที่ 5.2.1 ผลลัพธ์ของตัวแบบรวมทุกเมนูอาหารที่ถูกผีกິດด้วยรูปภาพขนาด 64X64.....	84

CHULALONGKORN
BUSINESS SCHOOL

FLAGSHIP FOR LIFE

สารบัญตาราง

ตารางที่ 3.1.1 รายชื่อเมนูอาหาร และจำนวนรูปภาพที่ใช้พัฒนาตัวแบบครั้งที่ 1 และ 2	26
ตารางที่ 3.2.1 การเปรียบเทียบอัตราส่วนผิดฟังก์ชัน MEAN ABSOLUTE ERROR (MAE)	30
ตารางที่ 3.3.1 จำนวนของรูปภาพและวิธีการปรับค่าพิกเซล	37
ตารางที่ 3.4.1 โครงสร้างของโผลเดอร์เก็บรูปภาพ	38
ตารางที่ 3.4.2 สัดส่วนการแบ่งหมวดหมู่ (การพัฒนาครั้งที่ 1)	41
ตารางที่ 3.4.3 สัดส่วนการแบ่งหมวดหมู่ (การพัฒนาครั้งที่ 2)	41
ตารางที่ 3.6.1 สรุปสถาปัตยกรรมโครงข่าย U-NET ที่มี VGG16 เป็นกระดูกสันหลัง	47
ตารางที่ 4.1.1.1 ตารางการประเมินผลตัวแบบเมนูภาษาเพราไกราดข้าว (ครั้งที่ 1)	55
ตารางที่ 4.1.2.1 ตารางการประเมินผลตัวแบบเมนูข้าวคลุกกะปิ (ครั้งที่ 1)	57
ตารางที่ 4.1.3.1 ตารางการประเมินผลตัวแบบเมนูข้าวมันไก่ (ครั้งที่ 1)	60
ตารางที่ 4.1.4.1 ตารางการประเมินผลตัวแบบเมนูกุ่ยเตี๋ยวะหมีเหลืองแห้ง (ครั้งที่ 1)	62
ตารางที่ 4.1.5.1 ตารางการประเมินผลตัวแบบเมนูผัดไทย (ครั้งที่ 1)	65
ตารางที่ 4.2.1.1 ตารางการประเมินผลตัวแบบเมนูข้าวคลุกกะปิ (พัฒนาครั้งที่ 2)	68
ตารางที่ 4.2.2.1 ตารางการประเมินผลตัวแบบเมนูข้าวมันไก่ (พัฒนาครั้งที่ 2)	69
ตารางที่ 4.2.3.1 ตารางการประเมินผลตัวแบบเมนูกุ่ยเตี๋ยวะหมีเหลืองแห้ง (พัฒนาครั้งที่ 2)	70

FLAGSHIP FOR LIFE

บทที่ 1 บทนำ

1.1) ความสำคัญและที่มาของโครงการ

อาหารเป็นสิ่งที่จำเป็นสำหรับมนุษย์ในการดำรงชีวิต เมื่อมนุษย์รับประทานอาหารจะเกิดกระบวนการย่อยอาหาร การดูดซึมสารอาหาร และการขับส่งสารอาหารไปยังอวัยวะต่าง ๆ ของร่างกาย เพื่อใช้ในการรักษาเสถียรภาพการทำงานของเซลล์อวัยวะต่าง ๆ ให้เป็นปกติ อาหารที่ถูกย่อยให้เป็นโมเลกุลที่เล็กลงจะเรียกว่า สารอาหาร (Nutrients) สารอาหารสามารถแบ่งออกเป็นสองกลุ่ม ได้แก่ สารอาหารหลัก (Macronutrients) และสารอาหารที่ต้องการในปริมาณน้อย (Micronutrients) สารอาหารหลัก ได้แก่ คาร์โบไฮเดรต, โปรตีน, และไขมัน ส่วนสารอาหารที่ต้องการในปริมาณน้อย ได้แก่ วิตามิน และแร่ธาตุ การจัดการปริมาณสารอาหารหลักอย่างมีประสิทธิภาพเป็นสิ่งสำคัญสำหรับการได้รับสารอาหารเข้าสู่ร่างกายอย่างสมดุล ป้องกันการขาดสารอาหาร และส่งเสริมสุขภาพที่ดี

แบบจำลองจานอาหาร (Plate Model) เป็นแนวทางที่ใช้ในการออกแบบจานอาหารเพื่อให้มีปริมาณอัตราส่วนสารอาหารที่สมดุล โดยทางกรมอนามัยและกระทรวงสาธารณสุขได้นำแนวทางดังกล่าวมาประยุกต์ใช้เป็นแนวทางการจัดจานอาหารแบบ $2:1:1$ ซึ่งเลือกใช้จานแบบกลม ขนาดเส้นผ่าศูนย์กลาง 9 นิ้ว จากนั้นแบ่งอัตราส่วนของจานกลมออกเป็น 4 ส่วนเท่า ๆ กัน แล้วจัดวางอาหารประเภทต่าง ๆ ใส่ในจาน ได้แก่ ผัก 2 ส่วน ข้าวเปลือก 1 ส่วน และเนื้อสัตว์ 1 ส่วน อย่างไรก็ตามการนำแบบจำลองอาหารมาใช้กับอาหารที่มีความหลากหลาย เช่น อาหารไทย อาจเป็นเรื่องที่ท้าทายสำหรับบุคคลที่ไม่เคยและนักวิเคราะห์หากไม่เครื่องมือวิเคราะห์ที่มีประสิทธิภาพ

จากความท้าทายในการจัดจานอาหารโดยใช้แบบจำลองอาหาร ผู้ศึกษาจึงมีความประสงค์ที่จะพัฒนาตัวแบบการระบุอัตราส่วนโภชนาการจากรูปภาพไทยตามหลักการของแบบจำลองจานอาหาร เพื่อให้บุคคลทั่วไปสามารถใช้ระบุอัตราส่วนของอาหารที่จัดว่าเป็นไปตามอัตราส่วนตามแบบจำลองอาหารหรือไม่ และช่วยให้นักวิเคราะห์อาหารสามารถใช้ในการตัดสินใจในกระบวนการระบุอัตราส่วนของโภชนาการได้อย่างแม่นยำมากขึ้น ตัวแบบดังกล่าวจะใช้หลักการของการเรียนรู้เชิงลึกที่เลียนแบบกระบวนการทำงานของระบบโครงข่ายประสาทในสมองมนุษย์ (Neural Network) การแบ่งส่วนความหมายของรูปภาพ โดยในการฝึกฝนตัวแบบจะใช้ข้อมูลรูปภาพอาหารไทย 5 ชนิด ได้แก่ ข้าวகະເພຣາ, ข້າວມັນໄກ໌ຕົມ, ข້າວຄຸກກະປີ, กໍວຍເຕີ່ຍວະໜີ່ເຫຼືອງແໜ້ງ, และຜັດໄທ ตัวแบบที่ได้ต้องสามารถระบุอัตราส่วนโภชนาการตามหลักการของแบบจำลองจานอาหารโดยมีประสิทธิภาพมากกว่าการระบุอัตราส่วนด้วยมนุษย์

1.2) วัตถุประสงค์โครงการ

- 1) เพื่อศึกษาเครื่องมือที่ใช้ในการพัฒนาตัวแบบระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร
- 2) เพื่อศึกษาลำดับวิธีการดำเนินการพัฒนาตัวแบบ ได้แก่ การสร้างป้ายกำกับระดับพิกเซล การวิเคราะห์ข้อมูล การเตรียมข้อมูลเพื่อนำไปฝึกฝนตัวแบบ การพัฒนาตัวแบบ การประเมินผลตัวแบบ และการนำตัวแบบไปใช้งานจริง
- 3) เพื่อพัฒนาตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร

1.3) ขอบเขตการศึกษา

- 1) เทคนิคที่ใช้ในการจัดทำตัวแบบในโครงการนี้คือ เทคนิคการเรียนรู้เชิงลึก เช่น โครงข่ายประสาทเทียม โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network) และการแบ่งส่วนความหมายของรูปภาพ เป็นต้น
- 2) การดำเนินการจัดทำตัวแบบในโครงการนี้ได้เลือกข้อมูลรูปภาพอาหารไทยจำนวน 5 ชนิด ได้แก่ กะเพราไก่ราดข้าว, ข้าวคุกกะบี, ข้าวมันไก่, ก๋วยเตี๋ยวบะหมี่เหลืองแห้ง, และผัดไทย โดยมีคุณลักษณะของข้อมูล ได้แก่ รูปภาพอาหารไทยทั้ง 5 ชนิด ซึ่งอาหาร อัตราส่วนของอาหารภายในงานโดยมีลักษณะเป็นไปตาม รูปแบบของแบบจำลองอาหาร รูปแบบอัตราส่วน ส่วนประกอบ ปริมาณ (กรัม) พลังงาน (กิโลแคลอรี) คาร์โบไฮเดรต (กรัม) โปรตีน (กรัม) และไขมัน (กรัม) โดยข้อมูลดังกล่าวถูกจัดเตรียมโดยคณะสหเวชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

1.4) วิธีการศึกษา

- 1) ศึกษาวิธีการและเครื่องมือที่ใช้ในการพัฒนาตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร
- 2) เลือกวิธีการและเครื่องมือที่เหมาะสมกับข้อมูลและปัญหา
- 3) สร้างป้ายกำกับระดับพิกเซลให้กับข้อมูลรูปภาพเพื่อนำพัฒนาตัวแบบ
- 4) พัฒนาตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร
- 5) ประเมินผลตัวแบบและเปรียบเทียบตัวแบบที่ได้พัฒนาขึ้น
- 6) นำตัวแบบไปใช้งานจริงผ่านเว็บแอปพลิเคชัน

7) สรุปผลการศึกษา ปัญหา และข้อเสนอแนะ

1.5) ประโยชน์ที่คาดว่าจะได้รับ

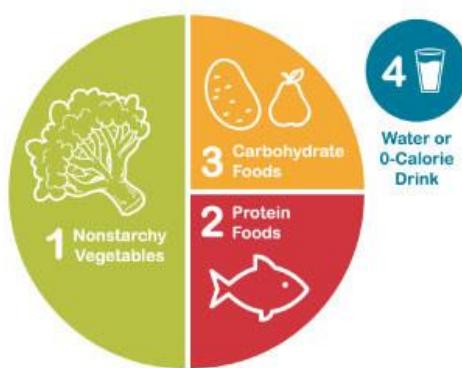
- 1) ได้ศึกษาและทำความเข้าใจเกี่ยวกับวิธีการและเครื่องมือที่ใช้ในการพัฒนาตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร
- 2) ได้ศึกษาวิธีการดำเนินการพัฒนาตัวแบบ ได้แก่ การสร้างป้ายกำกับระดับพิกเซล การวิเคราะห์ข้อมูล การเตรียมข้อมูลเพื่อนำไปฝึกฝนตัวแบบ การพัฒนาตัวแบบ การประเมินผลตัวแบบ และการนำตัวแบบไปใช้งานจริง
- 3) ตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหารที่สามารถช่วยบุคคลทั่วไปและนักวิเคราะห์อาหารระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยเบื้องต้นได้



บทที่ 2 แนวคิดหรือหลักการพื้นฐาน

2.1) แบบจำลองอาหาร (Plate Model)

แบบจำลองอาหารเป็นแนวทางที่ใช้ในการออกแบบอาหารเพื่อให้มีปริมาณอัตราส่วนสารอาหารที่สมดุล โดยทางกรมอนามัยและกระทรวงสาธารณสุขได้นำแนวทางดังกล่าวมาประยุกต์ใช้เป็นแนวทางการจัดอาหารแบบ $2:1:1$ ซึ่งเลือกใช้จานแบบกลม ขนาดเส้นผ่าศูนย์กลาง 9 นิ้ว จากนั้นแบ่งอัตราส่วนของจานกลมออกเป็น 4 ส่วนเท่า ๆ กัน และจัดวางอาหารประเภทต่าง ๆ ใส่ในจาน ได้แก่ ผัก 2 ส่วน ข้าวเปลือก 1 ส่วน และเนื้อสัตว์ 1 ส่วน



รูปที่ 2.1.1 การแบ่งปริมาณอัตราส่วนสารอาหารตามแบบจำลองอาหาร

ที่มา: <https://www.diabetesfoodhub.org/articles/what-is-the-diabetes-plate-method.html>

2.2) การเรียนรู้ของเครื่อง (Machine Learning)

การเรียนรู้ของเครื่องเป็นส่วนหนึ่งของปัญญาประดิษฐ์ (Artificial Intelligence) เป็นการศึกษาอัลกอริทึม (Algorithm) ของคอมพิวเตอร์ โดยอัลกอริทึมดังกล่าวจะสร้างแบบจำลองทางคณิตศาสตร์จากข้อมูลที่นำมาฝึกฝนเรียกว่า ข้อมูลชุดฝึก (Training Data) เพื่อใช้ในการจำแนก ทำนาย จัดกลุ่ม รวมถึงการหาความสัมพันธ์ในข้อมูล

กระบวนการทำงานของการเรียนรู้ของเครื่องคือ การนำข้อมูลเข้ามา (Input Data) และข้อมูลข้อออก (Output Data) เข้าสู่คอมพิวเตอร์ จากนั้นคอมพิวเตอร์จะประมวลผลร่วมกับอัลกอริทึม เพื่อค้นหารูปแบบ (Pattern) ที่เหมาะสมกับข้อมูลข้างต้นและแปลงสิ่งที่ค้นพบเหล่านั้นให้เป็นตัวแบบ จากนั้นผู้พัฒนาสามารถใช้ตัวแบบข้างต้นเพื่อการทำนายหรือตัดสินใจ แทนที่จะทำงานตามลำดับของคำสั่งโปรแกรมคอมพิวเตอร์ เมื่อมีการสั่งให้ทำงาน (Execute) ตามการเขียนโปรแกรมแบบดั้งเดิม (Traditional Programming)

การเรียนรู้ของเครื่องสามารถแบ่งประเภทของการเรียนรู้ได้เป็น 3 วิธี ได้แก่ การเรียนรู้แบบมีผู้สอน (Supervised Learning), การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning), และการเรียนรู้แบบเสริมกำลัง (Reinforcement Learning)

1) การเรียนรู้แบบมีผู้สอน (Supervised Learning)

การเรียนรู้แบบมีผู้สอนเป็นการเรียนรู้ของเครื่องที่จำเป็นต้องได้รับการสอนโดยมนุษย์ผ่านการสร้างป้ายกำกับ, เป้าหมาย (Target) หรือผลลัพธ์ (Output) ให้กับข้อมูลขาเข้า หรือข้อมูลชุดฝึกหัดจากนั้นเครื่องคอมพิวเตอร์จะเรียนรู้ผ่านข้อมูลข้างต้น ค้นหารูปแบบที่สามารถเชื่อมโยงข้อมูลขาเข้ากับข้อมูลขาออกได้

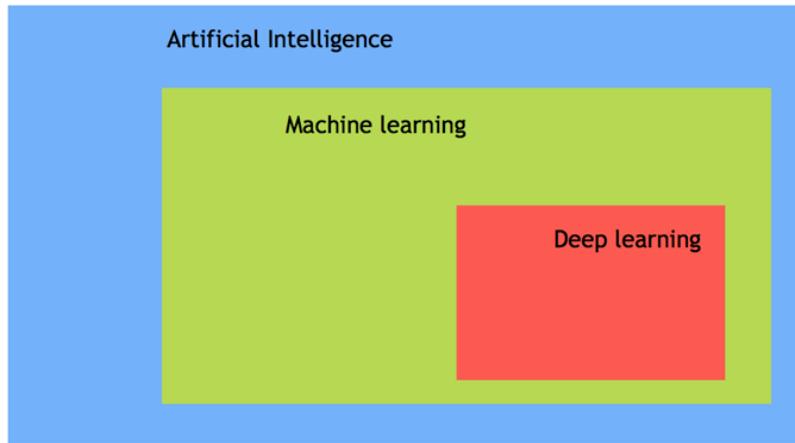
2) การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning)

การเรียนรู้แบบไม่มีผู้สอนเป็นการเรียนรู้ของเครื่องที่ไม่จำเป็นต้องได้รับการสอนโดยมนุษย์ ข้อมูลขาเข้า หรือข้อมูลชุดฝึก มีเพียงคุณลักษณะ (Feature) โดยคอมพิวเตอร์จะประมวลผลข้อมูลขาเข้าร่วมกับอัลกอริทึม การเรียนรู้แบบไม่มีผู้สอนจะพิจารณาข้อมูลเป็นเซต (Set) ของตัวแปรสุ่มจากนั้นจึงสร้างตัวแบบความหนาแน่นร่วมของชุดข้อมูล

3) การเรียนรู้แบบเสริมกำลัง (Reinforcement Learning)

การเรียนรู้แบบเสริมกำลังเป็นการเรียนรู้ของเครื่องที่เปลี่ยนไปตลอดเวลาตามสภาพแวดล้อมที่เปลี่ยนแปลงไป หลักการเรียนรู้แบบเสริมกำลังจะสร้างเป้าหมาย (Goal) และรางวัล (Reward) เพื่อกำหนดให้คอมพิวเตอร์ทำการกรรมต่าง ๆ เพื่อบรรลุเป้าหมาย แต่จะไม่มีผู้สอนโดยบอกว่ากิจกรรมที่คอมพิวเตอร์ทำอยู่นั้นเข้าใกล้เป้าหมายแล้วหรือไม่ และมีการกำหนดรางวัลให้กับคอมพิวเตอร์ที่ทำการกรรมที่ส่งผลให้เข้าใกล้เป้าหมาย คอมพิวเตอร์จะประมวลผลในทุกความเปลี่ยนแปลงในแต่ละรอบของการเรียนรู้เพื่อหาทางเลือกที่ดีที่สุดของแต่ละกิจกรรมตามแต่ละสถานการณ์

2.3) การเรียนรู้เชิงลึก (Deep Learning)



รูปที่ 2.3.1 การเรียนรู้เชิงลึกเป็นส่วนหนึ่งของการเรียนรู้ของเครื่อง
ที่มา: <https://www.thaiprogrammer.org/2018/12/การเรียนรู้ของเครื่องmachine-le/>

การเรียนรู้เชิงลึกเป็นส่วนหนึ่งของการเรียนรู้ของเครื่อง โดยใช้เทคนิคที่เลียนแบบกระบวนการทำงานของสมองมนุษย์เรียกว่า โครงข่ายประสาทเทียม (Artificial Neural Network: ANN) วิธีการที่ใช้ในการเรียนรู้อาจเป็นได้ทั้งการเรียนรู้แบบมีผู้สอน และการเรียนรู้แบบไม่มีผู้สอน

โครงข่ายประสาทเทียมได้รับแรงบันดาลใจจากโครงข่ายประสาทเทียมทางชีววิทยาที่มีอยู่ในสมองของมนุษย์หรือสัตว์ ประกอบไปด้วยเซลล์ประสาท (Neuron) หรือในทางเทคนิคจะถูกเรียกว่า โนนด (Node) แต่เซลล์ประสาทจะเชื่อมต่อกันเป็นโครงข่าย (Network) โดยโครงข่ายจะถูกแบ่งออกเป็น 3 ชั้น (Layer) ได้แก่

1) ชั้นรับข้อมูล (Input Layer)

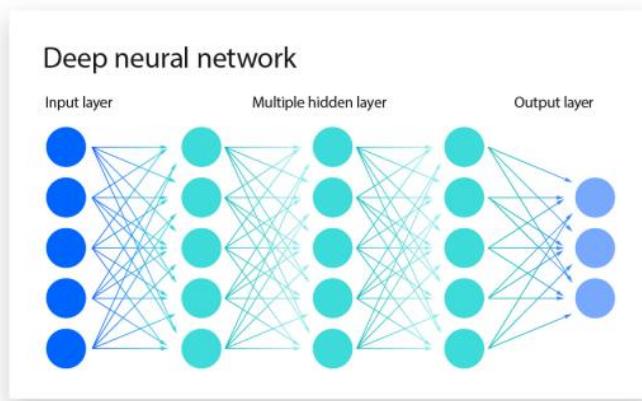
ชั้นรับข้อมูลมีหน้าที่รับข้อมูลเข้ามา โดยชั้นนี้จะมีจำนวนเซลล์ประสาทเท่ากับจำนวนคุณลักษณะของข้อมูล

2) ชั้นประมวลผล (Hidden Layer)

ชั้นประมวลผลมีหน้าที่รับข้อมูลจากชั้นรับข้อมูลเพื่อนำมาประมวลผลผ่านอัลกอริทึมที่ผู้พัฒนากำหนด โดยชั้นนี้สามารถมีจำนวนได้มากกว่า 1 ชั้น และแต่ละชั้นสามารถมีจำนวนเซลล์ประสาทเท่าไรก็ได้ตามที่ผู้พัฒนาต้องการ อย่างไรก็ตามการกำหนดจำนวนชั้นและจำนวนเซลล์ประสาทมากหรือน้อยเกินไปอาจส่งผลเสียต่อผลลัพธ์ที่เกิดขึ้น

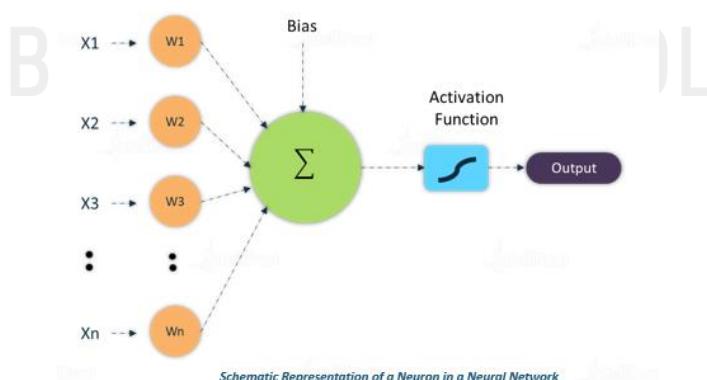
3) ชั้นผลลัพธ์ (Output Layer)

ชั้นผลลัพธ์มีหน้าที่รับข้อมูลจากชั้นประมวลผลเพื่อนำมาประมวลผลผ่านอัลกอริทึมที่ผู้พัฒนากำหนด โดยอัลกอริทึมจะสอดคล้องกับผลลัพธ์ที่ต้องการนำไปใช้



รูปที่ 2.3.2 ลักษณะของโครงข่ายประสาทเทียม
ที่มา: <https://www.ibm.com/topics/neural-networks>

ในแต่ละชั้นประมวลผลจะมีโหนดคำอ่าน (Bias Node) ที่ใช้เป็นค่าคงวนร่วมกับค่าถ่วงนำหนัก (Weight) ที่เป็นพารามิเตอร์การเรียนรู้ (Learning Parameters) ของโหนดอื่น ๆ ในชั้นนั้น ๆ เพื่อหาค่านำหนักรวม (Weighted Sum) เพื่อนำไปคำนวนผ่านฟังก์ชันกระตุ้น (Activation Function) ก่อนที่จะส่งไปคำนวนในชั้นถัดไป โดยค่าในโหนดคำอ่านและค่าถ่วงนำหนักในโหนดต่าง ๆ จะเปลี่ยนแปลงค่าตัวเองไปเรื่อย ๆ ระหว่างการเรียนรู้



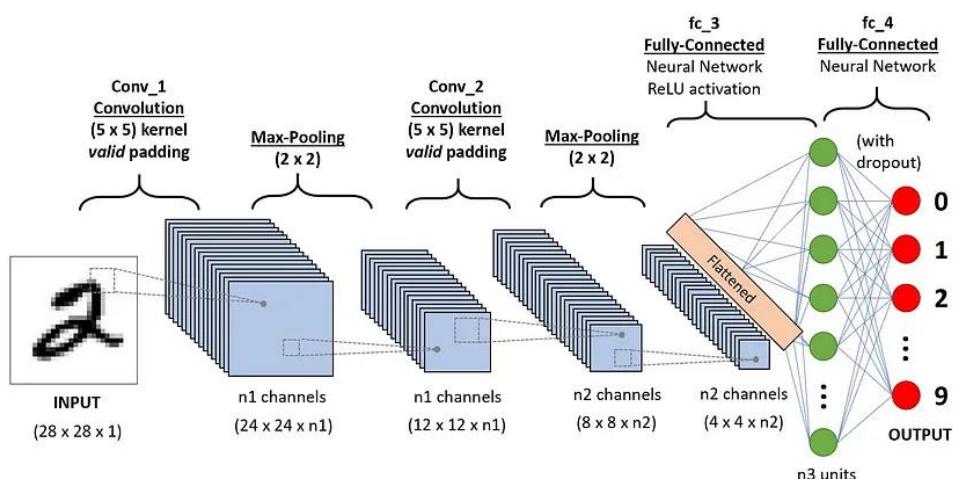
รูปที่ 2.3.3 กระบวนการทำงานของโครงข่ายประสาทเทียม
ที่มา: <https://intellipaat.com/community/253/role-of-bias-in-neural-networks>

ในขั้นตอนการเรียนรู้ ชั้นผลลัพธ์จะมีการรับค่าที่ได้จากการคำนวณผ่านฟังก์ชันกระตุ้นของชั้นประมวลผลชั้นสุดท้าย เพื่อนำมาผ่านฟังก์ชันกระตุ้นของชั้นตัวเองที่สอดคล้องกับผลลัพธ์ที่ต้องการนำไปใช้ และได้ออกมาเป็นผลลัพธ์ จำนวนจะนำผลลัพธ์ที่ได้มาคำนวณผ่านฟังก์ชันการสูญเสีย (Loss Function) ร่วมกับผลลัพธ์จริงของข้อมูลนั้น จากนั้นจะนำค่าที่ได้จากการคำนวณผ่านฟังก์ชันการสูญเสียมาระบุนย้อนกลับไปแต่ละโหนดในโครงข่ายเรียกวิธีนี้ว่า การแพร่ย้อนกลับ (Back Propagation) เพื่อเปลี่ยนแปลงค่าในโหนดลำเอียงและค่าถ่วงน้ำหนักในโหนดต่าง ๆ ให้เหมาะสมกับข้อมูลเพื่อที่จะได้นำไปใช้งานจริง

ในบริบทของคำว่า “ลีก” ในการเรียนรู้เชิงลึก หมายถึงการมีชั้นประมวลผลมากกว่า 1 ชั้นในโครงข่ายประสาทเทียม

2.4) โครงข่ายประสาทเทียมแบบconvโอลูชัน (Convolution Neural Network: CNN)

โครงข่ายประสาทเทียมแบบconvโอลูชัน (Convolution Neural Network: CNN) จัดเป็นการเรียนรู้เชิงลึกแบบหนึ่ง โดยจำลองการพฤติกรรมการมองเห็นของมนุษย์ที่มองพื้นที่ส่วนย่อย ๆ ของภาพหรือวัตถุ จากนั้นนำกลุ่มของพื้นที่ย่อย ๆ มาพسانกันเพื่อตรวจสอบดูว่าสิ่งที่กำลังมองอยู่นั้นคืออะไร โครงข่ายประสาทเทียมแบบconvโอลูชันสามารถสกัดคุณลักษณะเด่น (Feature Extraction) ของข้อมูลที่ไม่มีโครงสร้าง (Unstructured Data) เช่น ข้อความ, รูปภาพ, วิดีโอ เป็นต้น ได้ด้วยตัวเองโดยใช้ตัวกรอง (Filter)



รูปที่ 2.4.1 รูปแบบสถาปัตยกรรมโครงข่ายประสาทเทียมแบบconvโอลูชัน

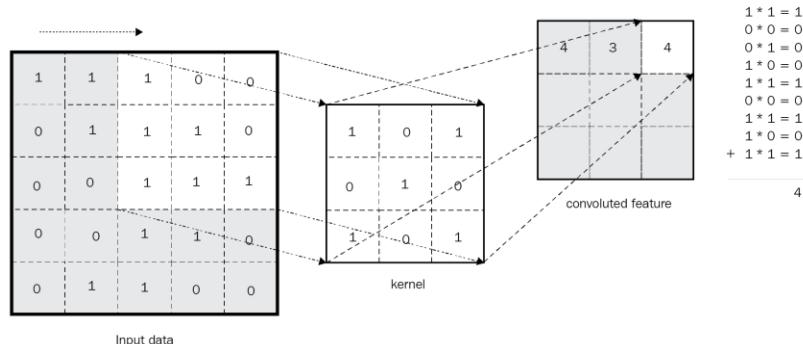
ที่มา: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

ตัวอย่างสถาปัตยกรรมโครงข่ายประสาทเทียมแบบ convolutional neural network (Convolution Neural Network Architecture: CNN Architecture) ที่ใช้ในงานประมวลผลรูปภาพ ((Image Classification)) มีขั้นตอนหลัก 2 ขั้นตอน ได้แก่ ขั้นตอนการ convolution (Convolution Stage) และขั้นตอนการเชื่อมต่ออย่างสมบูรณ์ (Fully Connected Stage) นอกจากนี้อาจมีเพิ่มขั้นตอนการ pooling (Pooling Stage) ระหว่างขั้นตอนการ convolution ของ convolutional neural network เพื่อเพิ่มความเร็วให้กับการเรียนรู้ของโครงข่าย แต่อาจส่งผลกระทบต่อผลลัพธ์

1) ขั้นตอนการ convolution (Convolution Stage)

เป็นขั้นตอนที่มีวัตถุประสงค์เพื่อสกัดคุณลักษณะของข้อมูล ขั้นตอนนี้จะมีการสร้างตัวกรองที่เรียกว่า Sliding Window (Kernel) มาคำนวณกับข้อมูลเพื่อสกัดคุณลักษณะเด่นข้างต้น

ตัวกรองแต่ละตัวจะมีจำนวนค่าพารามิเตอร์การเรียนรู้ขึ้นอยู่กับขนาดของตัวกรอง เช่น ตัวกรองขนาด 3×3 จะมีจำนวนค่าพารามิเตอร์การเรียนรู้ 9 ค่า และทุก ๆ ค่าจะเปลี่ยนแปลงตลอดช่วงเวลาการเรียนรู้ เพื่อที่จะได้ตัวกรองและค่าพารามิเตอร์การเรียนรู้ที่เหมาะสมกับข้อมูล



รูปที่ 2.4.2 การคูณระหว่างตัวกรองและรูปภาพ

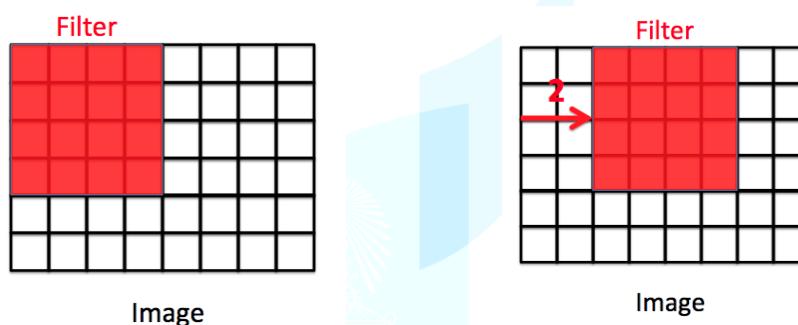
ที่มา: <https://www.oreilly.com/library/view/neural-networks-with/9781788397872/70ba8233-7507-474e-9958-4d2d338b5a44.xhtml>

ตัวอย่างข้อมูลประมวลรูปภาพที่ประกอบไปด้วย 3 ช่องสี (Channel) ได้แก่ สีแดง, สีเขียว, และสีน้ำเงิน พิกเซลที่แทนด้วยตัวเลขเพื่อบอกความเข้มของสีมีค่าตั้งแต่ 0 ถึง 255

ตัวกรองจะถูกทำบลังบันรูปภาพที่จุดขวางน จำกันจะทำการคูณแบบอาเรย์ (Array) กับพิกเซลของรูปภาพที่ถูกทำบันย แล่นำผลลัพธ์จากการคูณเมทริกซ์ดังกล่าวมาบวกกัน จำกันตัวกรองจะถูกเลื่อนไปบนพิกเซลอื่น ๆ ทางขวาจนครบทุกพิกเซลในภาพ โดยระยะในการเลื่อนจะขึ้นอยู่กับค่า Stride ที่จะกล่าวถึงในภายหลัง วิธีการนี้เรียกว่า convolution (Convolution) และเก็บค่าที่ได้ไว้ในเมทริกซ์ชุดใหม่ที่เรียกว่า ผังคุณลักษณะ (Feature Map)

ในขั้นตอนการสกัดคุณลักษณะจะต้องมีการทำหนดค่าต่าง ๆ ดังต่อไปนี้

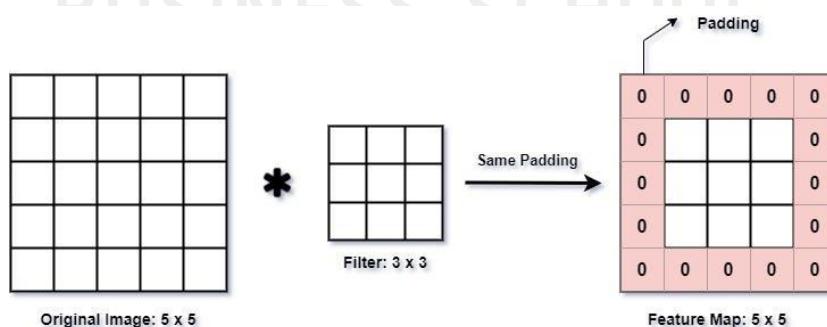
- 1.1) ขนาดเมทริกซ์ตัวกรองหรือ Sliding Window (Kernel) โดยทั่วไปจะมีขนาด 3×3 , 5×5 , หรือ 7×7 ขึ้นอยู่กับความต้องการในผลลัพธ์ ตัวอย่างเช่น หากตัวกรองมีขนาดเล็กจะสามารถสกัดคุณลักษณะของข้อมูลได้มากขึ้น ในทางตรงข้ามหากตัวกรองมีขนาดใหญ่จะสามารถสกัดคุณลักษณะของข้อมูลได้น้อยลง
- 1.2) Stride เป็นตัวกำหนดว่าต้องการจะเลื่อนเมทริกซ์ตัวกรองไปครั้งละกี่หน่วย ยกตัวอย่างเช่น หากข้อมูลเป็นรูปภาพ การกำหนดค่า Stride เท่ากับ 1 หมายความว่าเลื่อนเมทริกซ์ตัวกรองไปครั้งละ 1 พิกเซล



รูปที่ 2.4.3 การเลื่อนตัวกรองกับรูปภาพนำเข้า โดยกำหนดค่า Stride เท่ากับ 2

ที่มา: <https://medium.com/machine-learning-algorithms/what-is-stride-in-convolutional-neural-network-e3b4ae9baedb>

- 1.3) การเพิ่มหน่วยของคุณลักษณะ (Padding) ยิ่งจำนวนชั้นของการ convolution นานเท่าไร ขนาดของคุณลักษณะที่ถูกสกัดออกมาก็จะมีขนาดเล็กลงตามจำนวนชั้น ซึ่งอาจส่งผลให้ข้อมูลสูญหาย ตัวอย่างเช่น ขอบของรูปภาพอาจมีความสำคัญในการตัดสินใจ



รูปที่ 2.4.4 ผลลัพธ์จากการเพิ่มหน่วยของคุณลักษณะเพื่อคงขนาดคุณลักษณะ

ที่มา: <https://blog.gopenai.com/padding-in-neural-networks-why-and-how-b076ab0a4fc2>

2) ขั้นตอนการเชื่อมต่ออย่างสมบูรณ์ (Fully Connected Stage)

ขั้นตอนการเชื่อมต่ออย่างสมบูรณ์เป็นขั้นตอนที่ทำหน้าที่ในการจำแนก (Classification) ซึ่งจะใช้โครงสร้างเดียวกันกับโครงข่ายประสาทเทียม โดยชั้นรับข้อมูลจะรับผลลัพธ์สุดท้ายจากขั้นตอนการconvโวลูชัน หรือขั้นตอนการพูลลิ่ง โดยจะเริ่มจากการเปลี่ยนรูป (Re-Shape) เมทริกซ์ผังคุณลักษณะให้อยู่ในรูปแบบคอลัมน์เดียว เรียกกระบวนการนี้ว่า การทำให้แบนราบ (Flattening)

3) ขั้นตอนการพูลลิ่ง (Pooling Stage)

ขั้นตอนการพูลลิ่งเป็นขั้นตอนที่ใช้ลดขนาดมิติ (Downsample) ของผังคุณลักษณะ เพื่อให้ผังคุณลักษณะมีขนาดเล็กลงแต่ยังคงรักษารายละเอียดไว้ได้

การพูลลิ่งมีขั้นตอนการทำงานคล้ายกับการสกัดคุณลักษณะด้วยตัวกรอง กล่าวคือการพูลลิ่งจะเปรียบเสมือนตัวกรองชนิดหนึ่งที่มีการกำหนดขนาดตัวกรองและการStride โดยตัวกรองนี้จะทำการลงบนผังคุณลักษณะ และใช้ฟังก์ชันการคำนวณ เช่น การพูลลิ่งด้วยค่าสูงสุด (Max Pooling), การพูลลิ่งด้วยค่าเฉลี่ย (Average Pooling), การพูลลิ่งด้วยค่าต่ำสุด (Min Pooling) เป็นต้น โดยนำค่าของผังคุณลักษณะที่ถูกทำมาคำนวณตามฟังก์ชันที่กำหนด จากนั้นจะเลือกตัวกรองตามค่าStride ที่กำหนดไว้



รูปที่ 2.4.5 ผลลัพธ์จากการทำพูลลิ่งด้วยค่าสูงสุด

ที่มา: <https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/>

2.5) วิธีการจัดการตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting)

2.5.1) การเพิ่มความหลากหลายของข้อมูล (Data Augmentation)

การเพิ่มความหลากหลายของข้อมูลเป็นเทคนิคที่จะสร้างข้อมูลอย่างสุ่มจากข้อมูลที่มีอยู่โดยจะกำหนดให้อัลกอริทึมสร้างข้อมูลขึ้นมาใหม่ตามคุณลักษณะที่กำหนด ตัวอย่างเช่น ข้อมูลประเภทภาพสามารถกำหนดคุณลักษณะเพื่อสร้างข้อมูลใหม่ เช่น การบิดภาพ, การหมุนภาพ,

การย่อหรือขยายภาพ, การเปลี่ยนสีภาพ และการตัดบางส่วนของภาพออกไป เป็นต้น นอกจากการจัดการตัวแบบที่เข้ากันได้ดีมากเกินไปกับข้อมูลชุดฝึก (Overfitting) การเพิ่มความหลากหลายของข้อมูลยังสามารถใช้เพื่อเพิ่มจำนวนข้อมูลในกรณีที่จำนวนข้อมูลชุดฝึกน้อยเกินไป



รูปที่ 2.5.1.1 รูปภาพตั้งต้น (ข้าว) และผลลัพธ์จากการทำการเพิ่มความหลากหลายของข้อมูล

2.5.2) Dropout

Dropout เป็นเทคนิคที่ปิดการใช้งานของโหนดในชั้นประมวลผลอย่างสุ่ม โดยจะมีการกำหนดอัตรา Dropout เมื่อเริ่มต้นการเรียนรู้ และจะมีการสุ่มค่าความน่าจะเป็นในการเก็บรักษาโหนดใด ๆ ที่ค่าความน่าจะเป็นในการเก็บรักษาอย่างกว่าอัตรา Dropout โหนดนั้นจะถูกปิด

2.5.3) การหยุดการเรียนรู้ก่อนกำหนด (Early Stopping)

การหยุดการเรียนรู้ก่อนกำหนดเป็นวิธีที่จะหยุดการเรียนรู้ของตัวแบบก่อนกำหนดในกรณีที่ไม่มีการปรับปรุงในค่าที่สนใจ เช่น ค่าการสูญเสียของข้อมูลชุดตรวจทาน (Validation Loss) ค่าความแม่นยำของข้อมูลชุดตรวจทาน (Validation Accuracy) เป็นต้น นอกจากการจัดการตัวแบบที่เข้ากันได้ดีมากเกินไปกับข้อมูลชุดฝึก การหยุดการเรียนรู้ก่อนกำหนดยังช่วยลดต้นทุนในการคำนวณในช่วงการเรียนรู้

2.6) การแบ่งส่วนความหมายของรูปภาพ (Semantic Segmentation)

การแบ่งส่วนความหมายของรูปภาพ (Semantic Segmentation) เป็นจัดเป็นการเรียนรู้เชิงลึกในกลุ่มโครงข่ายประสาทเทียมแบบคอนโวลูชัน ดำเนินการโดยใช้แนวคิดการระบุตำแหน่ง (Localizing) ในการ

ค้นหาวัตถุและวัดกรอบล้อมรอบวัตถุ ต่อมาจะจำแนกประเภทของวัตถุ จากนั้นจะจัดกลุ่ม (Clustering) พิกเซลในภาพที่ถูกระบุตำแหน่ง

การแบ่งส่วนความหมายของรูปภาพสามารถนำมาใช้ประโยชน์ได้ในงานต่าง ๆ ตัวอย่างเช่น ยานพาหนะขับเคลื่อนอัตโนมัติ โดยยานพาหนะจะสามารถรับรู้สภาพแวดล้อมขณะเคลื่อนที่ได้ เช่น มุชช์ย์ ยานพาหนะคนอื่น เป็นต้น



รูปที่ 2.6.1 การแบ่งส่วนความหมายของรูปภาพกับการระบุวัตถุของยานพาหนะขับเคลื่อนอัตโนมัติ

ที่มา: <https://www.youtube.com/watch?v=ATlcEDSPWXY>

การแบ่งส่วนความหมายของรูปภาพสามารถทำได้ 3 วิธี (Yanming Guo et al., 2017) ได้แก่

- 1) การแบ่งส่วนความหมายของรูปภาพตามส่วน (Region-Based Semantic Segmentation)

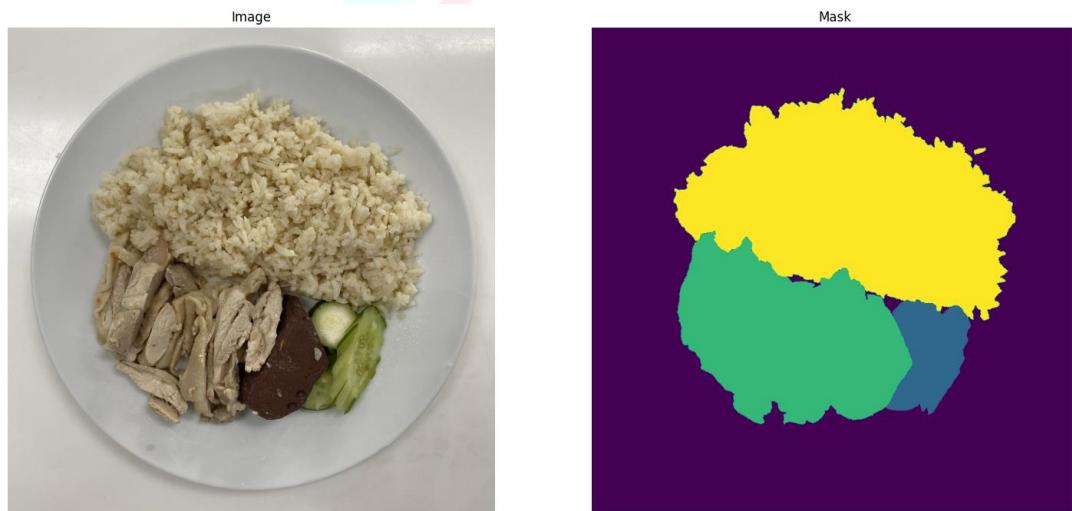
การแบ่งส่วนความหมายของรูปภาพตามส่วนเป็นวิธีที่คล้ายคลึงกับการตรวจจับวัตถุ (Object Detection) กล่าวคือแบ่งรูปภาพออกเป็นส่วนต่าง ๆ จากนั้นจะสกัดคุณลักษณะของแต่ละพื้นที่ผ่านโครงข่ายประสาทเทียมแบบคอนโวูลูชันที่ผ่านการฝึกฝนมาแล้ว เช่น R-CNN, Faster R-CNN เป็นต้น จากนั้นใช้ตัวแบบการจำแนก (Classification Model) เพื่อกำหนดป้ายกำกับให้แต่ละส่วน ซึ่งตัวแบบการจำแนกดังกล่าวอาจเป็นตัวแบบที่ผ่านการฝึกฝนมาแล้วหรือตัวแบบออกแบบ และฝึกฝนเอง สุดท้ายจะใช้วิธี Non-maximum Suppression กำจัดส่วนที่ซ้ำซ้อนหรือทับซ้อนกันเพื่อเก็บเฉพาะส่วนที่ไม่ทับซ้อนกันเท่านั้น

2) การแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวลูชันอย่างสมบูรณ์

(FCN-Based Semantic Segmentation)

การแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวลูชันอย่างสมบูรณ์เป็นวิธีที่ใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันที่สมบูรณ์ (Fully Convolutional Networks) เพื่อใช้ในการจำแนกระดับพิกเซล โดยโครงข่ายประสาทเทียมแบบคอนโวลูชันที่สมบูรณ์ประกอบไปด้วย ส่วนแรกคือ ส่วนการเข้ารหัส (Encoder section) ทำหน้าที่ในการสกัดคุณลักษณะ และลดขนาดเชิงพื้นที่ของข้อมูล ส่วนที่สองคือ ส่วนการถอดรหัส (Decoder Section) ทำหน้าที่ในการเพิ่มขนาดเชิงพื้นที่ของผังคุณลักษณะที่ได้รับจากส่วนการเข้ารหัส โดยทั้งสองส่วนจะถูกเชื่อมต่อกันโดยการข้ามการเชื่อมต่อ (Skip Connection) ทำหน้าที่ช่วยรักษารายละเอียดที่ข้อมูลในระหว่างการประมวลผลในส่วนการถอดรหัส โดยชั้นสุดท้ายจะใช้คอนโวลูชันแบบ 1×1 ทำหน้าที่ช่วยแปลงผังคุณลักษณะที่ผ่านการประมวลผลจากส่วนการถอดรหัสให้เป็นผลลัพธ์ที่ถูกจำแนกระดับพิกเซล

ข้อมูลที่ใช้ในวิธีฝึกฝนตัวแบบการแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวลูชันอย่างสมบูรณ์ประกอบไปด้วย ชุดรูปภาพ และหน้ากาก (Mask) ของแต่ละรูปภาพ หน้ากากของรูปภาพคือ รูปภาพที่มีป้ายกำกับระดับพิกเซลที่บ่งบอกถึงการแบ่งส่วนความหมายของรูปภาพว่าพิกเซลนั้นถูกจำแนกเป็นอะไร



รูปที่ 2.6.2 ข้อมูลรูปภาพและหน้ากาก

3) การแบ่งส่วนความหมายของรูปภาพตามผู้สอนที่อ่อนแอก (Weakly Supervised Semantic Segmentation)

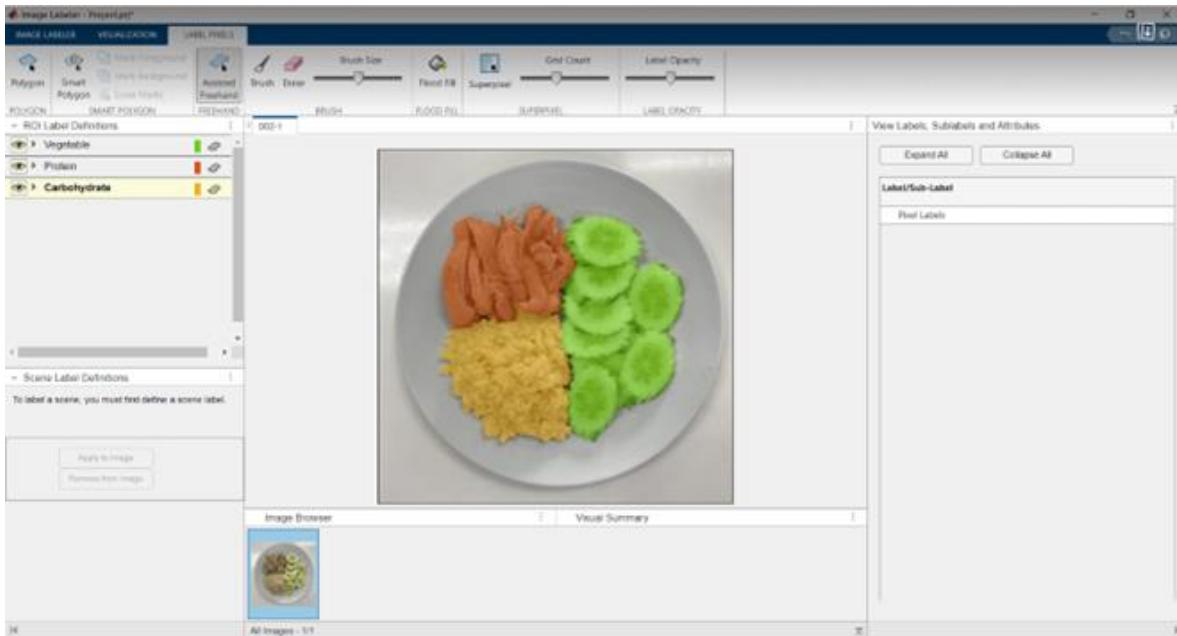
การแบ่งส่วนความหมายของรูปภาพตามผู้สอนที่อ่อนแอกเป็นวิธีที่คล้ายคลึงกับการแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวโลชันอย่างสมบูรณ์ แต่ต่างกันที่ข้อมูลที่ใช้ในวิธีฝึกฝนตัวแบบ เป็นจากการสร้างป้ายกำกับระดับพิกเซล หรือหน้ากากของแต่ละรูปภาพนั้นใช้เวลานานสำหรับข้อมูลที่มีจำนวนมาก เพื่อแก้ไขปัญหาดังกล่าว เราสามารถใช้ป้ายกำกับที่ไม่ละเอียดได้ เช่น ป้ายกำกับระดับรูปภาพ กล่องล้อมรอบวัตถุ เป็นต้น

2.7) การแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวโลชันอย่างสมบูรณ์ (FCN-Based Semantic Segmentation)

2.7.1) ลักษณะของข้อมูลที่ใช้พัฒนาตัวแบบการแบ่งส่วนความหมายของรูปภาพ

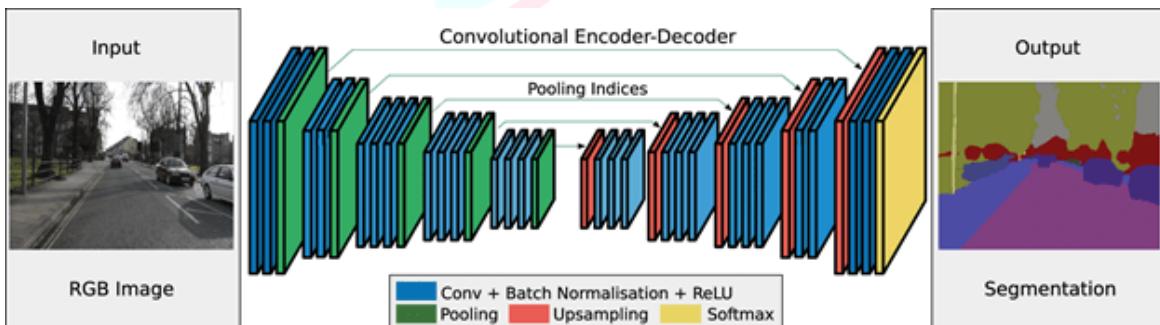
ข้อมูลที่ใช้ในวิธีพัฒนาตัวแบบประกอบไปด้วย ชุดรูปภาพ และหน้ากาก (Mask) ของแต่ละรูปภาพ หน้ากากของรูปภาพคือ รูปภาพที่มีป้ายกำกับระดับพิกเซลที่บ่งบอกว่าพิกเซลนั้นถูกจำแนกเป็นอะไร รูปภาพดังกล่าวจะมีค่าสีของพิกเซลขึ้นอยู่กับจำนวนเป้าหมายการทำนายทั้งหมดที่เป็นไปได้ ตัวอย่างเช่น ชนิดของโภชนาการ ได้แก่ ข้าวเบง, เนื้อสัตว์, และผัก รูปภาพจะมีค่าสีของพิกเซล 4 ค่า ได้แก่ 0, 1, 2, และ 3 โดยค่า 0 หมายถึงพื้นที่ไม่ได้จัดเป็นเป้าหมายใด ๆ

การสร้างรูปภาพหน้ากากสามารถทำได้ผ่านเครื่องมือการสร้างป้ายกำกับรูปภาพ (Image Labeling Tool) เช่น แมตแล็บ (Matrix Laboratory: MATLAB), เลเบลบ็อกซ์ (Labelbox) เป็นต้น



รูปที่ 2.7.1.1 หน้าจอของเครื่องมือแมตแล็บ (Matrix Laboratory: MATLAB)

2.7.2) ลักษณะของสถาปัตยกรรมการแบ่งส่วนความหมายของรูปภาพ



รูปที่ 2.7.2.1 สถาปัตยกรรมการแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวอลูชันอย่างสมบูรณ์
ที่มา: <https://neptune.ai/blog/image-segmentation>

สถาปัตยกรรมสำหรับการแบ่งส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบคอนโวอลูชันอย่างสมบูรณ์คือ โครงข่ายประสาทเทียมแบบคอนโวอลูชันที่สมบูรณ์ (Fully Convolutional Networks) ใช้ในการจำแนกระดับพิกเซล โดยโครงข่ายประสาทเทียมแบบคอนโวอลูชันที่สมบูรณ์ประกอบไปด้วย 4 องค์ประกอบ ได้แก่

1) ส่วนการเข้ารหัส (Encoder Section)

ส่วนการเข้ารหัสรายละเอียดที่เหมือนกับขั้นตอนการ convolutional neural network คือส่วนที่นำรูปภาพเข้ามาและผ่านชั้น convolutional layer และชั้น pooling layer แล้วส่งผลให้มีข้อมูลน้อยลง

2) ส่วนการถอดรหัส (Decoder Section)

ส่วนการถอดรหัสรายละเอียดที่เพิ่มความละเอียด (Upsampling) ของคุณลักษณะเด่นของรูปภาพที่ได้รับจากส่วนการเข้ารหัสกลับไปเป็นขนาดรูปภาพเดิม สิ่งนี้เป็นหัวใจสำคัญของการแปลงส่วนความหมายของรูปภาพตามโครงข่ายประสาทเทียมแบบ convolutional neural network ที่มีความสามารถในการรับรู้และจัดการข้อมูลที่ซับซ้อน เช่น การจำแนกประเภทของวัตถุในภาพ หรือการแปลงรูปภาพจากสีเป็นขาวดำ

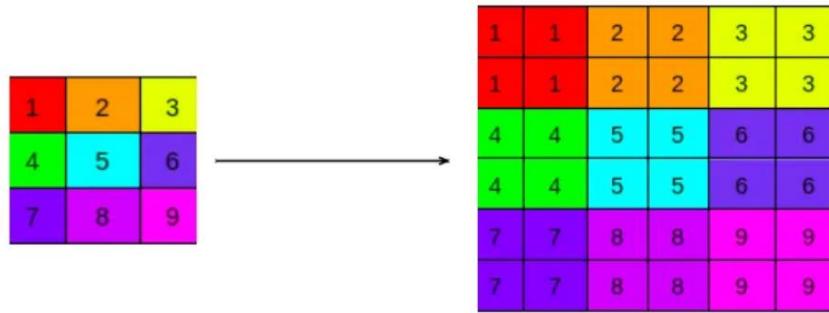
ตัวอย่างวิธีที่เกี่ยวข้องกับการเพิ่มความละเอียด ได้แก่

2.1) convolutional transpose (Transposed Convolutional)

convolutional transpose เป็นวิธีที่มีการดำเนินการตรงกันข้ามกับ convolutional คือ convolutional transpose เป็นการสกัดคุณลักษณะจากรูปภาพเพื่อให้ได้มาซึ่งผังคุณลักษณะ และส่งผลให้มีข้อมูลน้อยลง แต่ในทางตรงกันข้าม convolutional transpose เป็นการนำผังคุณลักษณะมาเพิ่มความละเอียดผ่านตัวกรองที่เหมือนกับ convolutional แต่ส่งผลให้สามารถเพิ่มมิติเชิงพื้นที่ของผังคุณลักษณะ อีกทั้งยังสามารถรวมผังคุณลักษณะที่มีระดับความละเอียดต่ำๆ ที่สูงขึ้นได้ ตัวกรองใน convolutional transpose เป็นแบบบีบบัดบีบ มีพารามิเตอร์การเรียนรู้ เช่น เดียว กับ convolutional แต่ส่งผลมีการเรียนรู้เกิดขึ้นที่ชั้น convolutional แบบบีบบัดบีบในช่วงฝึกฝนตัวแบบ

2.2) การเพิ่มความละเอียดสองมิติ (Two-dimensional Upsampling)

การเพิ่มความละเอียดสองมิติเป็นวิธีอย่างง่ายในการเพิ่มมิติเชิงพื้นที่ของผังคุณลักษณะ คือการอ้างอิงการขยายจากพิกเซลใกล้เคียง และไม่มีพารามิเตอร์การเรียนรู้ ดังนั้นจึงมีการดำเนินการที่รวดเร็ว



รูปที่ 2.7.2.2 ผลลัพธ์จากการใช้วิธีการเพิ่มความละเอียดสองมิติ

ที่มา: <https://medium.com/@suryatejamenta/upsampler-and-convolution-transpose-layers-7e4346c05bb6>

3) การข้ามการเชื่อมต่อ (Skip Connection)

การข้ามการเชื่อมต่อคือ การเชื่อมต่อที่ข้ามจากชั้นหนึ่งหรือหลายชั้นในโครงข่าย 譬如 ภาพเทียม ช่วยให้การส่งต่อของข้อมูลจากส่วนการเข้ารหัสไปยังส่วนการถอดรหัส ยังคงรักษารายละเอียดของข้อมูลดังเดิมไว้ได้ การข้ามการเชื่อมต่อช่วยในการบรรเทา ปัญหาการสูญหายของ Gradient Descent ในขั้นตอนการแพร่ย้อนกลับ

4) คอนโวลูชันแบบ 1×1 (1x1 Convolution)

คอนโวลูชันแบบ 1×1 เป็นชั้นคอนโวลูชันที่มีตัวกรองขนาด 1 คูณ 1 ทำหน้าที่ใน การรวมคุณลักษณะต่าง ๆ เข้าด้วยกันได้ ส่งผลให้ช่วยลดมิติเชิงความลึกของคุณลักษณะ ได้ นอกจากนี้ยังช่วยปรับมิติเชิงความลึกของคุณลักษณะ เพื่อให้สามารถนำคุณลักษณะที่ ถูกปรับมิติเชิงความลึกไปประมวลผลร่วมกับคุณลักษณะอื่น ๆ ได้

2.7.3) พังก์ชันการสูญเสีย (Loss Function)

1) พังก์ชัน Focal Loss

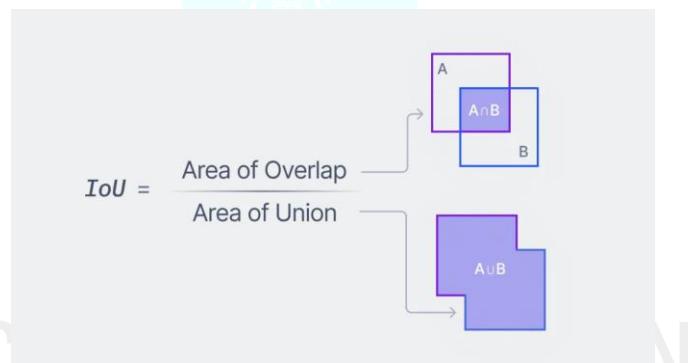
พังก์ชัน Focal Loss ช่วยในการจัดการกับปัญหาความไม่สมดุลของสัดส่วน เป้าหมายการทำนายทั้งหมดที่เป็นไปได้ โดยการกำหนดความสำคัญให้มากขึ้นกับ เป้าหมายที่จำแนกได้ยาก เช่น วัตถุที่มีสัญญาณรบกวน (Noise) เป็นต้น ดังนั้นพังก์ชัน Focal Loss จะช่วยลดผลการลงโทษตัวแบบหากจำแนกวัตถุง่ายผิดพลาด และเพิ่มผลการ ลงโทษตัวแบบหากจำแนกวัตถุยากผิดพลาด ส่งผลให้ตัวแบบเรียนรู้ที่จะจำแนกวัตถุยาก ได้ดียิ่งขึ้น

2) พังก์ชัน Dice Loss

พังก์ชัน Dice Loss ช่วยในการจัดการกับปัญหาความไม่สมดุลของสัดส่วน เป้าหมายการทำนายทั้งหมดที่เป็นไปได้ เช่นเดียวกับพังก์ชัน Focal Loss อย่างไรก็ตาม จะเน้นเฉพาะปัญหาความไม่สมดุลระหว่างพื้นหน้าและพื้นหลัง ที่ส่งผลกระทบต่อกระบวนการฝึกฝนตัวแบบ

2.7.4) การประเมินผลตัวแบบการแบ่งส่วนความหมายของรูปภาพ

ค่าเฉลี่ยของ Intersection over Union หรือ MIoU เป็นวิธีการประมาณผลตัวแบบที่ให้ผลลัพธ์เชิงพื้นที่ Mean Intersection over Union มีค่าอยู่ระหว่าง 0 ถึง 1 โดยยิ่งมีค่ามากนั้นหมายถึง ตัวแบบ ทำนายได้แม่นยำ ในการคำนวณจะใช้หน้าหากาที่ตัวแบบทำนายกับหน้าหากาที่แท้จริงของรูปภาพ จากนั้นนำทั้งสองหน้าหากามาเปรียบเทียบกันแยกตามแต่ละเป้าหมายการทำนายทั้งหมดที่เป็นไปได้ ตามรูปแบบสมการ ดังรูป



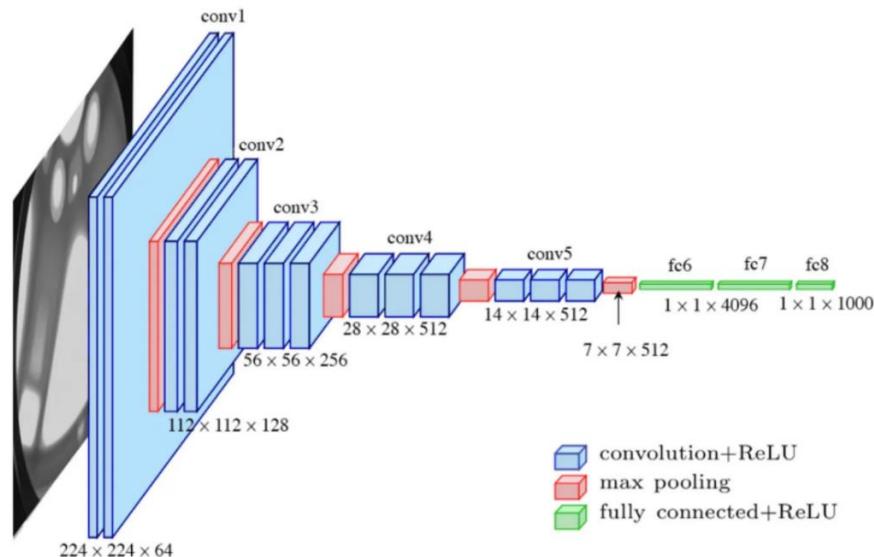
รูปที่ 2.7.4.1 รูปแบบสมการของ Intersection over Union
ที่มา: <https://www.v7labs.com/blog/intersection-over-union-guide>

ส่วน Area of Overlap หมายถึง ผลรวมของตำแหน่งพิกเซลเบ้าหมายได ๆ ของหน้าหากาที่ทำนายโดยตัวแบบได้ถูกต้อง โดยอ้างอิงจากตำแหน่งพิกเซลเบ้าหมายได ๆ ของหน้าหากาที่แท้จริงของรูปภาพ ส่วน Area of Union หมายถึง ผลรวมของตำแหน่งพิกเซลเบ้าหมายได ๆ ของหน้าหากาที่ทำนายโดยตัวแบบรวมกับตำแหน่งพิกเซลเบ้าหมายได ๆ ของหน้าหากาที่แท้จริงของรูปภาพ โดยไม่นับชั้นพิกเซลที่ซ้อนทับกัน

ในการหาค่าเฉลี่ยของ Intersection over Union จะนำค่า Intersection over Union ของแต่ละเป้าหมายการทำนายทั้งหมดที่เป็นไปได้มาหาค่าเฉลี่ย

2.8) สถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน

2.8.1) VGG16



รูปที่ 2.8.1.1 สถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน VGG16

ที่มา: <https://medium.com/mlearning-ai/an-overview-of-vgg16-and-nin-models-96e4bf398484>

VGG16 เป็นสถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน มีคำย่อมาจาก Visual Geometry Group 16 ได้รับการเสนอโดย Karen Simonyan และ Andrew Zisserman จาก Visual Geometry Group Lab ของ Oxford University โดยใช้สถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันกับข้อมูลชุด ImageNet จากรายการแข่งขัน ImageNet Large Scale Visual Recognition Challenge พ布ว่าให้ค่าความแม่นยำถึงร้อยละ 92.7

สถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชัน VGG16 ประกอบไปด้วย 22 ชั้น ได้แก่

1) ชั้นที่ 1: ชั้นรับข้อมูล (Input Layer)

ทำหน้าที่รับข้อมูลเป็นรูปภาพประเภทสี (Red Green Blue: RGB) ขนาด 224x224 พิกเซล จากนั้นข้อมูลรูปภาพจะถูกส่งต่อไปยังชั้นประมวลผลถัดไป

2) ชั้นที่ 2 และ 3: convolutional layer ที่ 1 (conv1)

แต่ละชั้นประกอบไปด้วยชั้น convolution โดยใช้ตัวกรองขนาด 3×3 และกำหนดมีการกำหนดจำนวนผังคุณลักษณะที่ต้องการสกัดออกมาจำนวน 64 ผัง

3) ชั้นที่ 4: ชั้นการพูลลิ่ง 1

ทำหน้าที่ลดขนาดผังคุณลักษณะ โดยใช้ฟังก์ชันพูลลิ่งด้วยค่าสูงสุด และขนาดตัวกรอง 2×2 จะได้ผลลัพธ์เป็นเมทริกซ์ผังคุณลักษณะขนาด $(112, 112, 64)$

4) ชั้นที่ 5 และ 6: convolutional layer ที่ 2 (conv2)

แต่ละชั้นประกอบไปด้วยชั้น convolution โดยใช้ตัวกรองขนาด 3×3 และกำหนดมีการกำหนดจำนวนผังคุณลักษณะที่ต้องการสกัดออกมาจำนวน 128 ผัง

5) ชั้นที่ 7: ชั้นการพูลลิ่ง 2

ทำหน้าที่ลดขนาดผังคุณลักษณะ โดยใช้ฟังก์ชันพูลลิ่งด้วยค่าสูงสุด และขนาดตัวกรอง 2×2 จะได้ผลลัพธ์เป็นเมทริกซ์ผังคุณลักษณะขนาด $(56, 56, 128)$

6) ชั้นที่ 8, 9, และ 10: convolutional layer ที่ 3 (conv3)

แต่ละชั้นประกอบไปด้วยชั้น convolution โดยใช้ตัวกรองขนาด 3×3 และกำหนดมีการกำหนดจำนวนผังคุณลักษณะที่ต้องการสกัดออกมาจำนวน 256 ผัง

7) ชั้นที่ 11: ชั้นการพูลลิ่ง 3

ทำหน้าที่ลดขนาดผังคุณลักษณะ โดยใช้ฟังก์ชันพูลลิ่งด้วยค่าสูงสุด และขนาดตัวกรอง 2×2 จะได้ผลลัพธ์เป็นเมทริกซ์ผังคุณลักษณะขนาด $(28, 28, 256)$

8) ชั้นที่ 12, 13, และ 14: convolutional layer ที่ 4 (conv4)

แต่ละชั้นประกอบไปด้วยชั้น convolution โดยใช้ตัวกรองขนาด 3×3 และกำหนดมีการกำหนดจำนวนผังคุณลักษณะที่ต้องการสกัดออกมาจำนวน 512 ผัง

9) ชั้นที่ 15: ชั้นการพูลลิ่ง 4

ทำหน้าที่ลดขนาดผังคุณลักษณะ โดยใช้ฟังก์ชันพูลลิ่งด้วยค่าสูงสุด และขนาดตัวกรอง 2×2 จะได้ผลลัพธ์เป็นเมทริกซ์ผังคุณลักษณะขนาด $(14, 14, 512)$

10) ชั้นที่ 16, 17, และ 18: คอนโวลูชันกลุ่มที่ 5 (conv5)

แต่ละชั้นประกอบไปด้วยชั้นคอนโวลูชัน โดยใช้ตัวกรองขนาด 3×3 และกำหนดมีการกำหนดจำนวนผังคุณลักษณะที่ต้องการสกัดออกมาจำนวน 512 ผัง

11) ชั้นที่ 19: ชั้นการพูลลิ่ง 5

ทำหน้าที่ลดขนาดผังคุณลักษณะ โดยใช้ฟังก์ชันพูลลิ่งด้วยค่าสูงสุด และขนาดตัวกรอง 2×2 จะได้ผลลัพธ์เป็นเมทริกซ์ผังคุณลักษณะขนาด $(7, 7, 512)$

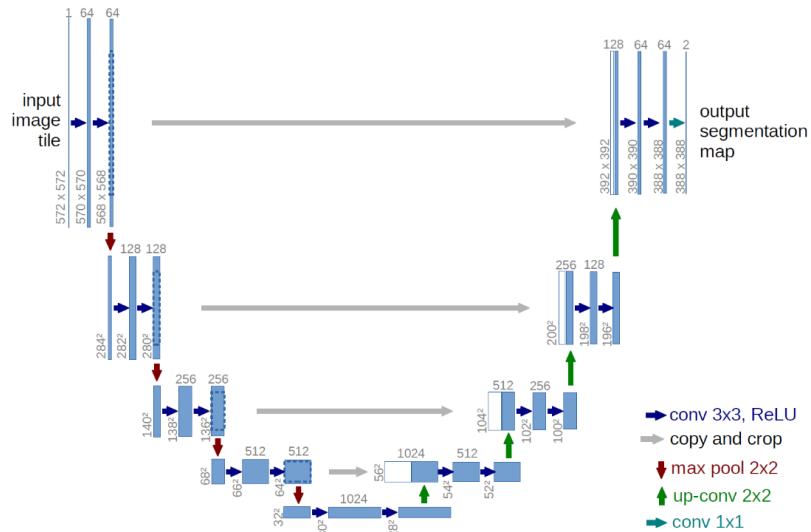
12) ชั้นที่ 20 และ 21: ชั้นการเชื่อมโยงแบบสมบูรณ์ (Fully Connected Layer)

แต่ละชั้นทำหน้าที่ในการจำแนก ประกอบไปด้วย 4,096 โนนด

13) ชั้นที่ 22: ชั้นผลลัพธ์

ถูกกำหนดให้จำนวนโนนดเท่ากับ 1,000 โนนด โดยใช้การคำนวณผลลัพธ์ด้วยฟังก์ชันเลขซึ่งกำลังที่ทำให้เป็นมาตรฐาน (Softargmax) หรือที่เรียกว่า ฟังก์ชันซอฟต์แมกซ์ (Softmax)

2.8.2) U-Net



รูปที่ 2.8.2.1 สถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชัน U-Net

ที่มา: <https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/>

U-Net เป็นสถาปัตยกรรมโครงข่ายประสาทเทียมแบบคอนโวลูชันที่พัฒนาขึ้นสำหรับการแบ่งส่วนความหมายของรูปภาพ ทางชีววิทยาทางการแพทย์ที่ภาควิชาชีววิทยาการคอมพิวเตอร์ของมหาวิทยาลัยไฟรบูร์ก (University of Freiburg) นอกจากนี้ยังถูกนำมาใช้งานกับตัวแบบการแพร่กระจาย (Diffusion Model) สำหรับการลดสัญญาณรบกวนในรูปภาพ (Denoising)

สถาปัตยกรรมโครงข่ายทั่วไปของ U-Net ประกอบไปด้วย 2 ส่วน ได้แก่ ส่วนแรก การเข้ารหัส (Encoder) หรืออาจเรียกว่า กระดูกสันหลัง (Backbone) และส่วนที่สอง การถอดรหัส (Decoder)

1) การเข้ารหัส (Encoder)

การเข้ารหัสเป็นส่วนที่ทำหน้าที่สกัดคุณลักษณะข้อมูลเรียกว่า ขั้นตอนการคอนโวลูชัน ในส่วนนี้ผู้พัฒนาสามารถกำหนดสถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันเองได้ หรืออาจใช้สถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันที่ถูกพัฒนาขึ้นมา เช่น VGG16 ชั้นที่ 2 ถึง 19 จากที่ได้กล่าวไปในหัวข้อที่แล้ว เป็นต้น การดำเนินการในส่วนนี้จะทำให้ขนาดของข้อมูลเชิงพื้นที่ลดลง ในขณะที่ข้อมูลเชิงคุณลักษณะจะเพิ่มขึ้น

2) การถอดรหัส (Decoder)

การถอดรหัสเป็นส่วนที่ทำหน้าที่สมมูลนั้นที่ได้จากการสกัดจากส่วนการเข้ารหัส หรืออาจเรียกว่าข้อมูลเชิงคุณลักษณะ ร่วมกับข้อมูลเชิงพื้นที่ที่ถูกส่งต่อมาก斯์ส่วนการถอดรหัสระหว่างกระบวนการเข้ารหัส โดยจะใช้เทคนิคการเพิ่มความละเอียด (Upsampling) ที่จะเพิ่มความละเอียดของผลลัพธ์

ที่มาของคำว่า U-Net คือ สถาปัตยกรรมโครงข่ายที่คล้ายกับตัวอักษร U เนื่องจากมีช่องทางการข้ามการเชื่อมต่อ ที่สมมาตรระหว่างส่วนการเข้ารหัสและส่วนการถอดรหัส ช่องทางการข้ามการเชื่อมต่อข้างต้นมีหน้าที่ส่งข้อมูลเชิงพื้นที่ไปประมวลผลร่วมกับข้อมูลเชิงคุณลักษณะ อีกทั้งยังช่วยในเรื่องการเผยแพร่องกลับ ทำให้โครงข่ายสามารถเรียนรู้ได้ดียิ่งขึ้น

2.9) เครื่องมือที่เกี่ยวข้องสำหรับการพัฒนา

โครงงานนี้มีวัตถุประสงค์เพื่อพัฒนาตัวแบบระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร ผู้ศึกษาได้เลือกใช้เครื่องมือในการพัฒนาดังต่อไปนี้

2.9.1) แมตแล็บ (Matrix Laboratory: MATLAB)

แมตแล็บผลิตโดยบริษัทแมตเวิร์กส์ (MathWorks) เป็นส่วนชุดคำสั่ง (Software) ในการคำนวณและการเขียนโปรแกรม มีความสามารถในด้านการพัฒนา อัลกอริทึม, การสร้างแบบจำลองทางคณิตศาสตร์ และอื่น ๆ ภายในโปรแกรมมีแอปพลิเคชันให้ผู้ใช้งานได้เลือกหลากหลาย หนึ่งในนั้น ได้แก่ Image Labeler เป็นแอปพลิเคชันที่เกี่ยวข้องกับการ ประมวลผลรูปภาพ (Image Processing)

ผู้ศึกษาได้ใช้แอปพลิเคชันดังกล่าวในการสร้างป้ายกำกับระดับพิกเซลให้กับรูปภาพที่จะถูกนำไปพัฒนาตัวแบบ

2.9.2) ไฟธอน 3 (Python3)

1) OpenCV

OpenCV เป็นไลบรารีส่วนชุดคำสั่งประเภทโอเพนซอร์ส (Open Source) พัฒนาโดยบริษัทอินเทล (Intel) ใช้สำหรับการเขียนโปรแกรมประเภทคอมพิวเตอร์วิทัค์ (Computer Vision) โดยการเพิ่มประสิทธิภาพของหน่วยประมวลผลกราฟิก (Graphics Processing Unit: GPU) สำหรับการทำงานแบบทันทีทันใด (Real Time)

2) เทนเซอร์ฟลว (TensorFlow)

แทนเซอร์ฟลวเป็นไลบรารีส่วนชุดคำสั่งประเภทโอเพนซอร์ส กล่าวคือฟรีไม่เสียค่าใช้จ่ายในการใช้งาน พัฒนาโดยบริษัทกูเกิล (Google) โดยมีวัตถุประสงค์คือ ช่วยในการเขียนโปรแกรมที่เกี่ยวข้องกับการพัฒนาตัวแบบการเรียนรู้ของเครื่อง และการเรียนรู้เชิงลึก

3) เคราส (Keras)

เคราสเป็นไลบรารีส่วนชุดคำสั่งประเภทโอเพนซอร์ส กล่าวคือฟรีไม่เสียค่าใช้จ่ายในการใช้งาน พัฒนาโดยบริษัทกูเกิลสำหรับการพัฒนาโครงข่ายประสาทเทียม สามารถทำงานร่วมกับแทนเซอร์ฟลว กล่าวคือเป็นองหลังของไลบรารีเคราสนั้นจะประกอบด้วย ไลบรารีแทนเซอร์ฟลว โดยมีวัตถุประสงค์คือ ให้ผู้ใช้สามารถพัฒนาตัวแบบด้วยการเรียนรู้เชิงลึกได้อย่างรวดเร็วและสะดวกมากยิ่งขึ้น

2.9.3) Django 4

Django เป็นเฟรมเวิร์ก (Framework) สำหรับการสร้างเว็บแอปพลิเคชัน ที่ถูกพัฒนาและดูแลโดย Django Software Foundation: DSF ซึ่งเป็นองค์กรอิสระที่ก่อตั้งขึ้นในสหรัฐอเมริกาในฐานะองค์กรไม่แสวงผลกำไร

วัตถุประสงค์ของ Django คือการทำให้การสร้างเว็บไซต์ที่ซับซ้อนและขับเคลื่อนด้วยฐานข้อมูลง่ายขึ้นตามหลักรูปแบบสถาปัตยกรรม Model–Template–Views: MTV โดยใช้ร่วมกับไพธอน (Python) แบบโอลเพนซอร์สฟรี

2.10) การทบทวนวรรณกรรม (Literature Review)

จากงานของ Yanming Guo, Yu Liu, Theodoros Georgiou และ Michael S. Lew (2017) ในหัวข้อ A review of semantic segmentation using deep neural networks ได้ศึกษาเกี่ยวกับความหมาย และวิธีต่าง ๆ ของการแบ่งส่วนความหมายของภาพ โดยผู้ศึกษาได้เรียนรู้ความหมาย และวิธีต่าง ๆ ของการแบ่งส่วนความหมายของภาพ และเลือกใช้วิธีที่เหมาะสมที่สุดในการดำเนินงาน



บทที่ 3 ชุดข้อมูลและการพัฒนาตัวแบบการเรียนรู้เชิงลึก

3.1) ลักษณะข้อมูล

ชุดข้อมูลที่นำมาใช้ในโครงการครั้งนี้เป็นชุดข้อมูลรูปภาพอาหารไทย ได้แก่ กะเพราไก่ราดข้าว, ข้าวคลุกกะปิ, ข้าวมันไก่, กวยเตี๋ยวbalance มีเหลืองแห้ง, และผัดไทย ซึ่งเป็นข้อมูลที่ไม่มีโครงสร้าง โดยมีที่มาจากการค้นหัวเวชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ชุดข้อมูลนี้ประกอบไปด้วย

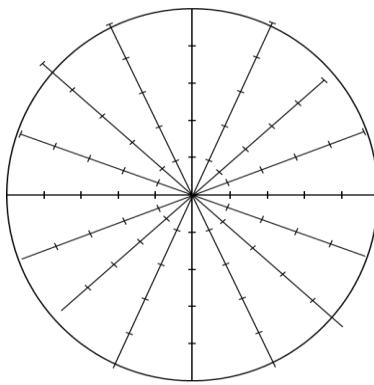
ตารางที่ 3.1.1 รายชื่อเมนูอาหาร และจำนวนรูปภาพที่ใช้พัฒนาตัวแบบครั้งที่ 1 และ 2

เมนูอาหาร	จำนวนรูปภาพ (พัฒนาครั้งที่ 1)	จำนวนรูปภาพ (พัฒนาครั้งที่ 2)
กะเพราไก่ราดข้าว	12	-
คลุกกะปิ	11	14
ข้าวมันไก่	13	16
กวยเตี๋ยวbalance มีเหลืองแห้ง	12	15
ผัดไทย	9	-

รูปภาพแต่ละรูปจะมีข้อมูลอัตราส่วนของโภชนาการที่วัดโดยการประมาณโดยใช้วงกลมอัตราส่วน ดังรูป

ภาพ	ชื่อ เมนูอาหาร	อตราส่วนของอาหารภายในจาน คิดจาก 100% (ขนาด 9 นิ้ว ห่างจากขอบ 1 นิ้ว ความสูง 3.5 เซนติเมตร)		
		ข้าวเปลือก	เนื้อสัตว์/ โปรตีน	ผัก
	ข้าวมันไก่	25.0	25.0	50.0

รูปที่ 3.1.1 รูปภาพเมนูข้าวมันไก่ และอัตราส่วนที่วัดโดยการประมาณโดยวงกลมอัตราส่วน

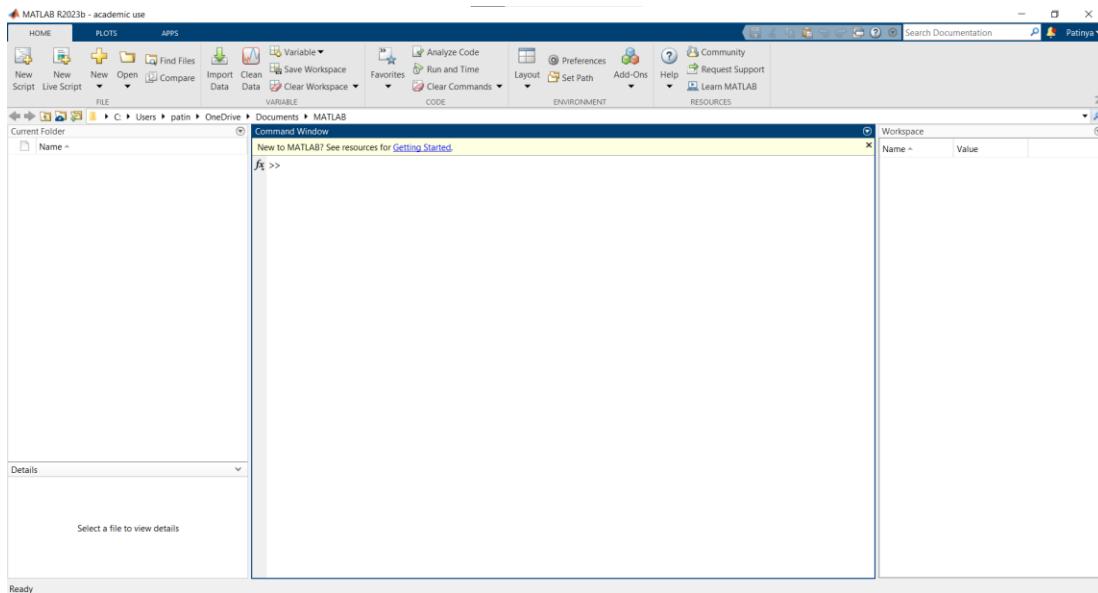


รูปที่ 3.1.2 เครื่องมือวงกลมอัตราส่วนที่ใช้ประมาณอัตราส่วนโภชนาการ

เนื่องจากอัตราส่วนโภชนาการที่วัดโดยวงกลมอัตราส่วนเป็นการประมาณ ดังนั้นผู้ศึกษาจึงมีความคิดว่าจะไม่นำอัตราส่วนโภชนาการที่มาใช้งาน และในการวัดอัตราส่วนโภชนาการจะใช้การนับจำนวนพิกเซลจากหน้ากากของรูปภาพแทน ซึ่งจะอธิบายในหัวข้อต่อไป

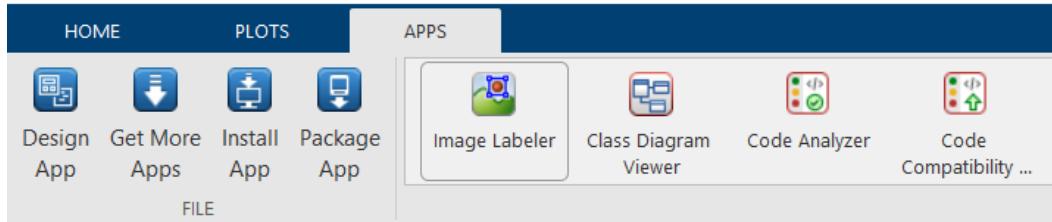
3.2) การสร้างหน้ากากของรูปภาพ และการคำนวณอัตราส่วนโภชนาการ

หน้ากากของรูปภาพใช้เป็นเป้าหมายในการพัฒนาตัวแบบการแบ่งส่วนความหมายของรูปภาพ โดยหนึ่งรูปภาพจะต้องมีหนึ่งหน้ากาก ในโครงงานครั้งนี้ ผู้ศึกษาได้ใช้ส่วนชุดคำสั่ง แมตแล็บ (Matrix Laboratory: MATLAB) ในการสร้างหน้ากากของแต่ละรูปภาพ



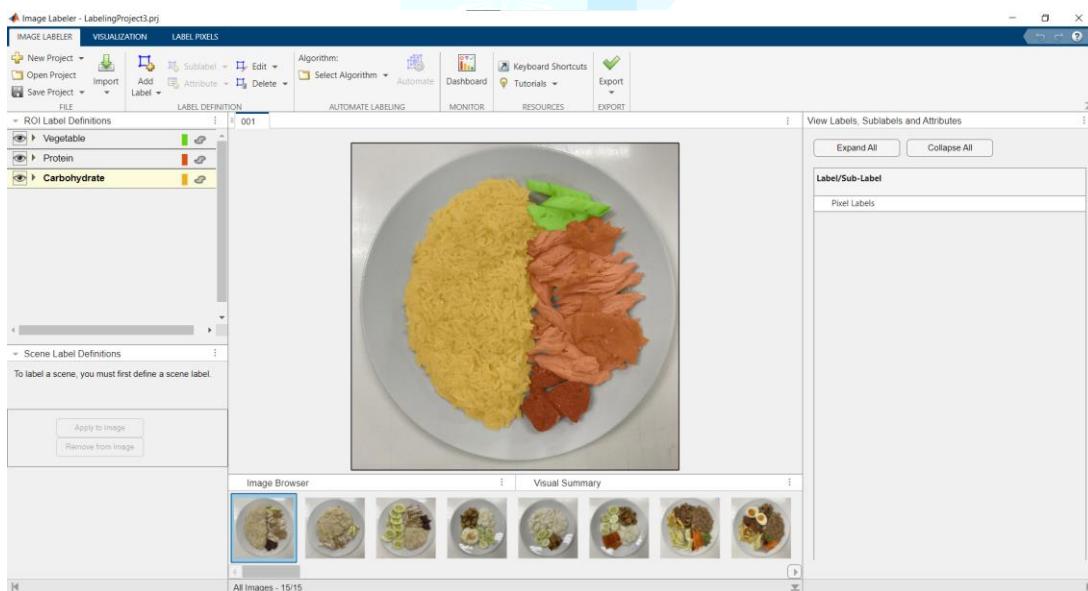
รูปที่ 3.2.1 หน้าจอของแมตแล็บ

ภายในส่วนชุดคำสั่งแมตแล็บมีแอปพลิเคชันต่าง ๆ ให้เลือกใช้งาน หนึ่งในนั้นคือ Image Labeler ซึ่งเป็นแอปพลิเคชันที่ใช้ในการทำงานเกี่ยวกับรูปภาพ เช่น สร้างหน้าหากา, วัดกรอบล้อมรอบวัตถุ เป็นต้น



รูปที่ 3.2.2 แท็บเครื่องมือของแมตแล็บ

ภายในแอปพลิเคชัน Image Labeler จะทำการเพิ่มเติมเข้าหมายการนำทางทั้งหมดที่เป็นไปได้ ได้แก่ ผัก (Vegetable), โปรตีน (Protein), และคาร์โบไฮเดรต (Carbohydrate) จากนั้นในแต่ละรูปภาพจะลากเส้นแบ่งส่วนตามเข้าหมายต่าง ๆ



รูปที่ 3.2.3 หน้าจอแอปพลิเคชัน Image Labeler

เมื่อลากเส้นแบ่งส่วนตามเข้าหมายต่าง ๆ แล้ว สามารถทำการนำออก (Export) หน้าหากาของรูปภาพได้ โดยหน้าหากาจะเป็นรูปภาพที่มีค่าสีของพิกเซลขึ้นอยู่กับจำนวนเข้าหมายการนำทางทั้งหมดที่เป็นไปได้ ได้แก่ ผัก, โปรตีน, และคาร์โบไฮเดรต หน้าหากาจะมีค่าสีของพิกเซล 4 ค่า ได้แก่ 0, 1, 2, และ 3 โดยค่า 0 หมายถึงพื้นที่ที่ไม่ได้จัดเป็นเข้าหมายใด ๆ ส่งผลให้มีอีกภาพหน้าหากาในส่วนชุดคำสั่งได้ ๆ จะเห็นภาพเป็นสีดำ

เราสามารถใช้ไลบรารี OpenCV และ Matplotlib ในภาษา Python ในการแสดงรูปภาพหน้ากากในชัดเจนได้

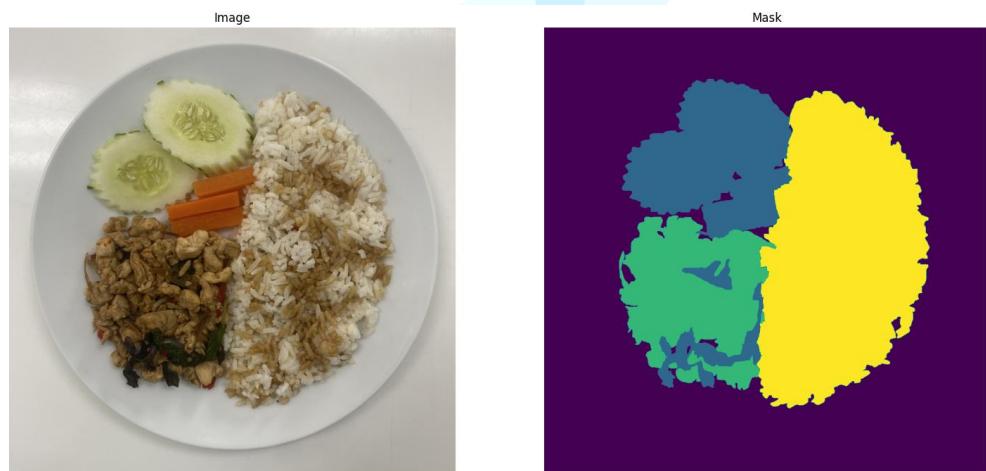
```
example_img = cv2.imread('Data/Images/1/I001/001-1.jpg', 1)
example_mask = cv2.imread('Data/Masks/1/L001/L001-1.png', 0)

plt.figure(figsize=(18, 12))

plt.subplot(121)
plt.imshow(cv2.cvtColor(example_img, cv2.COLOR_BGR2RGB))
plt.title('Image')
plt.axis('off')

plt.subplot(122)
plt.imshow(example_mask)
plt.title('Mask')
plt.axis('off')
```

รูปที่ 3.2.5 คำสั่งที่ใช้ในการแสดงรูปภาพอาหารและหน้ากากของรูปภาพนั้น



รูปที่ 3.2.6 ผลลัพธ์ของคำสั่งในรูปที่ 3.2.5 โดยสีน้ำเงินหมายถึงผัก สีเหลืองหมายถึง蛋白质 และสีเขียวหมายถึงโปรตีน

ในการคำนวณอัตราส่วนโภชนาการสามารถทำได้โดยการนับจำนวนพิกเซลของเป้าหมายการท่านายที่เป็นไปได้ของหน้ากาก ได้แก่ ผัก, โปรตีน, และคาร์โบไฮเดรต ดังรูปแบบการหาอัตราส่วนโปรตีนด้านล่าง

$$\text{อัตราส่วนโปรตีน} = \frac{\text{จำนวนพิกเซลโปรตีน}}{\text{จำนวนพิกเซลผัก} + \text{จำนวนพิกเซลโปรตีน} + \text{จำนวนพิกเซลคาร์โบไฮเดรต}}$$

เราสามารถใช้ไลบรารี Numpy ในภาษา Python ในการคำนวณอัตราส่วนโภชนาการได้

```

def calculateProportion(mask):
    total_proportion = 0
    segment_areas = {}
    categories = ['Carbohydrate', 'Protein', 'Vegetable']

    for label in np.unique(mask):
        area = np.sum(mask == label)
        segment_areas[label] = area

    segment_areas = {key: segment_areas[key] for key in [1, 2, 3]}

    total_area = sum(segment_areas.values())

    for label, area in segment_areas.items():
        proportion = (area / total_area) * 100
        total_proportion += proportion
        print(f'{categories[label - 1]} Proportion = {proportion:.2f}')

    print(f'Total Proportion = {total_proportion}')

```

รูปที่ 3.2.7 พังก์ชันคำสั่งที่ใช้ในการคำนวณอัตราส่วนโภชนาการได้

Carbohydrate Proportion = 27.71
Protein Proportion = 21.24
Vegetable Proportion = 51.05
Total Proportion = 100.0

รูปที่ 3.2.8 ผลลัพธ์ของคำสั่งในรูปที่ 3.2.7 ในการคำนวณอัตราส่วนโภชนาการของรูปภาพในรูปที่ 3.2.6

เมื่อนำอัตราส่วนที่คำนวณจากหน้ากากมาเปรียบเทียบกับอัตราส่วนเดิมที่คณะสหเวชศาสตร์จัดเตรียมให้ผ่านพังก์ชัน Mean Absolute Error (MAE) ได้ผลลัพธ์ดังตาราง

ตารางที่ 3.2.1 การเปรียบเทียบอัตราส่วนผ่านพังก์ชัน Mean Absolute Error (MAE)

Mean Absolute Error (MAE)			
เมนูอาหาร	คาร์โบไฮเดรต	โปรตีน	ผัก
กะเพราไก่ราดข้าว	2.13	8.21	3.15
ข้าวคลุกกะปิ	2.57	1.14	1.75
ข้าวมันไก่	7.57	9.24	2.58
ก๋วยเตี๋ยวบะหมี่เหลืองแห้ง	1.80	2.12	1.36
ผัดไทย	3.30	2.46	3.39

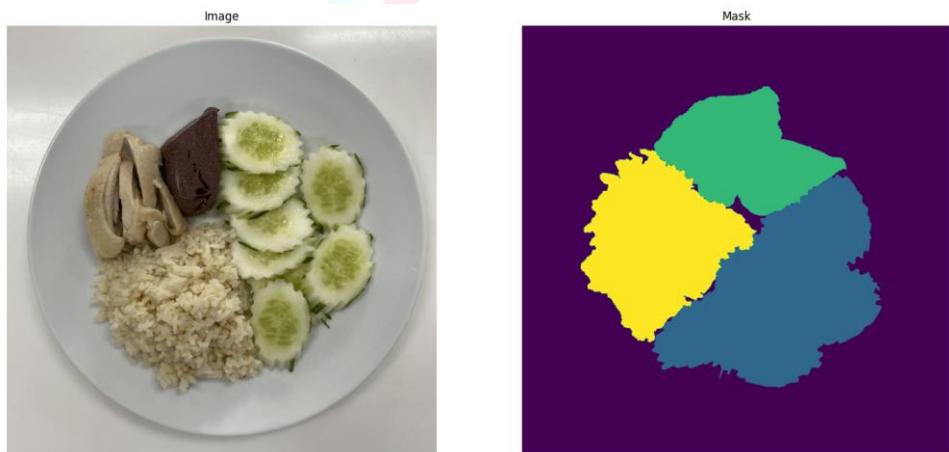
3.3) การเพิ่มความหลากหลายของข้อมูล

การเพิ่มความหลากหลายของข้อมูลเป็นเทคนิคที่จะสร้างข้อมูลอย่างสุ่มจากข้อมูลที่มีอยู่ โดยจะกำหนดให้อัลกอริทึมสร้างข้อมูลขึ้นมาใหม่ตามคุณลักษณะที่กำหนด ตัวอย่างเช่น ข้อมูลประเทกธุรูปภาพ สามารถกำหนดคุณลักษณะเพื่อสร้างข้อมูลใหม่ เช่น การบิดภาพ, การหมุนภาพ, การย่อหรือขยายภาพ, การเปลี่ยนสีภาพ และการตัดบางส่วนของภาพออกไป เป็นต้น

ในการเพิ่มความหลากหลายของข้อมูล ผู้ศึกษาได้เลือกใช้ไลบรารี OpenCV โดยแบ่งการเพิ่มความหลากหลายของข้อมูลออกเป็น 2 ส่วน ได้แก่

3.3.1) การย้ายตำแหน่งพิกเซล

การย้ายตำแหน่งพิกเซลของรูปภาพสามารถทำได้โดย การหมุน (Rotation), การบิด (Distortion), การสะท้อน (Reflection) เป็นต้น ใน การเปลี่ยนแปลงตำแหน่งพิกเซลของรูปภาพ ดังเดิม หากหน้าหากของรูปภาพนั้นไม่ถูกเปลี่ยนแปลงตำแหน่งของพิกเซลไปด้วยจะส่งผลให้ตำแหน่งการอธิบายเป้าหมายการทำนายทั้งหมดที่เป็นไปได้ของหน้าหากผิดพลาด ดังนั้นมีการเปลี่ยนแปลงตำแหน่งพิกเซลของรูปภาพใด ๆ จะต้องมีการเปลี่ยนแปลงตำแหน่งพิกเซลของหน้าหากของรูปภาพนั้น ๆ ด้วย



รูปที่ 3.3.1.1 ผลลัพธ์การย้ายตำแหน่งพิกเซลของรูปภาพ และไม่ย้ายตำแหน่งพิกเซลหน้าหาก

ผู้ศึกษาได้สร้างฟังก์ชัน `adjust_angle` ในไฟรอน โดยฟังก์ชันดังกล่าวทำหน้าที่ย้ายตำแหน่งพิกเซลของรูปภาพและหน้าหาก โดยมีองค์ประกอบดังนี้

```

def adjust_angle(image, mask):
    rotation_angle = np.random.randint(0, 360)
    rotation_matrix = cv2.getRotationMatrix2D((image.shape[1] // 2, image.shape[0] // 2), rotation_angle, 1.0)
    image_augmented = cv2.warpAffine(image, rotation_matrix, image.shape[1:-1], flags=cv2.INTER_LINEAR)
    mask_augmented = cv2.warpAffine(mask, rotation_matrix, mask.shape[1:-1], flags=cv2.INTER_NEAREST)

    flip = np.random.choice([0, 1])
    if flip == 1:
        image_augmented = cv2.flip(image_augmented, 1)
        mask_augmented = cv2.flip(mask_augmented, 1)

    return image_augmented, mask_augmented

```

รูปที่ 3.3.1.2 พังก์ชันคำสั่งที่ใช้ในการหมุนและการสะท้อน

1) การหมุนภาพ

เริ่มต้นจากการสุ่มค่าระหว่าง 0 และ 360 เพื่อใช้เป็นมุมองศาในการหมุน โดยเก็บไว้ในตัวแปร `rotation_angle` ต่อมาก็จะใช้เมทริกซ์การหมุนผ่าน `getRotationMatrix2D` ซึ่งเป็นพังก์ชันจากไลบรารี OpenCV จากนั้นใช้เมทริกซ์การหมุนผ่าน `warpAffine` ซึ่งเป็นพังก์ชันจากไลบรารี OpenCV เพื่อหมุนรูปภาพและหน้ากากตามมุมองศาที่ได้สุ่มขึ้นมา



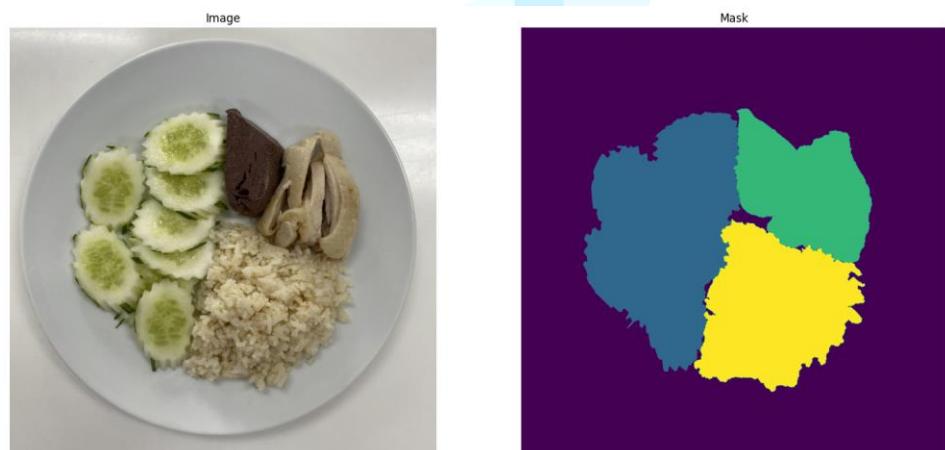
รูปที่ 3.3.1.3 ผลลัพธ์จากการหมุนภาพ

2) การสะท้อนภาพ

เริ่มต้นจากการสุ่มค่า 0 หรือ 1 เพื่อใช้เป็นตัวแปรกำหนดว่าจะมีการสะท้อนภาพหรือไม่ โดยเก็บไว้ในตัวแปร `flip` หาก `flip` มีค่าเท่ากับ 0 หมายความว่าจะไม่มีการสะท้อนภาพ และหาก `flip` มีค่าเท่ากับ 1 หมายความว่าจะมีการสะท้อนภาพ โดยการสะท้อนผ่าน `cv2.flip` ซึ่งเป็นพังก์ชันจากไลบรารี OpenCV



รูปที่ 3.3.1.4 รูปภาพและหน้ากากก่อนการสะท้อน



รูปที่ 3.3.1.5 ผลลัพธ์จากการสะท้อนรูปภาพจากรูปภาพและหน้ากากในรูปที่ 3.3.1.4

3.3.2) การปรับค่าพิกเซล

การปรับค่าพิกเซลของรูปภาพสามารถทำได้โดย การปรับอุณหภูมิ (Temperature), การปรับคอนทราสท์ (Contrast), การปรับความสว่าง (Brightness), การความคมชัด (Sharpness), การปรับความอิมตัว (Saturation) เป็นต้น เนื่องจากการปรับค่าพิกเซลไม่ได้มีการย้ายตำแหน่งพิกเซลแต่อย่างใด ดังนั้นจึงไม่ต้องปรับค่าพิกเซลของหน้ากาก

ผู้ศึกษาได้สร้างฟังก์ชัน `adjust_temperature` สำหรับการปรับอุณหภูมิ และ `adjust_freshness` สำหรับการปรับ คอนทราสท์, ความสว่าง, ความคมชัด, และความอิมตัว ในไฟชอน โดยทั้งสองฟังก์ชันทำหน้าที่ปรับค่าพิกเซลของรูปภาพ โดยมีองค์ประกอบดังนี้

```

def adjust_temperature(image):
    temperature_factor = np.random.uniform(1.1, 1.2)
    image[:, :, 0] = np.clip(image[:, :, 0] * temperature_factor, 0, 255)
    image[:, :, 2] = np.clip(image[:, :, 2] / temperature_factor, 0, 255)

    return image

```

รูปที่ 3.3.2.1 พังก์ชันคำสั่งที่ใช้ในการปรับอุณหภูมิ

```

def adjust_freshness(image):
    image = cv2.convertScaleAbs(image, alpha=1.2, beta=10)

    kernel_sharpening = np.array([[0, -1, 0],
                                  [-1, 5, -1],
                                  [0, -1, 0]])
    image = cv2.filter2D(image, -1, kernel_sharpening)

    image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    image[:, :, 1] = np.clip(image[:, :, 1] * 1.2, 0, 255)
    image = cv2.cvtColor(image, cv2.COLOR_HSV2BGR)

    return image

```

รูปที่ 3.3.2.2 พังก์ชันคำสั่งที่ใช้ในการปรับคุณภาพรูปภาพ ความสว่าง ความคมชัด และความอิ่มตัว

1) การปรับอุณหภูมิ

ผู้ศึกษาได้กำหนดให้มีการปรับอุณหภูมิของภาพให้มีความเย็นขึ้น กล่าวคือรูปภาพจะมีโทนสีฟ้ามากขึ้น เริ่มต้นจากการสุ่มค่าที่อยู่ระหว่าง 1.1 และ 1.2 เพื่อใช้เป็นปัจจัยในการปรับอุณหภูมิ โดยเก็บไว้ในตัวแปร `temperature_factor` จากนั้นจะนำค่าดังกล่าวมาคูณกับพิกเซลในช่องสีฟ้าและหารในช่องสีแดง การดำเนินการทางคณิตศาสตร์ดังกล่าวจะทำให้รูปภาพจะมีโทนสีฟ้ามากขึ้น นอกจากนี้ยังมีการใช้ `clip` ซึ่งเป็นพังก์ชันจากไลบรารี Numpy ทำหน้าที่ตรวจสอบหากผลการคูณหรือหาร โดยค่าที่น้อยกว่า 0 หรือมากกว่า 255 จะถูกปรับค่าเป็น 0 หรือ 255 ตามลำดับ



รูปที่ 3.3.2.3 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการปรับอุณหภูมิ (ขวา) ให้มีโทนเย็นมากขึ้น

2) การปรับค่อนทรายและความสว่าง

ผู้จัดได้กำหนดให้มีการเพิ่มค่อนทรายและความสว่าง โดยใช้ `convertScaleAbs` ซึ่งเป็นฟังก์ชันจากไลบรารี OpenCV โดยมีพารามิเตอร์ที่สำคัญสองตัว ได้แก่ `alpha` ที่ทำหน้าที่ควบคุมค่อนทราย หากเรากำหนดค่าอยู่ในช่วงปิด 0 ถึง 1 นั้นหมายถึงการลดค่อนทราย แต่หากเรามากกว่า 1 นั้นหมายถึงการเพิ่มค่อนทราย และ `beta` ที่ทำหน้าที่ควบคุมความสว่าง โดยสามารถมีค่าอยู่ระหว่าง -127 และ 127

ในการปรับค่อนทรายและความสว่าง ผู้จัดได้กำหนดให้มีการเพิ่มค่อนทรายและความสว่าง โดยกำหนด `alpha` เท่ากับ 1.2 และ `beta` เท่ากับ 10



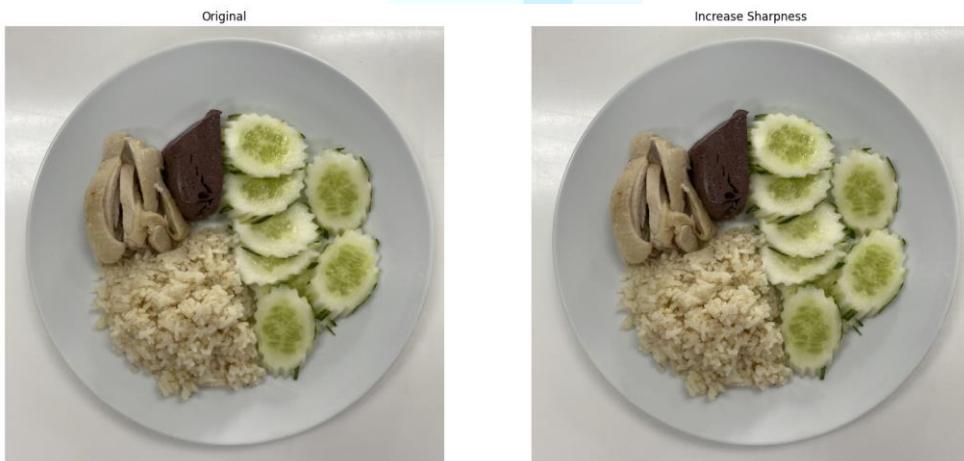
รูปที่ 3.3.2.4 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการเพิ่มค่อนทรายและความสว่าง (ขวา)

3) การปรับความคมชัด

การปรับความคมชัดของรูปภาพสามารถทำได้โดยนำรูปภาพมาดำเนินการผ่านตัวกรองที่เรียกว่า Sliding Window (Kernel) โดยการดำเนินการดังกล่าวมีความคล้ายกับการดำเนินการในขั้นตอนคอนโวลูชัน ผลลัพธ์จากการดำเนินการจะไม่ใช้ผังคุณลักษณะแต่อย่างใด แต่จะเป็นรูปภาพเดิมที่มีการปรับค่าพิกเซลตามลักษณะของเมทริกซ์ตัวกรอง โดยในการเพิ่มความคมชัดจะมีเมทริกซ์ตัวกรอง ดังนี้

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

โดยจะใช้ filter2D ซึ่งเป็นฟังก์ชันจากไลบรารี OpenCV ในการดำเนินการปรับความคมชัดของรูปภาพ



รูปที่ 3.3.2.5 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการเพิ่มความคมชัด (ขวา)

4) การปรับความอิ่มตัว

การปรับความอิ่มตัวของรูปภาพสามารถทำได้โดยแปลงรูปแบบของรูปภาพจาก BGR (Blue, Green, Red) เป็น HSV (Hue, Saturation, Value) โดยใช้ convertScaleAbs ซึ่งเป็นฟังก์ชันจากไลบรารี OpenCV ในการแปลงรูปแบบ ต่อมาก็จะเพิ่มความอิ่มตัวของรูปภาพโดยนำค่า 1.2 คูณกับช่องความอิ่มตัว โดยมีการใช้ clip ซึ่งเป็นฟังก์ชันจากไลบรารี Numpy ทำหน้าที่ตรวจสอบหากผลการคูณหรือหาร โดยค่าที่น้อยกว่า 0 หรือมากกว่า 255 จะถูกปรับค่าเป็น 0 หรือ 255 ตามลำดับ จากนั้นจะแปลงรูปแบบของรูปภาพจาก HSV กลับมาเป็น BGR



รูปที่ 3.3.2.6 รูปภาพเดิม (ซ้าย) กับผลลัพธ์จากการเพิ่มความอิ่มตัว (ขวา)

ในโครงการครั้งนี้ ผู้ศึกษาได้พัฒนาตัวแบบ 2 ครั้ง การพัฒนาครั้งแรกได้มีการเพิ่มความหลากหลายของข้อมูลโดยการย้ายตำแหน่งของพิกเซลเท่านั้น โดย 1 รูปภาพจะถูกสุ่มย้ายตำแหน่งของพิกเซลและสร้างเป็นรูปภาพ 9 ภาพ แต่ในการพัฒนาครั้งที่สองได้มีการเพิ่มความหลากหลายของข้อมูลทั้งการย้ายตำแหน่งของพิกเซลและการปรับค่าพิกเซล เพื่อให้ตัวแบบสามารถทำนายอัตราส่วนของรูปภาพที่ถูกถ่ายในสภาพแวดล้อมที่แตกต่างกันได้ดียิ่งขึ้น โดย 1 รูปภาพจะถูกสุ่มย้ายตำแหน่งของพิกเซลและสร้างเป็นรูปภาพ 14 ภาพ และ 14 รูปภาพจะถูกแบ่งออกเป็น 3 ส่วนในการปรับค่าพิกเซล ได้แก่

ตารางที่ 3.3.1 จำนวนของรูปภาพและวิธีการปรับค่าพิกเซล

	ไม่มีการปรับค่าพิกเซล	ปรับอุณหภูมิ	ปรับคอนทราสท์, ความสว่าง, ความคมชัด, และความอิ่มตัว
จำนวนภาพ	4	5	5

```
def augmentData(image, mask, n_samples, index):
    for i in range(n_samples):
        image_adjusted = image.copy()

        if i >= 9:
            image_adjusted = adjust_temperature(image_adjusted)
        elif i >= 4:
            image_adjusted = adjust_freshness(image_adjusted)

        image_augmented, mask_augmented = adjust_angle(image_adjusted, mask)

        files_length = len([f for f in os.listdir(f'./Data/Images/1/I{index}')])
        cv2.imwrite(f'./Data/Images/1/I{index}/{files_length + 1}.jpg', image_augmented)
        cv2.imwrite(f'./Data/Masks/1/L{index}/{files_length + 1}.png', mask_augmented)
```

รูปที่ 3.3.1 พิ้งก์ชันคำสั่งที่ใช้ในการเพิ่มความหลากหลายของข้อมูล

3.4) การนำเข้าข้อมูล

ในการพัฒนาตัวแบบ ผู้ศึกษาจะพัฒนาตัวแบบแยกตามแต่ละเมนูอาหาร เพื่อเพิ่มความเร็วในการพัฒนา และประสิทธิภาพของผลลัพธ์ โดยการพัฒนาจะถูกแบ่งออกเป็น 2 ครั้ง โดยมีรายละเอียดดังนี้

1) การพัฒนาครั้งที่ 1

เมนูอาหารที่ใช้ในการพัฒนาตัวแบบครั้งที่ 1 ประกอบไปด้วย ข้าวกระเพรา, ข้าวคลุกกะปิ, ข้าวมันไก่, ก๋วยเตี๋ยวเส้นบางมีเหลืองแห้ง, และผัดไทย และมีการเพิ่มความหลากหลายของรูปภาพ ได้แก่ การย้ายตำแหน่งพิกเซล โดยหนึ่งรูปภาพจะถูกเพิ่มความหลากหลายออกเป็น 9 รูปภาพ

2) การพัฒนาครั้งที่ 2

เมนูอาหารที่ใช้ในการพัฒนาตัวแบบครั้งที่ 2 ประกอบไปด้วย ข้าวคลุกกะปิ, ข้าวมันไก่, และก๋วยเตี๋ยวเส้นบางมีเหลืองแห้ง โดยมีการเพิ่มจำนวนรูปภาพตันของแต่ละเมนูจำนวน 3 รูปภาพ และมีการเพิ่มความหลากหลายของรูปภาพ ได้แก่ การย้ายตำแหน่งพิกเซล และการปรับค่าพิกเซล โดยหนึ่งรูปภาพจะถูกเพิ่มความหลากหลายออกเป็น 14 รูปภาพ

โครงสร้างของโฟลเดอร์เก็บรูปภาพประกอบไปด้วย 5 โฟลเดอร์ แต่ละโฟลเดอร์ประกอบไปด้วย โฟลเดอร์ย่อยตามจำนวนของรูปภาพแต่ละเมนู และภายในโฟลเดอร์ย่อยประกอบไปด้วยรูปภาพหลักและรูปภาพที่ถูกเพิ่มความหลากหลาย โดยโครงสร้างของโฟลเดอร์เก็บหน้ากากจะเหมือนกับโครงสร้างของโฟลเดอร์เก็บรูปภาพ สามารถสรุปได้ดังนี้

ตารางที่ 3.4.1 โครงสร้างของโฟลเดอร์เก็บรูปภาพ

ชื่อโฟลเดอร์	คำอธิบาย (เมนูอาหาร)	พัฒนาครั้งที่ 1		พัฒนาครั้งที่ 2	
		จำนวน โฟลเดอร์ย่อย	จำนวน รูปภาพ	จำนวน โฟลเดอร์ย่อย	จำนวน รูปภาพ
1	ข้าวกระเพรา	12	120	-	-
2	ข้าวคลุกกะปิ	11	110	14	210
3	ข้าวในไก่	13	130	16	240
4	ก๋วยเตี๋ยวเส้น บางมีเหลืองแห้ง	12	120	15	225
5	ผัดไทย	9	90	-	-

-  1
-  2
-  3
-  4
-  5

รูปที่ 3.4.1 โครงสร้างโฟลเดอร์เก็บรูปภาพ

-  I001
-  I002
-  I003
-  I004
-  I005
-  I006
-  I007
-  I008
-  I009
-  I010
-  I011
-  I012

รูปที่ 3.4.2 โครงสร้างโฟลเดอร์ย่อย



รูปที่ 3.4.3 รูปภาพในโฟลเดอร์ย่อย

ในการนำเข้ารูปภาพและนำ回去ผู้ศึกษาได้สร้างฟังก์ชัน `loadImageAndMask` ในไฟร่อน โดยมีการกำหนดตัวแปรนำเข้า (`Input Variable`) เป็นชื่อของโฟลเดอร์ที่เก็บรูปภาพ ดังนี้

```

def loadImageAndMask(folder):
    X_train = []
    X_val = []
    X_test = []
    y_train = []
    y_val = []
    y_test = []
    for i in range(0, len([f for f in os.listdir(f'C:\GitHub\Senior_Project\Senior Project\Data\Images\\{folder}')])):
        index = i + 1
        if len(str(index)) == 1:
            index = '0' + str(index)
        else:
            index = str(index)
    for j in range(len([f for f in os.listdir(f'C:\GitHub\Senior_Project\Senior Project\Data\Images\\{folder}\I{index}')])):
        img = cv2.imread(f'./Data/Images/{folder}/I{index}/{0}{index}-{(j+1)}.jpg', 1)
        mask = cv2.imread(f'./Data/Masks/{folder}/L{index}/L{index}-{(j+1)}.png', 0)
        img, mask = resize(img, mask)

        if i <= len([f for f in os.listdir(f'C:\GitHub\Senior_Project\Senior Project\Data\Images\\{folder}')]) - 5:
            X_train.append(img)
            y_train.append(mask)
        elif i <= len([f for f in os.listdir(f'C:\GitHub\Senior_Project\Senior Project\Data\Images\\{folder}')]) - 3:
            X_val.append(img)
            y_val.append(mask)
        else:
            X_test.append(img)
            y_test.append(mask)
    X_train = np.array(X_train)
    X_val = np.array(X_val)
    X_test = np.array(X_test)
    y_train = np.array(y_train)
    y_val = np.array(y_val)
    y_test = np.array(y_test)
    return X_train, X_val, X_test, y_train, y_val, y_test

```

รูปที่ 3.4.4 พังก์ชันคำสั่งที่ใช้ในการนำเข้ารูปภาพและหน้ากาก

รูปภาพและหน้ากากที่ถูกนำเข้าจะถูกลดขนาดลงเหลือ 128x128 จากเดิม 3024x3024 ผ่านพังก์ชัน resize นอกจากนี้ค่าพิกเซลของรูปภาพจะถูกปรับให้เป็นมาตรฐาน (Standardization) โดยหารด้วย 255

```

def resize(img, mask):
    img = cv2.resize(img, (128, 128)) / 255
    mask = cv2.resize(mask, (128, 128))
    return img, mask

```

รูปที่ 3.4.5 พังก์ชันคำสั่งที่ใช้ในการลดขนาดและปรับให้เป็นมาตรฐาน

ในพังก์ชันการนำเข้ารูปภาพและหน้ากากจะถูกจัดหมวดหมู่ออกเป็น 3 ประเภท ได้แก่ ชุดฝึก (X_train, y_train), ชุดตรวจทาน (X_val, y_val), และชุดทดสอบ (X_test, y_test) โดยสัดส่วนการแบ่งจะคิดตามความเหมาะสมของจำนวนไฟล์เดอร์อยู่ที่ใช้เก็บรูปภาพแต่ละเมนู ดังนี้

ตารางที่ 3.4.2 สัดส่วนการแบ่งหมวดหมู่ (การพัฒนาครั้งที่ 1)

เมนูอาหาร	ชุดฝึก (จำนวนไฟล์เดอร์)	ชุดตรวจสอบ (จำนวนไฟล์เดอร์)	ชุดทดสอบ (จำนวนไฟล์เดอร์)
ข้าวกระแสฯ	8	2	2
ข้าวมันไก่	7	2	2
ข้าวคลุกกะปิ	9	2	2
ก๋วยเตี๋ยวเส้นบางมี เหลืองแห้ง	8	2	2
ผัดไทย	7	1	1

ตารางที่ 3.4.3 สัดส่วนการแบ่งหมวดหมู่ (การพัฒนาครั้งที่ 2)

เมนูอาหาร	ชุดฝึก (จำนวนไฟล์เดอร์)	ชุดตรวจสอบ (จำนวนไฟล์เดอร์)	ชุดทดสอบ (จำนวนไฟล์เดอร์)
ข้าวมันไก่	10	2	2
ข้าวคลุกกะปิ	12	2	2
ก๋วยเตี๋ยวเส้นบางมี เหลืองแห้ง	11	2	2

ผลลัพธ์จากการเรียกใช้ฟังก์ชันนำเข้าข้อมูลคือ อาเรย์ (Array) ที่เก็บข้อมูลของรูปภาพแยกตามหมวดหมู่

```
x_train, x_val, x_test, y_train, y_val, y_test = loadImageAndMask('3')

print(f'x_train shape: {x_train.shape}, Type: {type(x_train)}')
print(f'x_val shape: {x_val.shape}, Type: {type(x_test)}')
print(f'x_test shape: {x_test.shape}, Type: {type(x_test)}')
print(f'y_train shape: {y_train.shape}, Type: {type(y_train)}')
print(f'y_val shape: {y_val.shape}, Type: {type(y_test)}')
print(f'y_test shape: {y_test.shape}, Type: {type(y_test)}')
✓ 39.1s

x_train shape: (180, 128, 128, 3), Type: <class 'numpy.ndarray'>
x_val shape: (30, 128, 128, 3), Type: <class 'numpy.ndarray'>
x_test shape: (30, 128, 128, 3), Type: <class 'numpy.ndarray'>
y_train shape: (180, 128, 128), Type: <class 'numpy.ndarray'>
y_val shape: (30, 128, 128), Type: <class 'numpy.ndarray'>
y_test shape: (30, 128, 128), Type: <class 'numpy.ndarray'>
```

รูปที่ 3.4.6 ผลลัพธ์จากการเรียกใช้ฟังก์ชันนำเข้าข้อมูลจากไฟล์เดอร์ 3 (ข้าวมันไก่)

เมื่อลองแสดงค่าอาเรย์ จะเห็นค่าพิกเซลของรูปภาพ

```
x_train
✓ 0.0s

array([[[[0.71764706, 0.7372549 , 0.74901961],
         [0.71372549, 0.73333333, 0.74509804],
         [0.7254902 , 0.74509804, 0.75686275],
         ...,
         [0.83529412, 0.84705882, 0.8627451 ],
         [0.81568627, 0.83529412, 0.84705882],
         [0.83137255, 0.85098039, 0.8627451 ]],

        [[0.71764706, 0.7372549 , 0.74901961],
         [0.70196078, 0.72156863, 0.73333333],
         [0.71372549, 0.73333333, 0.74509804],
         ...,
         [0.83529412, 0.85490196, 0.86666667],
         [0.84313725, 0.8627451 , 0.8745098 ],
         [0.82745098, 0.85098039, 0.85882353]],

        [[0.70980392, 0.72941176, 0.74117647],
         [0.71372549, 0.7372549 , 0.74509804],
         [0.71372549, 0.73333333, 0.74509804],
         ...,
         [0.8627451 , 0.88235294, 0.89411765],
         [0.85490196, 0.87843137, 0.88627451],
         [0.82745098, 0.84705882, 0.85882353]],

        ...,
```

รูปที่ 3.4.7 ค่าพิกเซลที่ถูกจัดเก็บในอาเรย์

3.5) การประมวลผลข้อมูลก่อนพัฒนาตัวแบบ (Data Preprocessing)

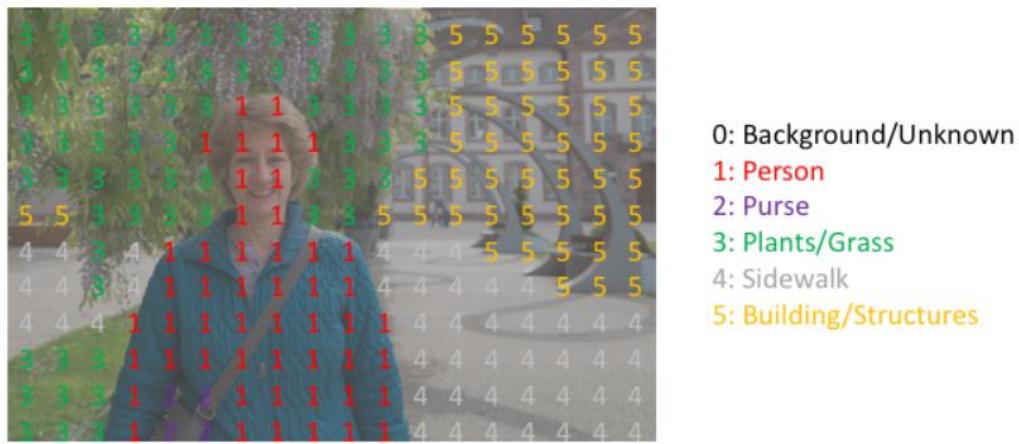
การประมวลผลข้อมูลก่อนพัฒนาตัวแบบแบ่งออกเป็น 2 ส่วนได้แก่

1) การประมวลผลรูปภาพ

การประมวลผลรูปภาพเป็นขั้นตอนที่ดำเนินการผ่านฟังก์ชันการนำเข้าข้อมูล “`train_datagen`” ได้แก่ การลดขนาดรูปภาพ และการปรับให้เป็นมาตรฐาน ดังรูปที่ 3.4.5

2) การประมวลผลหน้ากาก

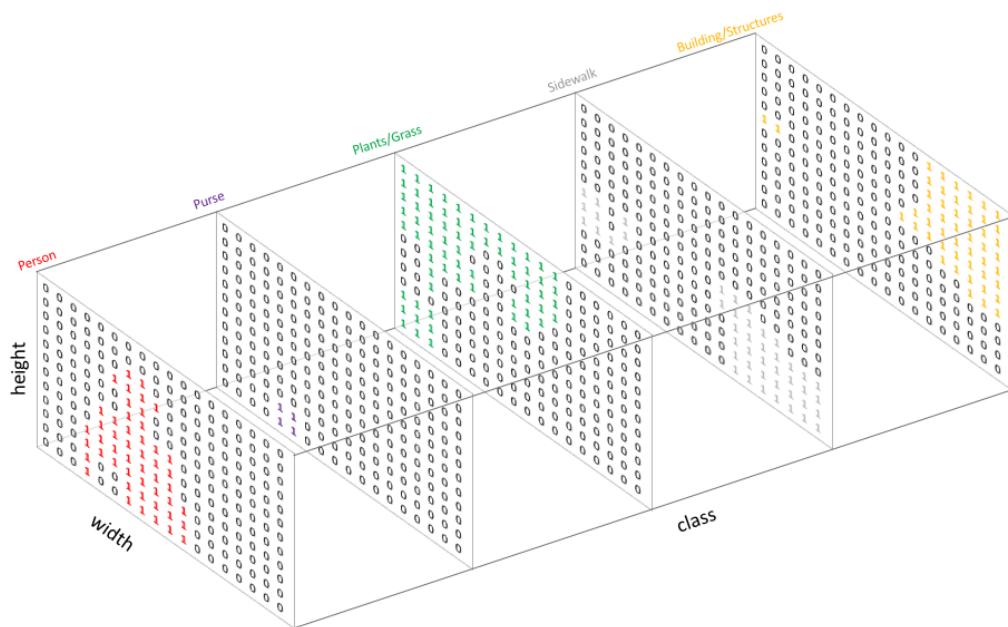
หน้ากากที่ถูกนำเข้าจะอยู่ในรูปแบบ 1 มิติ กล่าวคือค่าของพิกเซลจะแสดงถึงเป้าหมายการทํานายทั้งหมดที่เป็นไปได้



รูปที่ 3.5.1 ค่าพิกเซลของหน้ากากที่บ่งบอกถึงเป้าหมายการทำนายทั้งหมดที่เป็นไปได้

ที่มา: <https://www.jeremyjordan.me/semantic-segmentation/>

การที่จะนำหน้ากากไปพัฒนาตัวแบบควบคู่กับรูปภาพได้จะต้องผ่านกระบวนการ One-Hot encoding ก่อให้เกิดความซ้ำซ้อนในข้อมูลเดิม แต่จะช่วยให้เกิดจุดเด่นของภาพที่สำคัญ เช่น ใบไม้ หรือ บ้าน ให้เด่นชัดขึ้น ทำให้การเรียนรู้ของเครื่องจักรสามารถจับจังหวะของภาพได้ดีขึ้น



รูปที่ 3.5.2 ผลลัพธ์จากการทำ One-Hot encoding

ที่มา: <https://www.jeremyjordan.me/semantic-segmentation/>

ผู้ศึกษาได้สร้างฟังก์ชัน oneHotEncoder สำหรับการทำ One-Hot encoding ของหน้ากากได้ ดังนี้

```
def oneHotEncoder(masks):
    np.expand_dims(masks, axis=3)
    mask_oneHot = to_categorical(masks, num_classes=4)
    return mask_oneHot
```

รูปที่ 3.5.3 ฟังก์ชันคำสั่งการทำ One-Hot encoding

โดยอาศัยของหน้ากากจากการนำเข้าจะมีลักษณะเป็น (จำนวนข้อมูล, ความกว้าง, ความยาว) ทำให้ต้องใช้ expand_dims เปลี่ยนลักษณะเป็น (จำนวนข้อมูล, ความกว้าง, ความยาว, มิติ) หลังจากนั้นจะใช้ฟังก์ชัน to_categorical ซึ่งเป็นฟังก์ชันจากไลบรารี keras ในการทำ One-Hot encoding โดยผลลัพธ์สุดท้ายมิติของอาเรย์หน้ากากจะถูกขยายออกตามจำนวนจำานวนเป้าหมายการทำนายทั้งหมดที่เป็นไปได้

3.6) การพัฒนาตัวแบบ

```
def unet_vgg16(n_classes, IMG_HEIGHT, IMG_WIDTH, IMG_CHANNELS):
    inputs = Input((IMG_HEIGHT, IMG_WIDTH, IMG_CHANNELS))
    # ชั้นที่ 1
    c1 = Conv2D(64, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(inputs)
    c1 = Conv2D(64, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c1)
    p1 = MaxPool2D(2)(c1)
    p1 = Dropout(0.3)(p1)
    # ชั้นที่ 2
    c2 = Conv2D(128, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(p1)
    c2 = Conv2D(128, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c2)
    p2 = MaxPool2D(2)(c2)
    p2 = Dropout(0.3)(p2)
    # ชั้นที่ 3
    c3 = Conv2D(256, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(p2)
    c3 = Conv2D(256, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c3)
    c3 = Conv2D(256, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c3)
    p3 = MaxPool2D(2)(c3)
    p3 = Dropout(0.3)(p3)
    # ชั้นที่ 4
    c4 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(p3)
    c4 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c4)
    c4 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c4)
    p4 = MaxPool2D(2)(c4)
    p4 = Dropout(0.3)(p4)
    # ชั้นที่ 5
    c5 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(p4)
    c5 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c5)
    c5 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c5)
```

รูปที่ 3.6.1 สถาปัตยกรรมโครงข่าย U-Net ที่มี VGG16 เป็นกระดูกสันหลัง (ส่วนการเข้ารหัส)

```

# ชั้นที่ 6
c6 = Conv2DTranspose(512, 3, 2, padding = "same")(c5)
c6 = concatenate([c6, c4])
c6 = Dropout(0.3)(c6)
c6 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c6)
c6 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c6)
c6 = Conv2D(512, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c6)
# ชั้นที่ 7
c7 = Conv2DTranspose(256, 3, 2, padding = "same")(c6)
c7 = concatenate([c7, c3])
c7 = Dropout(0.3)(c7)
c7 = Conv2D(256, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c7)
c7 = Conv2D(256, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c7)
c7 = Conv2D(256, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c7)
# ชั้นที่ 8
c8 = Conv2DTranspose(128, 3, 2, padding = "same")(c7)
c8 = concatenate([c8, c2])
c8 = Dropout(0.3)(c8)
c8 = Conv2D(128, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c8)
c8 = Conv2D(128, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c8)
# ชั้นที่ 9
c9 = Conv2DTranspose(64, 3, 2, padding = "same")(c8)
c9 = concatenate([c9, c1])
c9 = Dropout(0.3)(c9)
c9 = Conv2D(64, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c9)
c9 = Conv2D(64, 3, padding = "same", activation = "relu", kernel_initializer = "he_normal")(c9)

outputs = Conv2D(n_classes, 1, padding = "same", activation = "softmax")(c9)

unet_model = Model(inputs, outputs)

return unet_model

```

รูปที่ 3.6.2 สถาปัตยกรรมโครงข่าย U-Net ที่มี VGG16 เป็นกระดูกสันหลัง (ส่วนการถอดรหัส และ convolutional แบบ 1x1)

ตามรูปแบบของสถาปัตยกรรมโครงข่ายประสาทเทียม U-Net ประกอบไปด้วย 4 ส่วน ได้แก่ ส่วนการเข้ารหัส, ส่วนการถอดรหัส, การข้ามการเชื่อมต่อ, และ convolutional แบบ 1x1

ดังนั้นผู้ศึกษาจึงสร้างตัวแบบโครงข่ายประสาทเทียม U-Net ที่มีสถาปัตยกรรม VGG16 เป็นกระดูกสันหลัง จากรูปที่ 3.6.1 และ 3.6.2 สามารถอธิบายได้ว่า

1) ส่วนการเข้ารหัส

ส่วนการเข้ารหัสจะเริ่มต้นจากการรับข้อมูลนำเข้าที่เป็นรูปภาพ โดยขนาดของข้อมูลนำเข้าสามารถปรับผันตามขนาดของรูปภาพได้ โดยขนาดของรูปภาพที่นำมาพัฒนาตัวแบบเท่ากับ 128x128 ประเภท RGB กล่าวคือมีขนาด (128, 128, 3) เมื่อนำรูปภาพเข้าสู่ตัวแบบแล้วจะดำเนินขั้นตอน convolutional ตามสถาปัตยกรรม VGG16 ตามที่ได้กล่าวในหัวข้อ 2.8.1 โดยขนาดของเมทริกซ์ตัวกรองเท่ากับ 3x3 และจำนวนผังคุณลักษณะที่สกัดออกมาที่ชั้นแรกเท่ากับ 64 จากนั้นจะเพิ่มขึ้นชั้นและสองเท่า จนสิ้นสุดที่จำนวน 512 มีการกำหนดค่า padding = same หมายความว่าส่งออกข้อมูลที่ได้ให้มีขนาดเท่าเดิมกับตอนที่รับเข้ามา กำหนดพังก์ชันกระตุ้นคือ Rectified Linear Units (ReLU) กำหนด kernel_initializer = "he_normal" ซึ่งเป็นหนึ่งในเทคนิคการกำหนดน้ำหนักของตัวกรอง และลดขนาดของผังคุณลักษณะลงครึ่งหนึ่งด้วยวิธีการพูลลิ่งด้วยค่าสูงสุด

2) ส่วนการถอดรหัส

ส่วนการถอดรหัสจะเริ่มต้นจากการรับผังคุณลักษณะจากส่วนเข้ารหัส ต่อมาจะดำเนินการเพิ่มความละเอียดโดยใช้วิธี convolutional แบบย้อนกลับผ่าน Conv2DTranspose ซึ่งเป็นฟังก์ชันจากไลบรารี keras โดยขนาดของเมทริกซ์ตัวกรองเท่ากับ 3×3 กำหนด Stride เท่ากับ 2 และกำหนดค่า padding = same หมายความว่าส่งออกข้อมูลที่ได้ให้มีขนาดเท่าเดิมกับตอนที่รับเข้ามา จากนั้นจะรวมผังคุณลักษณะจากการดำเนินการ convolutional แบบย้อนกลับกับผังคุณลักษณะจากส่วนการเข้ารหัสผ่านการข้ามการเชื่อมต่อ และนำผังคุณลักษณะที่รวมกันมาสกัดคุณลักษณะเด่นร่วมกันอีกในขั้นตอน convolutional โดยขนาดของเมทริกซ์ตัวกรองเท่ากับ 3×3 และจำนวนผังคุณลักษณะที่สกัดออกมาก็ที่ชั้นแรกเท่ากับจำนวนผังคุณลักษณะที่รับมาจากชั้นที่แล้ว มีการกำหนด padding = same, ฟังก์ชันกระตุ้นคือ Rectified Linear Units และใช้ kernel_initializer = "he_normal" จากนั้นส่งผังคุณลักษณะเด่นดังกล่าวให้ชั้นถัดไป

3) การข้ามการเชื่อมต่อ

ในส่วนของการถอดรหัสจะมีฟังก์ชัน concatenate ซึ่งทำหน้าที่เป็นการข้ามการเชื่อมต่อโดยฟังก์ชันดังกล่าวจะทำหน้าที่รวมผังคุณลักษณะของส่วนการเข้ารหัสและส่วนการถอดรหัสเข้าด้วยกันก่อนนำไปประมวลผลชั้นต่อไป โดยการข้ามการเชื่อมต่อจะมีอยู่ 4 ตำแหน่ง ได้แก่ ชั้นที่ 6 ทำหน้าที่รวมผังคุณลักษณะของชั้นที่ 4 และ 6, ชั้นที่ 7 ทำหน้าที่รวมผังคุณลักษณะของชั้นที่ 3 และ 7, ชั้นที่ 8 ทำหน้าที่รวมผังคุณลักษณะของชั้นที่ 2 และ 8, และชั้นที่ 9 ทำหน้าที่รวมผังคุณลักษณะของชั้นที่ 1 และ 9,

4) ค่อน convolutional แบบ 1×1

ในส่วนของค่อน convolutional แบบ 1×1 จะเป็นชั้นสุดท้ายของโครงข่ายประสาทเทียม U-Net มีหน้าที่รวมผังคุณลักษณะที่ได้รับจากส่วนการถอดรหัสออกมาเป็นผลลัพธ์การแบ่งส่วนความหมายโดยกำหนดจำนวนผังคุณลักษณะเท่ากับจำนวนเป้าหมายการทำนายทั้งหมดที่เป็นไปได้ กล่าวคือแต่ละผังคุณลักษณะจะเป็นผลการทำนายเป้าหมายใด ๆ แยกตามพิกเซล และจะมีการแจกแจงความน่าจะเป็นในการทำนายโดยใช้การคำนวนผลลัพธ์ด้วยฟังก์ชันเลขซึ่งกำลังที่ทำให้เป็นมาตรฐาน (activation = "softmax")

ระหว่างชั้นของโครงข่ายประสาทเทียม U-Net จะมีการป้องกันการเข้ากันได้มากเกินไปกับข้อมูลซ้ำฝึก (Overfitting) โดยใช้ Dropout ที่มีอัตราการ Dropout เท่ากับ 0.3

ตารางที่ 3.6.1 สรุปสถาปัตยกรรมโครงข่าย U-Net ที่มี VGG16 เป็นกรอบสันหลัง

ชื่อชั้น	ประเภทชั้น	รูปร่างข้อมูลขาออก	จำนวน พารามิเตอร์	รับข้อมูลมากจาก
input_1	Input Layer	(None, 128, 128, 3)	0	-
conv2d	Conv2D	(None, 128, 128, 64)	1792	input_1
conv2d_1	Conv2D	(None, 128, 128, 64)	36928	conv2d
max_pooling2d	MaxPooling2D	(None, 64, 64, 64)	0	conv2d_1
dropout	Dropout	(None, 64, 64, 64)	0	max_pooling2d
conv2d_2	Conv2D	(None, 64, 64, 128)	73856	dropout
conv2d_3	Conv2D	(None, 64, 64, 128)	147584	conv2d_2
max_pooling2d_1	MaxPooling2D	(None, 32, 32, 128)	0	conv2d_3
dropout_1	Dropout	(None, 32, 32, 128)	0	max_pooling2d_1
conv2d_4	Conv2D	(None, 32, 32, 256)	295168	dropout_1
conv2d_5	Conv2D	(None, 32, 32, 256)	590080	conv2d_4
conv2d_6	Conv2D	(None, 32, 32, 256)	590080	conv2d_5
max_pooling2d_2	MaxPooling2D	(None, 16, 16, 256)	0	conv2d_6
dropout_2	Dropout	(None, 16, 16, 256)	0	max_pooling2d_2
conv2d_7	Conv2D	(None, 16, 16, 512)	1180160	dropout_2
conv2d_8	Conv2D	(None, 16, 16, 512)	2359808	conv2d_7
conv2d_9	Conv2D	(None, 16, 16, 512)	2359808	conv2d_8
max_pooling2d_3	MaxPooling2D	(None, 8, 8, 512)	0	conv2d_9
dropout_3	Dropout	(None, 8, 8, 512)	0	max_pooling2d_3
conv2d_10	Conv2D	(None, 8, 8, 512)	2359808	dropout_3
conv2d_11	Conv2D	(None, 8, 8, 512)	2359808	conv2d_10
conv2d_12	Conv2D	(None, 8, 8, 512)	2359808	conv2d_11
conv2d_transpose	Conv2DTranspose	(None, 16, 16, 512)	2359808	conv2d_12
concatenate	Concatenate	(None, 16, 16, 1024)	0	conv2d_transpose, conv2d_9
dropout_4	Dropout	(None, 16, 16, 1024)	0	concatenate
conv2d_13	Conv2D	(None, 16, 16, 512)	4719104	dropout_4
conv2d_14	Conv2D	(None, 16, 16, 512)	2359808	conv2d_13

conv2d_15	Conv2D	(None, 16, 16, 512)	2359808	conv2d_14
conv2d_transpose_1	Conv2DTranspose	(None, 32, 32, 256)	1179904	conv2d_15
concatenate_1	Concatenate	(None, 32, 32, 512)	0	conv2d_transpose_1, conv2d_6
dropout_5	Dropout	(None, 32, 32, 512)	0	concatenate_1
conv2d_16	Conv2D	(None, 32, 32, 256)	1179904	dropout_5
conv2d_17	Conv2D	(None, 32, 32, 256)	590080	conv2d_16
conv2d_18	Conv2D	(None, 32, 32, 256)	590080	conv2d_17
conv2d_transpose_2	Conv2DTranspose	(None, 64, 64, 128)	295040	conv2d_18
concatenate_2	Concatenate	(None, 64, 64, 256)	0	conv2d_transpose_2, conv2d_3
dropout_6	Dropout	(None, 64, 64, 256)	0	concatenate_2
conv2d_19	Conv2D	(None, 64, 64, 128)	295040	dropout_6
conv2d_20	Conv2D	(None, 64, 64, 128)	147584	conv2d_19
conv2d_transpose_3	Conv2DTranspose	(None, 128, 128, 64)	73792	conv2d_20
concatenate_3	Concatenate	(None, 128, 128, 128)	0	conv2d_transpose_3, conv2d_1
dropout_7	Dropout	(None, 128, 128, 128)	0	concatenate_3
conv2d_21	Conv2D	(None, 128, 128, 64)	73792	dropout_7
conv2d_22	Conv2D	(None, 128, 128, 64)	36928	conv2d_21
conv2d_23	Conv2D	(None, 128, 128, 4)	260	conv2d_22

ในการเรียนรู้ของตัวแบบนั้น มีการกำหนดค่าพารามิเตอร์ต่าง ๆ ที่จำเป็นในการ Compile ตัวแบบ
ดังต่อไปนี้

```

dice_loss = segmentation_models.losses.DiceLoss(class_weights=np.array(class_weights))
focal_loss = segmentation_models.losses.CategoricalFocalLoss()
total_loss = dice_loss + (1 * focal_loss)

metrics = segmentation_models.metrics.IOUScore()

model.compile(optimizer='adam', loss=total_loss, metrics=[metrics])

```

รูปที่ 3.6.8 การกำหนดค่าพารามิเตอร์ที่จำเป็นในการ Compile ตัวแบบ

1) optimizer

Optimizer เป็นอัลกอริทึมที่มีหน้าที่ปรับค่าน้ำหนักพารามิเตอร์การเรียนรู้ในตัวแบบ เพื่อให้มีฟังก์ชันการสูญเสียต่ำที่สุด โดยผู้ศึกษาได้เลือกใช้ Optimizer ชื่อ “Adam”

2) loss

ฟังก์ชันการสูญเสียเป็นฟังก์ชันสำหรับใช้วัดค่าความผิดพลาดจากการลัพธ์ในแต่ละรอบของ การเรียนรู้ (Epoch) ของตัวแบบ โดยผู้ศึกษาได้เลือกใช้ฟังก์ชันการสูญเสียคือผลบวกของ Categorical Focal Loss และ Dice Loss เพื่อแก้ปัญหาความไม่สมดุลระหว่างพื้นหน้าและพื้นหลัง ที่ส่งผลกระทบต่อกระบวนการฝึกฝนตัวแบบ โดย Dice Loss จะต้องรับค่าน้ำหนักของเป้าหมายการทำนายทั้งหมดที่เป็นไปได้ของข้อมูลชุดฝึก ในการคำนวณรับค่าน้ำหนักของเป้าหมายการทำนาย ทั้งหมดที่เป็นไปได้ ผู้ศึกษาได้สร้างฟังก์ชัน classWeight โดยใช้ compute_class_weight ซึ่งเป็น ฟังก์ชันจากไลบรารี Scikit-learn

```
def classWeight(masks):
    mask_reshaped = masks.reshape(-1, 1)
    mask_reshaped = mask_reshaped.squeeze()
    class_weights = class_weight.compute_class_weight('balanced', classes=np.unique(mask_reshaped), y=mask_reshaped)
    return class_weights
```

รูปที่ 3.6.9 ฟังก์ชันคำสั่งที่ใช้ในการคำนวณค่าน้ำหนัก

```
class_weights = classWeight(y_train)
class_weights
✓ 0.5s
array([0.36630954, 3.24953281, 2.39995313, 1.83265308])
```

รูปที่ 3.6.10 ผลลัพธ์ของฟังก์ชันการคำนวณน้ำหนัก

3) metrics

เป็นค่าสำหรับวัดค่าความถูกต้องของตัวแบบ โดยผู้ศึกษาได้เลือกใช้ค่า MIoU ในการวัด ความถูกต้องผ่าน IOUScore ซึ่งเป็นฟังก์ชันจากไลบรารี segmentation_models

ในการฝึกฝนตัวแบบ ผู้ศึกษาได้กำหนดจำนวนรอบสูงสุดที่ 500 รอบ อย่างไรก็ตามเพื่อให้รองรับ เหตุฉุกเฉินระหว่างการฝึก เช่น ไฟฟ้าดับ เป็นต้น และเพื่อไม่ให้ประสิทธิภาพของการฝึกฝนตกลงเมื่อเวลา ผ่านไป ผู้ศึกษาได้กำหนดฟังก์ชันการ Callback เพื่อรับบัญชาดังกล่าว

```

callbacks = [
    tf.keras.callbacks.ModelCheckpoint(filepath='checkpoint/model_checkpoint.h5', save_best_only=True),
    tf.keras.callbacks.EarlyStopping(monitor='val_iou_score', patience=20, mode='max', restore_best_weights=True)
]

```

รูปที่ 3.6.11 พังก์ชันการ Callback

1) ModelCheckpoint

ModelCheckpoint คือพังก์ชันที่ใช้บันทึกตัวแบบที่ดีที่สุดในจำนวนรอบที่ผ่านไปในรูปแบบไฟล์นามสกุล .h5 เพื่อบอังกันเหตุฉุกเฉิน ทำให้หลังจากนั้นสามารถนำตัวแบบมาฝึกฝนต่อจากรอบที่ดีที่สุดได้

2) EarlyStopping

EarlyStopping คือพังก์ชันที่ใช้หยุดการเรียนรู้ของตัวแบบ หากการเรียนรู้นั้นไม่เปลี่ยนแปลงหรือแย่ลง โดยกำหนดตัวแปร monitor=val_iou_score, patience=20, และ mode='max' หมายความว่าให้ใช้ค่า MIoU ที่ประเมินโดยข้อมูลชุดทดสอบเป็นจุดตรวจสอบ หากภายใน 20 รอบ ค่าดังกล่าวไม่ค่าน้อยหรือเท่ากับค่าตั้งต้นจะหยุดการเรียนรู้ของตัวแบบทันที และกำหนดตัวแปร restore_best_weights=True หมายความว่าให้ย้อนผลลัพธ์การฝึกกลับไปที่รอบตั้งต้น

หลังจากการ Compile และกำหนดพังก์ชันการ Callback แล้ว ขั้นตอนถัดไปคือการฝึกฝนตัวแบบ

```

history = model.fit(x_train,
                     y_train,
                     batch_size=x_train.shape[0],
                     verbose=1,
                     epochs=500,
                     validation_data=(x_val, y_val),
                     callbacks=callbacks)

```

รูปที่ 3.6.12 พังก์ชันคำสั่งในการเริ่มฝึกฝนตัวแบบ

ในการฝึกฝนตัวแบบ เนื่องจากจำนวนข้อมูลมีไม่มาก และผู้ศึกษาต้องการให้ลักษณะของการเรียนรู้เป็นแบบเส้นตรง (Batch Gradient Descent) ดังนั้นจึงกำหนด batch_size ให้เท่ากับจำนวนข้อมูล กำหนด verbose=1 หมายความว่าให้มีการแสดงผลการฝึกฝนทุกรอบของการเรียนรู้ กำหนดจำนวนรอบการฝึกฝนเท่ากับ 500 รอบ และกำหนดพังก์ชันการ Callback ตามรูปที่ 3.6.11

```

Epoch 1/500
1/1 [=====] - 93s 93s/step - loss: 0.8662 - iou_score: 0.0401 - val_loss: 0.7501 - val_iou_score: 0.0981
Epoch 2/500
1/1 [=====] - 80s 80s/step - loss: 0.8507 - iou_score: 0.1353 - val_loss: 0.7010 - val_iou_score: 0.1516
Epoch 3/500
1/1 [=====] - 82s 82s/step - loss: 0.7154 - iou_score: 0.1422 - val_loss: 0.6799 - val_iou_score: 0.1748
Epoch 4/500
1/1 [=====] - 82s 82s/step - loss: 0.6943 - iou_score: 0.1624 - val_loss: 0.6546 - val_iou_score: 0.1840
Epoch 5/500
1/1 [=====] - 81s 81s/step - loss: 0.6671 - iou_score: 0.1774 - val_loss: 0.6109 - val_iou_score: 0.1909
Epoch 6/500
1/1 [=====] - 80s 80s/step - loss: 0.6243 - iou_score: 0.1843 - val_loss: 1.2586 - val_iou_score: 0.0460
Epoch 7/500
1/1 [=====] - 81s 81s/step - loss: 1.6080 - iou_score: 0.0341 - val_loss: 0.5827 - val_iou_score: 0.2117
Epoch 8/500
1/1 [=====] - 81s 81s/step - loss: 0.5806 - iou_score: 0.2248 - val_loss: 0.5488 - val_iou_score: 0.3447
Epoch 9/500
1/1 [=====] - 81s 81s/step - loss: 0.5542 - iou_score: 0.3188 - val_loss: 0.5363 - val_iou_score: 0.3302
Epoch 10/500
1/1 [=====] - 81s 81s/step - loss: 0.5377 - iou_score: 0.3387 - val_loss: 0.4657 - val_iou_score: 0.3599
Epoch 11/500
1/1 [=====] - 81s 81s/step - loss: 0.4819 - iou_score: 0.3407 - val_loss: 0.4262 - val_iou_score: 0.3463
Epoch 12/500
1/1 [=====] - 80s 80s/step - loss: 0.4504 - iou_score: 0.3236 - val_loss: 0.4002 - val_iou_score: 0.3471
Epoch 13/500
...
Epoch 265/500
1/1 [=====] - 77s 77s/step - loss: -0.7938 - iou_score: 0.9289 - val_loss: -0.7455 - val_iou_score: 0.8929
Epoch 266/500
1/1 [=====] - 77s 77s/step - loss: -0.7944 - iou_score: 0.9292 - val_loss: -0.7410 - val_iou_score: 0.8901

```

รูปที่ 3.6.13 การแสดงผลการฝึกฝนทุกรอบของการเรียนรู้

จากรูปที่ 3.6.13 จะเห็นว่าในระหว่างการของการฝึกฝนจะมีการแสดงค่าต่าง ๆ ดังต่อไปนี้

- เวลา (วินาที) / รอบ คือ เวลาที่ใช้ในการฝึกฝนรอบนั้น ๆ
- loss คือ ค่าความผิดพลาดของตัวแบบที่เกิดขึ้นในข้อมูลชุดฝึก
- iou_score คือ ค่า MIoU ของตัวแบบที่เกิดขึ้นกับข้อมูลชุดฝึก
- val_loss คือ ค่าความผิดพลาดของตัวแบบที่เกิดขึ้นในข้อมูลชุดตรวจทาน
- val_iou_score คือ ค่า MIoU ของตัวแบบที่เกิดขึ้นกับข้อมูลชุดตรวจทาน

เมื่อการฝึกฝนเสร็จสิ้น ผู้ศึกษาได้บันทึกตัวแบบผ่านฟังก์ชัน save ในรูปแบบไฟล์นามสกุล .h5

```
model.save('..../Models/มะหมีเหลืองแห้ง_3.h5')
```

รูปที่ 3.6.14 คำสั่งในการบันทึกตัวแบบ

3.7) การประเมินผลตัวแบบ (Evaluation)

เมื่อฝึกฝนตัวแบบเรียบร้อยแล้ว ผู้ศึกษาจึงต้องประเมินผลตัวแบบ เพื่อวัดประสิทธิภาพของตัวแบบ สำหรับการปรับปรุงและพัฒนาต่อไป

3.7.1) ค่า Mean Intersection over Union และ Loss ของข้อมูลชุดทดสอบ

การนำข้อมูลชุดทดสอบมาประเมินตัวแบบที่ฝึกฝนเรียบร้อยแล้วเป็นวิธีที่พื้นฐานที่สุด โดยนำข้อมูลชุดทดสอบมาดำเนินการผ่านฟังก์ชัน evaluate

```
_ , IoU = model.evaluate(x_test, y_test)  
print(f'Iou Score = {IoU}')
```

รูปที่ 3.7.1.1 คำสั่งในการประเมินผลตัวแบบด้วยข้อมูลชุดทดสอบ

```
1/1 [=====] - 3s 3s/step - loss: -0.5237 - iou_score: 0.8726  
Iou Score = 0.8726
```

รูปที่ 3.7.1.2 ผลลัพธ์ของคำสั่งประเมินผลตัวแบบด้วยข้อมูลชุดทดสอบ

3.7.2) กราฟเส้น (Line Graph)

การประเมินผลด้วยกราฟเส้นจะสามารถเห็นภาพรวมของพัฒนาการเรียนรู้ของตัวแบบได้ทั้งหมด และยังสามารถพิจารณาการเกิดปรากฏการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึกเบื้องต้นได้อีกด้วย

การสร้างกราฟจะใช้ประวัติ (History) ของการฝึกตัวแบบ โดยใช้ไลบรารี matplotlib ใน การสร้างกราฟ กราฟที่สร้างขึ้นแบ่งออกเป็น 2 รูปแบบ ได้แก่

- กราฟแสดงค่าความผิดพลาด (Loss) ของตัวแบบที่เกิดขึ้นในข้อมูลชุดฝึกฝน (Training Data) และข้อมูลชุดตรวจสอบ (Validation Data)
- กราฟแสดงค่า Mean Intersection over Union (MIoU) ของตัวแบบที่เกิดขึ้นในข้อมูลชุดฝึกฝน (Training Data) และข้อมูลชุดตรวจสอบ (Validation Data)

```

loss = history.history['loss']
val_loss = history.history['val_loss']
iou_score = history.history['iou_score']
val_iou_score = history.history['val_iou_score']
epochs = range(1, len(loss) + 1)

plt.figure(figsize=(10, 5))

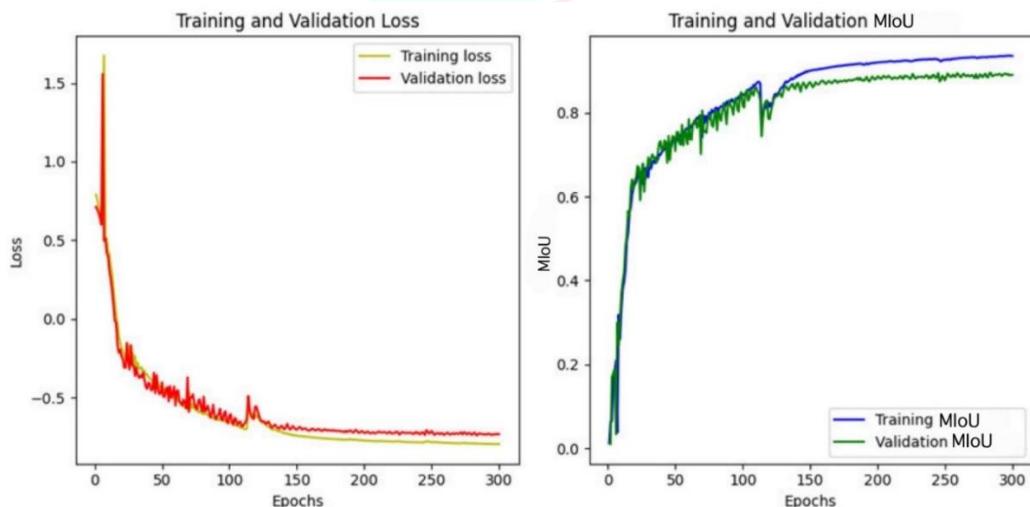
plt.subplot(1, 2, 1)
plt.plot(epochs, loss, 'y', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(epochs, iou_score, 'b', label='Training MIoU')
plt.plot(epochs, val_iou_score, 'g', label='Validation MIoU')
plt.title('Training and Validation MIoU')
plt.xlabel('Epochs')
plt.ylabel('MIoU')
plt.legend()

plt.tight_layout()
plt.show()

```

รูปที่ 3.7.2.1 คำสั่งที่ใช้ในการแสดงกราฟเส้น



รูปที่ 3.7.2.2 ตัวอย่างกราฟเส้นแสดงค่า Loss และ MIoU ของข้อมูลชุดฝึกฝนและตรวจสอบ

3.7.3) รูปหน้ากากที่ทำนายโดยตัวแบบ

การประเมินผลด้วยรูปหน้ากากที่ทำนายโดยตัวแบบเทียบกับหน้ากากจริงของรูปภาพนั้น จะทำให้สามารถเห็นชุดที่ตัวแบบทำนายผิดพลาด และสามารถตั้งข้อสังเกตุของการทำนายผิดพลาด

ได้ โดยจะใช้รูปภาพจากข้อมูลชุดทดสอบในการทำนาย และเทียบหน้ากากที่ทำนายกับหน้ากากจริง ในการนำข้อมูลชุดทดสอบไปให้ตัวแบบทำนายจะทำผ่านฟังก์ชัน predict จากนั้นจะดึงหน้ากากที่ตัวแบบทำนายมาแสดงเป็นรูปภาพสีควบคู่กับรูปภาพต้นและหน้ากากจริง

```
prediction = model.predict(input_img)
prediction_image = np.argmax(prediction, axis=3)[0]

plt.figure(figsize=(15, 10))
plt.subplot(231)
plt.title('Testing Image')
plt.imshow(input_img_plot)
plt.axis('off')

plt.subplot(232)
plt.title('Mask')
plt.imshow(input_mask)
plt.axis('off')

plt.subplot(233)
plt.title('Prediction on test image')
plt.imshow(prediction_image)
plt.axis('off')
plt.show()
```

รูปที่ 3.7.3.1 คำสั่งใช้ในการนำข้อมูลชุดทดสอบไปทำนาย และแสดงหน้ากากที่ตัวแบบทำนาย



รูปที่ 3.7.3.2 รูปภาพต้น (ซ้าย), หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)

บทที่ 4 การประเมินผล และการนำตัวแบบไปใช้งาน

4.1) การพัฒนาครั้งที่ 1

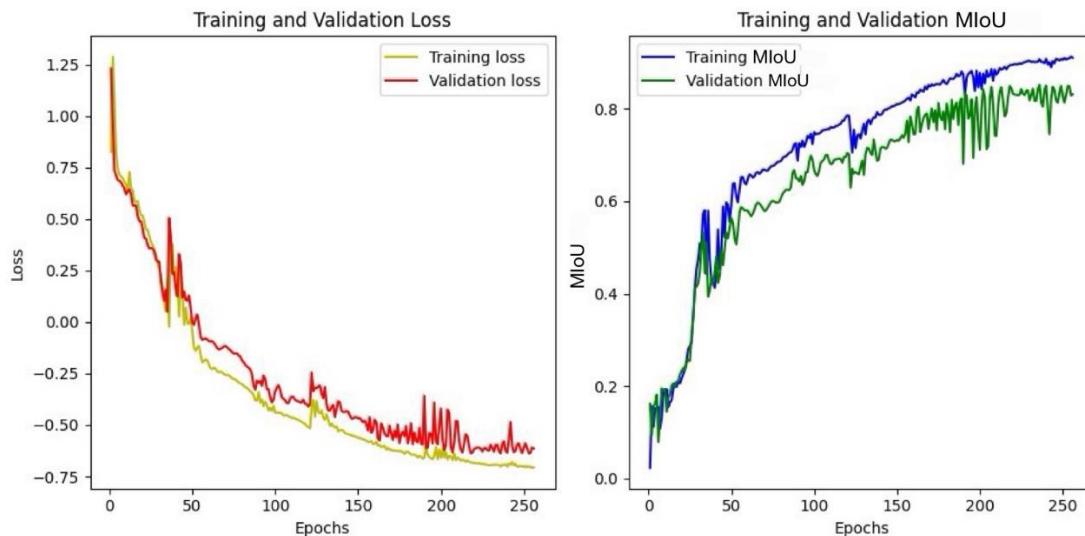
ในการพัฒนาครั้งที่ 1 ผู้ศึกษาได้ฝึกฝนตัวแบบแยกตามเมนูอาหาร 5 เมนู ได้แก่ กะเพราไก่ร้าดข้าว, ข้าวคลุกกะปิ, ข้าวมันไก่, ก๋วยเตี๋ยวบะหมี่เหลืองแห้ง, และผัดไทย โดยแต่ละเมนูจะฝึกฝนตัวแบบจำนวน 3 ตัวแบบ รวม 15 ตัวแบบ และเลือกตัวแบบที่ดีที่สุดของเมนูนั้น ๆ โดยมีการประเมินผลดังนี้

4.1.1) กะเพราไก่ร้าดข้าว

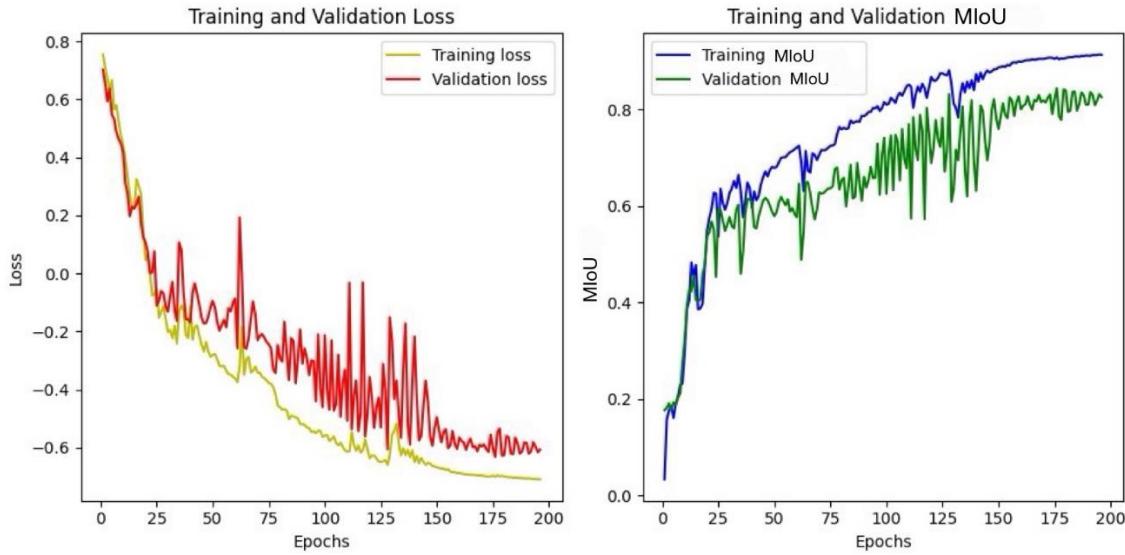
ตารางที่ 4.1.1.1 ตารางการประเมินผลตัวแบบเมนูกะเพราไก่ร้าดข้าว (ครั้งที่ 1)

ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.7008	-0.6378	-0.6662	0.9075	0.8521	0.8855
2	-0.6999	-0.6321	-0.6612	0.9072	0.8434	0.8803
3	-0.6752	-0.6104	-0.6747	0.8880	0.8287	0.8914

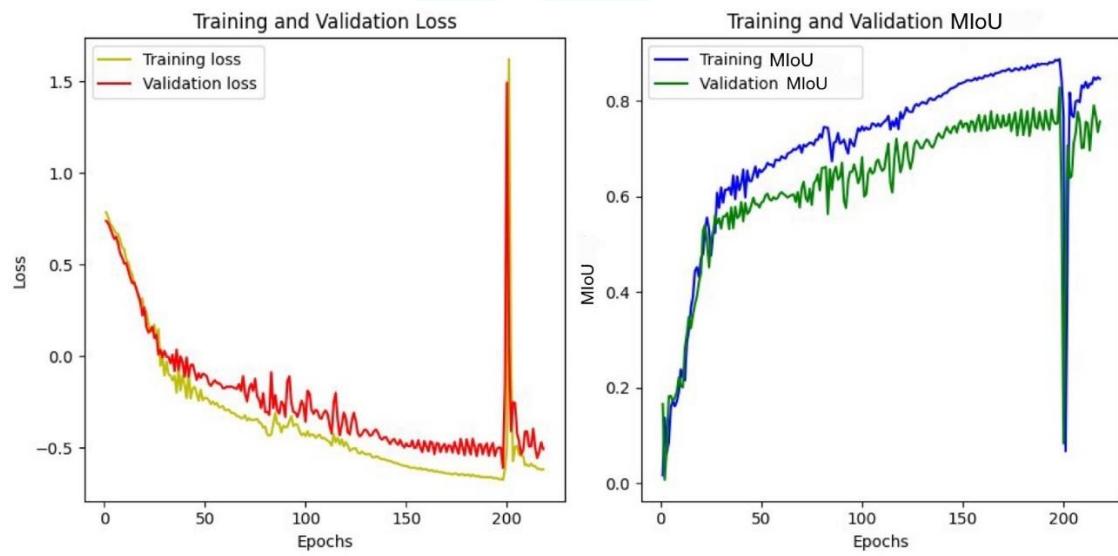
จากตารางที่ 4.1.1.1 พบร่วมกับตัวแบบที่ 3 มีค่า Mean Intersection over Union ของข้อมูลชุดทดสอบสูงที่สุดที่ 0.8914 รองลงมาคือตัวแบบที่ 1 มีค่า 0.8855 และลำดับสุดท้ายคือตัวแบบที่ 2 มีค่า 0.8803



รูปที่ 4.1.1.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูกะเพราไก่ร้าดข้าว (ตัวแบบที่ 1)



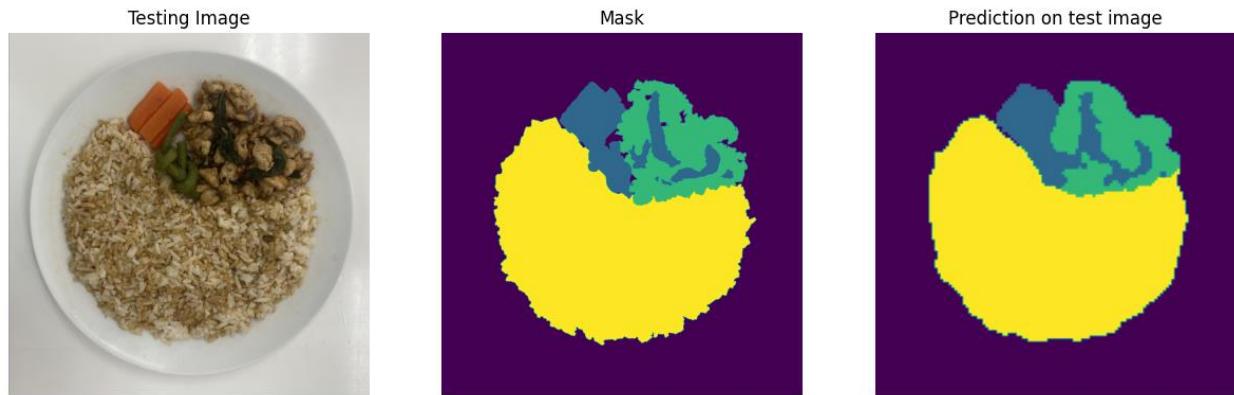
รูปที่ 4.1.1.2 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูกะเพราไกร้าดข้าว (ตัวแบบที่ 2)



รูปที่ 4.1.1.3 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูกะเพราไกร้าดข้าว (ตัวแบบที่ 3)

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจสอบ พบร่วมกันได้ว่าตัวแบบแบบไม่เกิดปรากฏการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้ดี

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและกราฟเส้น ผู้ศึกษาได้เลือกตัวแบบที่ 3 เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำงานหน้าหากของข้อมูลชุดทดสอบเทียบกับหน้าหากจริง ดังนี้



รูปที่ 4.1.1.4 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้าหากจริง (กลาง) และหน้าหากที่ตัวแบบทำงาน (ขวา)

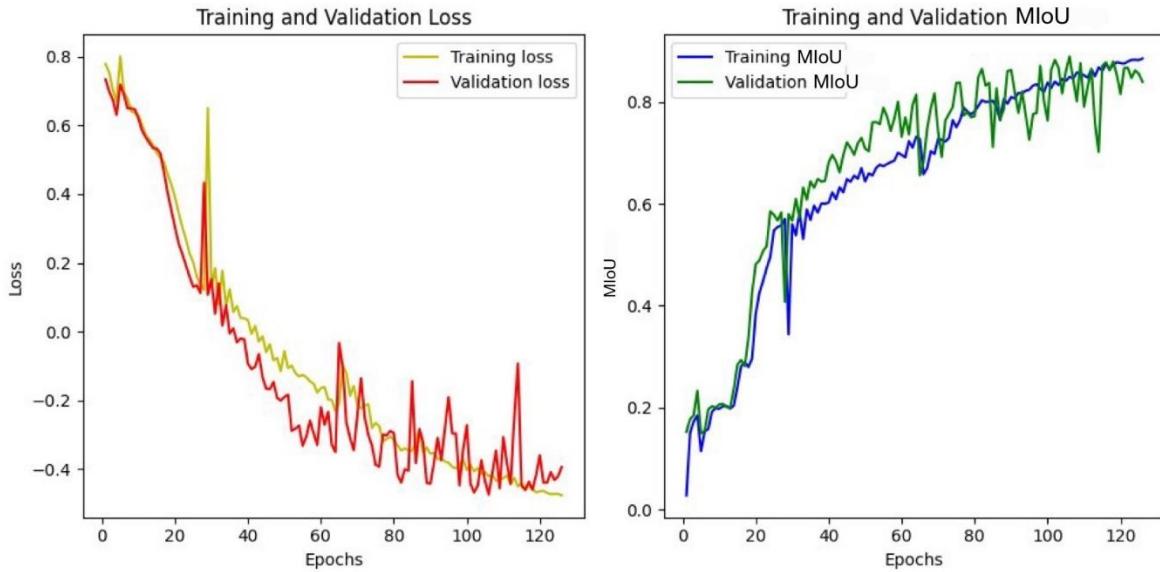
จากการเปรียบเทียบ พบร่วมกันว่าตัวแบบสามารถทำงานได้ดี แต่การทำนายไข่ไก่ (ผัก) บนเนื้อไก่ (โปรตีน) อาจมีความคาดเคลื่อน

4.1.2) ข้าวคลุกกะปิ

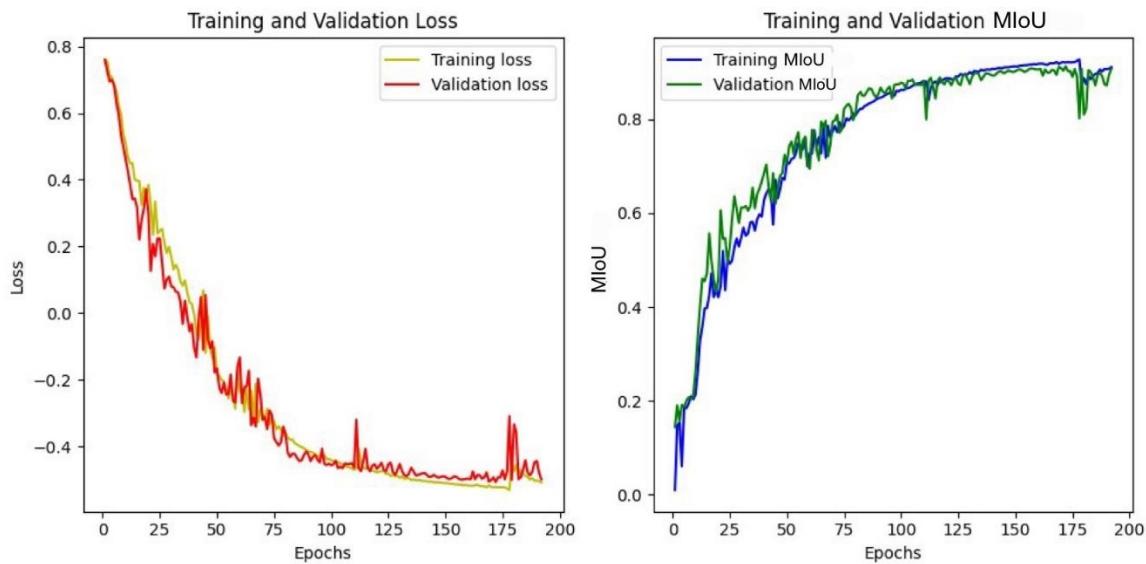
ตารางที่ 4.1.2.1 ตารางการประเมินผลตัวแบบเมนูข้าวคลุกกะปิ (ครั้งที่ 1)

ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.4205	-0.4736	-0.4039	0.8495	0.8893	0.8410
2	-0.5202	-0.5060	-0.4814	0.9270	0.9127	0.8955
3	-0.4611	-0.4873	-0.4556	0.8781	0.8992	0.8766

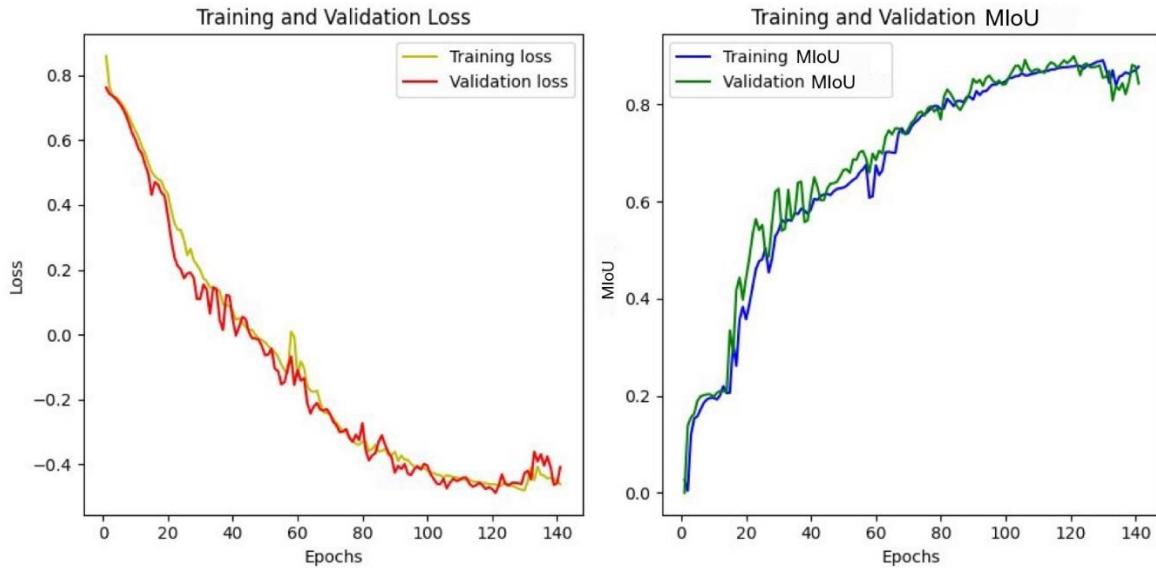
จากตารางที่ 4.1.2.1 พบร่วมกันว่าตัวแบบที่ 2 มีค่า Mean Intersection over Union ของข้อมูลชุดทดสอบสูงที่สุดที่ 0.8955 รองลงมาคือตัวแบบที่ 3 มีค่า 0.8766 และลำดับสุดท้ายคือตัวแบบที่ 1 มีค่า 0.8410



รูปที่ 4.1.2.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวคลุกกะปิ (ตัวแบบที่ 1)



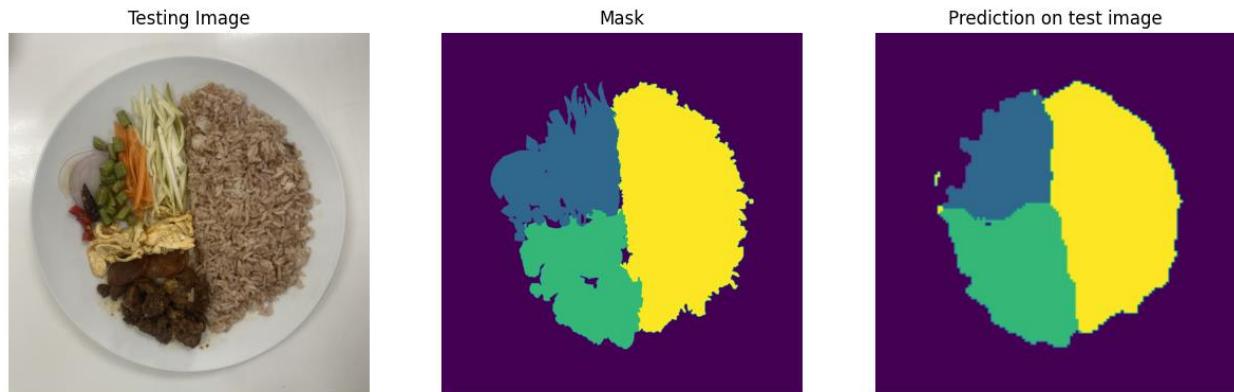
รูปที่ 4.1.2.2 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวคลุกกะปิ (ตัวแบบที่ 2)



รูปที่ 4.1.2.3 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวคลุกกะปิ (ตัวแบบที่ 3)

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจสอบ พบว่าตัวแบบแบบไม่เกิดปรากฏการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและภาพเส้น ผู้ศึกษาได้เลือกตัวแบบที่ 2 เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำนายหน้ากากของข้อมูลชุดทดสอบเทียบกับหน้ากากจริง ดังนี้



รูปที่ 4.1.2.4 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)

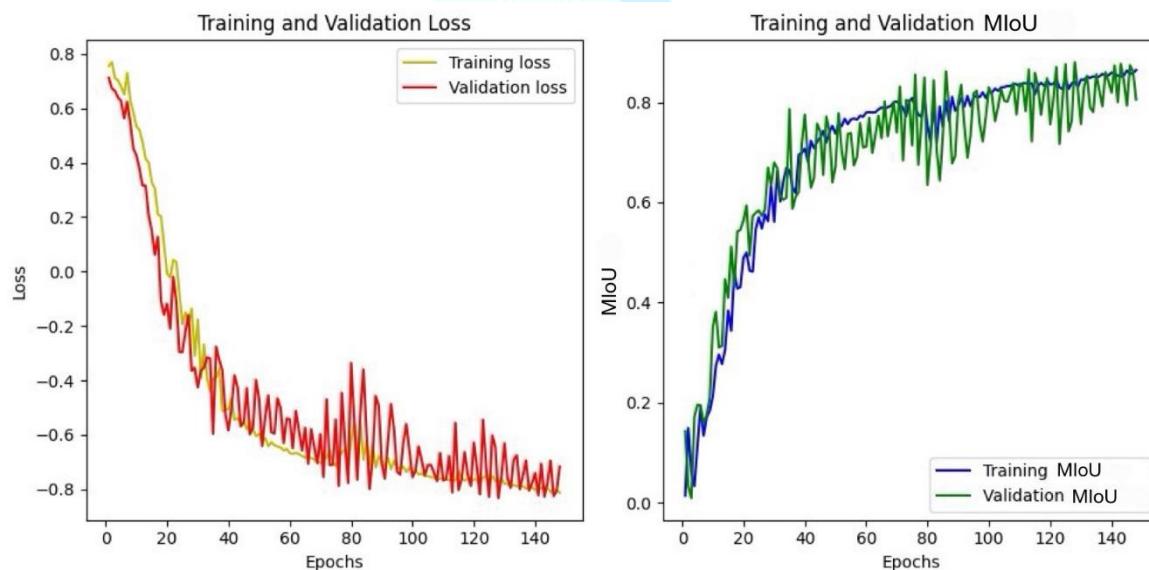
จากการเปรียบเทียบ พบร่วมกันแบบสามารถทำนายได้ดี แต่ไม่สามารถทำนายหัวหอม (ผัก) ได้

4.1.3) ข้าวมันไก่

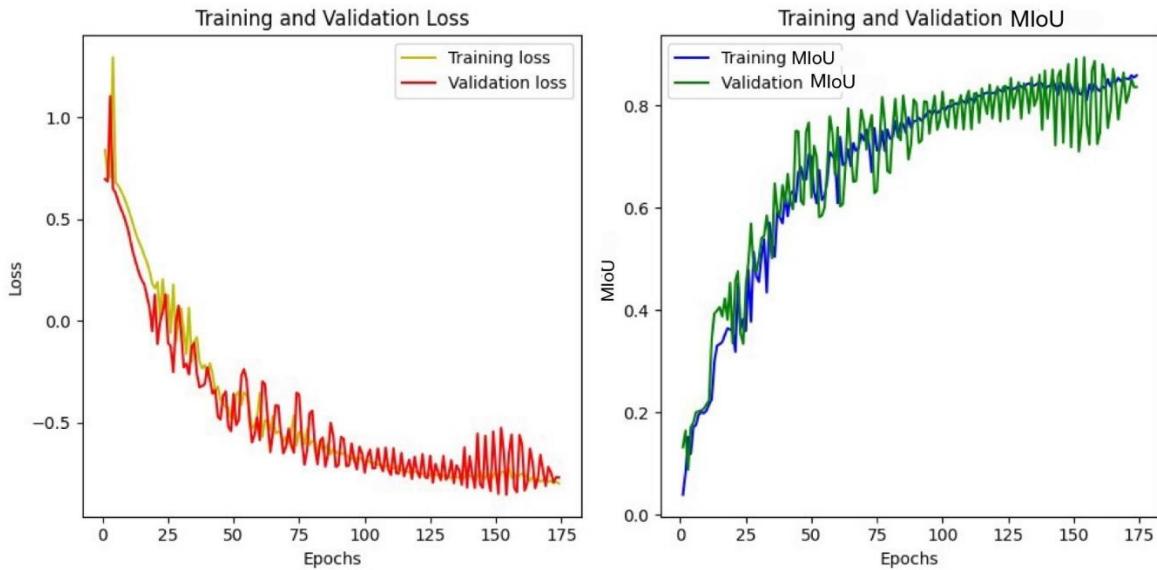
ตารางที่ 4.1.3.1 ตารางการประเมินผลตัวแบบเมนูข้าวมันไก่ (ครั้งที่ 1)

ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.7748	-0.8331	-0.8734	0.8411	0.8802	0.9113
2	-0.7738	-0.8548	-0.8067	0.8413	0.8936	0.8923
3	-0.7384	-0.8051	-0.8067	0.8220	0.8629	0.8742

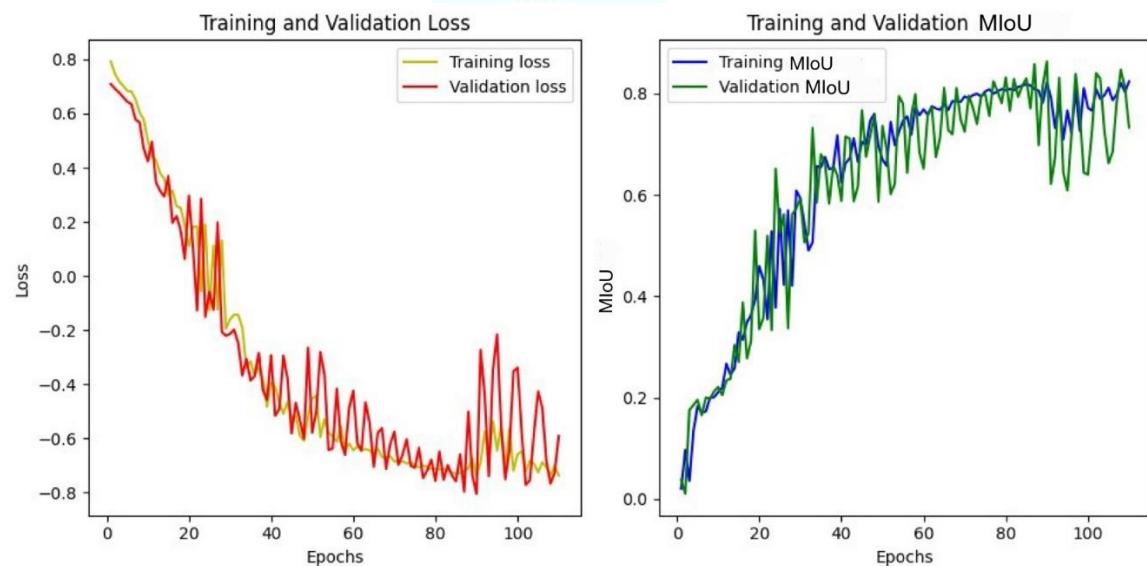
จากตารางที่ 4.1.1.3 พบร่วมกันแบบที่ 1 มีค่า Mean Intersection over Union ของข้อมูลชุดทดสอบสูงที่สุดที่ 0.9113 รองลงมาคือตัวแบบที่ 2 มีค่า 0.8923 และลำดับสุดท้ายคือตัวแบบที่ 3 มีค่า 0.8742



รูปที่ 4.1.3.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบการทำงานของเมนูข้าวมันไก่ (ตัวแบบที่ 1)



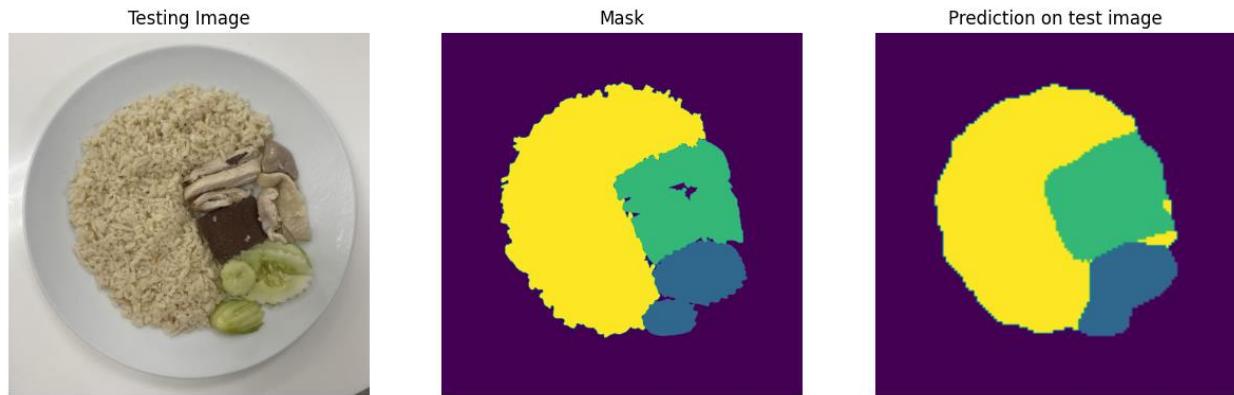
รูปที่ 4.1.3.2 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวมันไก่ (ตัวแบบที่ 2)



รูปที่ 4.1.3.3 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวมันไก่ (ตัวแบบที่ 3)

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจสอบ พบร่วมกันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้ดี

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและกราฟเส้น ผู้ศึกษาได้เลือกตัวแบบที่ 1 เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำงานหน้าหากของข้อมูลชุดทดสอบเทียบกับหน้าหากจริง ดังนี้



รูปที่ 4.1.3.4 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้าหากจริง (กลาง) และหน้าหากที่ตัวแบบทำงาน (ขวา)

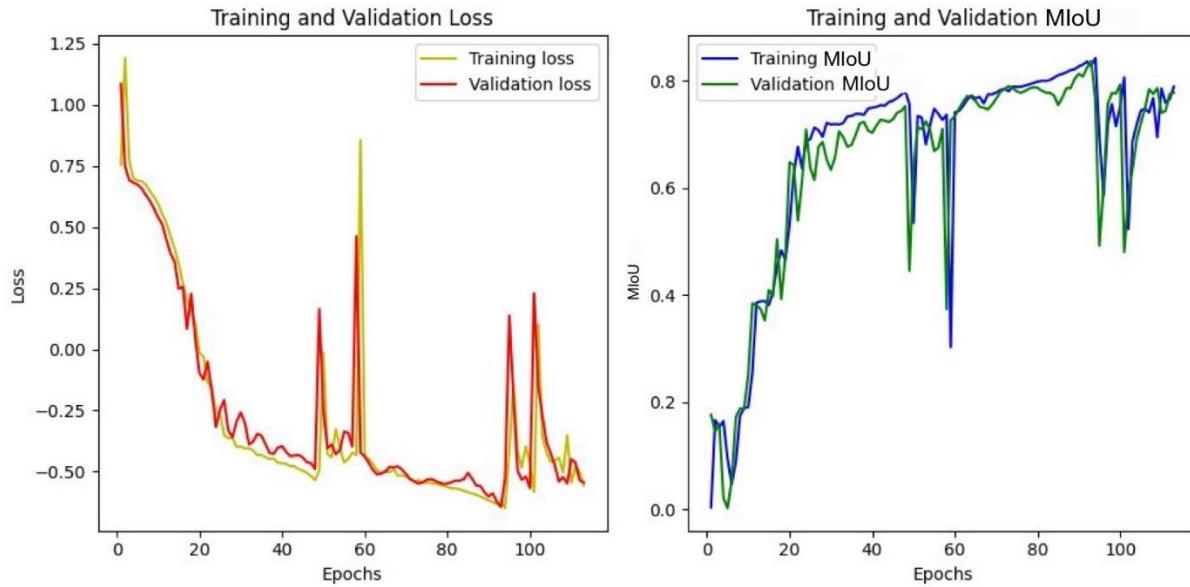
จากการเปรียบเทียบ พบร่วมกันว่าตัวแบบสามารถทำงานได้ดี แต่ในการทำงานหนังไก่ (โปรตีน) อาจผิดพลาดในบางจุด

4.1.4) กวยเตี๋ยวbalance มีเหลืองแห้ง

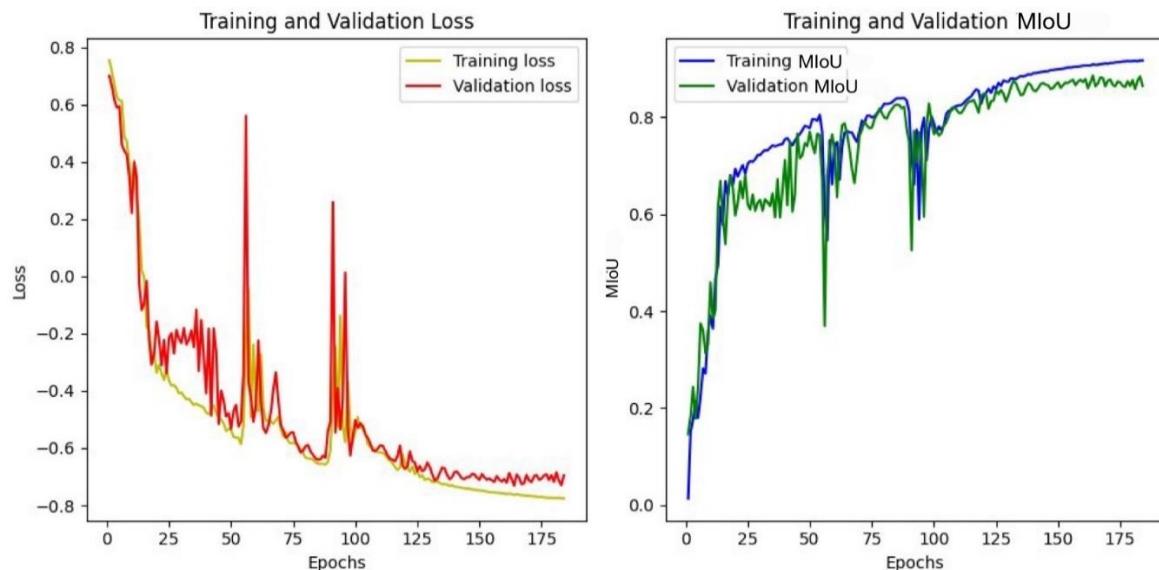
ตารางที่ 4.1.4.1 ตารางการประเมินผลตัวแบบเมนูกวยเตี๋ยวbalance มีเหลืองแห้ง (ครั้งที่ 1)

ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.7738	-0.8548	-0.8524	0.8413	0.8936	0.8983
2	-0.7639	-0.7314	-0.7407	0.9093	0.8862	0.8941
3	-0.7930	-0.7512	-0.7727	0.9285	0.8971	0.9145

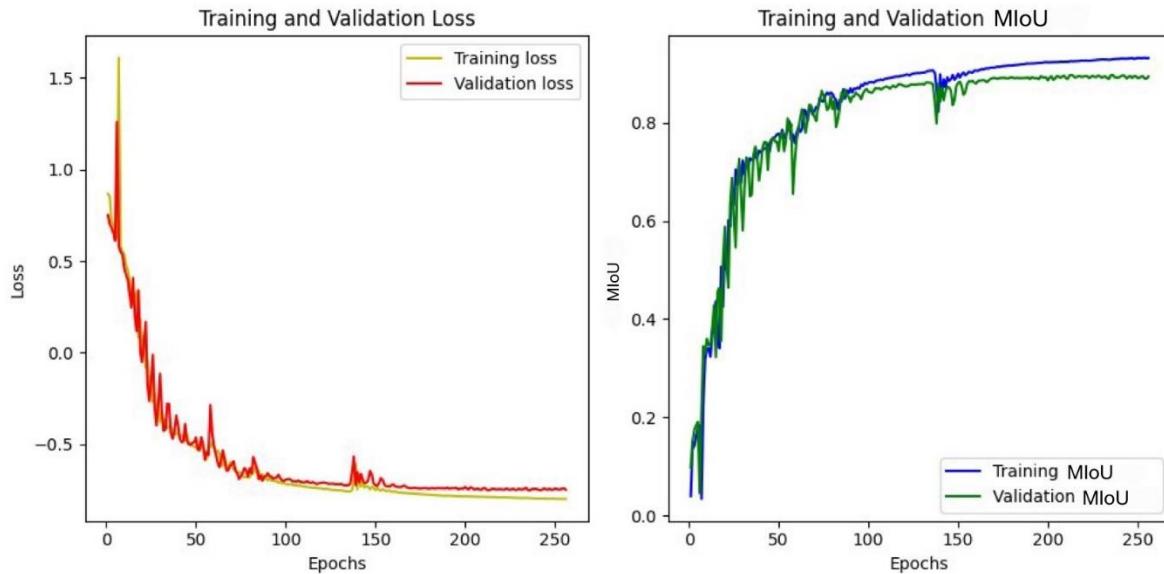
จากตารางที่ 4.1.4.1 พบร่วมกันว่าตัวแบบที่ 3 มีค่า Mean Intersection over Union ของข้อมูลชุดทดสอบสูงที่สุดที่ 0.9145 รองลงมาคือตัวแบบที่ 1 มีค่า 0.8983 และลำดับสุดท้ายคือตัวแบบที่ 2 มีค่า 0.8941



รูปที่ 4.1.4.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูภาษาไทยเว็บหนึ่ง (ตัวแบบที่ 1)



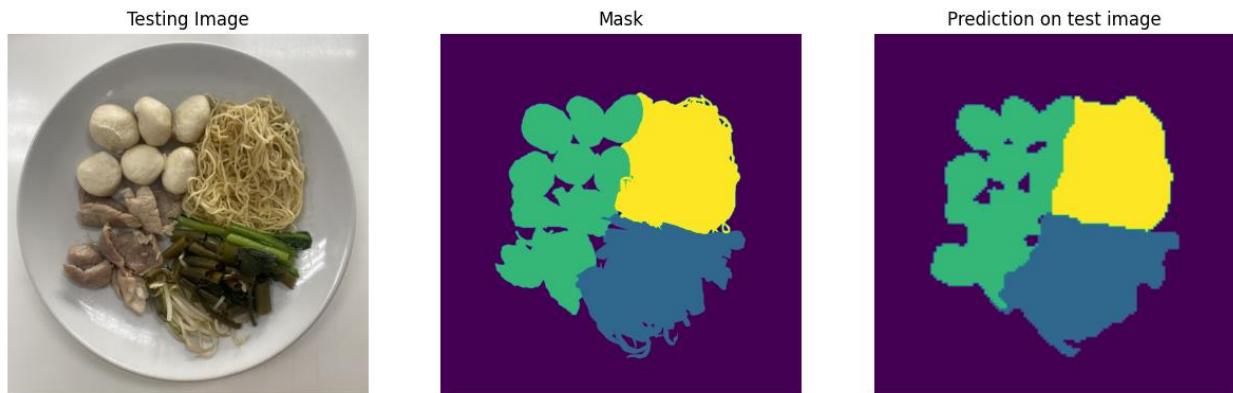
รูปที่ 4.1.4.2 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูภาษาไทยเว็บหนึ่ง (ตัวแบบที่ 2)



รูปที่ 4.1.4.3 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจงานของเมนูก๋วยเตี๋ยวbalance ให้ล่องแห้ง (ตัวแบบที่ 3)

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจงาน พบร่วมกันว่าตัวแบบแบบไม่เกิดประภัยการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้ดี

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและกราฟเสนอ ผู้ศึกษาได้เลือกตัวแบบที่ 3 เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำนายหน้ากากของข้อมูลชุดทดสอบเทียบกับหน้ากากจริง ดังนี้



รูปที่ 4.1.4.4 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)

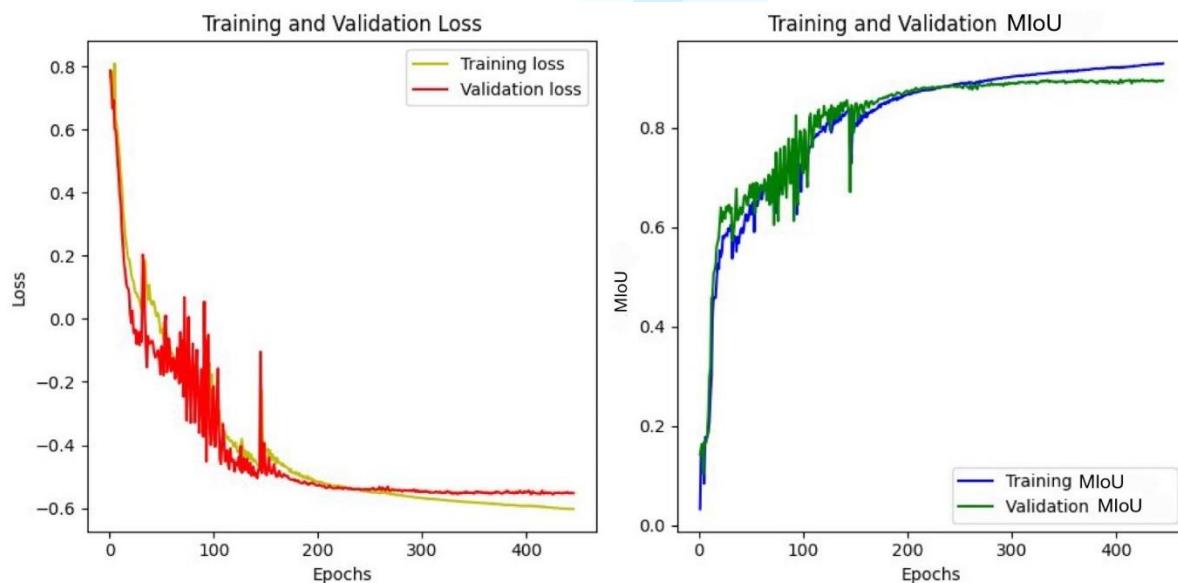
จากการเปรียบเทียบ พบว่าตัวแบบสามารถทำนายได้ดี

4.1.5) ผัดไทย

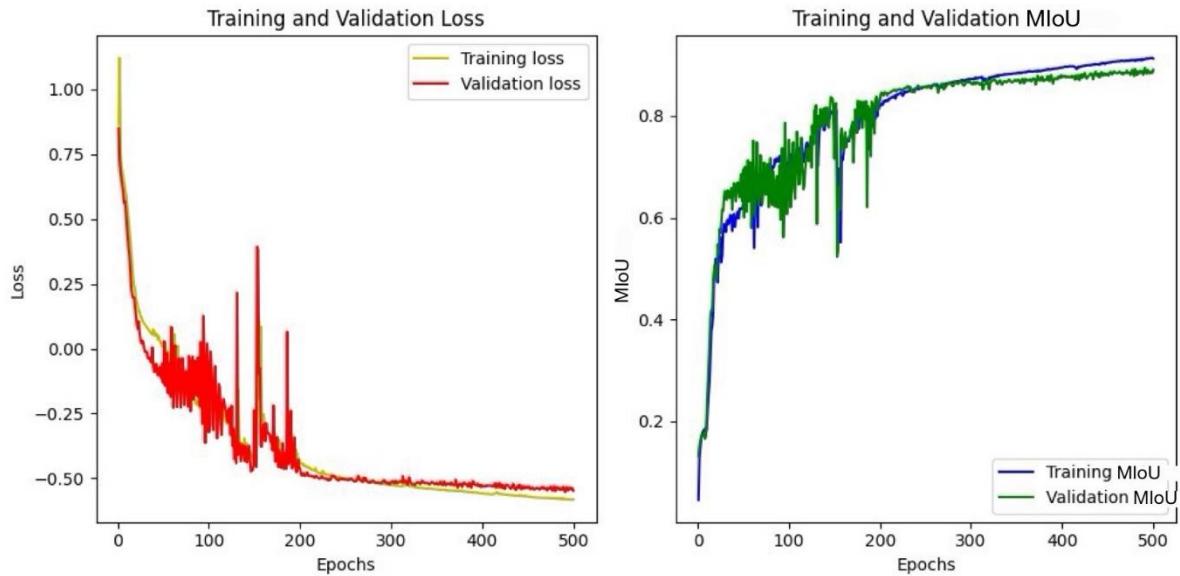
ตารางที่ 4.1.5.1 ตารางการประเมินผลตัวแบบเมนูผัดไทย (ครั้งที่ 1)

ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.5993	-0.5560	-0.5237	0.9267	0.8975	0.8726
2	-0.4344	-0.5022	-0.4430	0.8176	0.8545	0.8131
3	-0.5004	-0.5223	-0.4985	0.8633	0.8675	0.8563

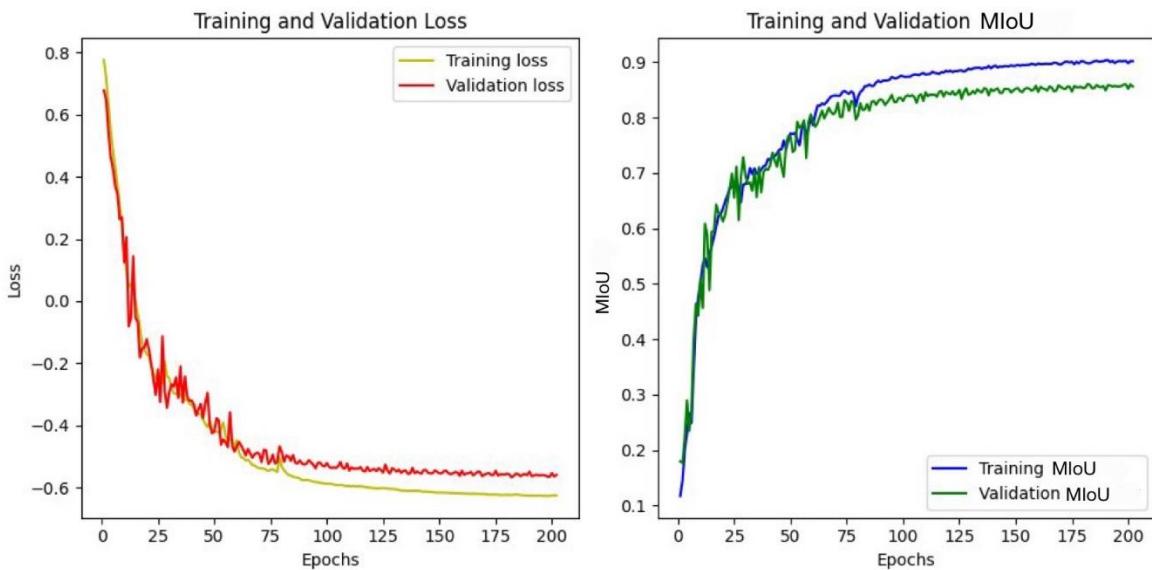
จากตารางที่ 4.1.5.1 พบร่วมกันว่าตัวแบบที่ 1 มีค่า Mean Intersection over Union ของข้อมูลชุดทดสอบสูงที่สุดที่ 0.8726 รองลงมาคือตัวแบบที่ 3 มีค่า 0.8563 และลำดับสุดท้ายคือตัวแบบที่ 2 มีค่า 0.8131



รูปที่ 4.1.5.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบงานของเมนูผัดไทย (ตัวแบบที่ 1)



รูปที่ 4.1.5.2 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมืองพัตไทย (ตัวแบบที่ 2)



รูปที่ 4.1.5.3 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมืองพัตไทย (ตัวแบบที่ 3)

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจสอบ พบร่วมแบบแบบไม่เกิด

ปรากฏการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและภาพเส้น ผู้ศึกษาได้เลือกตัวแบบที่ 1 เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำนายหน้าหากของข้อมูลชุดทดสอบเทียบกับหน้าหากจริง ดังนี้



รูปที่ 4.1.5.4 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้าหากจริง (กลาง) และหน้าหากที่ตัวแบบทำนาย (ขวา)

จากการเปรียบเทียบ พบว่าตัวแบบสามารถทำนายได้ดี แต่ทำนายกุ้ง (โปรตีน) ผิดพลาดในบางจุด

4.2) การพัฒนาครั้งที่ 2

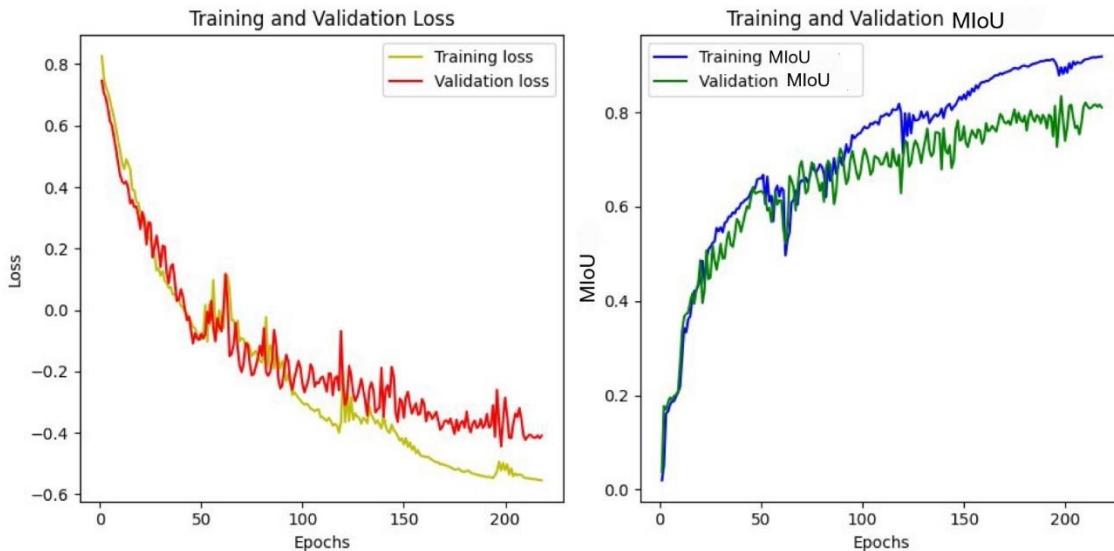
เมื่อทำตัวแบบจากการพัฒนาครั้งที่ 1 ไปทดลองทำนายอัตราส่วนโดยคณะสหเวชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย พบว่าตัวแบบเกิดปรากฏการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก ถึงแม้ว่าภาพเส้นจะไม่ปงบอกถึงปัญหาดังกล่าว เพราะว่าข้อมูลชุดทดสอบมีลักษณะเดียวกันกับข้อมูลชุดฝึก แต่ข้อมูลที่นำมาทดลองทำนายอัตราส่วนโดยคณะสหเวชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย มีลักษณะที่แตกต่างกันตามการตั้งค่าการถ่ายภาพ เช่น รูปภาพมีโทนสีชมพู เป็นต้น ดังนั้นจึงเกิดการพัฒนาตัวแบบครั้งที่ 2 เพื่อแก้ปัญหาดังกล่าว โดยมีการเพิ่มจำนวนข้อมูลชุดฝึกและเพิ่มวิธีการเพิ่มความหลากหลายของข้อมูล

ในการพัฒนาครั้งที่ 2 ผู้ศึกษาได้ฝึกฝนตัวแบบแยกตามเมนูอาหาร 3 เมนู ได้แก่ ข้าวคลุกกะปิ, ข้าวมันไก่, และผัดไทย โดยแต่ละเมนูจะฝึกฝนตัวแบบจำนวน 1 ตัวแบบ รวม 3 ตัวแบบ โดยมีการประเมินผลดังนี้

4.2.1) ข้าวคลุกกะปิ

ตารางที่ 4.2.1.1 ตารางการประเมินผลตัวแบบเมนูข้าวคลุกกะปิ (พัฒนาครั้งที่ 2)

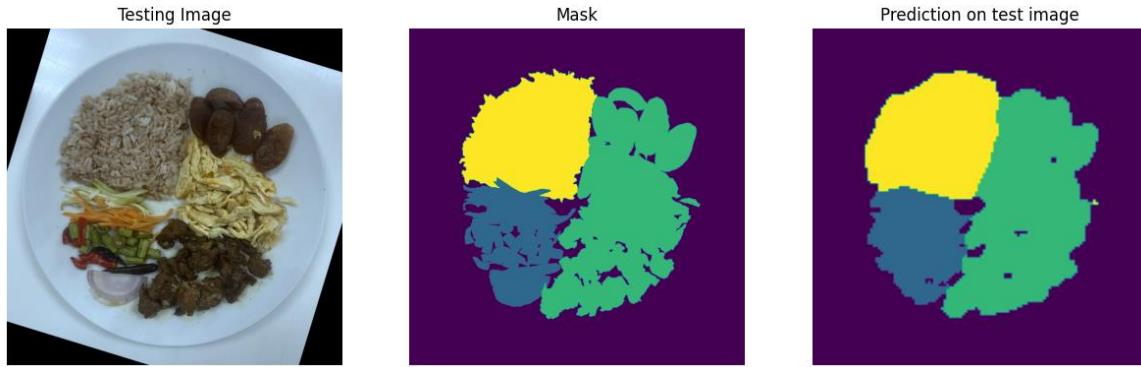
ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.5224	-0.4444	-0.4758	0.8950	0.8340	0.8674



รูปที่ 4.2.3.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวคลุกกะปิ

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจสอบ พบว่าตัวแบบเกิดปรากฏการณ์ตัวแบบที่เข้ากันได้ดีมากเกินไปกับข้อมูลชุดฝึก (Overfitting) เล็กน้อย แต่ไม่พบว่าตัวแบบไม่จำเพาะกับข้อมูล (Underfitting)

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและกราฟเส้น เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำนายหน้ากากของข้อมูลชุดทดสอบที่เป็นข้อมูลที่ถูกสุ่มขึ้นมาจากการเพิ่มความหลากหลายของข้อมูลเทียบกับหน้ากากจริง ดังนี้



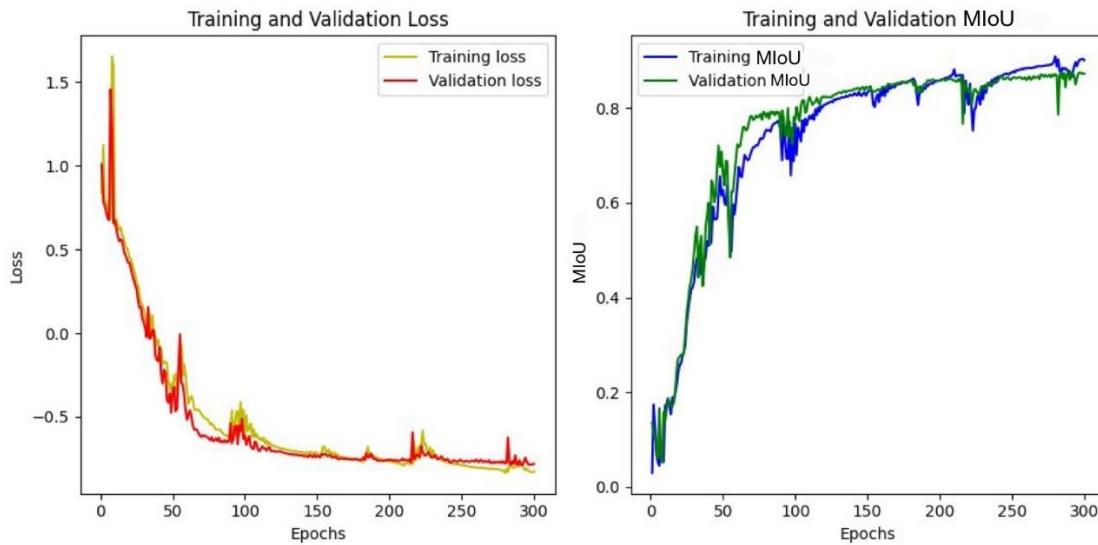
รูปที่ 4.2.1.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวคลุกกะปิ

จากการเปรียบเทียบ พบร่วมกันแบบสามารถทำนายได้ดี แต่อาจทำนายหัวหอม (ผัก) ผิดพลาดในบางจุด

4.2.2) ข้าวมันไก่

ตารางที่ 4.2.2.1 ตารางการประเมินผลตัวแบบเมนูข้าวมันไก่ (พัฒนาครั้งที่ 2)

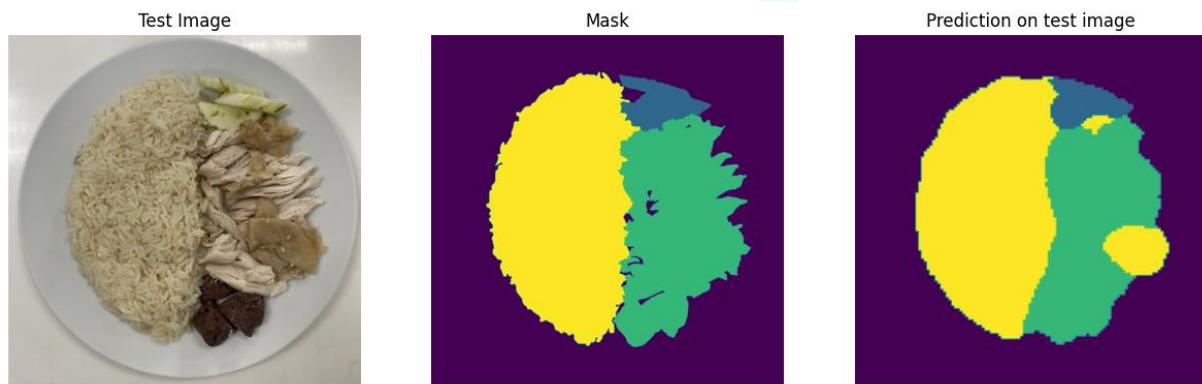
ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.5803	-0.5532	-0.4815	0.9120	0.8946	0.8479



รูปที่ 4.2.2.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูข้าวมันไก่

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจทาน พบร่วมกันแบบไม่เกิดประภูมิการณ์ตัวแบบที่เข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้ดี

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและภาพเส้น เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำนายหน้ากากของข้อมูลชุดทดสอบที่เป็นข้อมูลที่ถูกสุ่มขึ้นมาจากการเพิ่มความหลากหลายของข้อมูลเทียบกับหน้ากากจริง ดังนี้



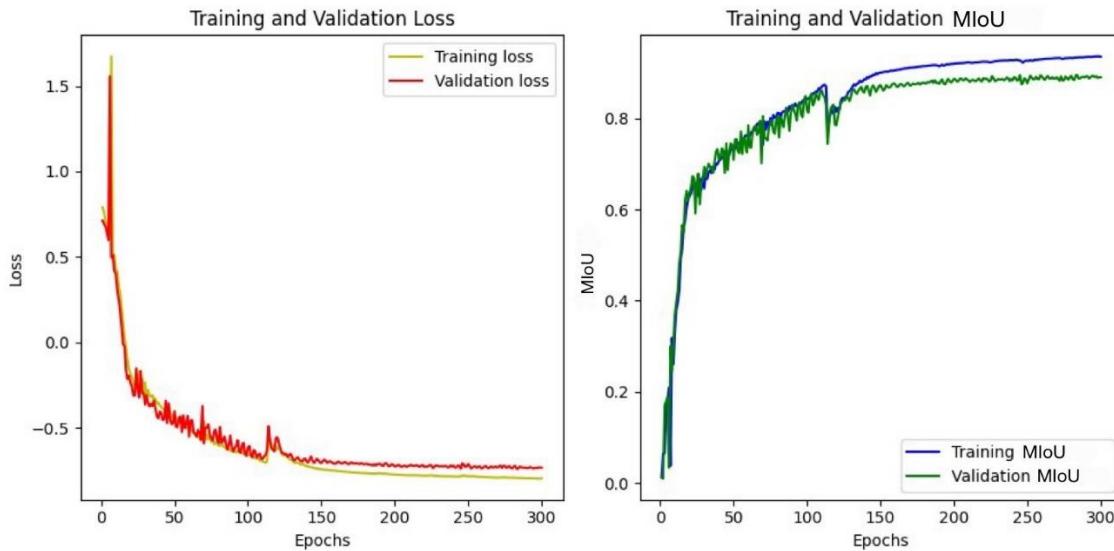
รูปที่ 4.2.3.2 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)

จากการเปรียบเทียบ พบว่าตัวแบบสามารถทำนายได้ดี แต่ทำนายหนังไก่ (โปรตีน) ผิดพลาดส่วนมาก

4.2.3) ก๋วยเตี๋ยวบางหมีเหลืองแห้ง

ตารางที่ 4.2.3.1 ตารางการประเมินผลตัวแบบเมนูก๋วยเตี๋ยวบางหมีเหลืองแห้ง (พัฒนารั้งที่ 2)

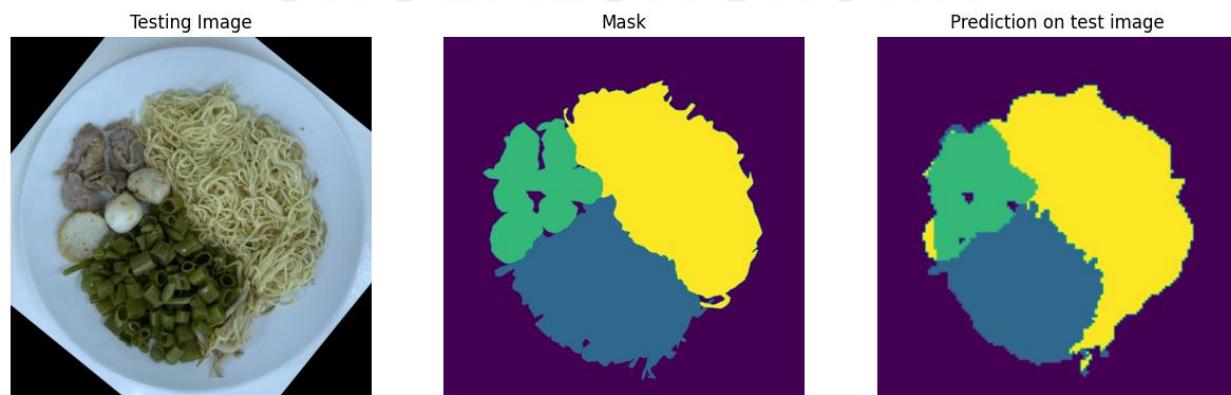
ตัวแบบที่	Train Loss	Validation Loss	Test Loss	Train MIoU	Validation MIoU	Test MIoU
1	-0.7927	-0.7384	-0.6927	0.9335	0.8948	0.8675



รูปที่ 4.2.3.1 กราฟแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) ของข้อมูลชุดฝึกและชุดตรวจสอบของเมนูก๋วยเตี๋ยวบะหมี่เหลืองแห้ง

จากการแสดงค่า Loss เทียบกับจำนวนรอบที่ฝึกฝน (Epoch) และกราฟแสดงค่า MIoU เทียบกับจำนวนรอบที่ฝึกฝนของข้อมูลชุดฝึกและชุดตรวจสอบ พบว่าตัวแบบแบบไม่เกิดปรากฏการณ์ตัวแบบที่เข้ากันได้ดีมากเกินไปกับข้อมูลชุดฝึก (Overfitting) และตัวแบบไม่จำเพาะกับข้อมูล (Underfitting) โดยตัวแบบสามารถเรียนรู้ได้ดี

จากการประเมินผลด้วยค่า Mean Intersection over Union ของข้อมูลชุดทดสอบและภาพเส้น เพื่อนำมาใช้งานจริง โดยมีการทดลองให้ตัวแบบทำนายหน้ากากของข้อมูลชุดทดสอบที่เป็นข้อมูลที่ถูกสุ่มขึ้นมาจากการเพิ่มความหลากหลายของข้อมูลเทียบกับหน้ากากจริง ดังนี้



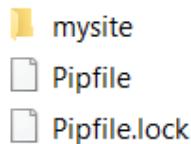
รูปที่ 4.2.3.2 เปรียบเทียบระหว่างรูปภาพ (ซ้าย) หน้ากากจริง (กลาง) และหน้ากากที่ตัวแบบทำนาย (ขวา)

จากการเปรียบเทียบ พบว่าตัวแบบสามารถทำนายได้ดี แต่ไม่สามารถทำนายถ้วงอก (ผัก) ได้ถูกต้อง และทำนายปลาเส้น (โปรดตีน) ผิดพลาดในบางจุด

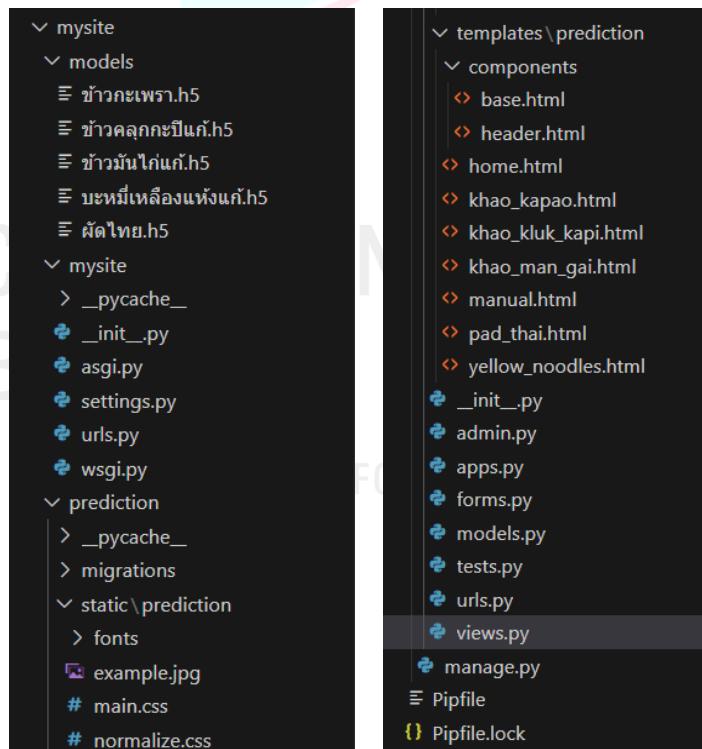
4.3) การนำตัวแบบไปใช้งาน

ตัวอย่างการนำตัวแบบไปใช้งานจริงคือ การให้บุคคลทั่วไปทดลองจัดจานอาหารใหม่อัตราส่วนโภชนาการอาหารใกล้เคียงกับแบบจำลองจากอาหารมากที่สุด จากนั้นให้ตัวแบบทำนายอัตราส่วนโภชนาการอาหารว่าบุคคลนั้นจัดจานอาหารได้ดีมากน้อยเพียงใด

โดยผู้ศึกษาได้จัดทำเว็บแอปพลิเคชันผ่าน Django Framework โดยใช้ pipenv เป็นสภาพแวดล้อมจำลองในการใช้งาน (Virtual environment)



รูปที่ 4.3.1 โฟลเดอร์เว็บแอปพลิเคชัน



รูปที่ 4.3.2 โครงสร้างโฟลเดอร์เว็บแอปพลิเคชัน

เริ่มต้นได้มีการสร้างแอปพลิเคชัน prediction และเพิ่มแอปพลิเคชันใหม่ที่ settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'prediction.apps.PredictionConfig',
]
```

รูปที่ 4.3.3 การเพิ่มแอปพลิเคชันใหม่ที่ settings.py

ต่อมา ได้มีการสร้างหน้าแรก (Home) ที่ views.py โดยจะแสดงส่วนต่อประสานกับผู้ใช้ตามที่ได้ออกแบบใน home.html

```
def home(request):
    return render(request, 'prediction/home.html')
```

รูปที่ 4.3.5 การสร้างหน้าแรกที่ views.py

```
{% extends 'prediction/components/base.html' %}
{% load static %}

{% block title %}หน้าแรก{% endblock %}

{% block content %}
    <main class="normal-content">
        <div>
            <h3 class="title">ໝາຍ</h3>
            <ul>
                <li style="margin-top: 2%; ">
                    <a href="{% url 'kaoKapao' %}">ຂ້າວກະເພາວ</a>
                </li>
                <li style="margin-top: 2%; ">
                    <a href="{% url 'kaoKluKapi' %}">ຂ້າວຄຸກກະປົກ</a>
                </li>
                <li style="margin-top: 2%; ">
                    <a href="{% url 'kaoManGai' %}">ຂ້າວນັ້ນໄກ</a>
                </li>
                <li style="margin-top: 2%; ">
                    <a href="{% url 'yellowNoodles' %}">ກຳຍເຕີຍວະໜີ່ເຫຼືອງແຮງ</a>
                </li>
                <li style="margin-top: 2%; ">
                    <a href="{% url 'padThai' %}">ຜັດໄທຍ</a>
                </li>
                <li style="margin-top: 2%; ">
                    <a href="{% url 'manual' %}">ຄູ່ມືອກາໃຈໜານ</a>
                </li>
            </ul>
        </div>
    </main>
{% endblock %}
```

รูปที่ 4.3.6 ไฟล์ Hypertext Markup Language (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าแรก



รูปที่ 4.3.7 หน้าจอหน้าแรกของเว็บแอปพลิเคชัน

ต่อมา สร้างหน้าทำนายอัตราส่วนโภชนาการอาหารแยกตามแต่ละเมนู โดยมีการเรียกใช้ฟอร์ม (Form) จาก forms.py เพื่อรับรูปภาพเข้ามาทำนาย

```
from django import forms

class ImageForm(forms.Form):
    image = forms.ImageField(label='เลือกรูปภาพ')
```

รูปที่ 4.3.8 ไฟล์ forms.py

CHULALONGKORN
BUSINESS SCHOOL

FLAGSHIP FOR LIFE

```

def khaoManGai(request):
    if request.method == 'POST':
        form = ImageForm(request.POST, request.FILES)
        if form.is_valid():
            uploaded_image = form.cleaned_data['image']
            image_data = uploaded_image.read()
            encoded_image = base64.b64encode(image_data)
            uploaded_image_base64 = encoded_image.decode("utf-8")

            image = cv2.imdecode(np.frombuffer(image_data, np.uint8), cv2.IMREAD_COLOR)

            model = load_model(os.path.join(settings.BASE_DIR, 'models', 'ข้าวมันไก่แกะ.h5'), compile=False)

            predicted_image = predictor(image, model)

            area = calculateProportion(predicted_image)
            area_percentage = {key + '_percentage': '{:0.2f}'.format(float(value) * 100) for key, value in area.items()}
            rFormat = ratioFormat(area_percentage)

            predicted_image = predicted_image.astype(np.uint8)
            resized_image = cv2.resize(predicted_image, (1280, 1280), interpolation=cv2.INTER_LANCZOS4)
            colored_predicted_image = apply_color_mapping(resized_image)
            _, buffer = cv2.imencode('.png', colored_predicted_image)
            predicted_image_base64 = base64.b64encode(buffer).decode("utf-8")

            context = {'form': form, 'image': uploaded_image_base64, 'prediction': predicted_image_base64, 'post': True, 'format': rFormat}
            context.update(area)
            context.update(area_percentage)
            return render(request, 'prediction\khao_man_gai.html', context)
    else:
        form = ImageForm()
        context = {'form': form}
    return render(request, 'prediction\khao_man_gai.html', context)

```

รูปที่ 4.3.9 การสร้างหน้าทำนายอัตราส่วนโภชนาการของข้าวมันไก่ที่ views.py

ต่อมา จะมีการนำเข้าตัวแบบที่ได้มีการฝึกฝนผ่าน load_model ซึ่งเป็นพังก์ชันของ keras

```
model = load_model(os.path.join(settings.BASE_DIR, 'models', 'ข้าวมันไก่แกะ.h5'), compile=False)
```

รูปที่ 4.3.10 คำสั่งที่ใช้ในการนำเข้าตัวแบบ

เมื่อมีการรับรูปภาพผ่านฟอร์มแล้ว ต่อมาจะเตรียมรูปภาพดังกล่าวเพื่อที่จะนำเข้าไปให้ตัวแบบทำนายโดยพังก์ชัน predictor ที่มีคำสั่ง ได้แก่ การลดขนาดลงเหลือ 128x128, การทำให้เป็นมาตราฐาน, การทำให้อยู่ในรูปแบบที่สามารถนำเข้าตัวแบบได้, นำรูปภาพนำเข้าไปให้ตัวแบบทำนาย, และส่งออกหน้าหากที่ถูกทำนาย

```

def predictor(img, model):
    X_size = 128
    y_size = 128
    image = []
    img = cv2.resize(img, (y_size, X_size))
    image.append(img)
    image = np.array(image)
    image = image / 255

    prediction = (model.predict(image))
    prediction_image = np.argmax(prediction, axis=3)[0]

    return prediction_image

```

รูปที่ 4.3.10 พังก์ชันคำสั่งที่ใช้ในการเตรียมรูปภาพและทำนาย

ต่อมา นำหน้าภาษาที่ถูกทำนายมาคำนวณค่าอัตราส่วนโภชนาการอาหารผ่านฟังก์ชัน calculateProportion เพื่อใช้แสดงผลบนหน้าเว็บแอปพลิเคชัน

```
def calculateProportion(mask):
    total_proportion = 0
    segment_areas = {}
    categories = ['Carbohydrate', 'Protein', 'Vegetable']

    for label in np.unique(mask):
        area = np.sum(mask == label)
        segment_areas[label] = area

    segment_areas = {key: segment_areas[key] for key in [1, 2, 3]}

    total_area = sum(segment_areas.values())

    for label, area in segment_areas.items():
        proportion = (area / total_area) * 100
        total_proportion += proportion
        print(f'{categories[label - 1]} Proportion = {proportion:.2f}')

    print(f'Total Proportion = {total_proportion}')
```

รูปที่ 4.3.11 ฟังก์ชันคำสั่งที่ใช้ในการคำนวณอัตราส่วนโภชนาการได้

ต่อมา นำค่าอัตราส่วนโภชนาการอาหารมาสร้างคำอธิบายประกอบ เพื่อความง่ายต่อการเข้าใจ โดยมีเกณฑ์ดังนี้

1) จานอาหารสมดุลตามอัตราส่วน 2:1:1

เมื่ออัตราส่วนทั้งสามชนิดโภชนาการมีค่าดังต่อไปนี้

- อัตราส่วนแป้งมีค่ามากกว่าหรือเท่ากับ 0.47 และน้อยกว่าหรือเท่ากับ 0.53
- อัตราส่วนโปรตีนมีค่ามากกว่าหรือเท่ากับ 0.22 และน้อยกว่าหรือเท่ากับ 0.28
- อัตราส่วนคาร์โบไฮเดรตมีค่ามากกว่าหรือเท่ากับ 0.22 และน้อยกว่าหรือเท่ากับ 0.28

2) จานอาหารมีส่วนของ (ชนิดโภชนาการ) น้อยกว่าอัตราส่วนที่ควรจะเป็น

- อัตราส่วนแป้งมีค่าน้อยกว่าหรือเท่ากับ 047
- อัตราส่วนโปรตีนมีค่าน้อยกว่าหรือเท่ากับ 0.22
- อัตราส่วนคาร์โบไฮเดรตมีค่าน้อยกว่าหรือเท่ากับ 0.22

3) จานอาหารมีส่วนของ (ชนิดโภชนาการ) มากกว่าอัตราส่วนที่ควรจะเป็น

- อัตราส่วนแป้งมีค่ามากกว่าหรือเท่ากับ 0.53
- อัตราส่วนโปรตีนมีค่ามากกว่าหรือเท่ากับ 0.28
- อัตราส่วนคาร์โบไฮเดรตมีค่ามากกว่าหรือเท่ากับ 0.28

```

def ratioFormat(area_dict):
    rFormat = []
    if 23 <= float(area_dict['Carbohydrate_percentage']) <= 28 and 23 <= float(area_dict['Protein_percentage']) <= 28:
        rFormat.append("จำนวนอาหารสมดุลตามอัตราส่วน 2:1:1")
    else:
        if float(area_dict['Carbohydrate_percentage']) > 28:
            rFormat.append('จำนวนอาหารมีส่วนของข้าวແປ່ງมากກວ່າอัตราส่วนที่ควรจะเป็น')
        elif float(area_dict['Carbohydrate_percentage']) < 22:
            rFormat.append('จำนวนอาหารมีส่วนของข้าวແປ່ງນ้อยກວ່າอัตราส่วนที่ควรจะเป็น')
        if float(area_dict['Protein_percentage']) > 28:
            rFormat.append('จำนวนอาหารมีส่วนของเนื้อสัตว์มากກວ່າอัตราส่วนที่ควรจะเป็น')
        elif float(area_dict['Protein_percentage']) < 22:
            rFormat.append('จำนวนอาหารมีส่วนของเนื้อสัตว์น้อยກວ່າอัตราส่วนที่ควรจะเป็น')
        if float(area_dict['Vegetable_percentage']) > 53:
            rFormat.append('จำนวนอาหารมีส่วนของผักมากກວ່າอัตราส่วนที่ควรจะเป็น')
        elif float(area_dict['Vegetable_percentage']) < 47:
            rFormat.append('จำนวนอาหารมีส่วนของผักน้อยກວ່າอัตราส่วนที่ควรจะเป็น')
    return rFormat

```

รูปที่ 4.3.12 พังก์ชันคำสั่งที่ใช้ในการสร้างคำอธิบายประกอบอัตราส่วน

ต่อมา นำหน้าหากำที่ถูกทำนาย ค่าอัตราส่วนโภชนาการอาหาร และคำอธิบาย แสดงผลบนหน้าเว็บแอปพลิเคชันผ่าน Hypertext Markup Language (HTML) ดังนี้

```

{% extends 'prediction/components/base.html' %}
{% load static %}

{% block title %}ข้าวมันไก่{% endblock %}

{% block content %}
    <main class="normal-content">
        <div style="margin-top: 2%;">
            <h1>ข้าวมันไก่</h1>
            <form id="image" method="POST" enctype="multipart/form-data">
                <div class="button button-primary" style="margin-top: 1%; ">
                    {% csrf_token %}
                    {{ form.image.label_tag }}
                    {{ form.image }}
                    <script>
                        document.getElementById("id_image").addEventListener("change", function() {
                            document.getElementById("image").submit();
                        });
                    </script>
                </div>
            </form>
        </div>
    </main>

```

รูปที่ 4.3.13 ไฟล์ Hypertext Markup Language (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าทำนายอัตราส่วนโภชนาการอาหารของเมนูข้าวมันไก่

```

{%- if post %}
    <div style="background-color: #FFFFFF; max-width: 40%; border-radius: 1em;">
        <h3 style="color: #027683; margin-left: 5%;">ส่วนเป็น = ข้าว澎湃</h3>
    </div>
    <div style="background-color: #FFFFFF; max-width: 40%; border-radius: 1em; margin-top: 0.5%;">
        <h3 style="color: #CF3140; margin-left: 5%;">สีแดง = เนื้อสัตว์</h3>
    </div>
    <div style="background-color: #FFFFFF; max-width: 40%; border-radius: 1em; margin-top: 0.5%;">
        <h3 style="color: #57A370; margin-left: 5%;">สีเขียว = ผัก</h3>
    </div>
    <div style="margin-top: 2%;">
        <div>
            {% if image %}
                
            {% endif %}
            {% if prediction %}
                
            {% endif %}
        </div>
    </div>
</div>

```

รูปที่ 4.3.14 ไฟล์ Hypertext Markup Language (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าทำนายอัตราส่วน
โภชนาการอาหารของเมนูข้าวมันไก่ (ต่อ)

```

<div>
    <div>
        <h3>ผลการท่านายสัดส่วน</h3>
        <h3>ข้าว澎湃 = {{ Carbohydrate }} หรือ {{ Carbohydrate_percentage }}%</h3>
        <h3>เนื้อสัตว์ = {{ Protein }} หรือ {{ Protein_percentage }}%</h3>
        <h3>ผัก = {{ Vegetable }} หรือ {{ Vegetable_percentage }}%</h3>
    </div>
</div>
<div>
    <h1>-----</h1>
</div>
<div>
    <div>
        <h3>คำแนะนำ (รูปแบบอัตราส่วน)</h3>
        {% for item in format %}
            <h3>{{ item }}</h3>
        {% endfor %}
    </div>
    {% endif %}
</div>
</main>
{% endblock %}

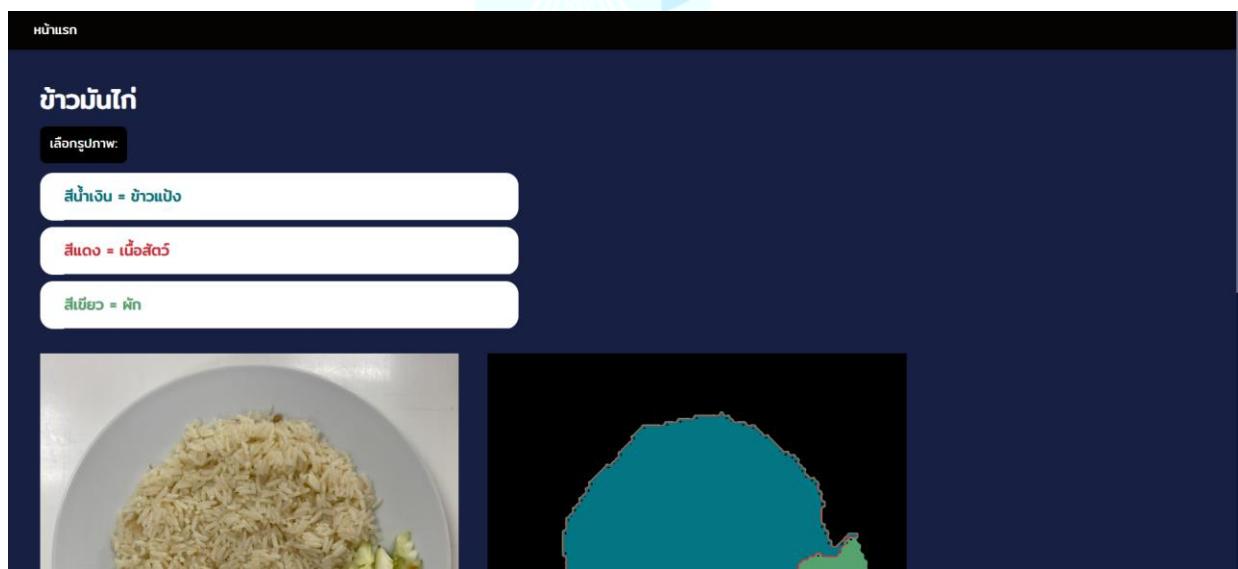
```

รูปที่ 4.3.15 ไฟล์ Hypertext Markup Language (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้ของหน้าทำนายอัตราส่วน
โภชนาการอาหารของเมนูข้าวมันไก่ (ต่อ)

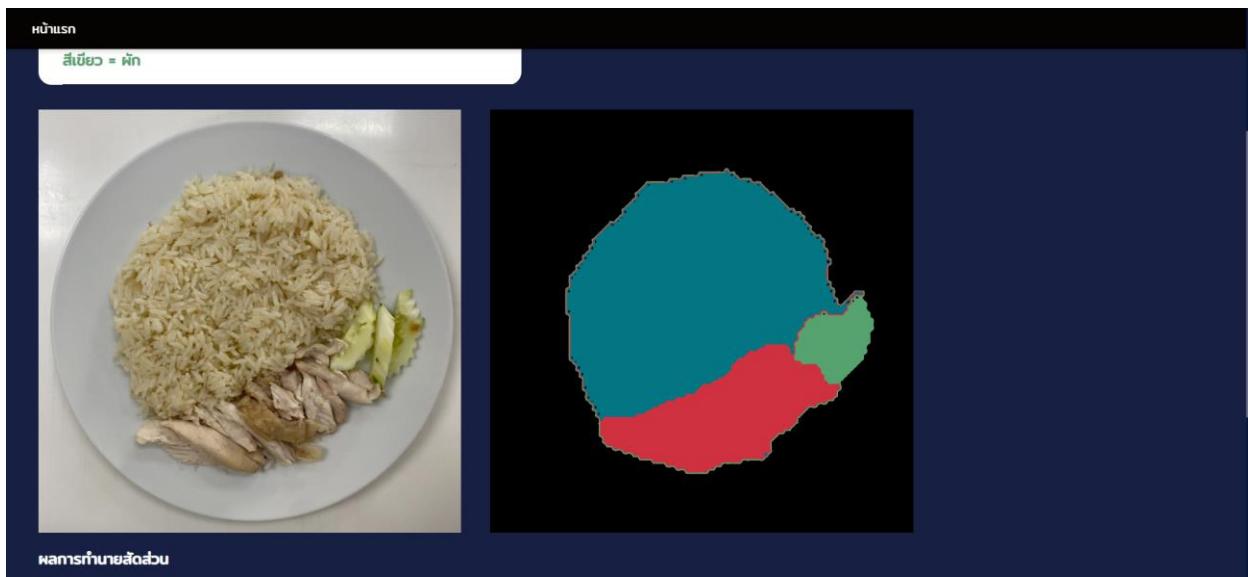
ตัวอย่างหน้าจอหน้าทามนายอัตราส่วนโภชนาการอาหารของเมนูข้าวมันไก่ ดังนี้



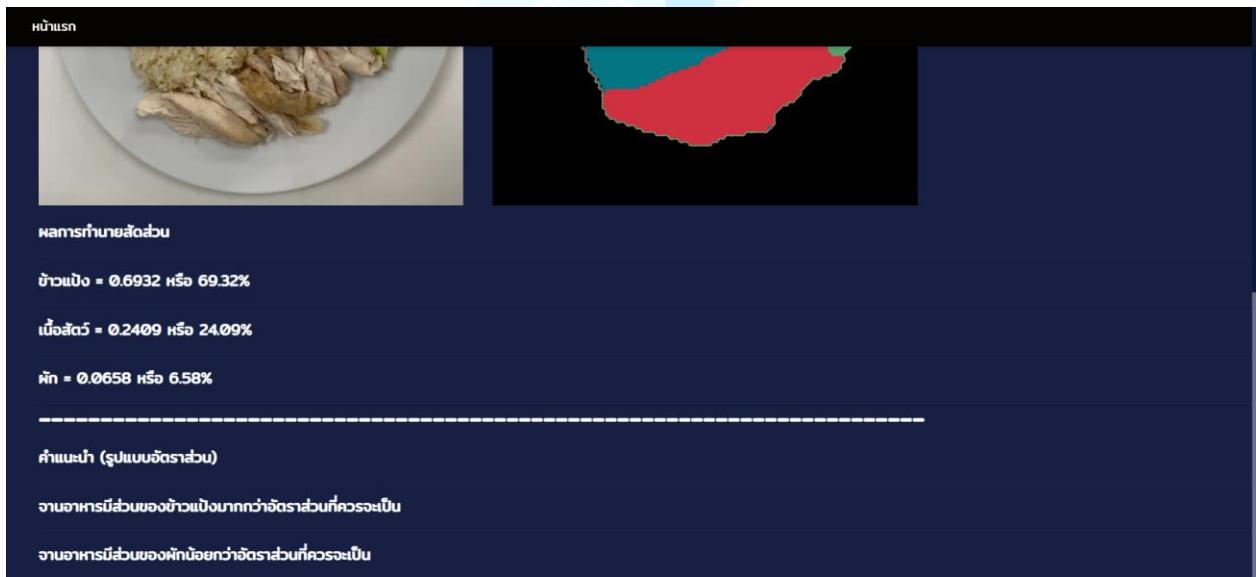
รูปที่ 3.4.16 หน้าจอหน้าทามนายอัตราส่วนโภชนาการอาหาร



รูปที่ 3.4.17 หน้าจอหน้าทามนายอัตราส่วนโภชนาการอาหาร (ต่อ)



รูปที่ 3.4.18 หน้าจอหน้าทำนายอัตราส่วนโภชนาการอาหาร (ต่อ)



รูปที่ 3.4.19 หน้าจอ หน้าจอนำเสนอการทำนายอัตราส่วนโภชนาการอาหาร (ต่อ)

ส่วนสุดท้ายของเว็บแอปพลิเคชันคือ คู่มือการใช้งานเว็บแอปพลิชัน มีการอธิบายถึงวัตถุประสงค์ ของเว็บแอปพลิชัน วิธีการใช้งาน และหมายเหตุ

```
def manual(request):
    return render(request, 'prediction\manual.html')
```

รูปที่ 4.3.20 การสร้างหน้าคู่มือการใช้งานที่ views.py

```

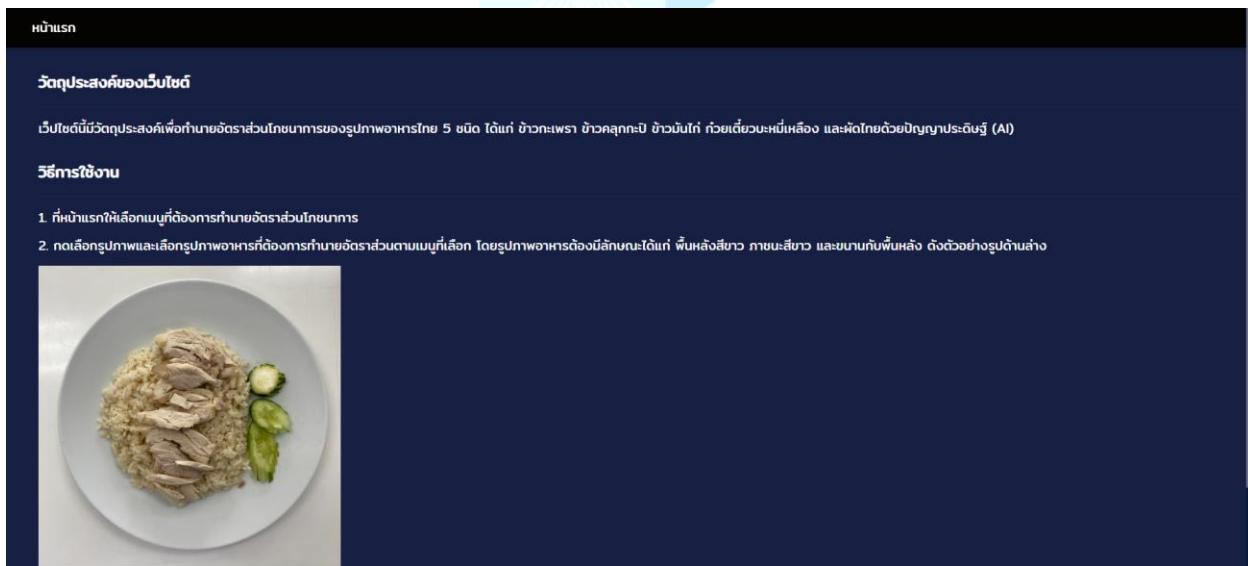
{% extends 'prediction/components/base.html' %}
{% load static %}

{% block title %}คู่มือการใช้งาน{% endblock %}

{% block content %}
    <main class="normal-content">
        <div>
            <h3 class="title">วัดคุณประสิทธิ์ของเว็บไซต์</h3>
            <div>
                <p>เว็บไซต์นี้มีวัดคุณประสิทธิ์เพื่อท่านนายอัครส่วนโภชนาการของรูปภาพอาหารไทย 5 ชนิด ได้แก่ ข้าว胭พร้า ข้าวคลุกกะปิ ข้าวบันไก่ กวยเตี๋ยวหมี่เหลือง และผัดไท</p>
                <h3>ธีมการใช้งาน</h3>
                <p>1. ที่หน้าเรียกใช้ล็อกเมนูที่ต้องการท่านนายอัครส่วนโภชนาการ</p>
                <p>2. กดเลือกรูปภาพและเลือกรูปภาพอาหารที่ต้องการท่านนายอัครส่วนเมนูที่เลือก โดยรูปภาพอาหารต้องมีลักษณะได้แก่ หัวหลังสีขาว ภาชนะสีขาว และขนาดก้น</p>
                <p></p>
                <p>หลังจากเลือกรูปภาพแล้วให้รออัปโหลดรูปภาพอาหาร รูปภาพที่ถูกท่านนายอัครส่วนโภชนาการ ต้องมีลักษณะไม่ตรงตามที่กดล่าสุด เช่น หัวหลังสีเขียว การท่านนายอัครส่วนอาหารคลื่อนและไม่แน่นยำ</p>
                </div>
                <ul>
                    <li style="margin-top: 2%;>
                        <a href="{% url 'home' %}" class="button button-primary">ย้อนกลับ</a>
                    </li>
                </ul>
            </div>
        </main>
    {% endblock %}

```

รูปที่ 4.3.21 ไฟล์ Hypertext Markup Language (HTML) ใช้แสดงส่วนต่อประสานกับผู้ใช้งานหน้าคู่มือการใช้งาน



รูปที่ 4.3.22 หน้าจอหน้าคู่มือการใช้งาน

หน้าแรก

วิธีการใช้งาน

1. ที่หน้าแรกให้เลือกเมนูที่ต้องการกรอกข้อมูล เช่น กินอาหาร ดื่มน้ำ ฯลฯ

2. กดเลือกรูปภาพและเลือกรูปภาพอาหารที่ต้องการกรอกข้อมูล เช่น กินอาหาร ดื่มน้ำ ฯลฯ

3. หลังจากเลือกรูปภาพแล้วให้รอสักครู่ ระบบจะแสดงรูปภาพที่ถูกกรอกข้อมูลแล้ว เช่น กินอาหาร ดื่มน้ำ ฯลฯ

หมายเหตุ: ถ้าหากรูปภาพที่เป็นมาใหม่ต้องรอนานกว่าเดิม ให้ลองลบไฟล์เดิมแล้วอัปโหลดใหม่

บันทึก

รูปที่ 4.3.23 หน้าจอหน้าคู่มือการใช้งาน (ต่อ)

CHULALONGKORN
BUSINESS SCHOOL
FLAGSHIP FOR LIFE

บทที่ 5 บทสรุป

5.1) สรุปผลการศึกษา

ผู้ศึกษาได้เรียนรู้ตามวัตถุประสงค์ที่ได้กล่าวไว้ข้างต้น ดังนี้

- เพื่อศึกษาเครื่องมือที่ใช้ในการพัฒนาตัวแบบระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร

ผู้ศึกษาได้มีความเข้าใจในทฤษฎีที่ใช้ในการพัฒนาตัวแบบระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร ได้แก่ การเรียนรู้เชิงลึก โครงข่ายประสาท เที่ยมแบบคอนโวโลชัน และการแบ่งส่วนความหมายของรูปภาพ อีกทั้งยังได้มีความเข้าใจในเครื่องมือ ได้แก่ แมตแล็บ และไลบรารีในภาษา Python ที่เกี่ยวข้อง

- เพื่อศึกษาลำดับวิธีการดำเนินการพัฒนาตัวแบบ ได้แก่ การสร้างป้ายกำกับระดับพิกเซล การวิเคราะห์ข้อมูล การเตรียมข้อมูลเพื่อนำไปฝึกฝนตัวแบบ การพัฒนาตัวแบบ การประเมินผลตัวแบบ และการนำตัวแบบไปใช้งานจริง

ผู้ศึกษาสามารถเรียนรู้และดำเนินตามลำดับวิธีการพัฒนาตัวแบบ ได้แก่ การสร้างป้ายกำกับระดับพิกเซลโดยแมตแล็บ การใช้ไลบรารีในภาษา Python ในการวิเคราะห์ข้อมูล สร้างความหลากหลายของข้อมูล เตรียมข้อมูล การพัฒนาตัวแบบ และการประเมินผลตัวแบบ และการใช้ Django ในการพัฒนาเว็บแอปพลิเคชันเพื่อนำตัวแบบไปใช้งานจริง

- เพื่อพัฒนาตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลองอาหาร

ผู้ศึกษาสามารถพัฒนาตัวแบบที่สามารถระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทย ตามหลักการของแบบจำลองอาหาร และได้นำตัวแบบให้คณบดี ศาสตราจารย์ จุฬาลงกรณ์ มหาวิทยาลัยทดลองใช้งาน ตัวแบบสามารถระบุอัตราส่วนโภชนาการได้ดี แต่ยังคงอยู่ภายใต้เงื่อนไขที่จำกัด

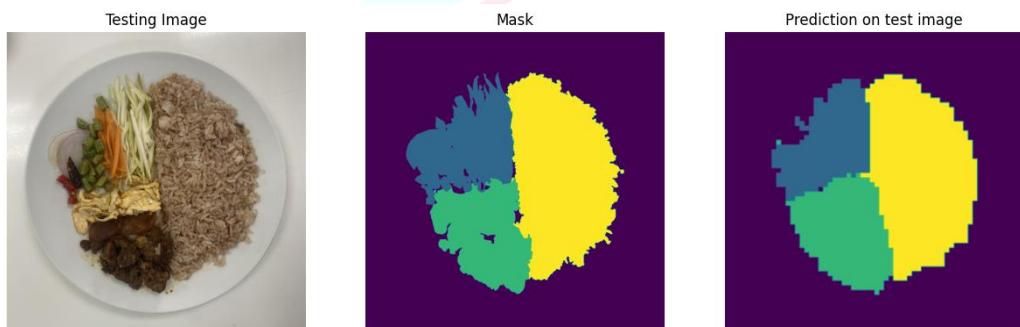
5.2) ข้อจำกัด และปัญหาที่เกิดขึ้น

1) จำนวนข้อมูลที่น้อยเกินไป

ข้อมูลรูปภาพที่นำมาพัฒนาตัวแบบครั้งที่ 1 มีจำนวน 67 รูปภาพ และครั้งที่ 2 มีจำนวน 76 รูปภาพ และทุกรูปภาพมีลักษณะที่เหมือนกัน ได้แก่ ภาชนะสีขาว และพื้นหลังสีขาว ส่งผลให้ตัวแบบที่ถูกฝึกฝนมีโอกาสเข้ากันได้มากเกินไปกับข้อมูลชุดฝึก (Overfitting) และมีความลำเอียง (Bias) ทำให้ตัวแบบมีโอกาสทำนายรูปภาพที่มีลักษณะนอกเหนือจากข้อมูลชุดฝึกได้ไม่แม่นยำ ใน การแก้ปัญหาเบื้องต้น ผู้ศึกษาได้ใช้วิธีเพิ่มความหลากหลายของข้อมูล เพื่อเพิ่มจำนวนรูปภาพ

2) ระยะเวลาการฝึกฝนตัวแบบที่นานเกินไป

เหตุผลที่ผู้ศึกษาพัฒนาตัวแบบแยกตามเมนูอาหาร แทนที่จะรวมทุกเมนูอาหารในตัวแบบเดียว เพราะว่าในการพัฒนาตัวแบบรวมทุกเมนูอาหารใช้ระยะเวลาที่ยาวนานมาก ในการแก้ไขเบื้องต้น ผู้จัดทำได้ลดขนาดรูปภาพที่นำไปฝึกฝนตัวแบบลงเหลือ 64×64 แต่ผลลัพธ์หน้าหากที่ถูกทำนายโดยตัวแบบมีความละเอียดที่หยาบ (มองเห็นพิกเซลชัดเจน) และยังคงใช้ระยะเวลาฝึกตัวแบบที่ยาวนานกว่าฝึกฝนตัวแบบแยกตามเมนู จากปัญหาดังกล่าวผู้ศึกษาจึงตัดสินใจพัฒนาตัวแบบแยกตามเมนูอาหาร



รูปที่ 5.2.1 ผลลัพธ์ของตัวแบบรวมทุกเมนูอาหารที่ถูกฝึกด้วยรูปภาพขนาด 64×64

3) ตัวแบบเข้ากันได้มากเกินไปกับข้อมูลชุดฝึก และมีความลำเอียง

ตัวแบบที่ถูกพัฒนาในครั้งที่ 1 เมื่อนำไปทดสอบโดยคณะสหเวชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย พบร่วมกับตัวแบบทำนายรูปภาพที่ถูกถ่ายในสภาพแวดล้อมที่แตกต่างจากข้อมูลชุดฝึกได้ไม่แม่นยำ ตัวอย่างเช่น รูปภาพที่ถูกถ่ายโดยโทรศัพท์มือถือรุ่นซัมซุงจะมีความอิ่มตัวที่สูงกว่าปกติ ทั้งนี้อาจเป็นเพราะโทรศัพท์มือถือแต่ละเครื่องมีการตั้งค่ากล้องถ่ายรูปแตกต่างกันออกไป ในการแก้ไขปัญหาเบื้องต้น ผู้ศึกษาได้เพิ่มวิธีในการเพิ่มความหลากหลายของข้อมูล เช่น เพิ่มความอิ่มตัว, เพิ่มความคมชัด เป็นต้น

- 4) ตัวแบบสามารถทำนายได้เฉพาะพื้นที่สามารถมองเห็นเท่านั้น ไม่สามารถทำนายพื้นที่มองไม่เห็น
ตัวแบบการระบุอัตราส่วนโภชนาการของรูปภาพอาหารไทยตามหลักการของแบบจำลอง
งานอาหารสามารถระบุอัตราส่วนได้เฉพาะพื้นที่ที่สามารถมองเห็นได้เท่านั้น แต่ไม่สามารถทำนาย
พื้นที่มองไม่เห็น ตัวอย่างเช่น ในการทำนายหน้ากากของเมนูข้าวมันไก่ หากลักษณะของงาน
อาหารจริงคือไก่ทับบนข้าว และรูปภาพที่นำมาทดสอบถูกถ่ายขนาดกับพื้นหลัง ส่งผลให้มองเห็น
แค่ไก่ในพื้นที่ดังกล่าว ตัวแบบจะสามารถทำนายได้เฉพาะไก่ที่อยู่บนข้าวเท่านั้น ไม่สามารถทำนาย
ข้าวที่อยู่ใต้ไก่ได้

5) การนำตัวแบบไปใช้งาน

จากข้อจำกัดที่ 4 รูปภาพที่นำมาให้ตัวแบบทำนายต้องมีการแยกส่วนของโภชนาการให้
ชัดเจนตัวแบบถึงจะทำนายได้ดี หากนำไปใช้งานกับรูปภาพจากร้านอาหารทั่วไปที่ส่วนของ
โภชนาการมีความซับซ้อนจะทำให้ตัวแบบทำนายได้ไม่ดี เพราะฉะนั้นตัวแบบนี้เหมาะสมกับบุคคล
ที่จดจำอาหารเองและต้องการทราบว่าจานอาหารที่ตนเองนั้นเป็นไปตามแบบจำลองงานอาหาร
หรือไม่

5.3) ข้อเสนอแนะ

1) การเพิ่มจำนวนและความหลากหลายของข้อมูล

ในการพัฒนาครั้งต่อไป ควรมีการเก็บข้อมูลรูปภาพให้มากขึ้น และมีลักษณะหรือ
สภาพแวดล้อมที่ครอบคลุม เพราžeข้อมูลรูปภาพที่นำมาพัฒนาตัวแบบมีจำนวน 76 รูปภาพ มี
ลักษณะมีเหมือนกัน ได้แก่ ภาชนะสีขาว และพื้นหลังสีขาว

2) การเพิ่มวิธีในการเพิ่มความหลากหลายของข้อมูล

ในโครงการครั้งนี้ผู้ศึกษาได้เลือกวิธีในการเพิ่มความหลากหลายของข้อมูล เช่น การหมุน,
การลดอุณหภูมิ, การเพิ่มความคมชัด เป็นต้น ใน การพัฒนาครั้งต่อไป อาจทดลองเพิ่มวิธีในการเพิ่ม
ความหลากหลายของข้อมูลให้มากขึ้น เช่น เบลอภาพ, เพิ่มอุณหภูมิ เป็นต้น เพื่อให้ตัวแบบสามารถ
เรียนรู้รูปภาพได้หลากหลายสภาพแวดล้อมมากขึ้น

3) การปรับเปลี่ยนสถาปัตยกรรมโครงข่ายประสาทเทียม

ในโครงการครั้งนี้ ผู้ศึกษาได้เลือกใช้สถาปัตยกรรมโครงข่ายประสาทเทียม U-Net ที่มี
VGG16 เป็นกระดูกสันหลังในการพัฒนาตัวแบบ ในการพัฒนาครั้งต่อไป อาจทดลองเลือกใช้

สถาปัตยกรรมโครงข่ายประสาทเทียนอื่น ๆ เช่น LinkNet, PSPNet เป็นต้น หรือเลือกใช้กราดูกสันหลังอื่น ๆ เช่น Inception, Xception, ResNet เป็นต้น

4) การเรียนรู้เชิงถ่ายทอด (Transfer Learning)

ในการพัฒนาครั้งต่อไป อาจพัฒนาตัวแบบโดยใช้วิธีการเรียนรู้เชิงถ่ายทอด กล่าวคือเป็นการเรียนรู้ของเครื่องที่นำตัวแบบที่ถูกฝึกฝนไว้แล้ว (Pre-Trained Model) มาปรับใช้ (Fine-Tune) กับชุดข้อมูลของเรา โดยมีการปรับค่าพารามิเตอร์การเรียนรู้แค่บางส่วนในกระบวนการฝึกฝน ส่งผลให้ช่วยประหยัดเวลาในการฝึกฝนตัวแบบ



บรรณานุกรม

ภาษาไทย

nessesence. การเรียนรู้ของเครื่อง (Machine Learning) และการเรียนรู้เชิงลึก (Deep Learning): ความแตกต่างระหว่างสองสิ่งนี้? [ออนไลน์]. 2561. แหล่งที่มา: <https://www.thaiprogrammer.org/2018/12/การเรียนรู้ของเครื่องmachine-le/> [2566, สิงหาคม]

Natthawat Phongchit. Convolutional Neural Network (CNN) คืออะไร. [ออนไลน์]. 2561. แหล่งที่มา: <https://medium.com/@natthawatphongchit/มาลองดูวิธีการคิดของ-cnn-กัน-e3f5d73eebaa> [2566, สิงหาคม]

Surapong Kanoktipsatharporn. Data Augmentation คืออะไร ประโยชน์ของ Data Augmentation ในการเทรน Deep Learning - Regularization ep.1. [ออนไลน์]. 2562. แหล่งที่มา: <https://medium.com/@natthawatphongchit/มาลองดูวิธีการคิดของ-cnn-กัน-e3f5d73eebaa> [2566, พฤษภาคม]

ภาษาอังกฤษ

The American Diabetes Association. What is the Diabetes Plate Method? [Online]. 2020. Available from: <https://www.diabetesfoodhub.org/articles/what-is-the-diabetes-plate-method.html> [2023, August]

Larry Hardesty. Explained: Neural networks. [Online]. 2017. Available from: <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414> [2023, August]

Sumit Saha. A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way. [Online]. 2018. Available from: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> [2023, August]

- Zoumana Keita. An Introduction to Convolutional Neural Networks (CNNs). [Online]. 2023. Available from: <https://www.datacamp.com/tutorial/introduction-to-convolutional-neural-networks-cnns> [2023, December]
- Ting-Hao Chen. What is “stride” in Convolutional Neural Network? [Online]. 2017. Available from: <https://medium.com/machine-learning-algorithms/what-is-stride-in-convolutional-neural-network-e3b4ae9baedb> [2023, September]
- Ashish Sanjay Raut. Padding in Neural Networks: Why and How? [Online]. 2023. Available from: <https://blog.gopenai.com/padding-in-neural-networks-why-and-how-b076ab0a4fc2> [2023, September]
- savyakhosla. CNN | Introduction to Pooling Layer. [Online]. 2023. Available from: <https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/> [2023, September]
- DigitalSreeni. 73 - Image Segmentation using U-Net - Part1 (What is U-net?). [Online]. 2019. Available from: https://youtu.be/azM57JuQpQI?si=BnJ7itSvdCUv_yDm [2023, September]
- DigitalSreeni. 74 - Image Segmentation using U-Net - Part 2 (Defining U-Net in Python using Keras). [Online]. 2019. Available from: https://youtu.be/68HR_eyzk00?si=wkx44sGQVkuxxgw5 [2023, September]
- DigitalSreeni. 75 - Image Segmentation using U-Net - Part 3 (What are trainable parameters?). [Online]. 2019. Available from: <https://youtu.be/sb0uglcqO2Y?si=V1VXlitw2-RhAgOP> [2023, September]
- DigitalSreeni. 76 - Image Segmentation using U-Net - Part 4 (Model fitting, checkpoints, and callbacks). [Online]. 2019. Available from: <https://youtu.be/0kiroPnV1tM?si=hb2AdkonkPdBLxAZ> [2023, September]
- DigitalSreeni. 77 - Image Segmentation using U-Net - Part 5 (Understanding the data). [Online]. 2019. Available from: https://youtu.be/cUHPL_dk17E?si=7sFAy4EL4vKir6oN [2023, September]

DigitalSreeni. [78 - Image Segmentation using U-Net - Part 6 \(Running the code and understanding results\)](#). [Online]. 2019. Available from: <https://youtu.be/RaswBvMnFvk?si=7-m8H1dl3BWs2tJV> [2023, September]

Karol Majek. [Tensorflow DeepLab v3 Xception Cityscapes](#). [Online]. 2018. Available from: <https://www.youtube.com/watch?v=ATlcEDSPWXY> [2023, November]

Derrick Mwiti, Katherine (Yi) Li. [Image Segmentation: Architectures, Losses, Datasets, and Frameworks](#). [Online]. 2023. Available from: <https://neptune.ai/blog/image-segmentation> [2023, November]

Surya Teja Menta. [Upsampling and Transposed Convolutions Layers](#). [Online]. 2022. Available from: <https://medium.com/@suryatejamenta/upsampler-and-convolution-transpose-layers-7e4346c05bb6> [2023, November]

Nilesh Barla. [The Beginner's Guide to Semantic Segmentation](#). [Online]. 2021. Available from: <https://www.v7labs.com/blog/semantic-segmentation-guide> [2023, November]

Yanming Guo, Yu Liu, Theodoros Georgiou, and Michael S. Lew. A review of semantic segmentation using deep neural networks. 2017

Naveen Joshi. [How Deep Learning Makes Semantic Segmentation More Precise](#). [Online]. 2021. Available from: <https://www.linkedin.com/pulse/how-deep-learning-makes-semantic-segmentation-more-precise-joshi/> [2023, November]

Jeremy Jordan. [An overview of semantic image segmentation](#). [Online]. 2018. Available from: <https://www.jeremyjordan.me/semantic-segmentation/> [2023, November]

Khuyen Le. [An overview of VGG16 and NiN models](#). [Online]. 2018. Available from: <https://lekhuyen.medium.com/an-overview-of-vgg16-and-nin-models-96e4bf398484> [2023, November]

CYBORG NITR. MIoU Calculation. [Online]. 2020. Available from:

<https://medium.com/@cyborg.team.nitr/miou-calculation-4875f918f4cb> [2023, December]

Deval Shah. Intersection over Union (IoU): Definition, Calculation, Code. [Online]. 2023. Available from: <https://www.v7labs.com/blog/intersection-over-union-guide> [2023, December]

ritvikmath. How this Little Matrix Sharpens your Images. [Online]. 2023. Available from: <https://www.youtube.com/watch?v=LCo69MDCGMs> [2024, February]

