การจำแนกความเสียหายรถยนต์โดยการเรียนรู้เชิงลึก

CAR DAMAGE CLASSIFICATION USING DEEP LEARNING

ธนัช เบญจอนุอาชา

บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

2564

การจำแนกความเสียหายรถยนต์โดยการเรียนรู้เชิงลึก

ธนัช เบญจอนุอาชา

สารนิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล
คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2564

CAR DAMAGE CLASSIFICATION USING DEEP LEARNING

THANUS BENJAANUARCHA

A Master's Project Submitted in Partial Fulfillment of the Requirements

for the Degree of MASTER OF SCIENCE

(Data Science)

Faculty of Science, Srinakharinwirot University

2021

สารนิพนธ์

เรื่อง

การจำแนกความเสียหายรถยนต์โดยการเรียนรู้เชิงลึก

ของ

ธนัช เบญจอนุอาชา

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล

ของมหาวิทยาลัยศรีนครินทรวิโรฒ

(รองศาสตราจารย์ นายแพทย์ฉัตรชัย เอกปัญญาสกุล)

คณบดีบัณฑิตวิทยาลัย

คณะกรรมการสอบปากเปล่าสารนิพนธ์

..... ที่ปรึกษาหลัก

(ผู้ช่วยศาสตราจารย์ ดร.วราภรณ์ วิยานนท์)

..... ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.อัครา ประโยชน์)

..... กรรมการ

(อาจารย์ ดร.วีระ สอิ้ง)

ชื่อเรื่อง	การจำแนกความเสียหายรถยนต์โดยการเรียนรู้เชิงลึก
ผู้วิจัย	ธนัช เบญจอนุอาชา
ปริญญา	วิทยาศาสตรมหาบัณฑิต
ปีการศึกษา	2564
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. วราภรณ์ วิยานนท์

งานวิจัยนี้มีวัตถุประสงค์เพื่อศึกษาการจำแนกรูปภาพรถยนต์ที่มีความเสียหายและไม่มีความเสียหาย โดยใช้เทคนิคการเรียนรู้เชิงลึก และพัฒนาแบบจำลองการจำแนกความเสียหายรถยนต์โดยโครงข่ายประสาทเทียมแบบสังวัฒนาการ หรือ Convolutional Neural Network (CNN) ได้แก่ VGG16, ResNet50 และ InceptionV3 ร่วมกับการใช้เทคนิคการเรียนรู้แบบถ่ายโอนซึ่งจำแนกออกเป็น 2 ประเภท ได้แก่ มีความเสียหาย และ ไม่มีความเสียหาย โดยได้ใช้ชุดข้อมูลจากเว็บไซต์ Kaggle ในการพัฒนาแบบจำลอง จากนั้นได้วัดประสิทธิภาพของแบบจำลองและปรับพารามิเตอร์ได้แก่ batch size, learning rate, pooling เพื่อให้ได้แบบจำลองที่มีประสิทธิภาพดีที่สุด ซึ่งผลลัพธ์ที่ได้คือโมเดล VGG16 มีประสิทธิภาพมากที่สุด วัดค่าความถูกต้อง (accuracy) เท่ากับ 83% ตามมาด้วยโมเดล InceptionV3 เท่ากับ 81% และ ResNet50 เท่ากับ 68%

คำสำคัญ : การเรียนรู้เชิงลึก, โครงสร้างประสาทเทียมแบบคอนโวลูชัน, การเรียนรู้แบบถ่ายโอน, การจำแนกรูปภาพ

Title	CAR DAMAGE CLASSIFICATION USING DEEP LEARNING
Author	THANUS BENJAANUARCHA
Degree	MASTER OF SCIENCE
Academic Year	2021
Thesis Advisor	Assistant Professor Dr. Waraporn Viyanon

The purpose of this research is to study the classification of damaged and undamaged car images using deep learning techniques in order to develop a car damage classification model using a convolutional neural network (CNN). The CNN used to develop the vehicle damage classification model were VGG16, ResNet50, and InceptionV3 using transfer learning techniques. The classification falls into two categories: damaged and non-damaged car. The datasets used for developing models are from Kaggle. To improve classification model accuracy batch size, learning rate, and pooling were selected for tuning parameters. The results showed that the accuracy of the VGG16 was 83 percent, which was higher than the accuracy of the other two methods.

Keyword : Deep Learning, Convolution Neural Network, Transfer Learning, Image Classification

กิตติกรรมประกาศ

การจัดทำวิจัยฉบับนี้สำเร็จลุล่วงไปได้ด้วยการให้ความช่วยเหลือแนะนำของอาจารย์ที่ปรึกษา ผศ.ดร.วราภรณ์ วิยานนท์ ที่ได้กรุณาให้คำแนะนำข้อคิดเห็น ตรวจสอบ และแก้ไขร่างสารนิพนธ์มาโดยตลอด และ อ.ดร.วีระ สะอิ้ง ให้การอนุเคราะห์ให้คำแนะนำ code และวิธีการทดลองในการทำ computer vision จึงขอขอบคุณมา ณ ที่นี้

ขอขอบคุณ คณาจารย์ทุกท่านในภาควิชาวิทยาการข้อมูล มหาวิทยาลัยศรีนครินทรวิโรฒ ที่แนะนำ ความรู้และแนวทางการทำสารนิพนธ์นี้

ธนัช เบญจอนุอาชา

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง

บทคัดย่อภาษาอังกฤษ ..... จ

กิตติกรรมประกาศ ..... ฉ

สารบัญ ..... ช

สารบัญตาราง ..... ญ

สารบัญรูปภาพ ..... ฎ

บทที่ 1 บทนำ 1

1.1 ความสำคัญและความเป็นมาของงานวิจัย 1

1.2 วัตถุประสงค์ 2

1.3 ขอบเขตการวิจัย 2

1.4 ประโยชน์ที่ได้รับจากงานวิจัย 2

บทที่ 2 วรรณกรรมและงานวิจัยที่เกี่ยวข้อง 3

2.1 การเรียนรู้เชิงลึก (Deep Learning) 3

2.2 โครงสร้างประสาทเทียมแบบคอนโวลูชั่น (Convolution Neural Network) 4

2.2.1 การสกัดคุณลักษณะ (Feature Extraction) 4

2.2.2 สไตรด์ (Stride) และแพดดิง (Padding) 4

2.2.3 พูลลิ่ง (Pooling) 6

2.2.4 Fully Connected Layer 6

2.3 การเรียนรู้แบบถ่ายโอน (Transfer Learning) 7

2.3.1 โครงสร้างการถ่ายโอนโมเดล VGG16 8

2.3.2 โครงสร้างการถ่ายโอนโมเดล ResNet50 9

2.3.3 โครงสร้างการถ่ายโอนโมเดล InceptionV3 10
2.4 การดัดแปลงข้อมูลต้นฉบับ (Data Augmentation) 10
2.5 การประเมินความแม่นยำของโมเดล 11
2.6 ขั้นตอนการดำเนินการเคลมประกันรถยนต์และเอกสารประกอบการเคลม 12
2.6.1 ขั้นตอนการเคลมประกันรถยนต์ 12
2.6.2 เอกสารประกอบการเคลม 13
2.7 งานวิจัยที่เกี่ยวข้อง 13
บทที่ 3 วิธีดำเนินการวิจัย 16
3.1 ขั้นตอนการดำเนินการวิจัย 16
3.1.1 การรวบรวมข้อมูล 16
3.1.2 การเตรียมข้อมูล (data preprocessing) และ การดัดแปลงข้อมูลต้นฉบับ (data augmentation) 18
3.1.3 การสกัดคุณลักษณะ (Feature extraction) 19
3.1.4 ขั้นตอนการจำแนกประเภท (Classification) 19
3.2 ขั้นตอนทดสอบ fine tune 20
3.3 สรุปขั้นตอนการทดลอง 21
3.4 การทดลองปรับพารามิเตอร์และเปรียบเทียบประสิทธิภาพ 22
3.4.1 การเปรียบเทียบระหว่าง max pooling และ average pooling 22
3.4.2 การเปรียบเทียบระหว่าง learning rate 0.0001 และ 0.001 23
3.4.3 การเปรียบเทียบระหว่าง Batch size ขนาดต่างๆ 24
3.5 การทดสอบ fine tune 28
3.5.1 โมเดล VGG16 28
3.5.2 โมเดล ResNet50 28

3.5.3 โมเดล InceptionV3 28
บทที่ 4 ผลการทดลอง 29
4.1 ผลลัพธ์ประสิทธิภาพผลลัพธ์ของโมเดลแต่ละโมเดล 29
4.1.1 ผลลัพธ์ประสิทธิภาพของโมเดล VGG16 29
4.1.2 ผลลัพธ์ประสิทธิภาพโมเดล VGG16 with data augmentation 30
4.1.3 ผลลัพธ์ประสิทธิภาพผลลัพธ์ของโมเดล ResNet 50 31
4.1.4 ผลลัพธ์ประสิทธิภาพโมเดล ResNet 50 with data augmentation 32
4.1.5 ผลลัพธ์ประสิทธิภาพผลลัพธ์ของโมเดล InceptionV3 33
4.1.6 ผลลัพธ์ประสิทธิภาพโมเดล InceptionV3 with data augmentation 34
4.2 การเปรียบเทียบผลลัพธ์ประสิทธิภาพของแต่ละโมเดล 35
4.3 ผลลัพธ์การเพิ่มประสิทธิภาพด้วยวิธีการ fine tune 38
บทที่ 5 สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ 40
5.1 สรุปผลการวิจัย 40
5.2 อภิปรายผลการวิจัย 41
5.3 ข้อเสนอแนะ 43
บรรณานุกรม 44
ภาคผนวก 46
ประวัติผู้เขียน 51

สารบัญตาราง

หน้า

ตาราง 1 จำนวนภาพการฝึก (train) และทดสอบ (test) จำแนกตามคลาส 17

ตาราง 2 เปรียบเทียบประสิทธิภาพความถูกต้องระหว่าง Max pooling และ Average pooling 23

ตาราง 3 เปรียบเทียบประสิทธิภาพความถูกต้องระหว่าง learning rate 0.001 และ 0.0001..... 23

ตาราง 4 เปรียบเทียบประสิทธิภาพผลลัพธ์ของ Batch size ขนาดต่างๆ 24

ตาราง 5 เปรียบเทียบผลลัพธ์ประสิทธิภาพของแต่ละโมเดล 35

ตาราง 6 เปรียบเทียบผลลัพธ์ประสิทธิภาพของแต่ละโมเดลหลังจากการทดลอง fine tune..... 38

สารบัญรูปภาพ

หน้า

ภาพประกอบ 1 ส่วนประกอบของโครงสร้างประสาทเทียม 3
ภาพประกอบ 2 แสดงโครงสร้าง Convolution Neural Network 4
ภาพประกอบ 3 การ Stride 5
ภาพประกอบ 4 การ padding 5
ภาพประกอบ 5 Max Pooling 6
ภาพประกอบ 6 Fully Connected Layer 7
ภาพประกอบ 7 การเปรียบเทียบระหว่าง Tradition ML และ Transfer Learning 8
ภาพประกอบ 8 สถาปัตยกรรม VGG 16 9
ภาพประกอบ 9 สถาปัตยกรรม Inception V3 10
ภาพประกอบ 10 Data Augmentation 11
ภาพประกอบ 11 ขั้นตอนการเคลมประกันรถยนต์ 12
ภาพประกอบ 12 โฟลเดอร์เก็บข้อมูล Train และ Test 17
ภาพประกอบ 13 ตัวอย่างรูปภาพชุดข้อมูล (Data set) 18
ภาพประกอบ 14 ชั้นของโมเดลที่ใช้ในการฝึกฝนในการจำแนกประเภท 20
ภาพประกอบ 15 แสดงชั้นของโมเดลที่ใช้ในการฝึกฝนในการ fine tune 21
ภาพประกอบ 16 ขั้นตอนการทดลอง 22
ภาพประกอบ 17 กราฟแสดง Accuracy และ Loss เปรียบเทียบกันระหว่าง learning rate RMSprop เท่ากับ 0.001 และ 0.0001 24
ภาพประกอบ 18 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss ของ Batch size ขนาดต่างๆ 27
ภาพประกอบ 19 classification report โมเดล VGG16 29

ภาพประกอบ 20 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล VGG16 30

ภาพประกอบ 21 classification report โมเดล VGG16 with data augmentation 30

ภาพประกอบ 22 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล VGG16 with data augmentation 31

ภาพประกอบ 23 classification report โมเดล Resnet 50 31

ภาพประกอบ 24 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล Resnet 50 32

ภาพประกอบ 25 classification report โมเดล ResNet 50 with data augmentation 32

ภาพประกอบ 26 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล ResNet 50 with data augmentation 33

ภาพประกอบ 27 classification report โมเดล InceptionV3 33

ภาพประกอบ 28 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล inceptionV3 34

ภาพประกอบ 29 classification report โมเดล InceptionV3 with data augmentation 34

ภาพประกอบ 30 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล InceptionV3 with data augmentation 35

ภาพประกอบ 31 confusion matrix ของโมเดลเดล VGG16, ResNet50 และ InceptionV3 36

ภาพประกอบ 32 confusion matrix ของโมเดลเดล VGG16, ResNet50 และ InceptionV3 with data augmentation 37

ภาพประกอบ 33 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล fine tune ทั้ง 6 โมเดล 39

ภาพประกอบ 34 Confusion matrix ของโมเดล VGG16 without augmentation 40

ภาพประกอบ 35 รูปภาพตัวอย่างโมเดล VGG16 without augmentation ทำนายผิด โมเดลทำนายเกิดความเสียหาย 42

ภาพประกอบ 36 รูปภาพตัวอย่างโมเดล VGG16 without augmentation ทำนายผิด โมเดล
ทำนายไม่เกิดความเสียหาย 43

บทที่ 1
บทนำ

1.1 ความสำคัญและความเป็นมาของงานวิจัย

ปัจจุบันในประเทศไทยอุตสาหกรรมประกันภัยเป็นอุตสาหกรรมที่มีการแข่งขันกันสูง โดยปีพ.ศ.2563 อัตราการเติบของเบี้ยประกันภัยอยู่ที่ 3.5% โดยมีเบี้ยรับรวมทั้งสิ้นอยู่ที่ 252,618,165,000 บาท สัดส่วนของประกันภัยรถยนต์ (Motor insurance) อยู่ที่ 57.80% หรือมีเบี้ยรับประมาณ 146,017,083,000 บาท (สำนักงานคณะกรรมการกำกับและส่งเสริมการประกอบธุรกิจประกันภัยและสำนักงานสภาพัฒนาการเศรษฐกิจและสังคมแห่งชาติ, 2563)ซึ่งจะเห็นได้ว่าประกันภัยรถยนต์มีความสำคัญอย่างยิ่งในธุรกิจประกันภัย

เนื่องจากประกันภัยรถยนต์มีเบี้ยรับที่มากนั้นบอกถึงจำนวนที่บริษัทประกันภัยต้องรับผิดชอบในกรณีมีเคลมประกันภัยเกิดขึ้นก็มีโอกาสมากขึ้นไปด้วย บริษัทประกันภัยจะสูญเสียเงินในส่วนของเคลมที่มีการจ่ายเคลมเกินกว่ายอดการเคลมที่แท้จริง (Claim Leak) ในปริมาณมากมายมหาศาลและในการแจ้งเคลมในปัจจุบันส่วนใหญ่ลูกค้าต้องแจ้งเคลมประกันภัยโดยโทรศัพท์แจ้งบริษัทประกัน บริษัทประกันภัยจะส่งเจ้าหน้าที่ลงพื้นที่เก็บหลักฐานถ่ายรูปความเสียหายรถยนต์ ดำเนินการเอกสารต่างๆและใช้เวลาในการพิจารณาเคลมซึ่งทำให้เสียเวลาเป็นอย่างมาก ดังนั้น หากมีเครื่องมือมาช่วยในการจำแนกความเสียหายก็จะช่วยลดภาระงานของเจ้าหน้าที่ลงได้ในการลงพื้นที่ งานวิจัยนี้เน้นการจำแนกความเสียหายจากรูปภาพเพื่อลดขั้นตอนการการลงพื้นที่และลดกระบวนการอื่นๆที่เกี่ยวข้อง เพื่อให้มีความแม่นยำในการวิเคราะห์จำแนกความเสียหาย บริษัทประกันภัยจึงมองหาการแก้ปัญหาด้วยนำเทคโนโลยี AI จากการเรียนรู้เครื่องและการเรียนรู้เชิงลึกสามารถช่วยแก้ปัญหาดังกล่าว

ในงานวิจัยนี้ใช้การเรียนรู้เชิงลึก (Deep Learning) ซึ่งเป็นเครื่องมือสำหรับการพัฒนาและสร้างแบบจำลองในการทำงานโดยไม่ต้องอาศัยความช่วยเหลือจากมนุษย์ โดยแสดงโครงข่ายประสาทเทียมแบบสังวัฒนาการ หรือ Convolutional Neural Network (CNN) ในการจำแนกรูปรถยนต์ ออกเป็น 2 Class ได้แก่ มีความเสียหาย และ ไม่มีความเสียหายโดยพิจารณาความเสียหายกายภาพ เช่น กันชนถูกชน ประตูถูกชน กระจกแตก ไฟหน้าแตก ไฟท้ายแตก หรือขีดข่วน เป็นต้น ในส่วนของข้อมูล dataset ได้นำมาจากเว็บไซต์ Kaggle (SHAH) เป็นชุมชนคนทำงานด้าน data science ที่ใหญ่ที่สุดในโลก อีกทั้งยังเป็นแหล่งให้นักวิทยาศาสตร์ข้อมูลมาแลกเปลี่ยนโมเดล จัดการแข่งขันด้านวิเคราะห์ข้อมูลและ machine learning ระหว่างกัน

1.2 วัตถุประสงค์

1. เพื่อศึกษากระบวนการและประยุกต์ใช้ Image classification โดย Deep learning
2. เพื่อให้สามารถจำแนกรูปภาพที่มีความเสียหาย และ รูปที่ไม่มีความเสียหาย
3. เพื่อสามารถตรวจสอบความเสียหายอัตโนมัติได้

1.3 ขอบเขตการวิจัย

1. สร้างโมเดลเพื่อจำแนกรูปรถยนต์ 2 Class รูปที่มีความเสียหาย และ รูปที่ไม่มีความเสียหาย

2. ใช้เทคนิค Transfer Learning ทำการเปรียบเทียบประสิทธิภาพโมเดล

1.4 ประโยชน์ที่ได้รับจากงานวิจัย

1.แบบจำลองการจำแนกความเสียหายสามารถนำไปใช้ในกระบวนการทำงานของพนักงานประกันภัยในการคัดกรองภาพถ่ายของรถยนต์ว่ามีความเสียหายหรือไม่ซึ่งเป็นการลดภาระของพนักงานประกันภัย

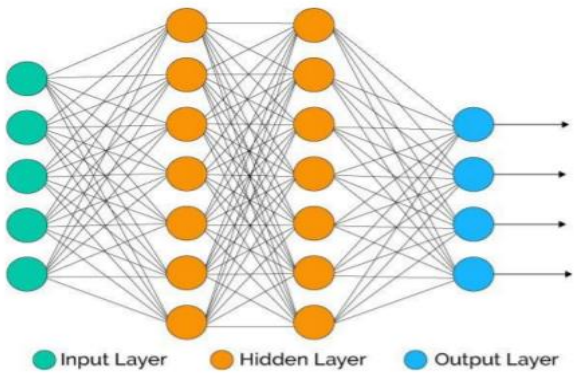
2.ลูกค้าประกันภัยได้รับการบริการเคลมได้รวดเร็ว

บทที่ 2
วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

ในงานวิจัยนี้ศึกษาโดยใช้องค์ความรู้ การเรียนรู้เชิงลึก(Deep learning) และโครงสร้างประสาทเทียมแบบคอนโวลูชั่น (Convolution Neural Network) เพื่อสกัดคุณลักษณะเฉพาะของข้อมูลรูปภาพ และใช้วิธีการเรียนรู้แบบถ่ายโอนซึ่งเป็นเทคนิคการเรียนรู้ของเครื่องด้วยการนำโครงสร้างของโมเดลที่ฝึกเรียบร้อยแล้วนำมาฝึกกับข้อมูลของงานวิจัยนี้ เพื่อให้ได้ผลลัพธ์

2.1 การเรียนรู้เชิงลึก (Deep Learning)

การเรียนรู้เชิงลึกเป็นการเลียนแบบการทำงานของโครงข่ายประสาท (neuron network) (O'Shea และ Nash, 2015) ในสมองของมนุษย์ ซึ่ง Algorithm ของ deep learning ถูกสร้างขึ้นจากการนำเอา neural network หลายๆชั้น (layer) มาซ้อนต่อกัน โดยมีส่วนประกอบด้วยชั้นต่างๆ ตามภาพประกอบ 1 ดังนี้ ชั้นแรกสุดทำหน้าที่ในการรับข้อมูล (input layer) และชั้นถัดมาจะเป็นชั้นที่อยู่ตรงกลางเรียกว่า hidden layer และชั้นสุดท้ายทำหน้าที่ส่งการประมวลผลลัพธ์ออกมา (output layer) ซึ่งใน hidden layer ต้องมีชั้นมากกว่า 2 ชั้นขึ้นไปจะถือว่าเป็น deep learning โดยใน hidden layer ของแต่ละชั้นจะเปรียบเสมือนเป็นเซลล์ประสาทจำนวนมาก ซึ่งมีหน้าที่ในการประมวลผล รับข้อมูลจากชั้นที่อยู่ก่อนหน้าและส่งข้อมูลที่ประมวลผลเสร็จแล้วไปยังชั้นถัดไป ยิ่งถ้าใส่ข้อมูล input มากเท่าไรก็สามารถสกัด feature ที่ซับซ้อนได้ดียิ่งขึ้น

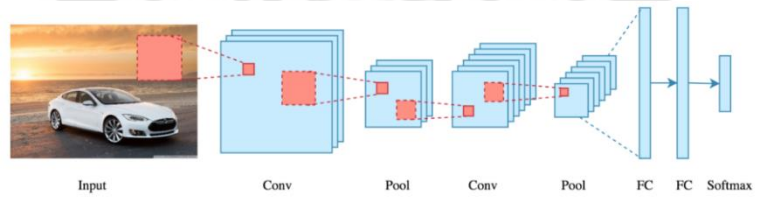


ภาพประกอบ 1 ส่วนประกอบของโครงสร้างประสาทเทียม

ที่มา : https://csit.nu.ac.th/kraisak/ds/ds/chapter07/Chapter07.pdf

2.2 โครงสร้างประสาทเทียมแบบคอนโวลูชั่น (Convolution Neural Network)

โครงสร้างประสาทเทียมแบบคอนโวลูชั่น (Convolution Neural Network: CNN)(O'Shea และ Nash, 2015) มีโครงสร้างเฉพาะตัว ทำการป้อนรูปภาพโครงสร้างซึ่งจะใช้การคำนวณทางคณิตศาสตร์ประกอบไปด้วยชั้นคอนโวลูชั่น (convolution layer) ชั้นพลูลิ่ง (pooling layer) และชั้นเชื่อมต่อกันอย่างสมบูรณ์ (fully connected layer) และผลลัพธ์ output มี activation function เป็น softmax ทำการจำแนกประเภทรูปภาพ โดยถูกออกแบบมาเพื่อการเพิ่มความสามารถในการสกัดเอาคุณลักษณะ (feature extraction) ที่สำคัญออกมา ดังภาพประกอบ 2 เช่น เส้นขอบของวัตถุต่างๆ เพื่อให้โมเดลสามารถเรียนรู้ลักษณะของภาพได้อย่างมีประสิทธิภาพและแม่นยำ โดยใช้ค่าพิกเซลซึ่งประกอบด้วย 3 ชาแนล (Channel) ได้แก่ สีแดง สีเขียว และ สีน้ำเงิน จะแทนค่าความเข้มสีด้วยตัวเลข 0 ถึง 255



ภาพประกอบ 2 แสดงโครงสร้าง Convolution Neural Network

ที่มา : https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

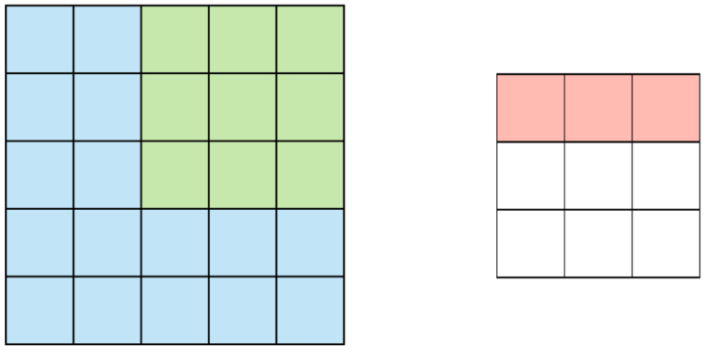
2.2.1 การสกัดคุณลักษณะ (Feature Extraction)

โครงสร้างของคอนโวลูชั่นจะใช้การคำนวณคณิตศาสตร์เพื่อสกัดเอาคุณลักษณะจากรูปภาพออกมาเรียกว่า เคอร์เนล (kernel) หรือ ตัวกรอง (filter) โดยตัวกรอง 1 ตัวสามารถดึงคุณลักษณะที่สนใจออกมาได้ 1 อย่าง ดังนั้นจึงต้องมีตัวกรองหลายตัวเพื่อสกัดหาคุณลักษณะทางพื้นที่มาประกอบกัน โดยตัวกรองจะถูกทาบลงในพิกเซลภาพแรกมีลักษณะเป็นตาราง 2 มิติที่มีขนาดตามพื้นที่ย่อยๆที่พิจารณา ตัวกรองจะเลื่อนไปจนครบทุกพิกเซลในภาพ และจะได้ผังคุณลักษณะ (feature map) ออกมา

2.2.2 สไตรด์ (Stride) และแพดดิง (Padding)

Stride เป็นตัวกำหนดว่าจะเลื่อนตัวกรองไปกี่ช่อง หากกำหนดค่า Stride มากยิ่งเลื่อนมากคุณลักษณะจะมีพื้นที่ซ้อนกันน้อยขึ้น โดยจากภาพประกอบ 3 เป็นการทำ Stride 1 คือ

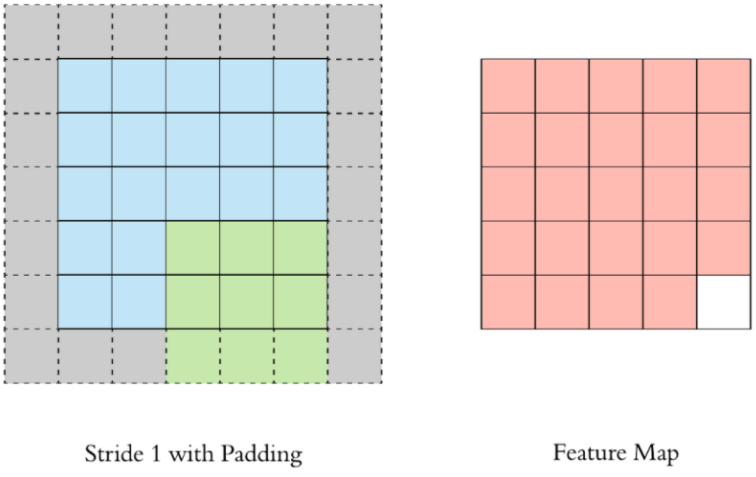
จะทำให้ตัวกรองที่มีขนาด 3x3 พื้นที่สีเขียว มาทาบและทำการเลื่อนไปทีละ 1 ช่อง จนครบทุกพิกเซลในภาพอินพุต และจะได้ผังคุณลักษณะ (feature map) ที่มีขนาดเล็กลง



ภาพประกอบ 3 การ Stride

ที่มา: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Padding เป็นการเพิ่มพื้นที่ไปรอบอินพุตสีเทาดังภาพประกอบ 4 เพื่อให้ขนาดของคุณลักษณะ (feature map) ที่ออกมาจะเท่ากับอินพุตและสามารถเก็บคุณลักษณะขอบของรูปภาพอินพุตได้

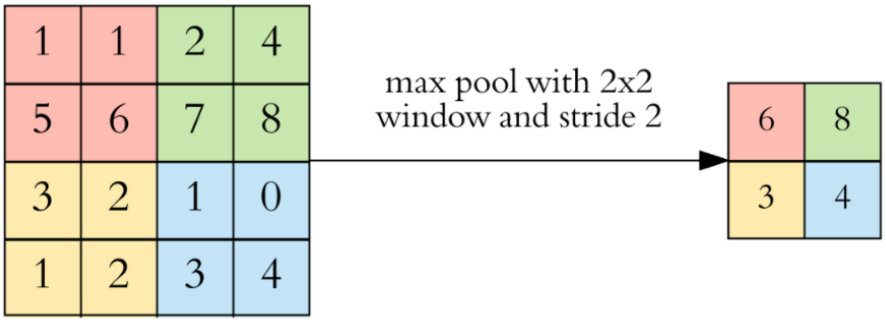


ภาพประกอบ 4 การ padding

ที่มา : https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

2.2.3 **พูลลิ่ง** (Pooling)

หลังจากดำเนินการทำคอนโวลูชั่นแล้ว Pooling จะช่วยลดมิติ (down sample) และจำนวนพารามิเตอร์ ทำให้ใช้เวลาในการเทรนน้อยลง ซึ่ง Pooling มีความสามารถในการย่อรูปแบบหนึ่ง โดย Max Pooling เป็นตัวกรองแบบหนึ่งที่หาค่าสูงสุดในบริเวณที่ตัวกรองทาบอยู่มาเป็นผลลัพธ์ โดยจะเตรียมตัวกรองในลักษณะเดียวกับการทำ Feature Extraction ของ CNN มาทาบบนข้อมูลแล้วเลือกค่าที่สูงที่สุดบนตัวกรองนั้นมาเป็นผลลัพธ์ใหม่ และจะเลื่อนตัวกรองไปตาม Stride ที่กำหนดไว้ โดยขนาดตัวกรองของการทำ max pooling ซึ่งนิยมเรียกกันว่า pool size จากภาพประกอบ 5 ได้กำหนดขนาด pool size เป็น 2x2 และหาค่าสูงสุดตามพื้นที่ นอกจาก Max pooling ยังมี Average pooling จะกรองเอาค่าเฉลี่ยในบริเวณที่ตัวกรองทาบอยู่มาเป็นผลลัพธ์

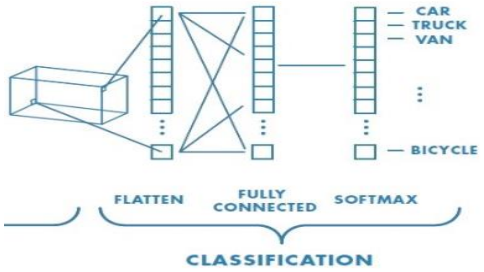


ภาพประกอบ 5 Max Pooling

ที่มา : https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

2.2.4 Fully Connected Layer

ชั้นนี้เป็นชั้นสุดท้าย ดังภาพประกอบ 6 ที่มีการเชื่อมต่อกันของแต่ละชั้นอย่างสมบูรณ์ (fully Connected Layer) โดยผลลัพธ์ที่ได้จากคอนโวลูชั่นและ Pooling ซึ่ง Fully Connected Layer จะรับคุณลักษณะเด่นที่ถูกสกัดออกมาเพื่อนำคุณลักษณะเด่นไปทำการจำแนกประเภท (classification)

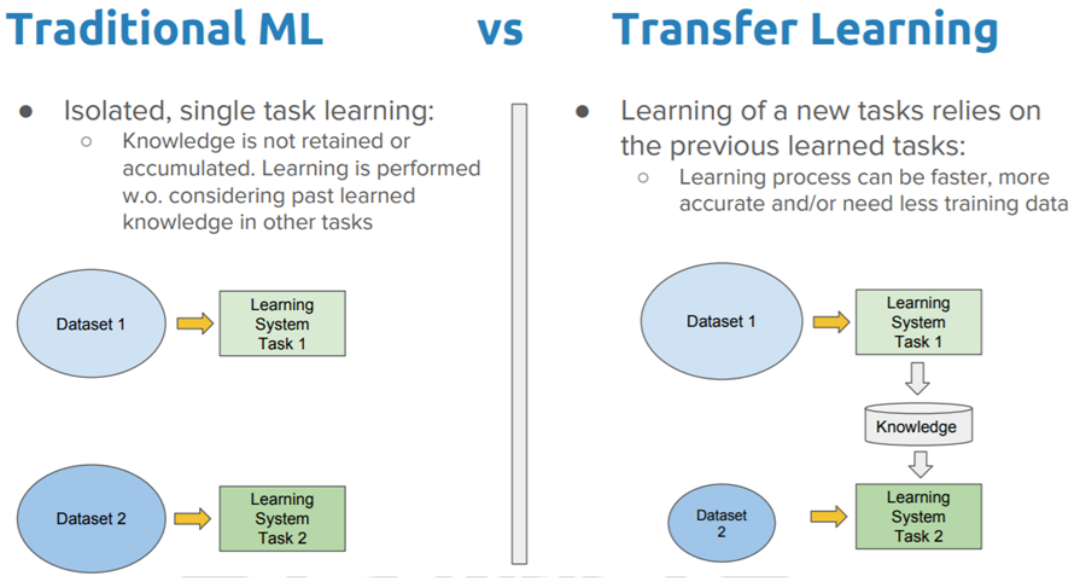


ภาพประกอบ 6 Fully Connected Layer

ที่มา : https://towardsml.com/2018/10/16/deep-learning-series-p2-understanding-convolutional-neural-networks/

2.3 การเรียนรู้แบบถ่ายโอน (Transfer Learning)

การเรียนรู้แบบถ่ายโอนเป็นเทคนิคการเรียนรู้ของเครื่อง ซึ่งใช้การเรียนรู้ด้วยการนำโครงสร้างของโมเดลที่ฝึกเรียบร้อยแล้วนำข้อมูลของใหม่มาฝึก จะช่วยเรื่องในการประหยัดเวลาอย่างมากเนื่องจากระยะเวลาในการฝึกตั้งแต่ต้นจนจบกระบวนการใช้เวลานานและมีความซับซ้อน อีกทั้งยังต้องใช้ ชุดข้อมูล (data set) ที่มีขนาดใหญ่และใช้เวลาในการประมวลผลหลายวันจนถึงหลายสัปดาห์ ดังภาพประกอบ 7 เป็นการเปรียบเทียบระหว่างการเรียนรู้ของเครื่องที่ไม่ใช้เทคนิคการเรียนรู้ของเครื่อง (traditional ML) และการเรียนรู้ของเครื่องใช้เทคนิคการถ่ายโอน (transfer Learning) โดยโมเดลที่นำมาใช้งานวิจัยนี้คือ VGG16, ResNet50 และ InceptionV3

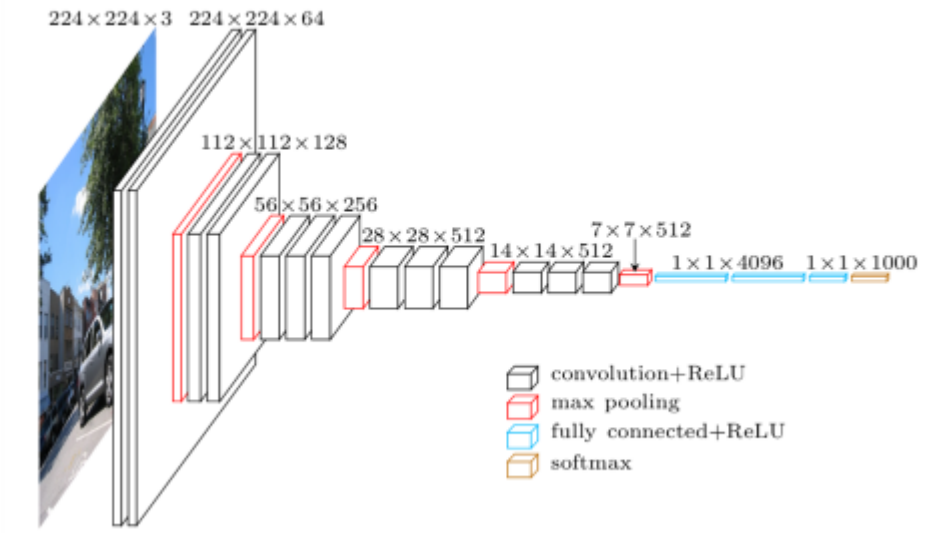


ภาพประกอบ 7 การเปรียบเทียบระหว่าง Tradition ML และ Transfer Learning

ที่มา : https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

2.3.1 โครงสร้างการถ่ายโอนโมเดล VGG16

Visual Geometry Group หรือ VGG16 (Simonyan และ Zisserman, 2014) โดยกลุ่มนักวิจัย Oxford ได้ทำการพัฒนาโครงสร้างสถาปัตยกรรมนี้ขึ้นมา และได้รับความสนใจมากจากการแข่งขัน ImageNet Large Scale Visual Recognition Challenge (ILSVRC) ปี ค.ศ. 2014 มีความแม่นยำในชุดข้อมูลทดสอบที่ 92.7% และเป็นที่นิยมจนถึงปัจจุบัน โดยสิ่งที่เป็นจุดเด่นของ VGG16 มีเอกลักษณ์เฉพาะแทนที่ hyperparameter จำนวนมาก เน้นไปที่การออกแบบชั้น conv2D 3x3 pixels การก้าว 1 stride และการใช้ same padding และ max pooling ขนาด 2x2 pixels การก้าว 2 stride แบบเดียวกันตลอดทั้งโครงสร้าง และมี 2 ชั้นสุดท้ายเป็นชั้นเชื่อมต่ออย่างสมบูรณ์ (Fully connected layer) VGG16 หมายถึงมี 16 ชั้นตามชื่อ ที่มีน้ำหนักเครือข่ายที่ใหญ่และมีพารามิเตอร์ประมาณ 138 ล้าน โครงสร้างสถาปัตยกรรม VGG16 ได้แสดงดังภาพประกอบ 8



ภาพประกอบ 8 สถาปัตยกรรม VGG 16

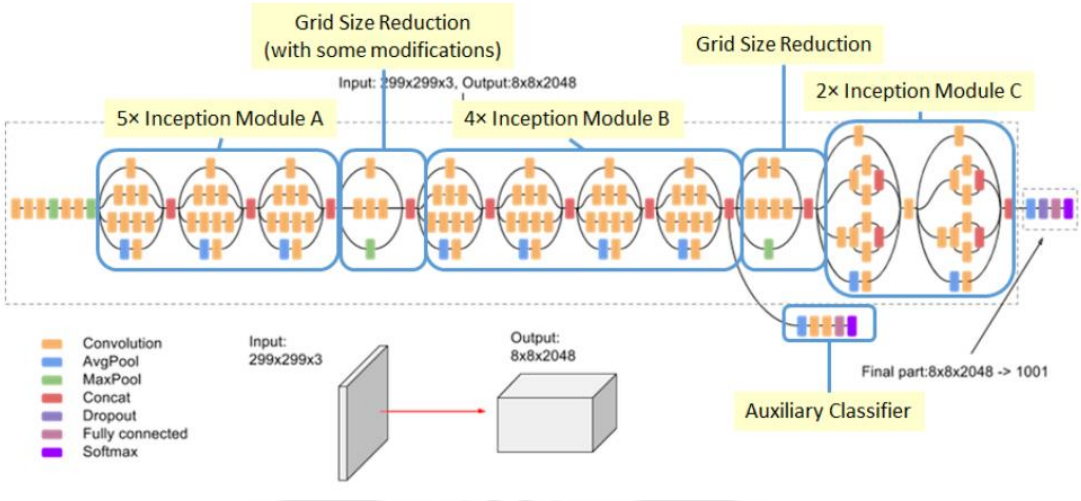
ที่มา https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7

2.3.2 โครงสร้างการถ่ายโอนโมเดล ResNet50

Deep Residual Networks หรือ ResNet (He, Zhang, Ren, และ Sun, 2016) ได้นำเสนอในงานวิจัย Deep residual learning for image recognition เป็นโครงสร้างที่มีขนาดใหญ่มีจำนวนชั้นในเครือข่ายถึง 152 ชั้น ถูกสร้างมาเพื่อแก้ปัญหา vanishing gradient คือปัญหาในระหว่างการเทรน Gradient มีขนาดเล็กลงเรื่อย ๆ จนเท่ากับ 0 ทำให้น้ำหนัก (weight) ไม่ถูกอัพเดทอีกต่อไป ทำให้โมเดลเทรนต่อไม่ได้ ซึ่งโครงสร้างสถาปัตยกรรมนี้ประกอบด้วย 4 บล็อกใหญ่ จำนวนชั้นที่มีพารามิเตอร์ใช้สำหรับการฝึกทั้งหมดจะเป็นจำนวนชั้นที่ใช้ในการเรียกชื่อของ Resnet เช่น Resnet 34, Resnet50, ResNet101, Resnet152 เป็นต้น โดยงานวิจัยนี้เลือกใช้ ResNet50 ซึ่งจะมี 50 ชั้น เป็น [3, 4, 6, 3] ซึ่งก็คือ (3+4+6+3)x3 = 48 ชั้น บวกชั้นคอนโวลูชั่นที่ติดกับชั้นอินพุต และชั้น Dense ที่ติดกับชั้นเอาต์พุต เข้าอีก 2 ชั้น รวมทั้งสิ้นเท่ากับ 50 ชั้น

2.3.3 โครงสร้างการถ่ายโอนโมเดล InceptionV3

โมเดลนี้พัฒนาโดย Google ถูกกล่าวในงานวิจัย InceptionV3 (1st ILSVRC 2015) เป็นโมเดลที่แสดงให้เห็นว่ามีความแม่นยำมากกว่า 78.1% ในชุดข้อมูล ImageNet โมเดลนี้มีแนวคิดมากมายที่นักวิจัยหลายคนพัฒนาขึ้นในช่วงหลายปีที่ผ่านมา มีพื้นฐานมาจากบทความต้นฉบับ (Szegedy, Vanhoucke, Ioffe, Shlens, และ Wojna, 2016) โดยการลดโครงสร้างภายในออกเป็น 5 ขั้นตอน คือ 1) Inception Module A จำนวน 5 Module, 2) Grid Size of Reductiion Step 1 จำนวน 1 Module, 3) Inception Module B จำนวน 4 Module, 4) Grid Size of Reduction Step 2 จำนวน 1 Moduel, 5) Inception Module C จำนวน 2 Module และ Head (8x8x2048) จำแนกประเภทโดยใช้ softmax function คำนวณเป็นความน่าจะเป็นในการจำแนกประเภท สามารถแยก output ได้ 1,000 Class ภาพประกอบ 9



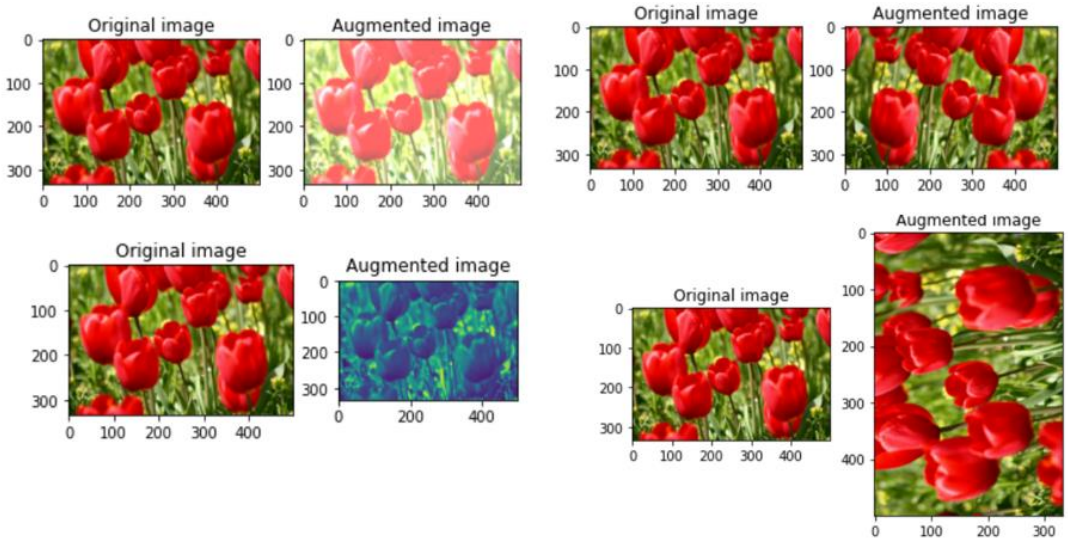
ภาพประกอบ 9 สถาปัตยกรรม Inception V3

ที่มา : https://stackoverflow.com/questions/54793602/tensorflow-hub-inception-v3-structure-compared-to-keras-inception-v3-structure

2.4 การดัดแปลงข้อมูลต้นฉบับ (Data Augmentation)

Data Augmentation คือ เป็นเทคนิคที่ใช้เพื่อเพิ่มปริมาณข้อมูลโดยการเพิ่มสำเนาที่แก้ไขเล็กน้อยของข้อมูลที่มีอยู่แล้วหรือข้อมูลสังเคราะห์ที่สร้างขึ้นใหม่จากข้อมูลที่มีอยู่ โดยการนำรูป มาย่อ ขยาย พลิกซ้าย ขวา ล่าง บน, หมุนซ้าย หมุนขวา, Crop มุม, ปรับสีเข้ม สีอ่อน, ปรับสว่าง ปรับมืด, ปรับ Contrast, เพิ่มลด noise, เบลอภาพ ดังภาพประกอบ 10 เพื่อเพิ่ม

ประสิทธิภาพของโมเดลโครงสร้างประสาทเทียมแบบคอนโวลูชัน และยังมีประโยชน์ในการลด Overfitting อีกด้วย



ภาพประกอบ 10 Data Augmentation

ที่มา : https://www.tensorflow.org/tutorials/images/data_augmentation

2.5 การประเมินความแม่นยำของโมเดล

การทำนายแบบ Classification วัดผลประสิทธิภาพโมเดล ในงานวิจัยนี้ใช้ค่า Confusion Matrix ซึ่งได้แก่ค่า Accuracy ดังสมการ (1), Precision ดังสมการ (2), Recall ดังสมการ (3), F1-Score ดังสมการ (4) ดังสมการดังนี้

Accuracy คือ การวัดความถูกต้องของโมเดล โดยพิจารณาทุกคลาส

$$\text{Accuracy} = \frac{\mathbf{TP+TN}}{\mathbf{TP+TN+FP+FN}} \qquad (1)$$

Precision คือ การวัดค่าความแม่นยำของโมเดล โดยพิจารณาแยกทีละคลาส

$$\text{Precision} = \frac{\mathbf{TP}}{\mathbf{TP+FP}} \qquad (2)$$

Recall คือ ค่าความระลึก วัดค่าความถูกต้องของโมเดล โดยพิจารณาแยกทีละคลาส

$$\text{Recall} = \frac{\mathbf{TP}}{\mathbf{TP+FN}} \qquad (3)$$

F1-Score คือ ค่าเฉลี่ยระหว่าง precision และ recall เพื่อวัดความสามารถของโมเดล

$$\text{F1-Score} = 2\,\frac{\mathbf{Precision\ x\ Recall}}{\mathbf{Precision+Recall}} \qquad (4)$$

โดยที่

True Position (TP) คือ โมเดลทำนายว่า จริง และคำตอบที่ถูกต้องมีค่าเป็น จริง โมเดลทำนายถูกต้อง

True Negative (TN) คือ โมเดลทำนายว่า เท็จ คำตอบที่ถูกต้องมีค่าเป็น เท็จ โมเดลทำนายถูกต้อง

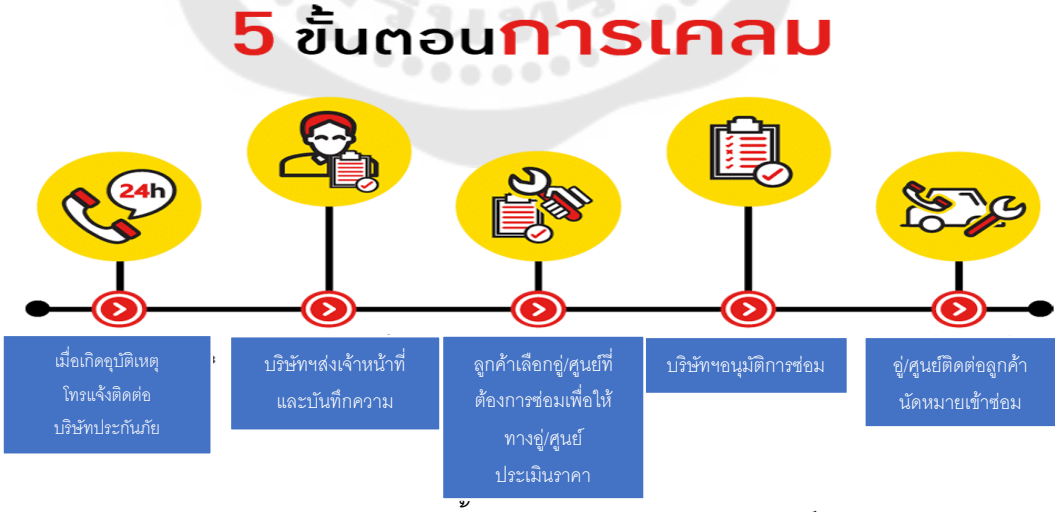
False Positive (FP) คือ โมเดลทำนายว่า จริง คำตอบที่ถูกต้องมีค่าเป็น เท็จ โมเดลทำนายผิด

False Negative (FN) คือ โมเดลทำนายว่า เท็จ คำตอบที่ถูกต้องมีค่าเป็น จริง โมเดลทำนายผิด

2.6 ขั้นตอนการดำเนินการเคลมประกันรถยนต์และเอกสารประกอบการเคลม

2.6.1 ขั้นตอนการเคลมประกันรถยนต์

ขั้นตอนการเคลมประกันรถยนต์ ดังภาพประกอบ 11 ประกอบด้วย 5 ขั้นตอน ได้แก่ ขั้นตอนที่1 เมื่อเกิดอุบัติเหตุโทรแจ้งติดต่อบริษัทประกัน ขั้นตอนที่ 2 จากนั้นบริษัทประกันจัดส่งเจ้าหน้าที่ถ่ายรูปและบันทึกความเสียหายที่เกิดขึ้น ขั้นตอนที่ 3 ลูกค้าเลือกอู่หรือศูนย์ที่ต้องการซ่อมเพื่อให้ทางอู่หรือศูนย์ประเมินราคา ขั้นตอนที่ 4 บริษัทประกันภัยอนุมัติการซ่อม ขั้นตอนที่ 5 อู่หรือศูนย์ติดต่อลูกค้าเพื่อนัดหมายเอารถยนต์เข้าซ่อม



ภาพประกอบ 11 ขั้นตอนการเคลมประกันรถยนต์

ที่มา : https://www.directasia.co.th/car-insurance-claims/what-to-do/

2.6.2 เอกสารประกอบการเคลม

เอกสารที่เกี่ยวข้องในการเคลมประกันรถยนต์มีดังนี้

1. ใบแจ้งเคลม พนักงานจะเป็นผู้เตรียมให้

2. ใบขับขี่ของผู้ขับขี่ขณะเกิดเหตุ

3. สมุดทะเบียนรถ เพื่อยืนยันว่ารถยนต์ที่เกิดความเสียหายตรงกับที่ทำประกัน

4. บันทึกความเสียหาย ถ่ายรูปประกอบรายการความเสียหายที่ทางเจ้าหน้าที่สำรวจภัยเป็นผู้ออกให้ฉบับจริง

5. ชื่อและรายละเอียดพยาน (ถ้ามี) พยานที่เห็นเหตุการณ์ในที่เกิดเหตุ เพื่อสามารถซักถามข้อมูลเพิ่มเติม

6. รูปถ่ายและเอกสารอื่นๆ กรณีบาดเจ็บต้องมีใบรับรองแพทย์และใบเสร็จรับเงินต้นฉบับ

2.7 งานวิจัยที่เกี่ยวข้อง

จากการศึกษางานวิจัยที่เกี่ยวข้อง มีงานวิจัยเกี่ยวกับรูปภาพรถยนต์ที่มีความเสียหายและน่าสนใจนำมาศึกษา โดยจะกล่าวถึงงานวิจัย 3 งานดังนี้

1.) Kalpesh Patil. Mandar KulKarni. Anand Sriraman and Shirish Karande (2017) (Patil, Kulkarni, Sriraman, และ Karande, 2017) ในบทความนี้ใช้เทคนิคการเรียนรู้เชิงลึก (Deep learning) ในการจำแนกประเภทความเสียหายจากรูปภาพรถยนต์ (Classifications of car damages) โดยใช้กระบวนการ Convolutional Neural Network (CNN) ด้วยข้อมูลที่มีปริมาณน้อยทำให้ต้องมีการทำ data augmentation เป็นการเพิ่มข้อมูลด้วยการย่อ ขยาย หมุนซ้าย/ขวา Flip ซ้าย/ขวา/บน/ล่าง Crop มุมปรับสีเข้ม/จืด ปรับแสง สว่าง/มืด ปรับ Contrast ปรับ Perspective เพิ่ม/ลด Noise เบลอภาพ เพื่อเพิ่มปริมาณรูปภาพและเปรียบเทียบประสิทธิภาพได้ว่าการไม่ทำ data augmentation ได้ประสิทธิภาพที่ดีขึ้นกว่าการทำการ data augmentation ในการทดสอบกระบวนการ Convolutional Neural Network (CNN) ใช้โครงสร้างต่างๆเช่น Car, Inception, Alexnet, VGG19, VGG16 ,Resnet ผลลัพธ์ของการทดลองวิธีที่ดีที่สุดคือ Resnet ประสิทธิภาพความแม่นยำที่ 88.24% โดยไม่ได้ทำการ data augmentation และได้ประสิทธิภาพความแม่นยำที่ดีที่ 89.53% ด้วยการใช้การรวมวิธีการ transfer and ensemble learning

2.) Ranjodh Singh, Meghna P Ayyar, Tata Venkata Sri Pavan, Sandeep Gosain, Rajiv Ratn Shah (2019) (Singh, Ayyar, Pavan, Gosain, และ Shah, 2019) ในประเทศอินเดียมียานพาหนะประมาณ 230 ล้านคัน ส่งผลให้ธุรกิจประกันภัยได้เป็นตลาดที่กำลัง

เติบโต และกรณีเรียกร้องค่าสินไหมก็จะใช้วิธีแบบเดิม คือต้องใช้เจ้าหน้าที่ลงสำรวจเพื่อตรวจสอบและทำการประเมินค่าเสียหายและค่าสินไหมทดแทนด้วนจำนวนรถที่เยอะในปัจจุบันทำให้กระบวนการต่างๆใช้เวลารอคอยนาน จึงได้เสนอระบบ end to end เพื่อทำให้กระบวนการนี้เป็นไปโดยอัตโนมัติอันจะเป็นประโยชน์ต่อทั้งบริษัทและลูกค้า ระบบนี้จะถ่ายภาพรถที่เสียหายป้อนข้อมูลและให้ข้อมูลที่เกี่ยวข้อง เช่นชิ้นส่วนที่เสียหายและให้ประมาณการขอบเขตความเสียหาย โดยทำการศึกษาความเสียหายของรถยนต์จากรูปภาพได้ทดลองโดยการฝึกโมเดลการเรียนรู้เชิงลึกต่างๆ ออกแบบตัวทำนายความเสียหายของรถยนต์ แยกแต่ละงานออกเป็นโมดูลต่างๆ โมดูลแรกตรวจจับและกำหนดตำแหน่งชิ้นส่วนในภาพรถยนต์ การตรวจจับชิ้นส่วนจำเป็นต้องระบุชิ้นส่วนที่เสียหาย โดยใช้โมเดล Mask R-CNN, PANet และใช้ทั้งสองอย่าง Ensemble โมดูลที่สองจำแนกชิ้นส่วนที่ตรวจพบโดยโมดูลแรกคือมีเสียหายหรือไม่เสียหาย เมื่อตรวจพบชิ้นส่วนโดยใช้โมดูลแรก กรองส่วนที่ไม่เสียหายออก โดยใช้ตัวแยกประเภทภาพโมเดล VGG16 เพื่อจำแนกชิ้นส่วนที่เสียหายและไม่เสียหาย เรียกเครือข่ายนี้ว่า Damage Net (D-Net) และคาดการณ์ขอบเขตความเสียหาย ได้ทำการทดสอบกับชุดข้อมูล COCO ประสิทธิภาพ Parts MRCNN, Part PA Net, Ensemble(Parts MRCNN+Part PA)แ ล ะ Damage MRCNN ผ ล คะแนน mAP (mean Average Precision) 0.35, 0.32, 0.38, 0.40 ตามลำดับ Parts MRCNN มีประสิทธิภาพดีกว่า Part PA Net การทดสอบ D-Net สำหรับการแยกชิ้นส่วนโดย Ensemble พบว่ามีความแม่นยำถึง 85.6% มีประสิทธิภาพดีพอโดยการทดลอง CNN อื่น ๆเช่น ResNet และ Inception V3

3) Phyu Mar Kyu and Kuntpong Woraratpanya (2020)(Kyu แ ล ะ Woraratpanya, 2020) ทำการศึกษาซึ่งในปัจจุบัน อุตสาหกรรมรถยนต์เติบโตซึ่งมีผลโดยตรงเกี่ยวข้องกับจำนวนอุบัติเหตุที่เพิ่มขึ้น ดังนั้นบริษัทประกันต่างๆเผชิญกับการเรียกร้องเคลมและการแก้ไขปัญหาเคลมที่เรียกร้องสูงเกิดความเป็นจริง จึงใช้ปัญญาประดิษฐ์ (AI) แมชชีนเลิร์นนิ่งและอัลกอริธึมการเรียนรู้เชิงลึกมามาช่วยแก้ปัญหาได้ปัญหาเหล่านี้สำหรับอุตสาหกรรมประกันภัย โดยบทความนี้จะใช้อัลกอริธึมการเรียนรู้เชิงลึก VGG16 และ VGG19 ในการตรวจจับและประเมินความเสียหายรถยนต์ ประเมินตำแหน่ง และความรุนแรง โดยได้พบว่า โมเดล CNN ได้ผ่านการอบรมด้วยชุดข้อมูล ImageNet และตามด้วยการปรับ fine tuning เพื่อเพิ่มประสิทธิภาพ ผลลัพธ์ที่ได้ ความแม่นยำในการแยก damaged detection ของ VGG19 อยู่ที่ 95.22% และ VGG16 อยู่ที่ 94.56% ความแม่นยำในการแยกตำแหน่ง (ส่วนหน้า, ส่วนข้าง, ส่วนหลัง) VGG19 อยู่ที่ 76.48% และVGG16 อยู่ที่ 74.39% และความแม่นยำในความรุนแรง (ความรุนแรงน้อย, ปานกลาง ,มาก)

VGG19 อยู่ที่ 58.48% และ VGG16 อยู่ที่ 54.8% จากผลลัพธ์ประสิทธิภาพของ VGG19 นั้นดีกว่า VGG16

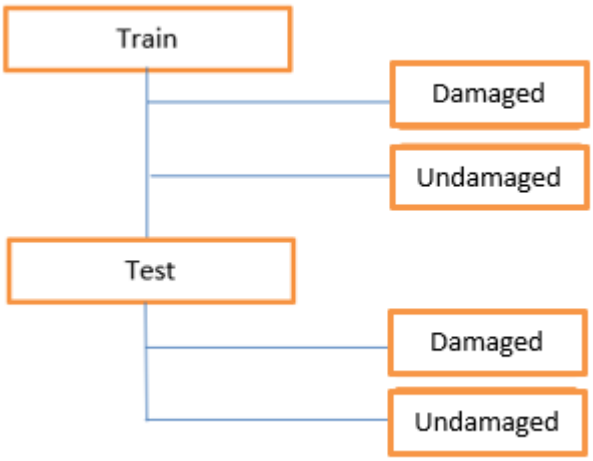
บทที่ 3
วิธีดำเนินการวิจัย

การศึกษางานวิจัยนี้เป็นการวิจัยเพื่อการจำแนกรูปภาพที่มีความเสียหาย และ รูปที่ไม่มีความเสียหายเพื่อช่วยแบ่งเบาภาระของเจ้าหน้าที่ประกันภัยในการลงสำรวจภัย โดยใช้โครงสร้างประสาทเทียมแบบคอนโวลูชัน (Convolution neural network) ใช้เทคนิคการถ่ายโอน 3 โครงสร้าง (transfer learning) ได้แก่ 1) โครงสร้างการถ่ายโอนโมเดล VGG16 2) โครงสร้างการถ่ายโอนโมเดล ResNet50 3) โครงสร้างการถ่ายโอนโมเดล InceptionV3 วิธีการและขั้นตอนในการดำเนินการวิจัย เพื่อใช้ในการสรุปผลของงานวิจัยดังนี้

3.1 ขั้นตอนการดำเนินการวิจัย

3.1.1 การรวบรวมข้อมูล

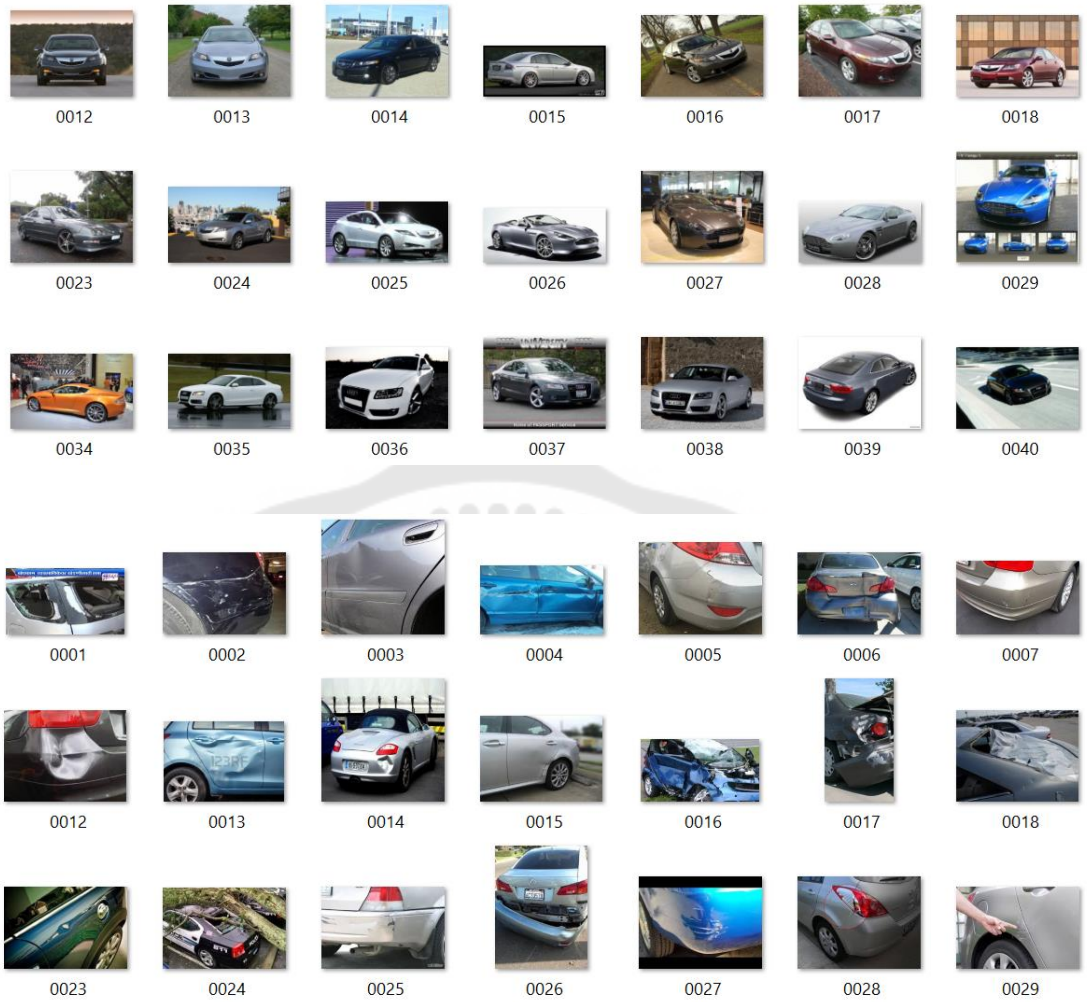
งานวิจัยนี้เก็บรวบรวมข้อมูลจากเว็บไซต์ kaggle เป็นหลัก (SHAH) โดยค้นหารูปภาพรถยนต์ทั้งรูปภาพที่มีความเสียหาย และไม่มีความเสียหาย ทำความสะอาดข้อมูลโดยการลบรูปภาพที่ไม่ชัดเจน ลบรูปภาพที่มีส่วนที่ไม่เกี่ยวข้องออก เช่น คน หรือ วัตถุต่างๆ จากนั้นนำมาสร้าง data set เป็น train และ test อย่างละโฟลเดอร์ และ ภายในโฟลเดอร์ train จะมีโฟลเดอร์ undamaged ที่มีรูปภาพรถยนต์ที่ไม่มีความเสียหายอยู่จำนวน 920 รูปภาพ และ โฟลเดอร์ damaged รูปภาพรถยนต์ที่มีความเสียหายจำนวน 920 รูปภาพ และในส่วนของข้อมูล test ก็มี 2 โฟลเดอร์ undamaged มีรูปภาพจำนวน 230 รูปภาพ และ โฟลเดอร์ damaged มีรูปภาพจำนวน 230 รูปภาพ เช่นกัน แสดงแผนผังโฟลเดอร์เก็บข้อมูลดังภาพประกอบ 12 สรุปดังตาราง 1 และแสดงตัวอย่างรูปภาพชุดข้อมูล (Data set) รถยนต์ที่มีความเสียหายไม่มีความเสียหายดังภาพประกอบ 13



ภาพประกอบ 12 โฟลเดอร์เก็บข้อมูล Train และ Test

ตาราง 1 จำนวนภาพการฝึก (train) และทดสอบ (test) จำแนกตามคลาส

Class	Training dataset (จำนวนรูป)	Test dataset (จำนวนรูป)
Undamaged	920	230
Damaged	920	230
Total	1,840	460



ภาพประกอบ 13 ตัวอย่างรูปภาพชุดข้อมูล (Data set)

3.1.2 การเตรียมข้อมูล (data preprocessing) และ การดัดแปลงข้อมูลต้นฉบับ (data augmentation)

ขั้นตอนการเตรียมข้อมูลจากที่ได้เก็บข้อมูลและแยกโฟลเดอร์ train และ test แล้วทำการเตรียมข้อมูลโดยโปรแกรมไพธอน (python) ใช้คำสั่ง ImageDataGenerator ในการปรับขนาดรูปภาพ (target_size) ให้เป็น 128x128x3 และปรับค่าพิกเซลของรูปภาพแทนค่าความเข้มสีประกอบไปด้วย สีแดง สีเขียวและสีน้ำเงิน ด้วยตัวเลขระหว่าง 0 ถึง 255 (rescale) หลังจากนั้นนำข้อมูลไปดัดแปลงข้อมูลต้นฉบับ (data augmentation) ปรับข้อมูลด้วยคำสั่งดังนี้ ปรับหมุนภาพ 40 องศา (rotation_range = 40) ปรับเลื่อนรูปภาพแนวกว้าง 20% (width_shift_range = 0.2) ปรับเลื่อนรูปภาพแนวสูง 20% (height_shift_range = 0.2) เฉือนรูปภาพ 20% (shear_range = 0.2) ปรับขยายรูปภาพ 20% (zoom_range = 0.2) ปรับพลิกรูปภาพกลับจากซ้ายไปขวาหรือขวา

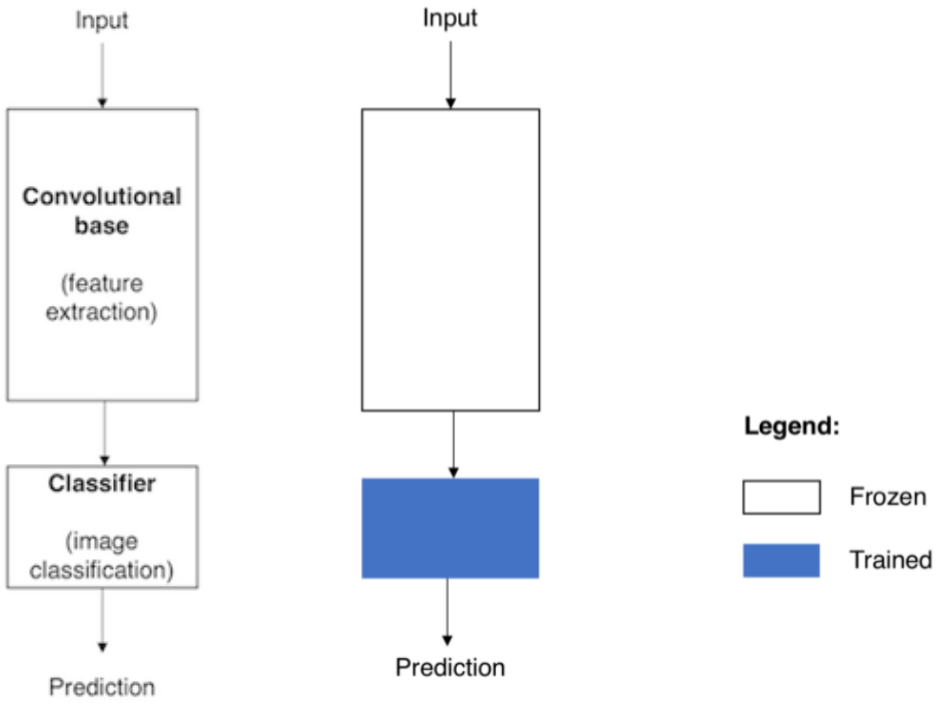
ไปซ้าย (horizontal_flip=True) และคำสั่งเติมพิกเซลที่ขาดหายไปหลังจากการย้ายหรือเฉือนออกด้วยพิกเซลใกล้เคียง (fill_mode=nearest) และไม่แปลงข้อมูลต้นฉบับ (without data augmentation) โดยสำหรับข้อมูลที่ไม่มีการแปลงข้อมูลต้นฉบับหลังจากขั้นตอนปรับขนาดรูปภาพให้เป็นพิกเซลแล้วข้ามขั้นตอนการแปลงข้อมูลต้นฉบับ จะนำข้อมูลไปสกัดคุณลักษณะ (feature extraction) ต่อไป

3.1.3 การสกัดคุณลักษณะ (Feature extraction)

จากบทที่ 2.2.1 ที่อธิบายการทำงานของการสกัดคุณลักษณะ (Feature extraction) ผู้วิจัยได้ใช้การเรียนรู้แบบถ่ายโอน(Transfer Learning) ทั้ง 3 โครงสร้างซึ่งเป็นโครงสร้างประสาทเทียมแบบคอนโวลูชัน (Convolution neural network) ได้แก่ VGG16, Resnet50, InceptionV3 ซึ่งใช้ weight ข้อมูลจาก ImageNet (Krizhevsky, Sutskever, และ Hinton, 2017)

3.1.4 ขั้นตอนการจำแนกประเภท (Classification)

การจำแนกประเภท (Classification) เป็นขั้นตอนสุดท้ายที่รับข้อมูลจากการสกัดคุณลักษณะ (Feature extraction) ซึ่งเป็นโครงสร้างคอนโวลูชัน (Convolution network) ส่วนนี้จะยังไม่มีการฝึก ยังใช้ weight ข้อมูลจากฐานข้อมูล ImageNet และใช้การเรียนรู้แบบถ่ายโอน (Transfer Learning) ส่งคุณลักษณะ (Feature extraction) มายังส่วนการจำแนกประเภท (Classification) ในส่วนนี้ได้ดำเนิน custom head โดยการเพิ่มชั้น pooling และ outputs dense = 1 ในการจำแนกประเภท (Classification) โดยใช้ activation sigmoid ในส่วนนี้จะฝึกฝน(train) ด้วยชุดข้อมูล (Data Set) เพื่อให้โมเดลเรียนรู้และสามารถทำนายชุดข้อมูลที่เตรียมสำหรับวิจัยได้ในส่วนสีฟ้าดังภาพประกอบ 14

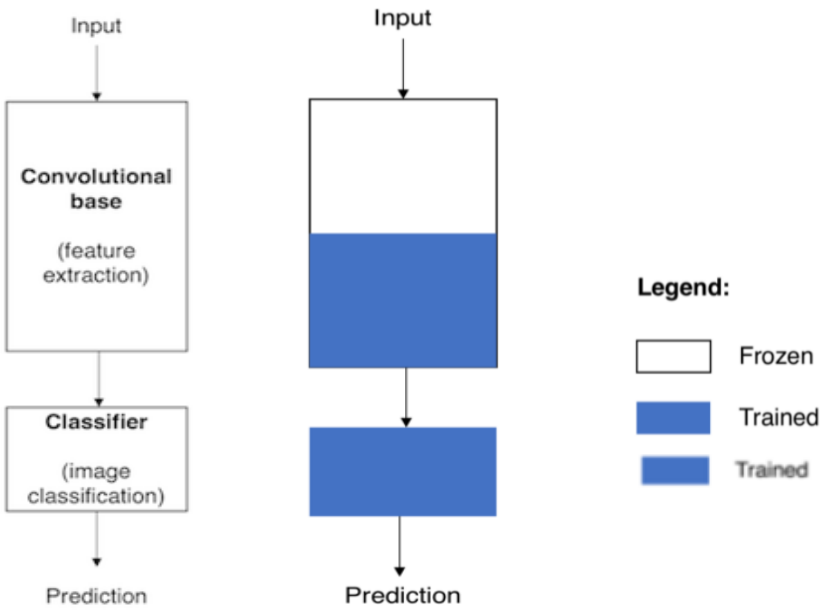


ภาพประกอบ 14 ชั้นของโมเดลที่ใช้ในการฝึกฝนในการจำแนกประเภท

ที่มา : https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751

3.2 **ขั้นตอนทดสอบ** fine tune

ในขั้นตอนนี้หลังจากได้ผลลัพธ์ทุกโมเดลแล้วจะดำเนินการทดสอบ fine tune เพิ่มเพื่อทดลองปรับประสิทธิภาพโมเดล โดยจะทำการนำชุดข้อมูล (data set) เข้าไปเทรนในชั้น (layer) ท้ายๆของโครงสร้างประสาทเทียมแบบคอนโวลูชัน (Convolution neural network) ทั้ง 3 โครงสร้าง ส่วนสีฟ้าตามภาพประกอบ 15

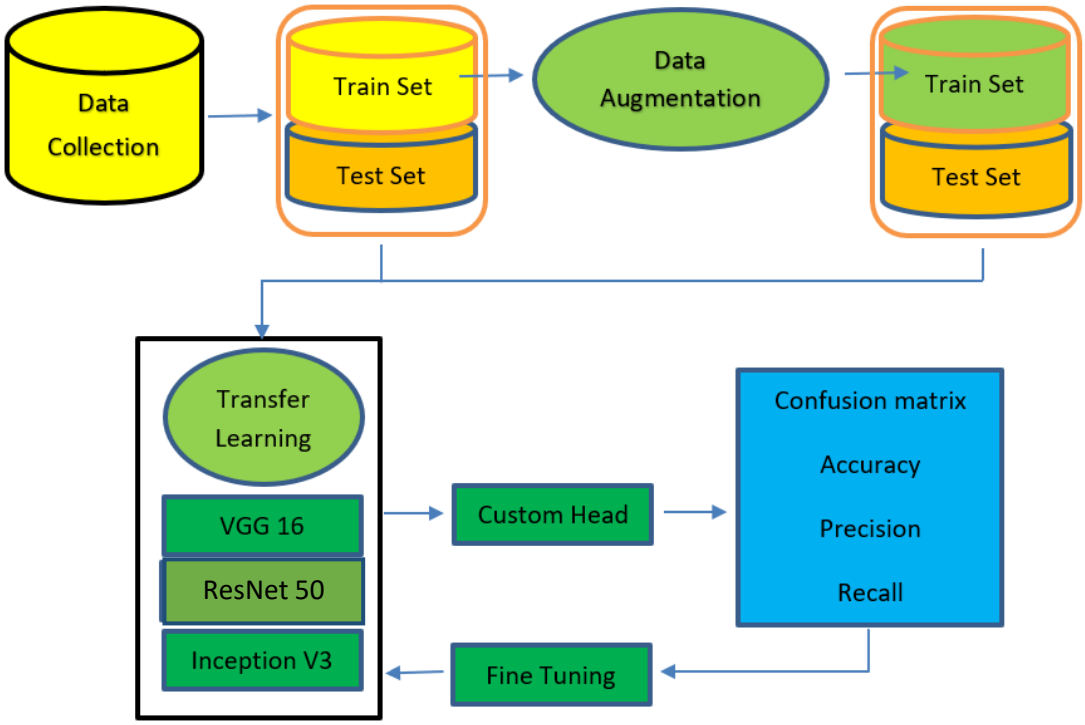


ภาพประกอบ 15 แสดงชั้นของโมเดลที่ใช้ในการฝึกฝนในการ fine tune

ที่มา : https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751

3.3 สรุปขั้นตอนการทดลอง

ขั้นตอนกระบวนการทดลองดำเนินการอธิบายตามภาพประกอบ 16 ขั้นตอนแรกดำเนินการเก็บข้อมูลตามที่ได้กล่าวในข้อ 3.1.1 และ 3.1.2 หลังจากนั้นนำข้อมูลชุดฝึกฝน (train set) ทำการแปลงข้อมูล (data augmentation) นำเข้าโมเดล transfer learning 3 โมเดล ได้แก่ Inception V3, VGG16, ResNet 50 ทำการ custom head โดยเพิ่มชั้น pooling และ dense = 1 เปรียบเทียบประสิทธิภาพ และปรับพารามิเตอร์ และดำเนินการ fine tune เพื่อปรับปรุงประสิทธิภาพ



ภาพประกอบ 16 ขั้นตอนการทดลอง

3.4 การทดลองปรับพารามิเตอร์และเปรียบเทียบประสิทธิภาพ

ขั้นตอนนี้ได้ทำการทดลองเพื่อปรับจูนพารามิเตอร์ต่างๆ ได้แก่ max pooling, average pooling, batch size, learning rate ของ optimizer เพื่อปรับให้โมเดลมีประสิทธิภาพที่ดีที่สุด โดยจะเปรียบเทียบผลพารามิเตอร์ต่างๆ ดังนี้

3.4.1 การเปรียบเทียบระหว่าง max pooling และ average pooling

Pooling layer นี้อยู่ในส่วน Classification ตามที่ได้ทำการเพิ่มเข้าไปในขั้นตอนที่ 3.4 โดยกำหนดพารามิเตอร์ต่างๆ ดังนี้ Batch size กำหนดที่ 256 ค่า Learning rate เป็น RMSprop เท่ากับ 0.0001 และกำหนด กำหนดรอบการเรียนรู้ที่ 500 รอบ (epoch) และใช้โมเดล VGG16 ในการทดสอบซึ่งผลที่ได้จากการทดลองเปรียบเทียบประสิทธิภาพพบว่า max pooling layer นั้นทำให้โมเดลมีประสิทธิภาพดีกว่า ดังตาราง 2

ตาราง 2 เปรียบเทียบประสิทธิภาพความถูกต้องระหว่าง Max pooling และ Average pooling

pooling	Max pooling	Average pooling
accuracy	0.81	0.80

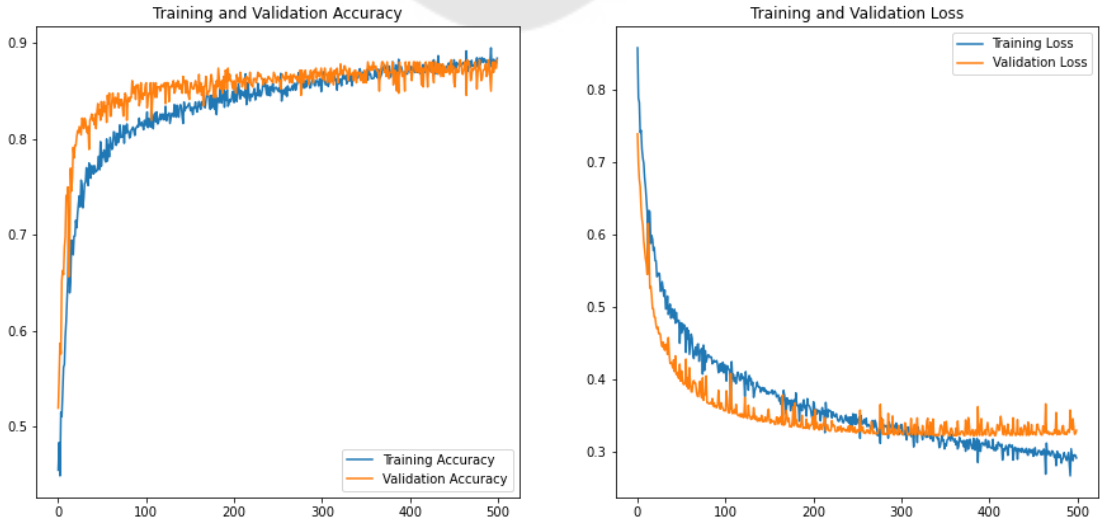
3.4.2 การเปรียบเทียบระหว่าง learning rate 0.0001 และ 0.001

ในการทดสอบเปรียบเทียบประสิทธิภาพของพารามิเตอร์ learning rate RMSprop โดยได้ทดลองสอบการเปรียบเทียบค่าที่ 0.0001 และ 0.001 พบว่า accuracy เท่ากับ 0.81 และ 0.80 ตามลำดับ ตารางที่ 3 โมเดลมีประสิทธิภาพที่ใกล้เคียงกัน และเมื่อมาพิจารณากราฟของ learning rate RMSprop เท่ากับ 0.001 พบว่าเกิด overfit จึงได้ผลสรุปว่า learning rate RMSprop เท่ากับ 0.0001 นั้นมีประสิทธิภาพที่ดีกว่า เนื่องจากโมเดลค่อยๆเปลี่ยนแปลงที่ละน้อยๆในในการเรียนรู้แต่ละรอบทำให้กราฟมีความนิ่งกว่า ดังภาพประกอบ 17

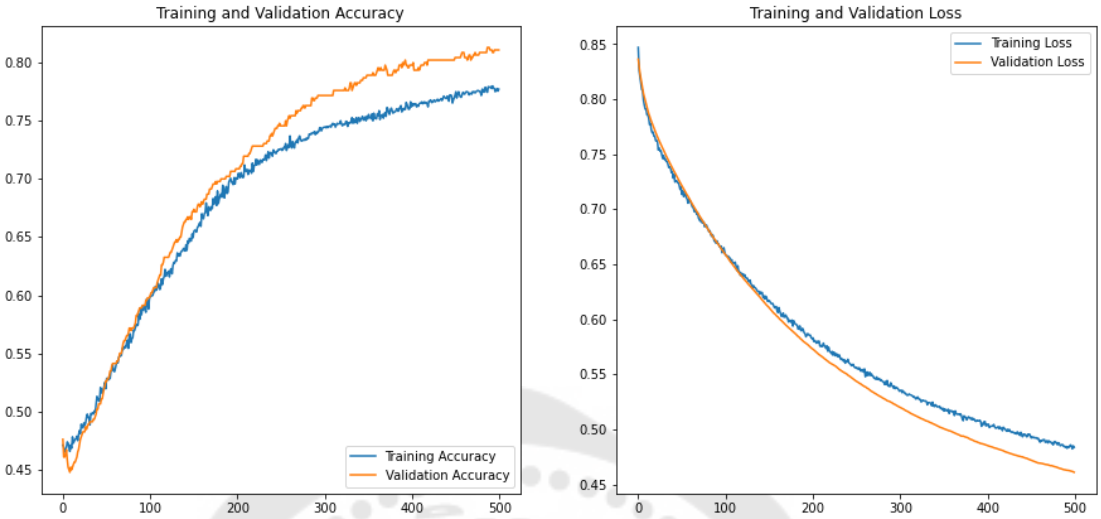
ตาราง 3 เปรียบเทียบประสิทธิภาพความถูกต้องระหว่าง learning rate 0.001 และ 0.0001

Optimizer	Learning rate = 0.001	Learning rate = 0.0001
Accuracy	0.80	0.81

Learning rate RMSprop เท่ากับ = 0.001



Learning rate RMSprop เท่ากับ = 0.001



ภาพประกอบ 17 กราฟแสดง Accuracy และ Loss เปรียบเทียบกันระหว่าง learning rate RMSprop เท่ากับ 0.001 และ 0.0001

3.4.3 การเปรียบเทียบระหว่าง Batch size ขนาดต่าง ๆ

จากการทดสอบปรับพารามิเตอร์ก่อนหน้านี้เราพบว่า พารามิเตอร์ pooling และ optimizer ที่ดีที่สุดสำหรับชุดข้อมูลนี้ (data set) คือ max pooling และ optimizer RMSprop learning rate 0.0001 การทดสอบ batch จึงตั้งค่าพารามิเตอร์เหมือนกันดังนี้ กำหนด pooling เป็น max pooling และ optimizer คือ RMSprop learning rate 0.0001

ตาราง 4 เปรียบเทียบประสิทธิภาพผลลัพธ์ของ Batch size ขนาดต่างๆ

Batch size	Without Augmentation			With Augmentation		
	Accuracy	Precision	Recall	Accuracy	Precision	Recall
16	0.87	0.87	0.87	0.80	0.81	0.80
128	0.83	0.83	0.83	0.80	0.81	0.80
256	0.81	0.81	0.81	0.76	0.77	0.76

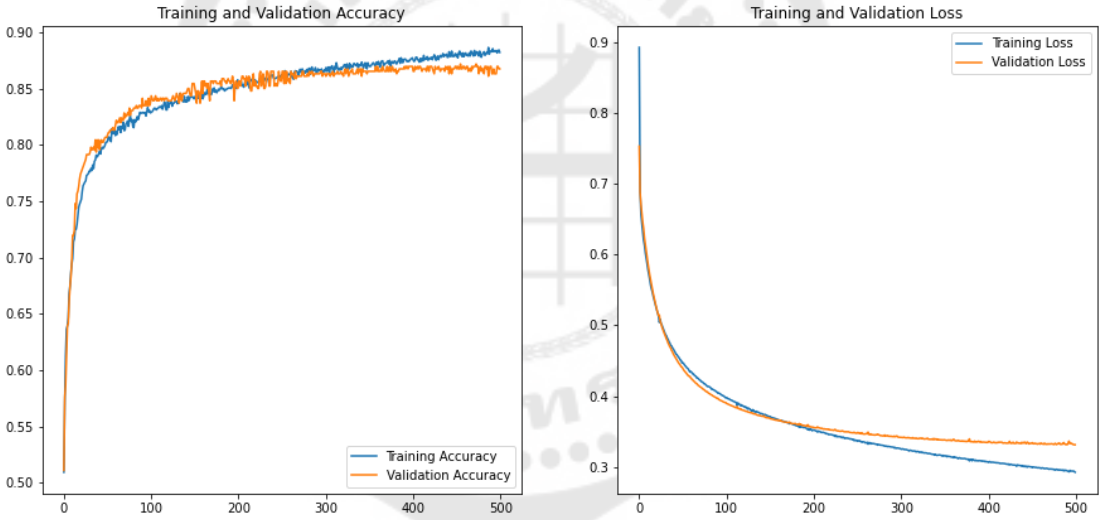
**สำหรับ Batch size 512 เนื่องจากข้อจำกัดของทรัพยากรคอมพิวเตอร์ (kernel dead) ทำให้ไม่สามารถทดสอบ batch size 512 ได้

จากผลการทดลองตาราง 4 เมื่อเปรียบเทียบค่า Batch size ขนาดต่างๆ พบว่า ความแม่นยำ (accuracy) batch size 16, 128, 256 มีค่า 0.87, 0.83, 0.81 ตามลำดับ และ 0.80, 0.80, 0.76 สำหรับการดัดแปลงข้อมูลต้นฉบับ (data augmentation) และเมื่อพิจารณาจากกราฟความสัมพันธ์ระหว่างค่าความแม่นยำ (accuracy) และค่าความสูญเสีย (loss) พบว่า Batch ขนาด 16 เกิด over fitting สำหรับ batch ที่ 128 และ 256 กราฟมีการเรียนรู้ที่ดี ภาพประกอบ 18 เมื่อพิจารณาค่าความแม่นยำ (Accuracy) ดังนั้นจึงสรุปได้สว่า batch size เหมาะสมกับชุดข้อมูล (data set) นี้คือ ขนาด 128

โมเดล VGG16

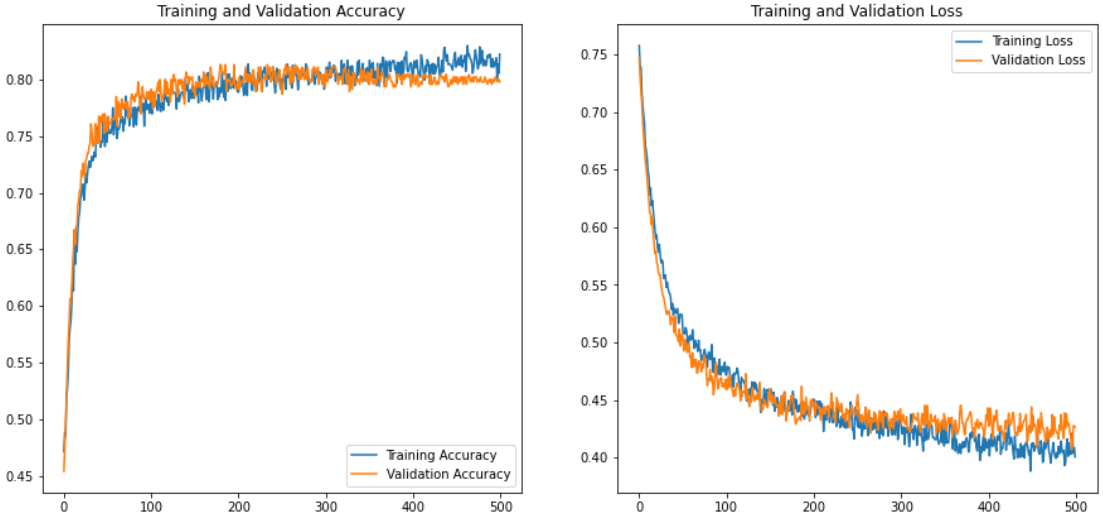
Batch 16

Without data augmentation



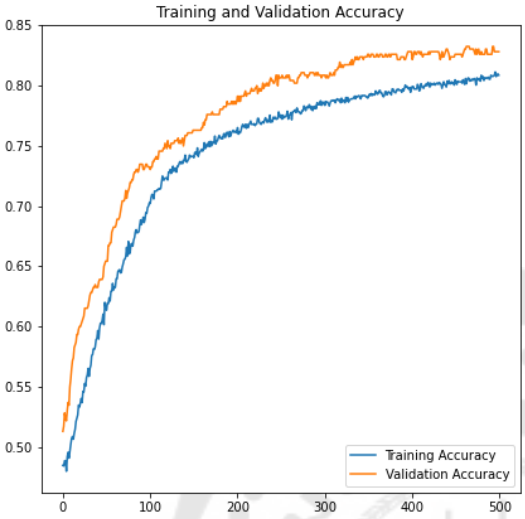
Batch 16

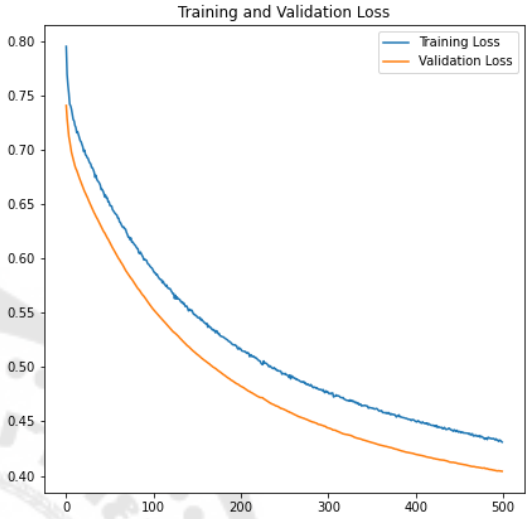
With data augmentation



Batch 128

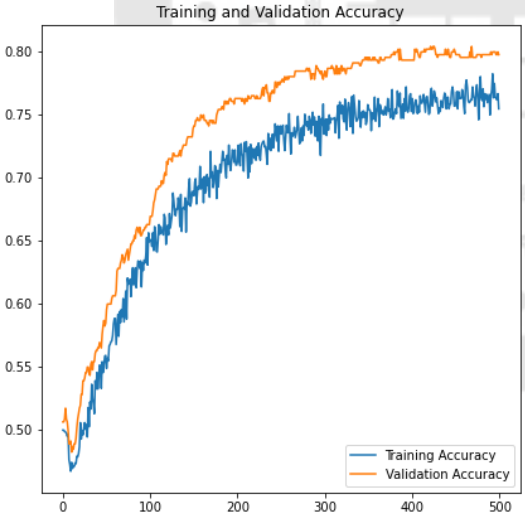
Without data augmentation

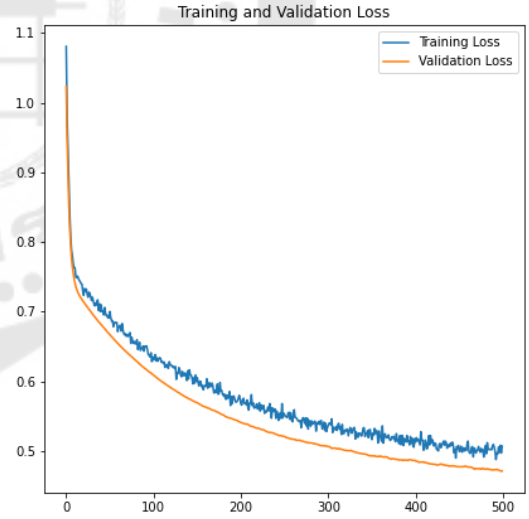




Batch 128

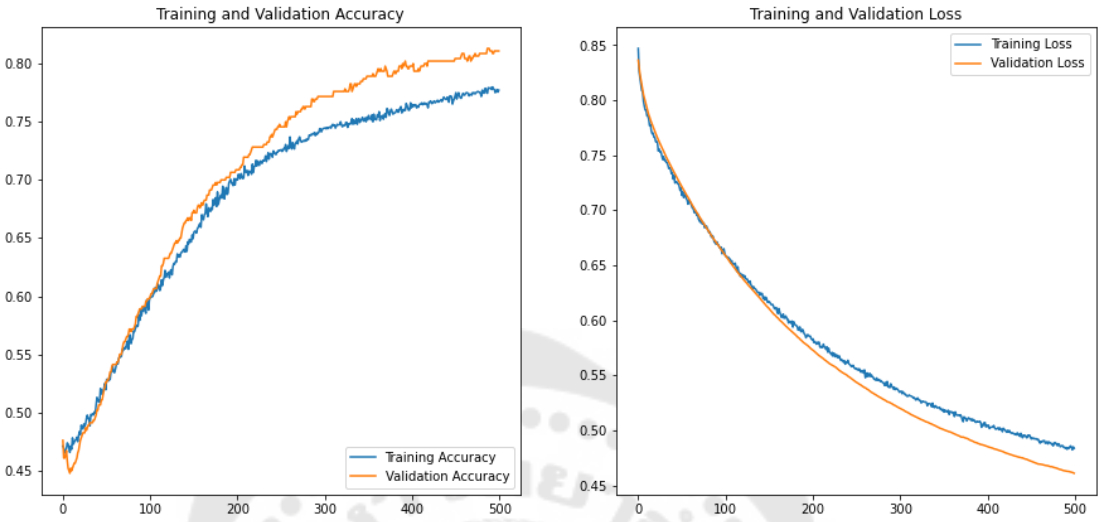
With data augmentation





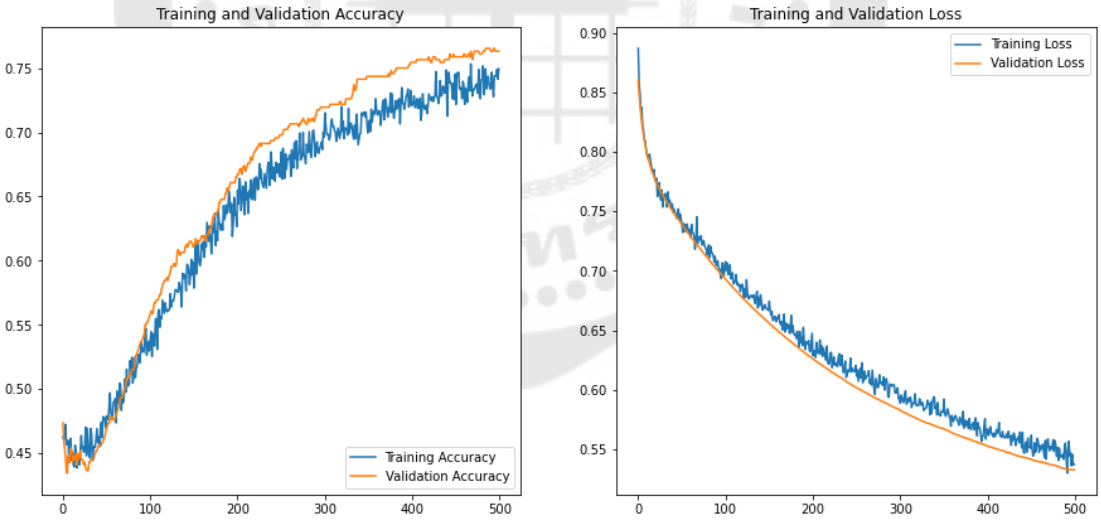
Batch 256

Without data augmentation



Batch 256

With data augmentation



ภาพประกอบ 18 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss ของ Batch size ขนาดต่างๆ

จากการทดสอบปรับเพื่อทดสอบพารามิเตอร์เบื้องต้น ได้พารามิเตอร์ที่ทำให้โมเดลมีประสิทธิภาพมากที่สุด ดังนี้ max pooling RMSprop Learning เท่ากับ 0.0001 และ Batch size เท่ากับ 128 ดังนั้นผู้วิจัยดำเนินการตั้งค่าพารามิเตอร์ดังกล่าวในแต่ละโมเดล ผลลัพธ์ที่ได้นั้นจะกล่าวในหัวข้อผลการวิจัยและอภิปรายผลการวิจัย

3.5 **การทดสอบ** fine tune

จากการที่ได้ทำการทดสอบปรับพรามิเตอร์ได้ผลลัพธ์ทุกโมเดลแล้วในหัวข้อนี้ดำเนินการทดสอบ fine tune เพื่อเพิ่มประสิทธิภาพของโมเดล โดยจะทำการนำชุดข้อมูล (data set) เข้าไปเทรนในชั้น (layer) ท้ายๆ ของโครงสร้างทั้ง 3 โครงสร้าง ตามที่กล่าวในหัวข้อ 3.2 โดยดำเนินการดังนี้

3.5.1 **โมเดล** VGG16

จากโมเดล VGG16 มีจำนวนชั้น (layer) ของโมเดลมีทั้งหมด 19 ชั้น (layer) รวม max pooling layer และ outputs dense ซึ่งรวมทั้งหมด 21 ชั้น ทำการปรับแต่งชั้นเพื่อฝึก (train) โมเดลตั้งแต่ชั้นที่ 16 ถึงชั้นที่ 21 รวมทั้งหมด 6 ชั้น

3.5.2 **โมเดล** ResNet50

จากโมเดล ResNet50 มีจำนวนชั้น (layer) ของโมเดลมีทั้งหมด 175 ชั้น (layer) รวม max pooling layer และ outputs dense ซึ่งรวมทั้งหมด 177 ชั้น ทำการปรับแต่งชั้นเพื่อฝึก (train) โมเดลตั้งแต่ชั้นที่ 144 ถึงชั้นที่ 177 รวมทั้งหมด 34 ชั้น

3.5.3 **โมเดล** InceptionV3

จากโมเดล InceptionV3 มีจำนวนชั้น (layer) ของโมเดลมีทั้งหมด 311 ชั้น (layer) รวม max pooling layer และ outputs dense ซึ่งรวมทั้งหมด 313 ชั้น ทำการปรับแต่งชั้นเพื่อฝึก (train) โมเดลตั้งแต่ชั้นที่ 281 ถึงชั้นที่ 313 รวมทั้งหมด 33 ชั้น

โดยทุกโมเดลได้การทำการ compile โมเดล optimizer RMSprop learning late 0.0001 ทำการฝึก 500 epoch ตามเดิม และกำหนดให้สิ้นสุดการฝึกโดยใช้ early stopping กำหนดค่า loss ใน test set หากไม่ลดลง 3 รอบสุดท้ายของการฝึก (patience = 3) ผลลัพธ์ที่ได้นั้นจะกล่าวในหัวข้อผลการทดลองในหัวข้อถัดไป

บทที่ 4
ผลการทดลอง

จากการเตรียมข้อมูลในบทที่ 3 และในบทนี้จะดำเนินการเทรนโมเดลแบบถ่ายโอน (transfer learning) ทั้ง 3 โมเดลได้แก่ InceptionV3, VGG16 และ ResNet 50 โดยใช้ weight จาก ImageNet และทำการ custom head เพิ่ม pooling layer ใช้ max pooling layer และกำหนดชั้นสุดท้ายเป็น dense เท่ากับ 1 เพื่อกำหนด output โดยใช้ activation sigmoid และ compile model โดยใช้ optimizer เป็น RMSprop learning rate = 0.0001 loss เท่ากับ binary crossentropy วัดประสิทธิภาพการเทรนระหว่างการฝึก แต่ละรอบโดยใช้ metric เท่ากับ accuracy และทำการเทรน 500 epochs

4.1 ผลลัพธ์ประสิทธิภาพผลลัพธ์ของโมเดลแต่ละโมเดล

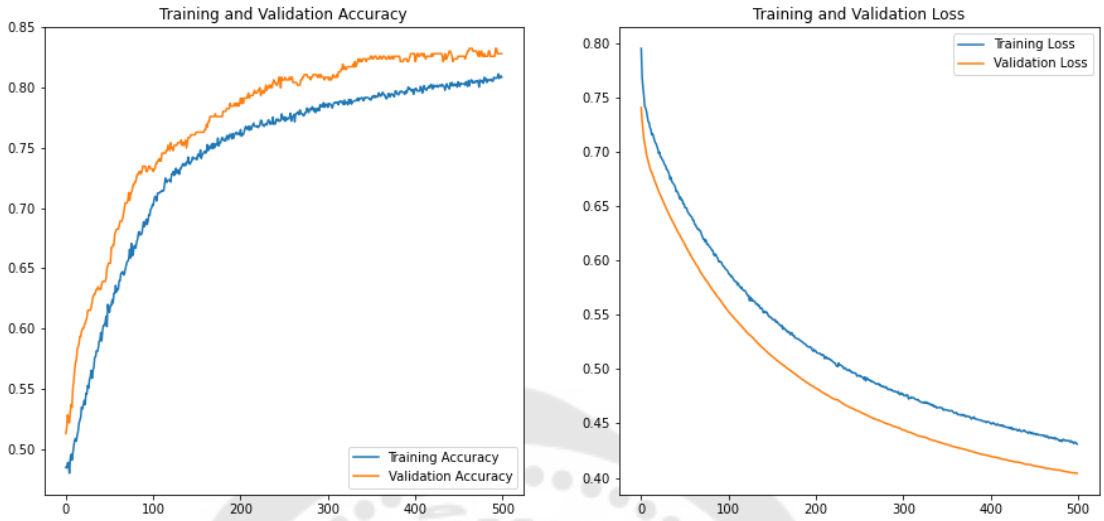
จากการทดลองเทรนโมเดลทั้ง 3 โมเดล ทำการเปรียบเทียบผลลัพธ์ประสิทธิภาพโมเดล ได้ผลการทดลองตามดังนี้

4.1.1 ผลลัพธ์ประสิทธิภาพของโมเดล VGG16

ผลการทำลองที่ได้สำหรับโมเดล VGG16 ได้ผลลัพธ์ accuracy เท่ากับ 0.83 Precision เท่ากับ 0.83 และ Recall เท่ากับ 0.83 (ภาพประกอบ19) และได้ผลลัพธ์กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs (ภาพประกอบ 20) มีค่า accuracy validation ความแม่นยำ เท่ากับ 0.83 และ loss validation ค่าความผิดพลาด เท่ากับ 0.40

classification_report	precision	recall	f1-score	support
Undamage	0.83	0.83	0.83	230
Damage	0.83	0.83	0.83	230
accuracy			0.83	460
macro avg	0.83	0.83	0.83	460
weighted avg	0.83	0.83	0.83	460

ภาพประกอบ 19 classification report โมเดล VGG16



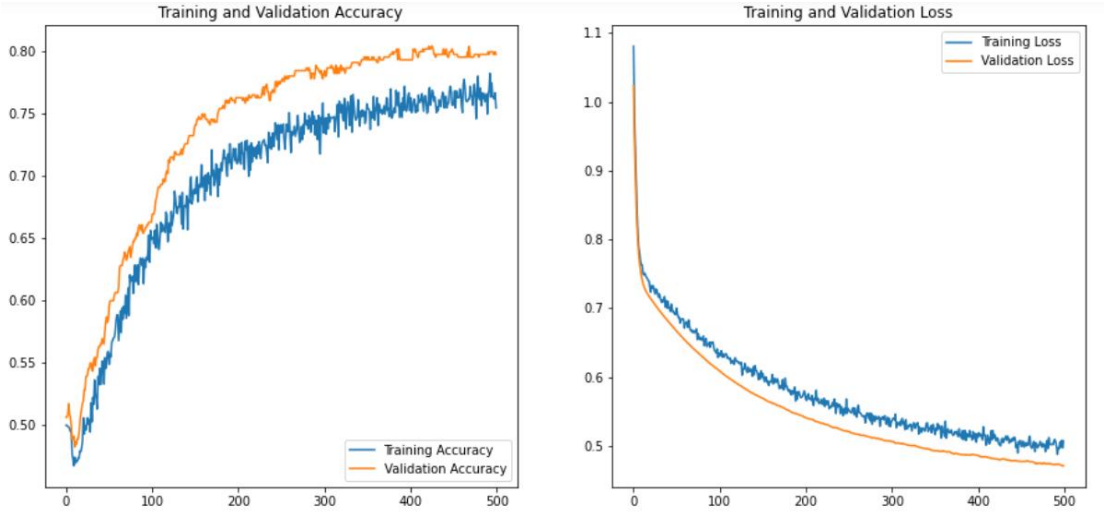
ภาพประกอบ 20 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล VGG16

4.1.2 **ผลลัพธ์ประสิทธิภาพโมเดล** VGG16 with data augmentation

ผลการทำลองที่ได้สำหรับโมเดล VGG16 โดยในขั้นตอน preprocessing ได้เพิ่มขั้นตอนการทำ data augmentation กับข้อมูลเทรนนิ่ง(training set) ได้ผลลัพธ์ accuracy เท่ากับ 0.80 Precision เท่ากับ 0.81 และ Recall เท่ากับ 0.80 (ภาพประกอบ 21) และได้ผลลัพธ์กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs (ภาพประกอบ 22) มีค่า accuracy validation ความแม่นยำ เท่ากับ 0.80 และ loss validation ค่าความผิดพลาด เท่ากับ 0.47

classification_report	precision	recall	f1-score	support
0	0.75	0.90	0.82	230
1	0.88	0.69	0.77	230
accuracy			0.80	460
macro avg	0.81	0.80	0.80	460
weighted avg	0.81	0.80	0.80	460

ภาพประกอบ 21 classification report โมเดล VGG16 with data augmentation



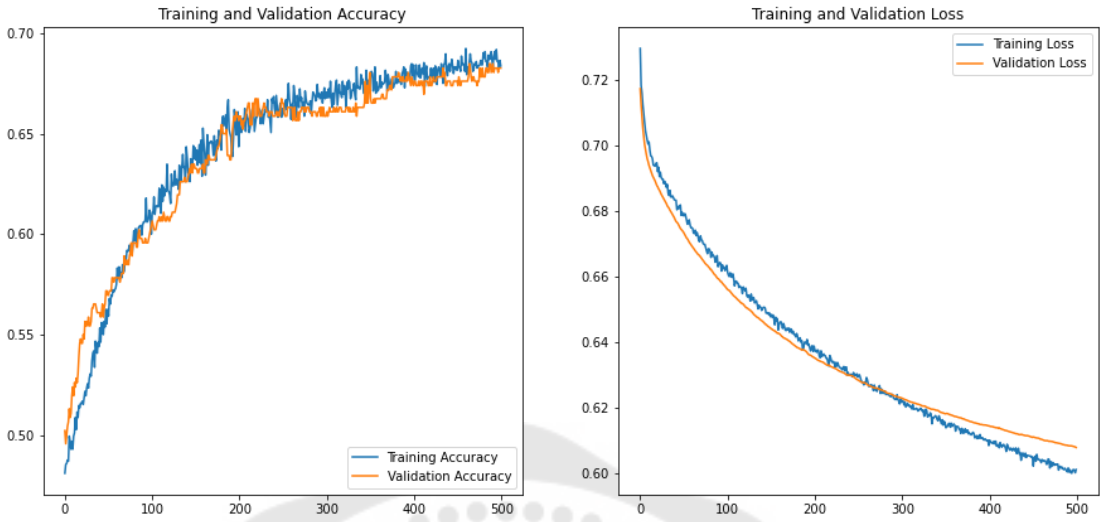
ภาพประกอบ 22 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล VGG16 with data augmentation

4.1.3 ผลลัพธ์ประสิทธิภาพผลลัพธ์ของโมเดล ResNet 50

ผลการทำลองที่ได้สำหรับโมเดล ResNet 50 ได้ผลลัพธ์ accuracy เท่ากับ 0.68 Precision เท่ากับ 0.69 และ Recall เท่ากับ 0.68 (ภาพประกอบ 23) และได้ผลลัพธ์กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs (ภาพประกอบ 24) มีค่า accuracy validation ความแม่นยำ เท่ากับ 0.68 และ loss validation ค่าความผิดพลาด เท่ากับ 0.61

classification_report	precision	recall	f1-score	support
0	0.71	0.62	0.66	230
1	0.66	0.75	0.70	230
accuracy			0.68	460
macro avg	0.69	0.68	0.68	460
weighted avg	0.69	0.68	0.68	460

ภาพประกอบ 23 classification report โมเดล Resnet 50



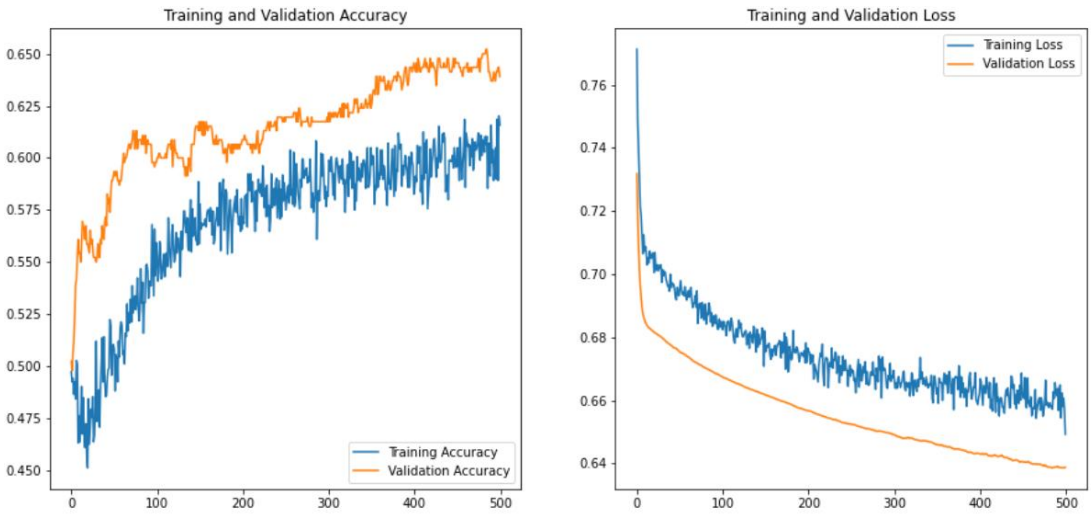
ภาพประกอบ 24 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล Resnet 50

4.1.4 **ผลลัพธ์ประสิทธิภาพโมเดล** ResNet 50 with data augmentation

ผลการทำลองที่ได้สำหรับโมเดล ResNet 50 โดยในขั้นตอน preprocessing ได้เพิ่มขั้นตอนการทำ data augmentation กับข้อมูลเทรน(training set) ได้ผลลัพธ์ accuracy เท่ากับ 0.64 Precision เท่ากับ 0.64 และ Recall เท่ากับ 0.64 (ภาพประกอบ 25) และได้ผลลัพธ์กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs (ภาพประกอบ 26) มีค่า accuracy validation ความแม่นยำ เท่ากับ 0.64 และ loss validation ค่าความผิดพลาด เท่ากับ 0.64

classification_report	precision	recall	f1-score	support
0	0.63	0.66	0.65	230
1	0.65	0.62	0.63	230
accuracy			0.64	460
macro avg	0.64	0.64	0.64	460
weighted avg	0.64	0.64	0.64	460

ภาพประกอบ 25 classification report โมเดล ResNet 50 with data augmentation



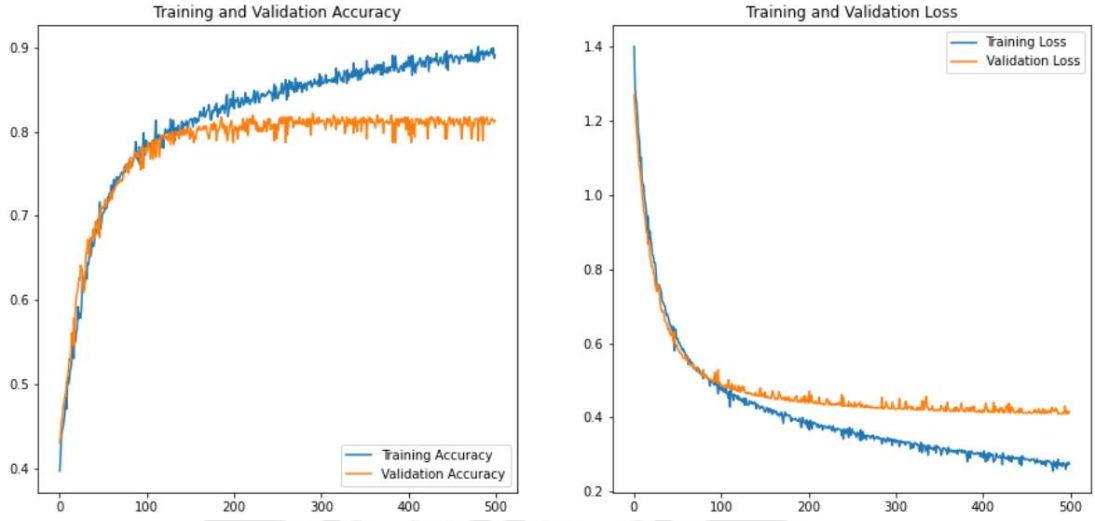
ภาพประกอบ 26 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล ResNet 50 with data augmentation

4.1.5 ผลลัพธ์ประสิทธิภาพผลลัพธ์ของโมเดล InceptionV3

ผลการทำลองที่ได้สำหรับโมเดล InceptionV3 ได้ผลลัพธ์ accuracy เท่ากับ 0.81 Precision เท่ากับ 0.82 และRecall เท่ากับ 0.81 (ภาพประกอบ 27) และได้ผลลัพธ์กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs (ภาพประกอบ 28) มีค่า accuracy validation ความแม่นยำ เท่ากับ 0.81 และ loss validation ค่าความผิดพลาด เท่ากับ 0.42

classification_report	precision	recall	f1-score	support
0	0.78	0.87	0.82	230
1	0.86	0.75	0.80	230
accuracy			0.81	460
macro avg	0.82	0.81	0.81	460
weighted avg	0.82	0.81	0.81	460

ภาพประกอบ 27 classification report โมเดล InceptionV3



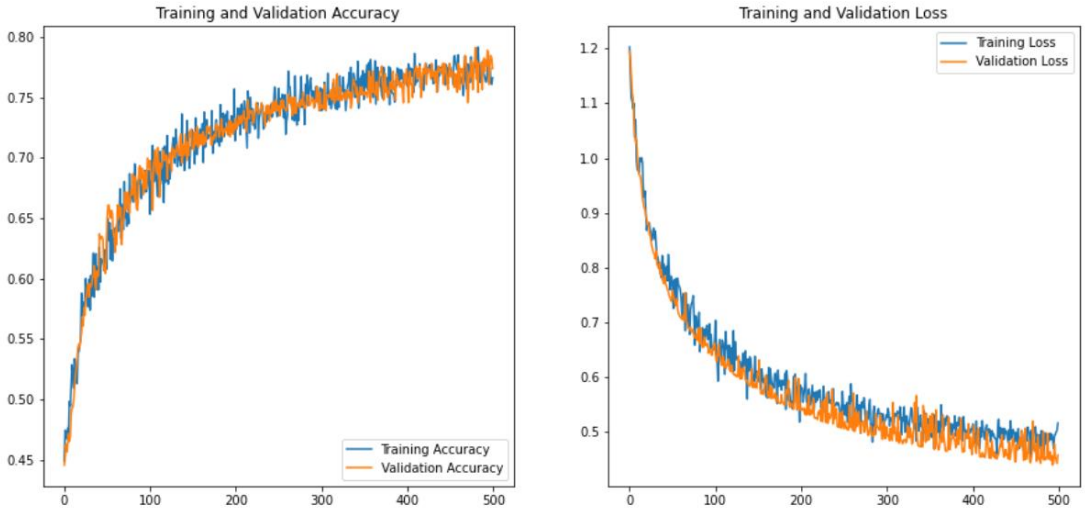
ภาพประกอบ 28 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล inceptionV3

4.1.6 **ผลลัพธ์ประสิทธิภาพโมเดล** InceptionV3 with data augmentation

ผลการทำลองที่ได้สำหรับโมเดล InceptionV3 โดยในขั้นตอน preprocessing ได้เพิ่มขั้นตอนการทำ data augmentation กับข้อมูลเทรน(training set) ได้ผลลัพธ์ accuracy เท่ากับ 0.77 Precision เท่ากับ 0.78 และ Recall เท่ากับ 0.77 (ภาพประกอบ 29) และได้ผลลัพธ์กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs (ภาพประกอบ 30) มีค่า accuracy validation ความแม่นยำ เท่ากับ 0.77 และ loss validation ค่าความผิดพลาด เท่ากับ 0.46

classification_report	precision	recall	f1-score	support
0	0.73	0.87	0.79	230
1	0.84	0.68	0.75	230
accuracy			0.77	460
macro avg	0.78	0.77	0.77	460
weighted avg	0.78	0.77	0.77	460

ภาพประกอบ 29 classification report โมเดล InceptionV3 with data augmentation



ภาพประกอบ 30 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล InceptionV3 with data augmentation

4.2 การเปรียบเทียบผลลัพธ์ประสิทธิภาพของแต่ละโมเดล

จากตาราง 5 นั้นได้แสดงให้เห็นว่า โมเดล VGG16 มีประสิทธิภาพดีที่สุดสำหรับข้อมูลชุดนี้ และรองลงมาคือ InceptionV3 และ Resnet50 ตามลำดับ โดยมี accuracy 0.84 0.81 และ 0.68 ตามลำดับ และสำหรับ preprocessing ที่เพิ่ม data augmentation โมเดลที่มีประสิทธิภาพดีที่สุดคือ VGG16 InceptionV3 และ ResNet50 ตามลำดับ โดยมี accuracy 0.80 0.77 และ 0.64

ตาราง 5 เปรียบเทียบผลลัพธ์ประสิทธิภาพของแต่ละโมเดล

Model	Without Augmentation			With Augmentation		
	Accuracy	Precision	Recall	Accuracy	Precision	Recall
VGG 16	**0.83**	0.83	0.83	**0.80**	0.81	0.80
Resnet 50	0.68	0.69	0.68	0.64	0.64	0.64
InceptionV3	0.81	0.82	0.81	0.77	0.78	0.77

จากผลการทดลองสรุปได้จาก data set ที่ใช้เป็นข้อมูลทดสอบสรุปได้ดังนี้ (ภาพประกอบ 31)

โมเดล VGG16

ทายรถยนต์ที่ไม่มีความเสียหายถูกต้อง 190 คัน ทายผิด 40 คัน

ทายรถยนต์ที่มีความเสียหายถูกต้อง 191 คัน ทายผิด 39 คัน

รวมทายถูกต้องทั้งหมด 381 คัน ทายผิด 79 คัน

โมเดล ResNet50

ทายรถยนต์ที่ไม่มีความเสียหายถูกต้อง 142 คัน ทายผิด 88 คัน

ทายรถยนต์ที่มีความเสียหายถูกต้อง 172 คัน ทายผิด 58 คัน

รวมทายถูกต้องทั้งหมด 314 คัน ทายผิด 146 คัน

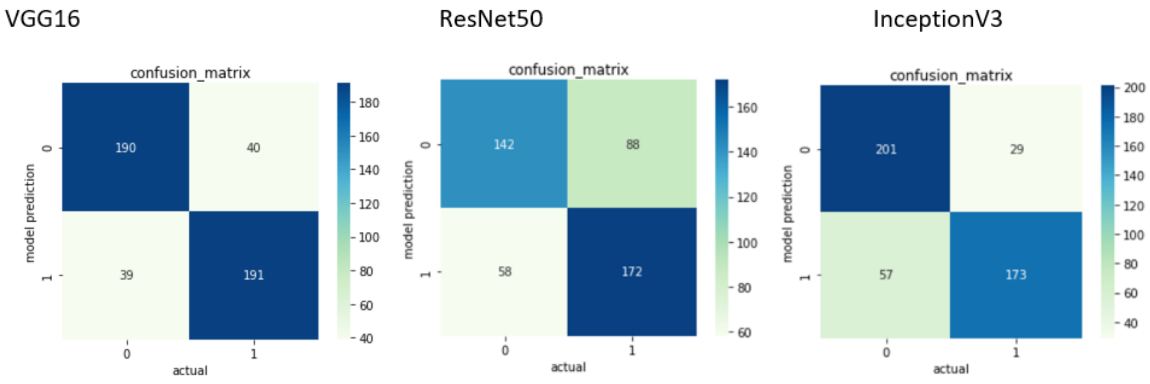
โมเดล InceptionV3

ทายรถยนต์ที่ไม่มีความเสียหายถูกต้อง 201 คัน ทายผิด 29 คัน

ทายรถยนต์ที่มีความเสียหายถูกต้อง 173 คัน ทายผิด 57 คัน

รวมทายถูกต้องทั้งหมด 374 คัน ทายผิด 86 คัน

ผลลัพธ์ confusion matrix ในแต่ละโมเดลดังนี้



ภาพประกอบ 31 confusion matrix ของโมเดลเดล VGG16, ResNet50 และ InceptionV3

จากผลการทดลองเพิ่ม data augmentation ในข้อมูลการเทรนสรุปได้จาก data set ที่ใช้เป็นข้อมูลทดสอบสรุปได้ดังนี้ (ภาพประกอบ 32)

โมเดล VGG16

ทายรถยนต์ที่ไม่มีความเสียหายถูกต้อง 208 คัน ทายผิด 22 คัน

ทายรถยนต์ที่มีความเสียหายถูกต้อง 159 คัน ทายผิด 71 คัน

รวมทายถูกต้องทั้งหมด 367 คัน ทายผิด 93 คัน

โมเดล ResNet50

ทายรถยนต์ที่ไม่มีความเสียหายถูกต้อง 152 คัน ทายผิด 78 คัน

ทายรถยนต์ที่มีความเสียหายถูกต้อง 142 คัน ทายผิด 88 คัน

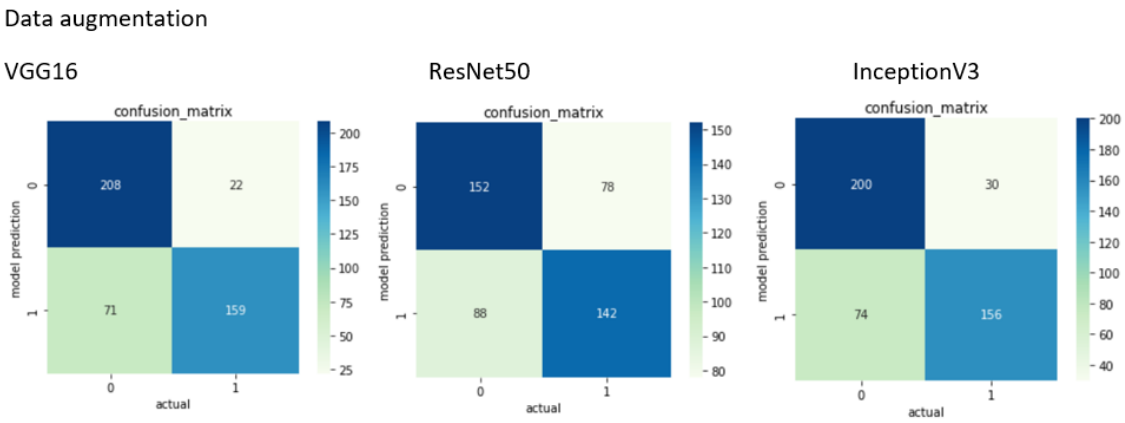
รวมทายถูกต้องทั้งหมด 294 คัน ทายผิด 166 คัน

โมเดล InceptionV3

ทายรถยนต์ที่ไม่มีความเสียหายถูกต้อง 200 คัน ทายผิด 30 คัน

ทายรถยนต์ที่มีความเสียหายถูกต้อง 126 คัน ทายผิด 74 คัน

รวมทายถูกต้องทั้งหมด 326 คัน ทายผิด 104 คัน



ภาพประกอบ 32 confusion matrix ของโมเดลเดล VGG16, ResNet50 และ InceptionV3 with data augmentation

4.3 ผลลัพธ์การเพิ่มประสิทธิภาพด้วยวิธีการ fine tune

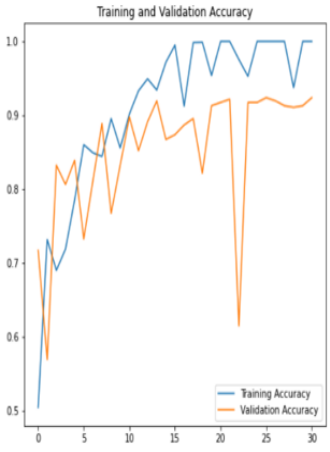
ผลการทำลอง fine tune โดยใช้คำสั่ง Early Stopping ให้สิ้นสุดการฝึก โดยกำหนด monitor ที่ค่า loss หาก loss ไม่เปลี่ยนแปลง 3 epoch โมเดลจะหยุดการฝึกลง จากตาราง 6 นั้นได้แสดงให้เห็นว่า โมเดล VGG16 จำนวน 31 epoch มีประสิทธิภาพดีที่สุดสำหรับข้อมูลชุดนี้ และรองลงมาคือ InceptionV3 จำนวน 160 epoch และ Resnet50 จำนวน 21 epoch โดยมี accuracy 0.92, 0.69 และ 0.55 ตามลำดับ และสำหรับ preprocessing ที่ทำการดัดแปลงข้อมูลต้นฉบับ (data augmentation) โมเดลที่มีประสิทธิภาพดีที่สุดคือ VGG16 จำนวน 11 epoch InceptionV3 จำนวน 22 epoch และ ResNet50 จำนวน 19 epoch โดยมี accuracy 0.85, 0.60 และ 0.50 ตามลำดับ

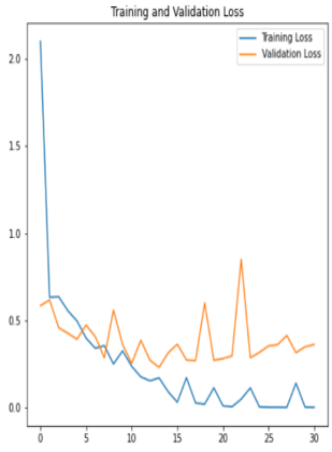
ตาราง 6 เปรียบเทียบผลลัพธ์ประสิทธิภาพของแต่ละโมเดลหลังจากการทดลอง fine tune

Model	Without Augmentation				With Augmentation			
	Accuracy	Precision	Recall	Epoch	Accuracy	Precision	Recall	Epoch
VGG 16	**0.92**	0.92	0.92	31	**0.85**	0.80	0.80	11
Resnet 50	0.55	0.54	0.52	21	0.50	0.50	0.50	19
InceptionV3	0.69	0.62	0.58	160	0.60	0.53	0.44	22

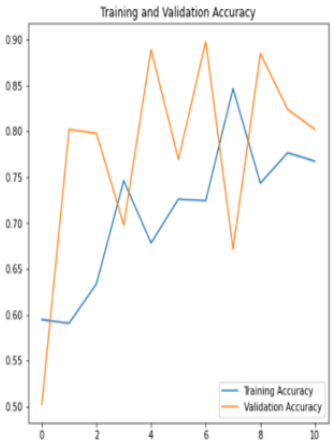
ทั้งนี้เมื่อพิจารณากราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล ทั้ง 6 โมเดล ภาพประกอบ 33 จะเห็นได้ว่าโมเดลมีกราฟเรียนรู้ที่ไม่นิ่งและเกิด overfitting เนื่องจากปริมาณรูปภาพในการฝึก (Train) มีปริมาณไม่เยอะเมื่อเทียบกับชุดข้อมูล ImageNet ที่เป็นฐานข้อมูลสำหรับการฝึกโมเดล VGG16, InceptionV3 และ ResNet50 ซึ่งเป็นโมเดลที่ถูกฝึก (Pre-train) มาเป็นอย่างดีแล้วทั้งสิ้น

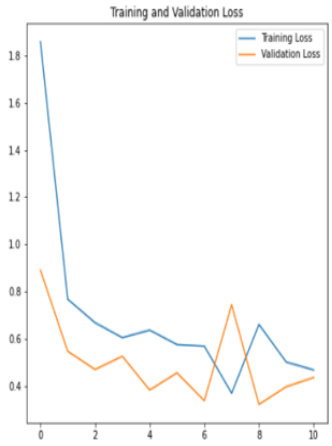
VGG16 without data augmentation



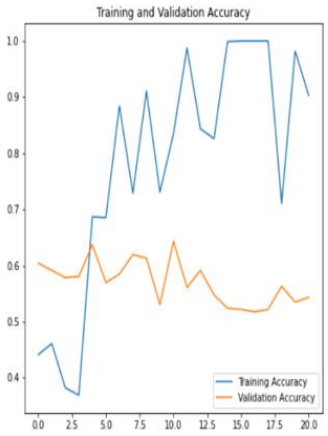


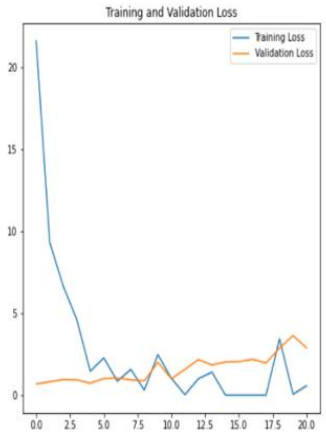
VGG16 with data augmentation



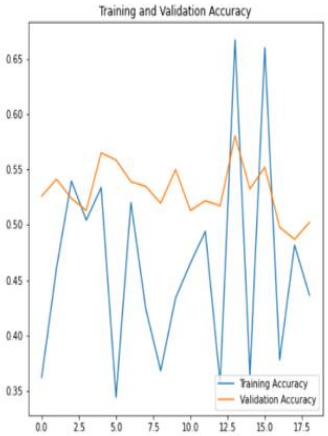


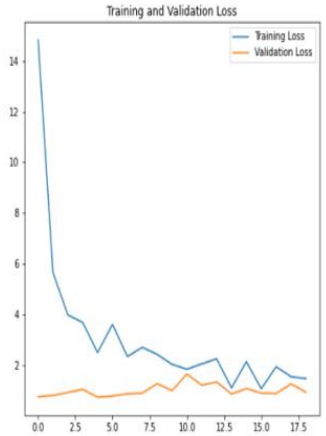
ResNet50 without data augmentation



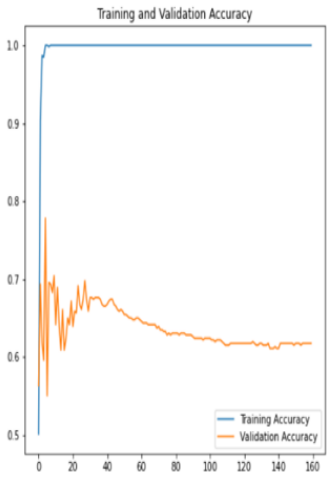


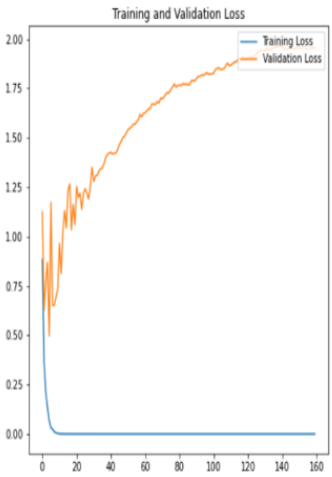
ResNet50 with data augmentation



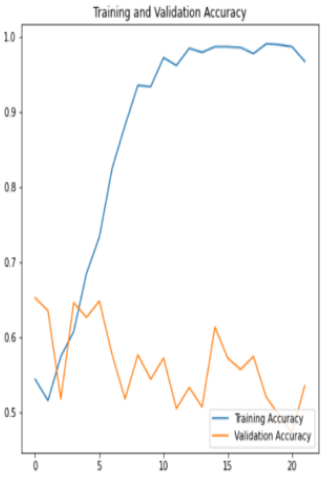


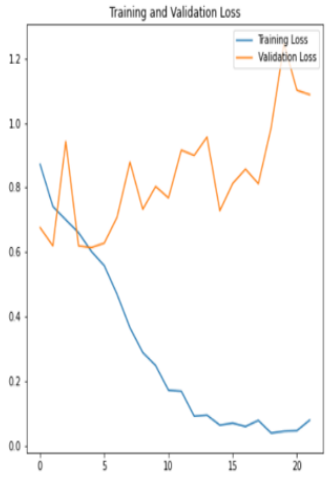
InceptionV3 without data augmentation





InceptionV3 with data augmentation





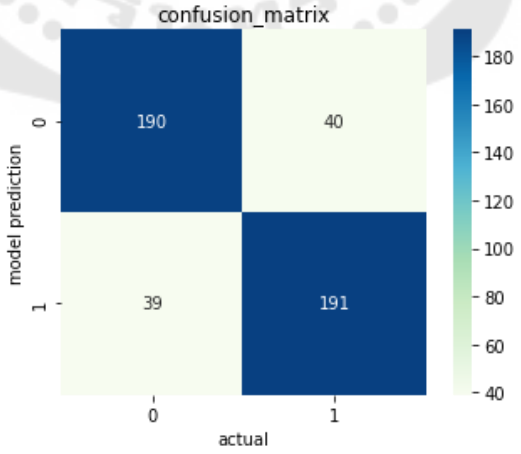
ภาพประกอบ 33 กราฟแสดงความสัมพันธ์ระหว่าง accuracy และ loss กับ epochs ของโมเดล fine tune ทั้ง 6 โมเดล

บทที่ 5
สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ

งานวิจัยในสารนิพนธ์นี้ได้กล่าวถึงการจำแนกรูปภาพรถยนต์ที่มีความเสียหาย และ รูปที่ไม่มีความเสียหาย โดยการเรียนรู้เชิงลึกโดยใช้การเรียนรู้แบบถ่ายโอน (transfer learning) ได้เก็บรวบรวมข้อมูลจากเว็บไซต์ kaggle เป็นหลัก

5.1 สรุปผลการวิจัย

งานวิจัยนี้เป็นการศึกษาวิธีการจำแนกประเภทรูปภาพรถยนต์ที่มีความเสียหาย และไม่มีความเสียหายโดยใช้การเรียนรู้เชิงลึก (Deep Learning) ใช้วิธีการเรียนรู้แบบถ่ายโอน (transfer learning) ได้นำโครงสร้างสถาปัตยกรรม 3 โครงสร้าง ได้แก่ VGG16, ResNet50, InceptionV3 ได้ผลลัพธ์ที่ดีที่สุดคือโมเดล VGG16 เป็นโมเดลที่มีประสิทธิภาพสูงที่สุดในการเปรียบเทียบทั้งหมด 6 โมเดล สำหรับ data set ชุดนี้ สามารถทำนายจำแนกรูปภาพรถยนต์ที่มีความเสียหายและ รูปที่ไม่มีความเสียหายได้ดี ซึ่งสามารถวัดประสิทธิภาพมีผลลัพธ์ดังนี้ Accuracy = 0.83, Precision = 0.83, Recall = 0.83, และ F1 = 0.83 อยู่ในเกณฑ์ที่ดี ตามภาพประกอบ 34 แสดงผลConfusion matrix สามารถจำแนกรถยนต์ที่ไม่มีความเสียหายถูกต้อง 190 คัน ทายผิด 40 คัน และรถยนต์ที่มีความเสียหายถูกต้อง 191 คัน ทายผิด 39 คัน รวมทายถูกต้องทั้งหมด 381 คัน ทายผิดทั้งหมด 79 คัน จากข้อมูลทดสอบ 460 รูปภาพ

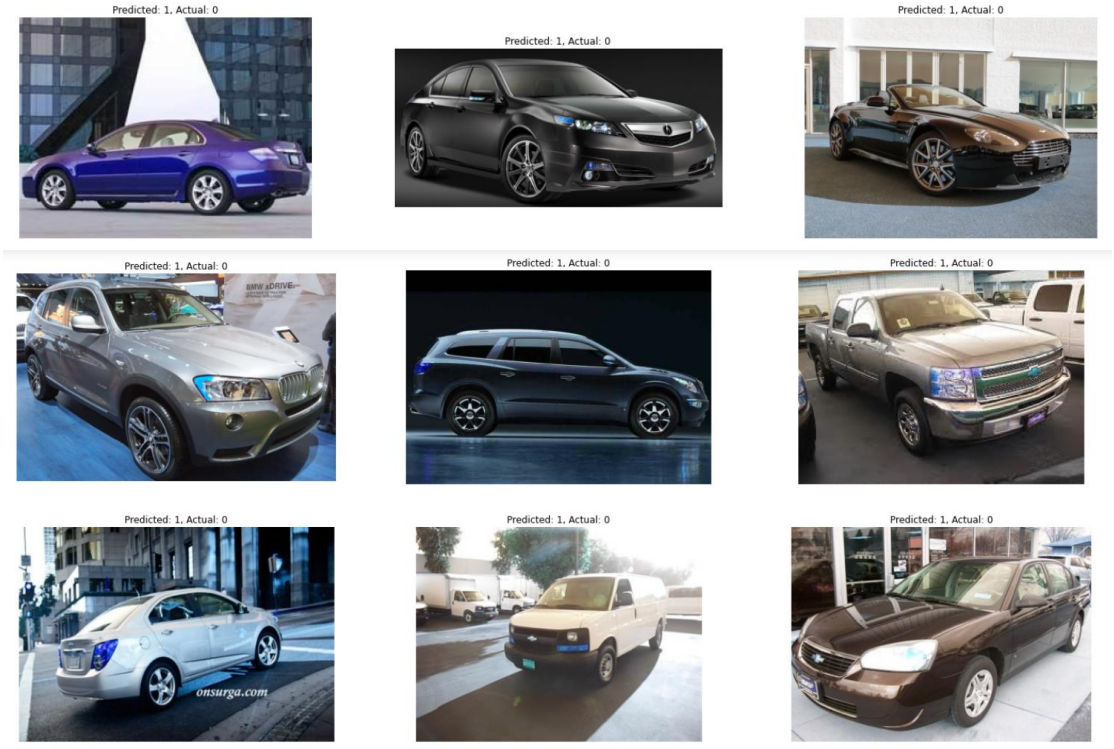


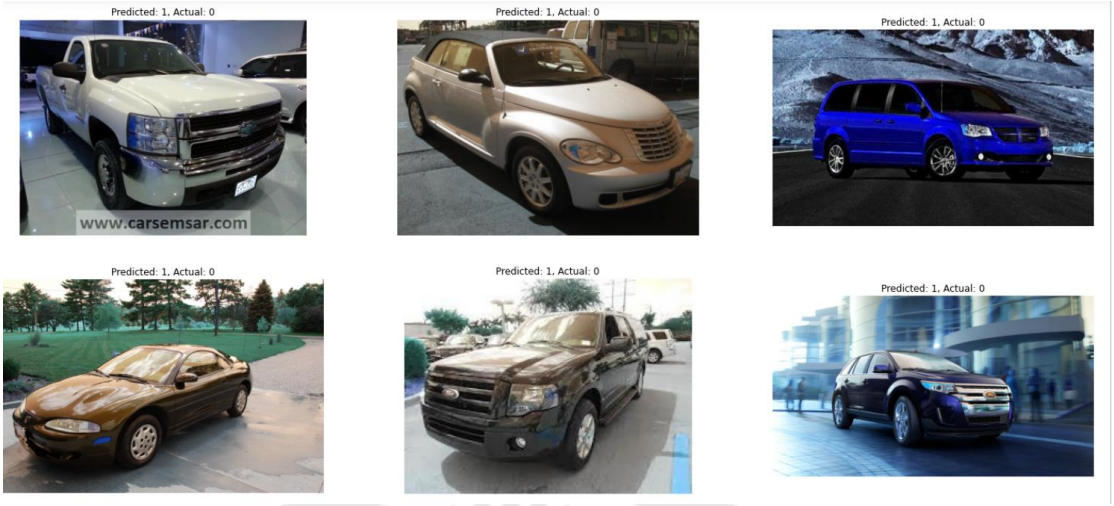
ภาพประกอบ 34 Confusion matrix ของโมเดล VGG16 without augmentation

สำหรับการศึกษานี้สามารถนำมาใช้กับขั้นตอนการดำเนินการเคลมที่ได้กล่าวในหัวข้อ 2.6.1 ในขั้นตอนที่ 2 ซึ่งจะมีส่วนช่วยให้พนักงานไม่ต้องลงเสียเวลาลงพื้นที่เกิดอุบัติเหตุ เช่นหากเกิดอุบัติเหตุที่มีความเสียหายที่ไม่รุนแรงมากหนักและไม่มีคู่กรณีหรือสามารถตกลงกับคู่กรณีได้ลูกค้าประกันก็ทำการแจ้งเคลมและส่งรูปถ่ายอุบัติเหตุให้แก่พนักงานได้โดนที่พนักงานไม่ต้องลงพื้นที่สำรวจอุบัติเหตุ ในงานวิจัยนี้เป็นเพียงจุดเริ่มต้นของการพัฒนาด้านการเคลมประกันรถยนต์ยังเป็นเพียงช่วยเหลือแบ่งเบาหน้าที่ของพนักงานสำรวจภัยเท่านั้น แต่หากอนาคตได้ทำการวิจัยต่อให้สามารถตรวจสอบความเสียหายและประเมินค่าซ่อมได้อย่างแม่นยำก็จะช่วยให้ขั้นตอนดำเนินการเคลมมีความรวดเร็วมากขึ้นอีกทั้งยังป้องกันการฉ้อโกงของศูนย์หรืออู่ที่ประมาณความเสียหายเกินกว่าความเป็นจริงซึ่งจะป้องกันและแก้ไขปัญหาได้

5.2 อภิปรายผลการวิจัย

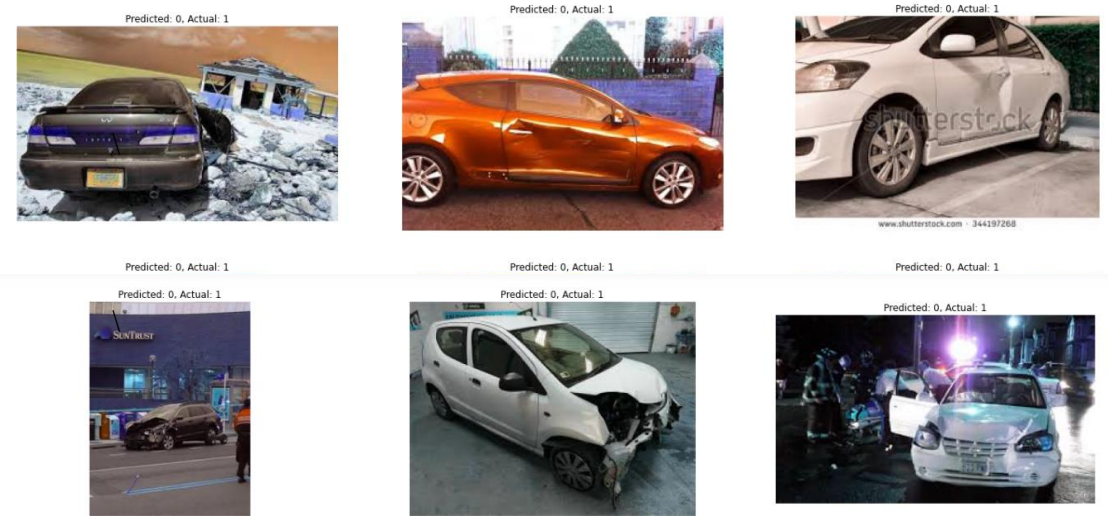
จากข้อสรุปผลการศึกษานั้นเมื่อมาวิเคราะห์รูปภาพที่โมเดลได้ทำนายผิดนั้น ตามภาพประกอบ 35 โมเดลได้ทำการทำนายว่ารถยนต์มีความเสียหาย แต่ที่ถูกต้องคือไม่มีความเสียหาย เมื่อนำรูปภาพมาวิเคราะห์จะเห็นได้ว่ารูปภาพที่ทำนายผิดนั้นรูปภาพส่วนมากที่ทำนายผิดจะเป็นรูปภาพที่รถยนต์ที่มีลักษณะอยู่ในตำแหน่งเฉียงหรือเอียงในมุมประมาณ 45 องศา

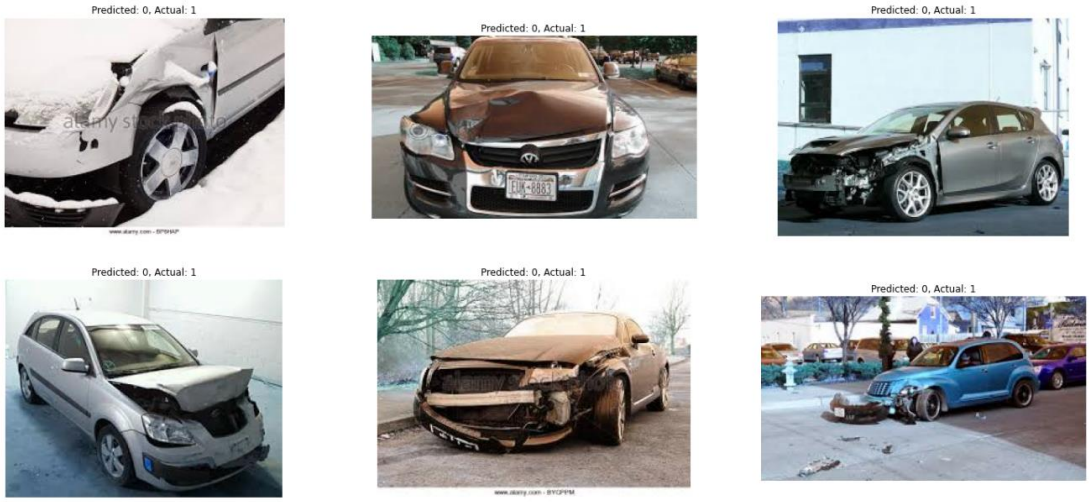




ภาพประกอบ 35 รูปภาพตัวอย่างโมเดล VGG16 without augmentation ทำนายผิด โมเดลทำนายเกิดความเสียหาย

ส่วนที่โมเดลทำนายว่าไม่เกิดความเสียหาย แต่จริงๆแล้วเกิดความเสียหาย ดังภาพประกอบ 36 จากการวิเคราะห์รูปภาพที่เกิดความเสียหายนั้นมีรูปภาพที่เกิดเหตุหลายลักษณะและมีความชัดเจนของความเสียหายแต่โมเดลไม่สามารถที่จะทำนายได้ถูกต้อง ซึ่งยังไม่สามารถสรุปสาเหตุว่าโมเดลทำนายผิดเกิดจากรูปภาพในลักษณะใด จึงต้องทำการศึกษาในอนาคตต่อไป





ภาพประกอบ 36 รูปภาพตัวอย่างโมเดล VGG16 without augmentation ทำนายผิด โมเดลทำนายไม่เกิดความเสียหาย

5.3 ข้อเสนอแนะ

5.3.1 ศึกษาโมเดลเพิ่มเติมโดยใช้หลายๆโมเดลมาทดสอบวัดเปรียบเทียบประสิทธิภาพโดยเฉพาะโมเดลใหม่ๆที่มีประสิทธิภาพที่ดีเช่น EfficientNet

5.3.2 ศึกษาทำวิจัยเพิ่มเติมให้สามารถประเมินค่าเสียหายเป็นจำนวนเงินได้โดยอัตโนมัติเพื่อประมาณการความเสียหายให้ลูกค้าได้ทันที

5.3.3 ทดลองปรับจูนพารามิเตอร์ต่างๆ เพื่อเพิ่มประสิทธิภาพของโมเดล

5.3.4 ใช้คอมพิวเตอร์ที่มีการ์ดจอประสิทธิภาพสูงเพื่อประสิทธิภาพในการทำการวิจัย

บรรณานุกรม

He, K., Zhang, X., Ren, S., และ Sun, J. (2016, 27-30 June 2016). Deep Residual Learning for Image Recognition. Paper presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Krizhevsky, A., Sutskever, I., และ Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM,* 60(6), 84-90.

Kyu, P. M., และ Woraratpanya, K. (2020). *Car Damage Detection and Classification*. Paper presented at the Proceedings of the 11th International Conference on Advances in Information Technology.

O'Shea, K., และ Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv e-prints*.

Patil, K., Kulkarni, M., Sriraman, A., และ Karande, S. (2017). *Deep Learning Based Car Damage Classification*. Paper presented at the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA).

SHAH, A. Car damage detection. สืบค้นจาก https://www.kaggle.com/anujms/car-damage-detection

Simonyan, K., และ Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* 1409.1556.

Singh, R., Ayyar, M. P., Pavan, T. V. S., Gosain, S., และ Shah, R. R. (2019, 11-13 Sept. 2019). *Automating Car Insurance Claims Using Deep Learning Techniques.* Paper presented at the 2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., และ Wojna, Z. (2016, 27-30 June 2016). *Rethinking the Inception Architecture for Computer Vision.* Paper presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

สำนักงานคณะกรรมการกำกับและส่งเสริมการประกอบธุรกิจประกันภัยและสำนักงานสภา

พัฒนาการเศรษฐกิจและสังคมแห่งชาติ. (2563). เบี้ยประกันภัยรับโดยตรงของธุรกิจประกันวินาศภัย ประจำไตรมาส : ม.ค.-ธ.ค. ปี 2563. สืบค้นจาก https://www.oic.or.th/th/industry/92073

ภาคผนวก

Import library

```
import tensorflow as tf
import pandas as pd
import numpy as np
import cv2
import os

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, GlobalAveragePooling2D, Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam

from keras.models import load_model

import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns

from sklearn.metrics import  accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

กำหนด path ข้อมูล

```
dataset_path_new = "./data2"

train_dir = os.path.join(dataset_path_new, "train")
validation_dir = os.path.join(dataset_path_new, "test")
```

ใช้คำสั่ง ImageDataGenerator

```
data_gen_train = ImageDataGenerator(rescale=1/255.)
data_gen_valid = ImageDataGenerator(rescale=1/255.)
```

```
train_generator = data_gen_train.flow_from_directory(train_dir, target_size=(128,128), batch_size=128, class_mode="binary",
valid_generator = data_gen_valid.flow_from_directory(validation_dir, target_size=(128,128), batch_size=128, class_mode="bina
```

Trainning model VGG16 (หากต้องการเรียนเรียกโมเดลอื่นสามารถเปลี่ยน Code ส่วนนี้)

กำหนด ขนาดรูปภาพ 128x128x3

include_top=False ไม่เอาส่วน head ของโมเดล

```
base_model = tf.keras.applications.vgg16.VGG16(input_shape=(128, 128, 3), include_top=False, weights="imagenet")
```

```
base_model.summary()
```

...

```
base_model.trainable = False
```

Add custom head

Max pooling layer dense layer (sigmoid)

เพิ่มส่วน head เพื่อฝึกฝนข้อมูล

```
max_pooling_layer = tf.keras.layers.GlobalMaxPooling2D()(base_model.output)
prediction_layer = tf.keras.layers.Dense(units=1, activation='sigmoid')(max_pooling_layer)
```

```
model = tf.keras.models.Model(inputs=base_model.input, outputs=prediction_layer)
```

```
model.summary()
```

Train model

Optimizers RMSprop learning=0.001

```
model.compile(optimizer=tf.keras.optimizers.RMSprop(lr=0.0001),loss="binary_crossentropy", metrics=["accuracy"])
```

```
...
```

```
modelVGG16=model.fit_generator(train_generator, epochs=500, validation_data=valid_generator)
```

แสดงกราฟ accuracy Loss

```
import matplotlib.pyplot as plt
acc = modelVGG16.history['accuracy']
val_acc = modelVGG16.history['val_accuracy']
loss = modelVGG16.history['loss']
val_loss = modelVGG16.history['val_loss']

epochs_range = range(500)

plt.figure(figsize=(15, 15))
plt.subplot(2, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(2, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

Save model เพื่อเรียกใช้งานครั้งต่อไปโดยไม่ต้องฝึกใหม่

```
# save model for later
model.save('./model/VGG16data2_max_batch128.h5')
```

คำสั่ง load โมเดล ใช้งาน

```
loaded_model = load_model('./model/VGG16data2_max_batch128.h5')
```

คำสั่ง evaluate

```
# print evaluation score
score = loaded_model.evaluate(valid_generator)
print('Loss: {0} Accuracy: {1}'.format(*score))
```

```
# predict validation set
pred = loaded_model.predict(valid_generator)
```

```
# true y and predicted y
y_true = valid_generator.classes
y_pred = (pred.flatten() > 0.5).astype('int32')
```

```
print('Accuracy score:\n', accuracy_score(y_pred, y_true))
```

สร้าง confusion matrix

```
mat = confusion_matrix(y_true, y_pred)

axes = sns.heatmap(mat, square=True, annot=True,
                   fmt='d', cbar=True, cmap=plt.cm.GnBu)

axes.set_xlabel('actual')
axes.set_ylabel('model prediction')

axes.set_title('confusion_matrix')
```

Classification report

```
target_names = ['Undamage', 'Damage']
print('classification_report', classification_report(y_true , y_pred , target_names=target_names))
```

แสดงรูปภาพที่โมเดลทำนายผิด

```
img_index = 1

plt.figure(figsize=(25, 400))

for t, p, i in zip(y_true, y_pred, range(len(y_true))):

    # true_y not equal to predict_y
    if (t != p):
        folder = '00-undamage' if t == 0 else '01-damage'
        ext = 'jpg' if t == 0 else 'JPEG'
        img_number = i + 1 if t == 0 else i + 1 - 230
        img_path = './data2/test/{}/{:04}.{}'.format(folder, img_number, ext)
        image = cv2.imread(img_path, cv2.COLOR_BGR2RGB)

        # image file type is incorrect
        if (image is None):
            continue

        sp = plt.subplot(66, 3, img_index)
        sp.axis('Off')
        sp.set_title('Predicted: {}, Actual: {}'.format(p, t))
        sp.imshow(image)

        img_index += 1

plt.show()
```

แสดงรูปภาพที่โมเดลทำนายถูก

```
img_index = 1

plt.figure(figsize=(25, 800))

for t, p, i in zip(y_true, y_pred, range(len(y_true))):

    # true_y not equal to predict_y
    if (t == p):
        folder = '00-undamage' if t == 0 else '01-damage'
        ext = 'jpg' if t == 0 else 'JPEG'
        img_number = i + 1 if t == 0 else i + 1 - 230
        img_path = './data2/test/{}/{:04}.{}'.format(folder, img_number, ext)
        image = cv2.imread(img_path, cv2.COLOR_BGR2RGB)

        # image file type is incorrect
        if (image is None):
            continue

        sp = plt.subplot(199, 3, img_index)
        sp.axis('Off')
        sp.set_title('Predicted: {}, Actual: {}'.format(p, t))
        sp.imshow(image)

        img_index += 1

plt.show()
```

Fine tuning

คำสั่งกำหนดให้โมเดลสามารถฝึกฝนข้อมูลของเราได้

```
loaded_model.trainable = True
```

กำหนด layer ให้โมเดลชั้นแรกถึง 15 ไม่สามารถฝึกฝนได้

คำสั่งนี้จะทำให้โมเดลเทรนเฉพาะ ชั้นที่ 16 ถึงชั้น 21 ชั้นสุดท้าย

```
for layer in loaded_model.layers[:15]:
    layer.trainable = False
```

คำสั่งกำหนดให้โมเดลหยุดฝึกฝนหาก loss ไม่เปลี่ยนแปลงใน 3 epoch

```
callback = tf.keras.callbacks.EarlyStopping(monitor='loss', patience=3)
```

Complie โมเดล

```
loaded_model.compile(optimizer=tf.keras.optimizers.RMSprop(lr=0.0001),
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

ฝึกโมเดล 500 epoch

```
model_ft_VGG16=loaded_model.fit_generator(train_generator, epochs=500, validation_data=valid_generator,callbacks=[callback])
```

หลังจากนั้น evaluate, confusion ตาม code ด้านบนใหม่เพื่อแสดงผลลัพธ์ของโมเดลที่ได้ดำเนินการ fine tune

ประวัติผู้เขียน

ชื่อ-สกุล	ธนัช เบญจอนุอาชา
วัน เดือน ปี เกิด	15 ธันวาคม 2529
สถานที่เกิด	กรุงเทพฯ
วุฒิการศึกษา	พ.ศ.2553 วิทยาศาสตร์บัณฑิต สาขาสถิติ จาก มหาวิทยาลัยธรรมศาสตร์