

การเปรียบเทียบสมรรถนะของตัวแบบการจำแนกรูปภาพภัยพิบัติ
COMPARISON OF MODELS' PERFORMANCE FOR DISASTER IMAGE
CLASSIFICATION



การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูลและการวิเคราะห์
ศูนย์วิเคราะห์ข้อมูลดิจิทัลอัจฉริยะพระจอมเกล้าลาดกระบัง
คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2566

KMITL--2023-SC-M-017-033

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COMPARISON OF MODELS' PERFORMANCE FOR DISASTER IMAGE CLASSIFICATION

KOSOL RUJIMETHABHAS

AN INDEPENDENT STUDY SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE
IN DATA SCIENCE AND ANALYTICS

KMITL DIGITAL ANALYTICS AND INTELLIGENCE CENTER SCHOOL OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2023

KMITL--2023-SC-M-017-033

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



COPYRIGHT 2023

SCHOOL OF SCIENCE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อการค้นคว้าอิสระ	การเปรียบเทียบสมรรถนะของตัวแบบการจำแนกรูปภาพ ภัยพิบัติ
ชื่อนักศึกษา	นายโกศล รุจิเมธาภาส
รหัสประจำตัว	64605018
ปริญญา	วิทยาศาสตรมหาบัณฑิต (วิทยาการข้อมูลและการวิเคราะห์) ศูนย์วิเคราะห์ข้อมูลดิจิทัลอัจฉริยะพระจอมเกล้าลาดกระบัง
พ.ศ.	2566

อาจารย์ที่ปรึกษาการค้นคว้าอิสระ รองศาสตราจารย์ ดร.ละออ บุญเกษม

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อศึกษาและทำการเปรียบเทียบสมรรถนะตัวแบบโครงข่ายประสาทเทียมคอนโวลูชันนอลแบบมีการถ่ายโอนการเรียนรู้ที่เหมาะสมกับข้อมูลรูปภาพภัยพิบัติจากแพลตฟอร์มอินเทอร์เน็ตและสื่อสังคมออนไลน์ ให้ผลลัพธ์เป็นข้อมูลเบื้องต้นเพื่อสนับสนุนการตัดสินใจตอบสนองต่อภัยพิบัติแต่ละประเภทที่เกิดขึ้น สำหรับหน่วยงานที่เกี่ยวข้องในการตอบสนองต่อเหตุการณ์ โดยเลือกใช้ตัวแบบที่เป็นที่นิยมในปัจจุบันและมีประสิทธิภาพสูงจำนวน 3 ตัวแบบ ประกอบด้วยสถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 จาก TensorFlow Hub ทำการจำแนกประเภทรูปภาพภัยพิบัติ 4 ประเภท ประกอบด้วยความเสียหายที่เกิดขึ้นกับสิ่งปลูกสร้าง ไฟไหม้ น้ำท่วม และไม่มี ความเสียหาย จากชุดข้อมูล Crisis Vision Benchmark (CVB) จำนวน 5,124 รูป และ Aerial Image Database for Emergency Response Applications (AIDER) จำนวน 5,948 รูป ปรับปรุงชุดข้อมูลให้สมดุล ฝึกด้วยตัวแบบมาตรฐาน จากนั้นเพิ่มการปรับปรุงตัวแบบด้วยวิธีการเพิ่มข้อมูล (Data Augmentation) การฝึกด้วยตัวแบบด้วยชุดข้อมูลปัจจุบัน (Fine Tuning) และทั้ง 2 วิธีร่วมกัน ประเมินประสิทธิภาพตัวแบบด้วยเมทริกซ์ความสับสน (Confusion Matrix) ค่าการเรียกคืน (Recall) และค่าความถูกต้อง (Accuracy) โดยตัวแบบมาตรฐาน EfficientNet V2 ที่ฝึกด้วยข้อมูลชุด CVB ในขั้นตอนการฝึกให้ค่าความถูกต้องที่ 94% ค่าการสูญเสียที่ 55% ดีที่สุดเป็นอันดับ 2 ในขั้นตอนการทดสอบด้วยชุดข้อมูลปัจจุบันมีค่าเฉลี่ยการเรียกคืนที่ 86% ค่าความถูกต้องที่ 86% ดีที่สุดเป็นอันดับ 1 และขั้นตอนการทดสอบด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อนให้ค่าความถูกต้องสูงสุดที่ 88% และค่าการสูญเสียต่ำสุดที่ 64% ดีที่สุดเป็นอันดับ 1 เหมาะที่จะนำมาใช้ในการจำแนกประเภทรูปภาพภัยพิบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสำคัญ : การถ่ายโอนการเรียนรู้, การเปรียบเทียบสมรรถนะ, โครงข่ายประสาทเทียมคอนโวลูชัน
นอล, รูปภาพภัยพิบัติ, EfficientNet V2, Inception V3, MobileNet V3, TensorFlow Hub



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Independent Study Title	Comparison of Models' Performance for Disaster Image Classification
Student Name	Kosol Rujimethabhas
Student ID	64605018
Degree	Master of Science (Data Science and Analytics) KMITL-Digital Analytics and Intelligence Center
Year	2023
Independent Study Advisor	Assoc.Prof.Dr.Laor Boongasame

Abstract

The purpose of this research is to study and compare the performance of a convolutional neural network model with transfer learning for disaster image data from internet and social media platforms. The results provide preliminary information to support decision-making in responding to each type of disaster. Three popular, high-performance models were used: EfficientNet V2, Inception V3, and MobileNet V3 architectures from TensorFlow Hub. Disaster image classification was performed for four categories: building damage, fire, flood, and no damage. The models were trained on the Crisis Vision Benchmark (CVB) dataset of 5,124 images and the Aerial Image Database for Emergency Response Applications (AIDER) dataset of 5,948 images. The models were first trained with a standard model. Then, they were enhanced by data augmentation, fine tuning, and a combination of the two methods. The model performance was assessed by the confusion matrix, recall, and accuracy. The standard model EfficientNet V2 trained on the CVB dataset in the training phase yielded an accuracy of 94% and a loss of 55%. In the testing phase with the current dataset, it had a mean recall of 86% and an accuracy of 86%. In the testing phase with never-before-seen datasets, it had the highest accuracy of 88% and the lowest loss of 64%.

Overall, the results showed that the EfficientNet V2 model with transfer learning is suitable for disaster image classification. It achieved high accuracy and low loss on both the current and never-before-seen datasets.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Keywords : Convolutional Neural Network, Disaster Image, EfficientNet V2, Inception V3, MobileNet V3, Performance Comparison, TensorFlow Hub, Transfer Learning



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

การวิจัยเรื่องการเปรียบเทียบประสิทธิภาพโครงข่ายประสาทเทียมคอนโวลูชันนอลแบบมีการถ่ายโอนการเรียนรู้ สำหรับใช้ในการจำแนกรูปภาพภัยพิบัตินี้ ประสบความสำเร็จลุล่วงไปได้ด้วยดีและเป็นไปตามแผนงานที่กำหนดไว้ ข้าพเจ้าขอขอบพระคุณอาจารย์ที่ปรึกษา รศ.ดร.ละออ บุญเกษม และอาจารย์ที่ปรึกษาการค้นคว้าอิสระร่วม ดร.อัศเดช อุดมชัยพร สำหรับการเสียสละเวลาของอาจารย์ทั้งสองท่าน ในการให้คำปรึกษา ความรู้ การชี้แนะแนวทางแก้ปัญหา และการนำความรู้จากหลักสูตรที่เรียนมาประยุกต์ให้ใช้เกิดประโยชน์สูงสุด เช่นเดียวกับคำอุปมาที่กล่าวถึงคนแคะที่ยืนอยู่บนไหล่ของยักษ์ ซึ่งทำให้ผู้วิจัยได้มีความรู้ มุมมองและวิสัยทัศน์ที่กว้างขึ้นกว่าเดิม และจะนำความรู้ที่ได้ไปต่อยอดเพื่อช่วยพัฒนาสังคมและประเทศต่อไปในอนาคต

ขอขอบพระคุณคุณพ่อ คุณแม่ และครอบครัวของข้าพเจ้าสำหรับแรงบันดาลใจและกำลังใจที่ทำให้ข้าพเจ้าพยายามทำให้สำเร็จตามเป้าหมายที่ตั้งไว้

และสุดท้ายนี้ ขอขอบพระคุณคณาจารย์ทุกท่านจากทั้งภายในและภายนอกมหาวิทยาลัยที่ได้ฝึกอบรมให้ความรู้แก่ข้าพเจ้า และขอขอบพระคุณศูนย์วิเคราะห์ข้อมูลดิจิทัลอัจฉริยะพระจอมเกล้าลาดกระบัง สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่เปิดโอกาสให้ข้าพเจ้าได้เข้ามาศึกษาเทคโนโลยีและความรู้เพิ่มเติมด้านวิทยาการข้อมูล

นายโกศล รุจิเมธาภาส

สารบัญ

หน้า

บทคัดย่อภาษาไทย	ก
บทคัดย่อภาษาอังกฤษ	ค
กิตติกรรมประกาศ	จ
สารบัญ	ฉ
สารบัญตาราง	ญ
สารบัญรูป	ฎ
บทที่ 1 บทนำ	1
1.1 ความเป็นมาและความสำคัญของปัญหา	1
1.2 วัตถุประสงค์ของงานวิจัย	2
1.3 ขอบเขตของงานวิจัย	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ	2
1.5 ขั้นตอนการดำเนินงาน	3
บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง	4
2.1 การเรียนรู้เชิงลึก (Deep Learning)	4
2.2 อัลกอริทึมเกรเดียนต์เดสเซนท์ (Gradient Descent Algorithm)	7
2.2.1 สรุปขั้นตอนการทำงานของเกรเดียนต์เดสเซนท์ได้ดังนี้	9
2.2.2 ประเภทของเกรเดียนต์เดสเซนท์	11
2.2.3 ปัญหาของเกรเดียนต์เดสเซนท์	12
2.3 ตัวประมาณค่าโมเมนต์ชนิดปรับปรุงตัวเองได้ (Adaptive Moment Estimator) หรือชื่อย่อ	
อดัมออฟติไมเซอร์ (Adam Optimizer)	14
2.3.1 โมเมนตัม (Momentum)	14
2.3.2 ค่าเฉลี่ยกำลังสองของการแพร่กระจาย (Root Mean Square Propagation – RMSP)	15
2.4 ปัญหาโอเวอร์ฟิตติงและอันเดอร์ฟิตติง (Overfitting and Underfitting)	17
2.4.1 ปัญหาโอเวอร์ฟิตติง (Overfitting)	17
2.4.2 ปัญหาอันเดอร์ฟิตติง (Underfitting)	18
2.4.3 เทคนิคการแก้ปัญหาโอเวอร์ฟิตติงสำหรับโครงข่ายประสาทเทียมการเรียนรู้เชิงลึก	19
2.5 การตรวจสอบความถูกต้องของตัวแบบในระหว่างการฝึก (Model Validation)	23

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ไว้เพื่อใช้ในการศึกษาเท่านั้น ไม่อนุญาตให้เผยแพร่โดยไม่ได้รับอนุญาต

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.1 การแบ่งชุดข้อมูลฝึกและทดสอบตามสัดส่วน (Train/Test Split หรือ Hold-Out Validation)	23
2.5.2 การตรวจสอบแบบข้าม K-Fold (K-Fold Cross-Validation)	24
2.5.3 การตรวจสอบแบบข้าม K-Fold แบบมีการแบ่งชั้น (Stratified K-Fold Cross-Validation)	24
2.5.4 การตรวจสอบข้ามแบบเว้นครั้งเดียว (Leave-One-Out Validation)	25
2.6 โครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (Convolutional Neural Network)	26
2.6.1 ชั้นคอนโวลูชันนอล (Convolutional Layer)	26
2.6.2 ชั้นการรวมกัน (Pooling Layer)	30
2.6.3 ชั้นการเชื่อมต่อสมบูรณ์ (Fully Connected Layer)	31
2.7 การสูญเสียเอนโทรปีข้ามหมวดหมู่ (Categorical Cross-Entropy Loss) หรือการสูญเสีย Softmax (Softmax Loss)	33
2.8 การถ่ายโอนการเรียนรู้ (Transfer Learning)	35
2.8.1 การฝึกล่วงหน้า (Pre-Training)	36
2.8.2 การสกัดคุณสมบัติ (Feature Extraction)	36
2.8.3 การปรับแต่งอย่างละเอียด (Fine-Tuning)	36
2.9 สถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันนอล	36
2.9.1 สถาปัตยกรรม EfficientNet V2	37
2.9.2 สถาปัตยกรรม Inception v3	39
2.9.3 สถาปัตยกรรม MobileNetV3	40
2.10 การประเมินผลประสิทธิภาพของตัวแบบ (Performance Metrics)	41
2.10.1 เมทริกซ์ความสับสน (Confusion Matrix)	41
2.10.2 ตัวชี้วัดประสิทธิภาพ	42
2.11 งานวิจัยที่เกี่ยวข้อง	43
2.11.1 EmergencyNet: Efficient Aerial Image Classification for Drone-Based Emergency Monitoring Using Atrous Convolutional Feature Fusion โดย C. Kyrkou [38]	44
2.11.2 Exploring The Role Of Deep Neural Networks For Post-Disaster Decision Support โดย N. Chaudhuri et al [39]	46
2.11.3 Multi-Modality Helps In Crisis Management: An Attention-Based Deep Learning Approach Of Leveraging Text For Image Classification โดย Z. Ahmad et al [40]	47

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11.4 Transfer Learning-Based Automatic Hurricane Damage Detection Using Satellite Images โดย S.Kaur et al [41]	49
บทที่ 3 วิธีการดำเนินงานวิจัย	51
3.1 ระเบียบวิธีวิจัย	51
3.1.1 ศึกษางานวิจัยที่เกี่ยวข้องในการจำแนกรูปภาพภัยพิบัติ	51
3.1.2 ศึกษาวิธีการจำแนกข้อมูลรูปภาพด้วยตัวแบบการเรียนรู้เชิงลึก (Deep Learning) วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) การนำมาประยุกต์ใช้ และเลือกตัวแบบที่ต้องการ	52
3.1.3 รวบรวมข้อมูลรูปภาพภัยพิบัติ	53
3.1.4 ปรับปรุงข้อมูลรูปภาพภัยพิบัติ (Data Preprocessing)	57
3.1.5 การทดลองจำแนกข้อมูลด้วยวิธีการเรียนรู้เชิงลึกร่วมกับวิธีการถ่ายโอนการเรียนรู้	61
3.1.6 ประเมินผลการทดลองและปรับปรุงแก้ไขตัวแบบให้มีประสิทธิภาพที่ดีขึ้น	80
3.1.7 วิเคราะห์ สรุปผลการทดลอง และให้ข้อเสนอแนะ	80
บทที่ 4 ผลการวิจัยและการอภิปรายผล	81
4.1 ผลการทดลองการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม EfficientNet V2 กับชุดข้อมูล CVB และชุดข้อมูล AIDER	81
4.1.1 ผลลัพธ์จากการฝึกตัวแบบ EfficientNet V2	82
4.1.2 ผลลัพธ์จากการทดสอบตัวแบบ EfficientNet V2	82
4.1.3 ผลลัพธ์จากการทดสอบตัวแบบ EfficientNet V2 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	83
4.2 ผลการทดลองการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม Inception V3 กับชุดข้อมูล CVB และชุดข้อมูล AIDER	84
4.2.1 ผลลัพธ์จากการฝึกตัวแบบ Inception V3	84
4.2.2 ผลลัพธ์จากการทดสอบตัวแบบ Inception V3	85
4.2.3 ผลลัพธ์จากการทดสอบตัวแบบ Inception V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	85
4.3 ผลการทดลองการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม MobileNet V3 กับชุดข้อมูล CVB และชุดข้อมูล AIDER	86
4.3.1 ผลลัพธ์จากการฝึกตัวแบบ MobileNet V3	86
4.3.2 ผลลัพธ์จากการทดสอบตัวแบบ MobileNet V3	87
4.3.3 ผลลัพธ์จากการทดสอบตัวแบบ MobileNet V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	88
4.4 ปัญหาที่พบในการจำแนกประเภทรูปภาพภัยพิบัติ	88
4.4.1 ปัญหาการใช้ข้อมูลที่เหมาะสมในการฝึกตัวแบบ	88
4.4.2 ปัญหาจากการใช้งาน Tensorflow	88
4.4.3 ปัญหาด้านทรัพยากรการประมวลผลที่จำกัด	89

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 การอภิปรายผล	89
4.5.1 องค์ความรู้จากการทดลองสถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 แบบใช้วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) จาก Tensorflow Hub	89
4.5.2 องค์ความรู้จากการเลือกใช้ชุดข้อมูลที่เหมาะสมในการฝึกตัวแบบ	90
บทที่ 5 สรุปผลการวิจัยและข้อเสนอแนะ	92
5.1 สรุปผลการดำเนินงาน	92
5.2 ข้อเสนอแนะ	93
5.2.1 ทดลองสร้างตัวแบบด้วยชุดข้อมูลสมระหว่าง CVB และ AIDER	93
5.2.2 ทดลองสร้างตัวแบบด้วยชุดข้อมูลสมรูปภาพและข้อความ (Multimodal Dataset)	94
5.2.3 สร้างระบบ Social Listening Platform	94
เอกสารอ้างอิง	95



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตารางที่	หน้า
ตารางที่ 1.1 ตารางแสดงจำนวนข้อมูลรูปภาพจาก CVB และ AIDER	2
ตารางที่ 2.1 ตารางเปรียบเทียบเพื่อในการเลือกใช้เรกูลาร์ไรเซชัน L1 และ L2 [12]	20
ตารางที่ 3.1 ประเภทและจำนวนรูปภาพจากชุดข้อมูล CVB ที่นำมาใช้ในงานวิจัยนี้	55
ตารางที่ 3.2 ประเภทและจำนวนรูปภาพจากชุดข้อมูล AIDER ที่นำมาใช้ในงานวิจัยนี้	56
ตารางที่ 3.3 จำนวนรูปของชุดข้อมูล CVB หลังทำการแก้ไขปัญหาข้อมูลไม่สมดุลและทำการแบ่งชุดข้อมูล	63
ตารางที่ 3.4 จำนวนรูปของชุดข้อมูล AIDER หลังทำการแก้ไขปัญหาข้อมูลไม่สมดุลและทำการแบ่งชุดข้อมูล	63
ตารางที่ 3.5 รูปแบบ One Hot ของข้อมูลคำตอบ	68
ตารางที่ 3.6 ประเภทและจำนวนรูปของชุดข้อมูล eval_ds	80
ตารางที่ 4.1 ประสิทธิภาพการฝึกตัวแบบ EfficientNet V2	82
ตารางที่ 4.2 ประสิทธิภาพการทดสอบตัวแบบ EfficientNet V2	83
ตารางที่ 4.3 ประสิทธิภาพการทดสอบตัวแบบ EfficientNet V2 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	83
ตารางที่ 4.4 ประสิทธิภาพการฝึกตัวแบบ Inception V3	84
ตารางที่ 4.5 ประสิทธิภาพการทดสอบตัวแบบ Inception V3	85
ตารางที่ 4.6 ประสิทธิภาพการทดสอบตัวแบบ Inception V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	86
ตารางที่ 4.7 ประสิทธิภาพการฝึกตัวแบบ MobileNet V3	86
ตารางที่ 4.8 ประสิทธิภาพการทดสอบตัวแบบ MobileNet V3	87
ตารางที่ 4.9 ประสิทธิภาพการทดสอบตัวแบบ MobileNet V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	88
ตารางที่ 4.10 เปรียบเทียบประสิทธิภาพการทดลองตัวแบบ EfficientNet V2, Inception V3 และ MobileNet V3	90
ตารางที่ 4.11 เปรียบเทียบประสิทธิภาพการทดสอบตัวแบบ EfficientNet V2, Inception V3 และ MobileNet V3 มาตรฐาน ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน	91

สารบัญรูป

รูปที่	หน้า
รูปที่ 2.1 ความสัมพันธ์ระหว่าง Deep Learning Machine Learning และ Artificial Intelligence [1]	5
รูปที่ 2.2 การทำงานของเซลล์ประสาทมนุษย์และเซลล์ประสาทเทียม [2]	5
รูปที่ 2.3 การทำงานของ Deep Learning กับข้อมูลรูปภาพ [1]	6
รูปที่ 2.4 การทำงานโครงข่ายประสาทเทียมการเรียนรู้เชิงลึก [3]	7
รูปที่ 2.5 การทำงานของ Gradient Descent [5]	9
รูปที่ 2.6 ความแตกต่างของ Learning Rate [5]	10
รูปที่ 2.7 ผลกระทบต่อการสูญเสียที่ค่า Learning Rate แตกต่างกัน [6]	10
รูปที่ 2.8 จุดต่ำสุดสัมพัทธ์ (Local Minima) และจุดอานม้า (Saddle Point) [5]	13
รูปที่ 2.9 การเปรียบเทียบตัวเพิ่มประสิทธิภาพที่ใช้สำหรับการฝึกเพื่อปรับปรุงประสิทธิภาพของโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลภาพ MNIST [8]	17
รูปที่ 2.10 ตัวอย่างของปัญหาอันเดอร์ฟิตติงและโอเวอร์ฟิตติง [11]	19
รูปที่ 2.11 ผลการการปรับ Regularization ใน Deep Learning [12]	20
รูปที่ 2.12 การทำ Dropout ใน Deep Learning [13]	21
รูปที่ 2.13 การทำ Data Augmentation ใน Deep Learning [14]	22
รูปที่ 2.14 การแบ่งชุดข้อมูลแบบ Train/Test Split หรือ Hold-Out Validation [16]	23
รูปที่ 2.15 การแบ่งชุดข้อมูลแบบ K-Fold Cross-Validation [16]	24
รูปที่ 2.16 การสุ่มแบบ Stratified [17]	25
รูปที่ 2.17 การแบ่งชุดข้อมูลแบบ Leave-One-Out Validation [16]	25
รูปที่ 2.18 สถาปัตยกรรม CNN [18]	26
รูปที่ 2.19 การทำ Dot Product ของเมทริกซ์รูปภาพและตัวกรองเพื่อสร้างฟังก์ชันลักษณะ [19]	27
รูปที่ 2.20 การทำ Convolution เพื่อสร้างเมทริกซ์ Activation Map รูปเมทริกซ์สีน้ำเงินคือรูปภาพขาเข้า และเมทริกซ์สีเทาคือ Activation Map [19]	28
รูปที่ 2.21 ภาพรวมการทำคอนโวลูชันนอล โดยตัวกรองจะเลื่อนไปตามเมทริกซ์ของรูปภาพขาเข้าจนครบทุกด้าน เพื่อสร้างเมทริกซ์ฟังก์ชันลักษณะขึ้นมา [20]	28
รูปที่ 2.22 กราฟฟังก์ชันกระตุ้น ReLU [21]	29
รูปที่ 2.23 การทำงานของฟังก์ชันกระตุ้น ReLU [22]	30
รูปที่ 2.24 ขั้นตอนการทำ Max Pooling [23]	30
รูปที่ 2.25 ขั้นตอนการทำ Average Pooling [23]	31

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.26 การปรับลดมิติของฟังก์ชันลักษณะลง (Flattening) ให้เป็นเวกเตอร์เดี่ยว (Single Vector)	32
รูปที่ 2.27 โครงสร้างชั้น Fully Connected และฟังก์ชันกระตุ้น Softmax สำหรับใช้จำแนกข้อมูล	33
รูปที่ 2.28 การจำแนกหลายคลาส [28]	34
รูปที่ 2.29 Categorical Cross Entropy Loss หรือ Softmax Loss [29]	34
รูปที่ 2.30 โครงสร้าง Fused-MBConv ใน EfficientNet V2 [31]	38
รูปที่ 2.31 สถาปัตยกรรมพื้นฐานของโครงข่าย EfficientNet-B0 [30]	38
รูปที่ 2.32 ประสิทธิภาพความถูกต้องของ EfficientNet V2 [32]	39
รูปที่ 2.33 ภาพรวมโครงข่ายสถาปัตยกรรม Inception V3 [34]	40
รูปที่ 2.34 ประสิทธิภาพความถูกต้องของ MobileNet V3 [36]	41
รูปที่ 2.35 เมทริกซ์ความสับสน (Confusion Matrix) [37]	42
รูปที่ 2.36 ประสิทธิภาพของสถาปัตยกรรม EmergencyNet	45
รูปที่ 2.37 Confusion Matrix จากการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม EmergencyNet	45
รูปที่ 2.38 ประสิทธิภาพของ Deep Learning และ Machine Learning ในการจำแนกรูปภาพภัยพิบัติ	47
รูปที่ 2.39 ประสิทธิภาพของ Deep Learning กับชุดข้อมูล Class I และ Class II	47
รูปที่ 2.40 สถาปัตยกรรม Attention-Based Deep Learning สำหรับชุดข้อมูล Multi-Modality	48
รูปที่ 2.41 ประสิทธิภาพของ Deep Learning แบบใช้และไม่ใช้ Attention-Based	49
รูปที่ 2.42 สถาปัตยกรรม CNN ที่มีการเพิ่มขึ้น Dense Layer ชั้นที่ 2	50
รูปที่ 2.43 ประสิทธิภาพของ DenseNet121 MobileNet V2 Inception V3 และ VGG16 แบบไม่มีและมีการเพิ่มขึ้น Dense Layer ชั้นที่ 2	51
รูปที่ 3.1 ระเบียบวิธีวิจัยการจำแนกรูปภาพภัยพิบัติ	51
รูปที่ 3.2 TensorFlow Hub	52
รูปที่ 3.3 ขั้นตอนการจัดเก็บข้อมูลในงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning	53
รูปที่ 3.4 ประเภทและจำนวนรูปภาพจากชุดข้อมูลในงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning	54
รูปที่ 3.5 ตัวอย่างรูปภาพแต่ละประเภทจากชุดข้อมูลในงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning	54
รูปที่ 3.6 การกระจายตัวของชุดข้อมูล CVB	55
รูปที่ 3.7 ตัวอย่างรูปภาพแต่ละประเภทจากชุดข้อมูล AIDER	56

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.8 การกระจายตัวของชุดข้อมูล AIDER	57
รูปที่ 3.9 ตัวอย่างภาพที่ถูกสุ่มเลือกจากชุดข้อมูล CVB	59
รูปที่ 3.10 ตัวอย่างภาพที่ถูกสุ่มเลือกจากชุดข้อมูล AIDER	60
รูปที่ 3.11 รายงานประสิทธิภาพการทำนายจากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)	65
รูปที่ 3.12 ค่าคำตอบจริงเปรียบเทียบกับคำตอบจากการทำนายของตัวแบบจากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)	65
รูปที่ 3.13 ความแม่นยำของตัวแบบในขั้นการฝึกและประเมินตัวแบบจากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)	66
รูปที่ 3.14 เมทริกซ์ความสับสน (Confusion Matrix) จากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)	67
รูปที่ 3.15 ผลการฝึกตัวแบบ EfficientNet V2 ด้วยชุดข้อมูล CVB ที่ Epoch = 10	74
รูปที่ 3.16 ตัวอย่างกราฟค่าความถูกต้อง Training และ Validation ของตัวแบบ	75
รูปที่ 3.17 ตัวอย่างกราฟค่าการสูญเสีย Training และ Validation ของตัวแบบ	76
รูปที่ 3.18 โครงสร้างโฟลเดอร์ของชุดข้อมูล eval_ds	80

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันโลกได้เผชิญกับสภาวะอากาศรุนแรงสุดขีด (Extreme Weather) จากการเปลี่ยนแปลงอุณหภูมิพื้นผิวโลกที่ร้อนขึ้นเร็วกว่าเดิม ซึ่งเป็นผลจากการกระทำของมนุษย์ในการแข่งขันด้านอุตสาหกรรมที่สูงขึ้น ก่อให้เกิดผลกระทบต่องานแวดล้อมในทางลบเพิ่มมากขึ้น ทั้งนี้ยังส่งผลกระทบต่อเนื้อทำให้เกิดสภาวะภัยพิบัติจากธรรมชาติที่มีความรุนแรงเพิ่มมากยิ่งขึ้น คาดการณ์ได้ยาก มีความถี่ในการอุบัติสูง สร้างความเสียหายอย่างมากต่อชีวิต ทรัพย์สิน และโครงสร้างพื้นฐาน เช่น ปรากฏการณ์เอลนีโญ ลานีญา กระแสน้ำอุ่นเปลี่ยนทิศทาง ภัยพิบัติทางธรรมชาติจากน้ำท่วมใหญ่ ฝนตกรุนแรงและต่อเนื่องยาวนาน การเกิดพายุที่มีความถี่สูง ไฟป่าที่รุกรามกินพื้นที่เป็นบริเวณกว้าง คลื่นความร้อน คลื่นความเย็น ความแห้งแล้งยาวนาน แผ่นดินไหว สึนามิ และ ดินโคลนถล่ม เป็นต้น

ในขณะเดียวกัน มนุษย์มีการพัฒนาด้านการเข้าถึงอินเทอร์เน็ตและใช้บริการข้อมูลในการสื่อสารผ่านแพลตฟอร์มต่างๆ มากยิ่งขึ้น โดยเฉพาะสื่อสังคมออนไลน์ (Social Network Service) เพื่อใช้ในการแสดงออกถึงอารมณ์ความรู้สึก ความคิด สถานะทางสังคม และสถานการณ์ที่ตนเองกำลังเผชิญอยู่รอบตัว ประกอบด้วยข้อมูลในรูปแบบข้อความตัวอักษร รูปภาพ และวิดีโอ ข้อมูลเหล่านี้จึงอาจจะเป็นประโยชน์ในการใช้แจ้งข้อมูลเหตุการณ์ เวลา สถานที่ การร้องขอความช่วยเหลือ แจ้งสถานะความปลอดภัย เมื่อเกิดเหตุการณ์ผิดปกติหรือมีอันตรายเกิดขึ้นกับตัวผู้ใช้งาน อีกทั้งยังเป็นเครื่องมือในการสื่อสารที่รวดเร็วและมีประสิทธิภาพระหว่างผู้ใช้งานกับครอบครัว กลุ่มเพื่อน หรือเครือข่ายสาธารณะที่ผู้ใช้งานเชื่อมต่ออยู่

จากข้อมูลรายงานเหตุการณ์ผ่านสื่อสังคมออนไลน์ที่กล่าวมาข้างต้น เป็นช่องทางหนึ่งที่เป็นประโยชน์ในการช่วยสนับสนุนข้อมูลเหตุการณ์ให้กับหน่วยงานที่เกี่ยวข้องในการตอบสนองต่อเหตุการณ์ภัยพิบัติได้ ทั้งในด้านการเตรียมความพร้อมของทรัพยากรให้รวดเร็ว เหมาะสม และครอบคลุมต่อสถานการณ์ ทั้งนี้การประยุกต์และเลือกใช้ตัวแบบการเรียนรู้เชิงลึก (Deep Learning) ที่มีประสิทธิภาพดี และมีระยะเวลาในการประมวลผลที่รวดเร็ว ร่วมกับวิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) จากตัวแบบที่ผ่านการฝึกมาล่วงหน้า (Pre-trained Model) จะช่วยให้ระบบสามารถจำแนกข้อมูลรูปภาพจากภัยพิบัติได้มีประสิทธิภาพสูงขึ้นและรวดเร็วยิ่งขึ้น ซึ่งเป็นโอกาสที่ดีในการนำไปใช้เป็นข้อมูลเพื่อประกอบการตัดสินใจให้กับหน่วยงานที่เกี่ยวข้องต่อไป

1.2 วัตถุประสงค์ของงานวิจัย

1. เพื่อศึกษาวิธีที่เหมาะสมในการจัดการและการจำแนกข้อมูลรูปภาพภัยพิบัติ
2. เพื่อเปรียบเทียบประสิทธิภาพของตัวแบบการเรียนรู้เชิงลึก (Deep Learning) ด้วยวิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) ในแบบต่างๆ
3. เพื่อปรับปรุงตัวแบบให้มีประสิทธิภาพดียิ่งขึ้น

1.3 ขอบเขตของงานวิจัย

1. ทำการทดสอบประสิทธิภาพการจำแนกชนิดของรูปภาพภัยพิบัติด้วยตัวแบบการเรียนรู้เชิงลึก (Deep Learning) ด้วยวิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) จำนวน 3 ตัวแบบ ประกอบด้วย EfficientNet Inception และ MobileNet
2. ใช้ข้อมูลรูปภาพชุดทดลองจาก Crisis Vision Benchmark (CVB) และ Aerial Image Database for Emergency Response Applications (AIDER) ซึ่งเป็นข้อมูลที่ได้มาจากอินเทอร์เน็ตและสื่อสังคมออนไลน์ที่มีการระบุคำตอบของข้อมูลไว้แล้ว (Labeled Datasets) โดยแบ่งเป็นชนิดข้อมูลรูปภาพเป็น 4 กลุ่มประกอบด้วย รูปภาพความเสียหายของสิ่งปลูกสร้างจำนวน 1,929 รูป รูปภาพความเสียหายจากไฟไหม้ 870 รูป รูปภาพความเสียหายจากน้ำท่วม 911 รูป และรูปภาพสถานการณ์ที่ไม่ได้รับความเสียหายจำนวน 4,390 รูป เนื่องจากชุดข้อมูลไม่สมดุล (Imbalanced Datasets) จึงทำการปรับชุดข้อมูลให้มีสมดุล (Balanced Datasets) โดยทำการสุ่มเลือกข้อมูลรูปภาพจากชุดข้อมูล CVB กลุ่มละ 349 รูป และสุ่มเลือกข้อมูลรูปภาพจากชุดข้อมูล AIDER กลุ่มละ 500 รูป เพื่อใช้เป็นตัวแทนข้อมูลจากแต่ละกลุ่ม

ตารางที่ 1.1 ตารางแสดงจำนวนข้อมูลรูปภาพจาก CVB และ AIDER

ประเภทความเสียหาย	จำนวนรูป CVB	จำนวนรูป AIDER
สิ่งปลูกสร้าง	1,418 รูป	511 รูป
ไฟไหม้	349 รูป	521 รูป
น้ำท่วม	385 รูป	526 รูป
ไม่มีความเสียหาย	2,972 รูป	4,390 รูป

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้องค์ความรู้เพื่อเลือกใช้วิธีที่เหมาะสมในการจัดการและการจำแนกข้อมูลรูปภาพภัยพิบัติ
2. ได้ข้อมูลการเปรียบเทียบประสิทธิภาพของตัวแบบการเรียนรู้เชิงลึก (Deep Learning) ด้วยวิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) ที่ได้รับความนิยม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. ได้วิธีการเพื่อช่วยปรับปรุงตัวแบบให้มีประสิทธิภาพดียิ่งขึ้น

1.5 ขั้นตอนการดำเนินงาน

1. ศึกษางานวิจัยที่เกี่ยวข้อง
2. ศึกษาวิธีการจำแนกข้อมูลรูปภาพวิธีการเรียนรู้เชิงลึก (Deep Learning)
3. ศึกษาวิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) และการนำมาประยุกต์ใช้
4. ศึกษาและเลือกตัวแบบที่ต้องการนำมาใช้
5. เก็บรวบรวมข้อมูลรูปภาพภัยพิบัติ
6. ทำการทดลองปรับปรุงข้อมูลรูปภาพภัยพิบัติ
7. ทำการทดลองจำแนกข้อมูลด้วยวิธีการเรียนรู้เชิงลึก (Deep Learning) ร่วมกับวิธีการถ่ายโอนการเรียนรู้ (Transfer Learning)
8. ประเมินผลการทดลองและปรับปรุงแก้ไขตัวแบบให้มีประสิทธิภาพที่ดีขึ้น
9. วิเคราะห์และสรุปผลการทดลอง พร้อมให้ข้อเสนอแนะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

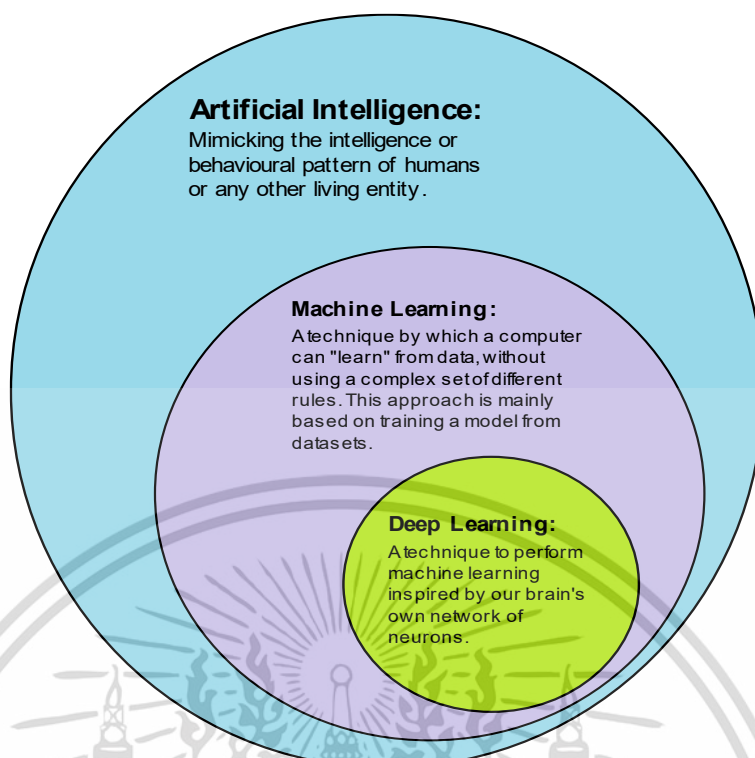
บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องซึ่งประกอบไปด้วยการเรียนรู้เชิงลึก (Deep Learning) โครงข่ายประสาทเทียมการเรียนรู้เชิงลึก (Deep Learning Neural Network) ชนิดของตัวปรับปรุงประสิทธิภาพ (Optimizer) การแก้ปัญหาอันเดอร์ฟิตติ้งและโอเวอร์ฟิตติ้ง (Underfitting and Overfitting Resolving Technique) โครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (Convolutional Neural Network - CNN) การสูญเสียเอนโทรปีข้ามหมวดหมู่ (Categorical Cross-Entropy Loss) การถ่ายโอนการเรียนรู้ (Transfer Learning) สถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 การประเมินประสิทธิภาพตัวแบบ (Performance Metrics) และงานวิจัยที่เกี่ยวข้อง

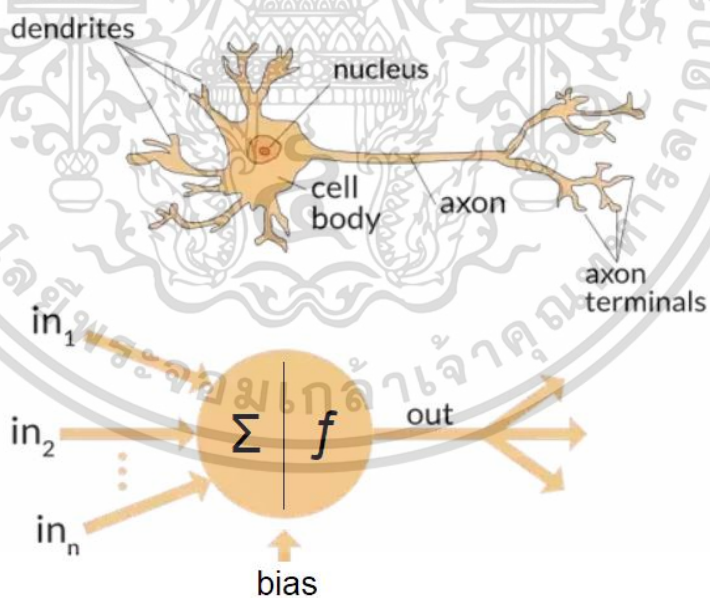
2.1 การเรียนรู้เชิงลึก (Deep Learning)

การเรียนรู้เชิงลึก (Deep Learning) เป็นส่วนหนึ่งของการเรียนรู้ของเครื่อง (Machine Learning) ซึ่งเน้นการฝึกโครงข่ายประสาทเทียม (Artificial Neural Network – ANN) แบบหลายชั้น (Multiple-Layers) มีโครงสร้างและการทำงานเลียนแบบสมองของมนุษย์ โดยใช้เซลล์ประสาทเทียม (Artificial Neuron) หลายชั้นที่เชื่อมต่อกัน เพื่อประมวลผลระหว่างข้อมูลเข้า (Input Data) ค่าน้ำหนัก (Weight) และค่าความเอนเอียง (Bias) โดยทำการเรียนรู้ สกัดรูปแบบที่ซับซ้อน มีความสามารถในการแปลผลและอธิบายข้อมูลที่ซับซ้อนได้ ปัจจุบันการเรียนรู้เชิงลึกเป็นที่นิยมในการใช้ทำนายผลของชุดข้อมูลขนาดใหญ่ จากประสิทธิภาพในการค้นหาคุณลักษณะที่ซับซ้อนของข้อมูลได้โดยอัตโนมัติและมีความถูกต้องสูง



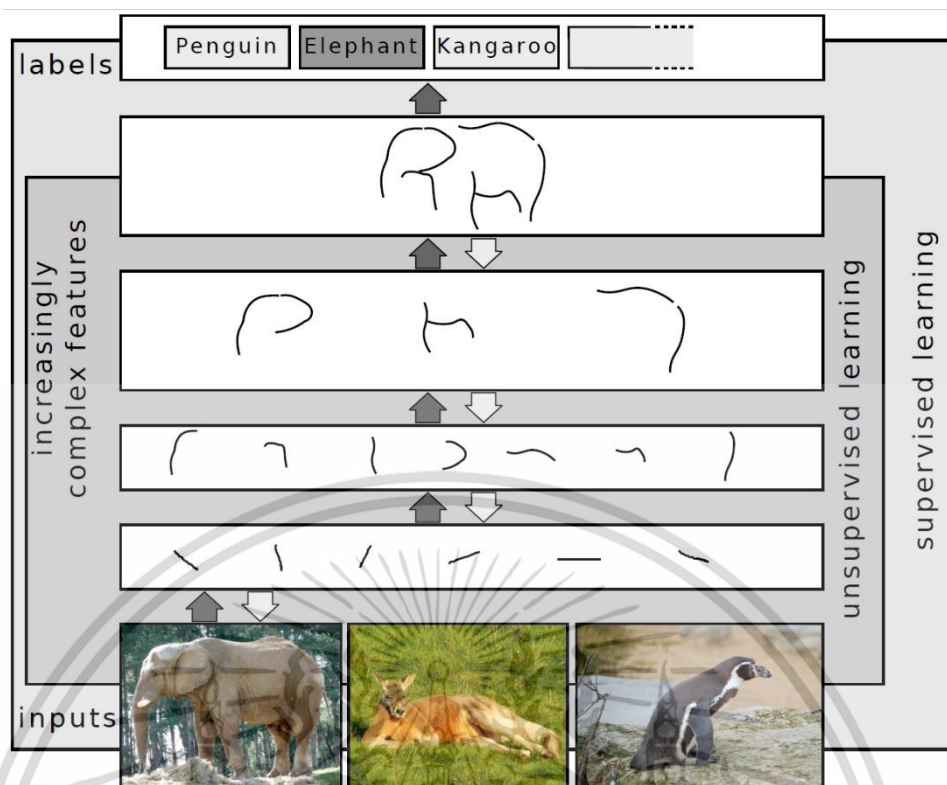
รูปที่ 2.1 ความสัมพันธ์ระหว่าง Deep Learning Machine Learning และ Artificial Intelligence

[1]



รูปที่ 2.2 การทำงานของเซลล์ประสาทมนุษย์และเซลล์ประสาทเทียม [2]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



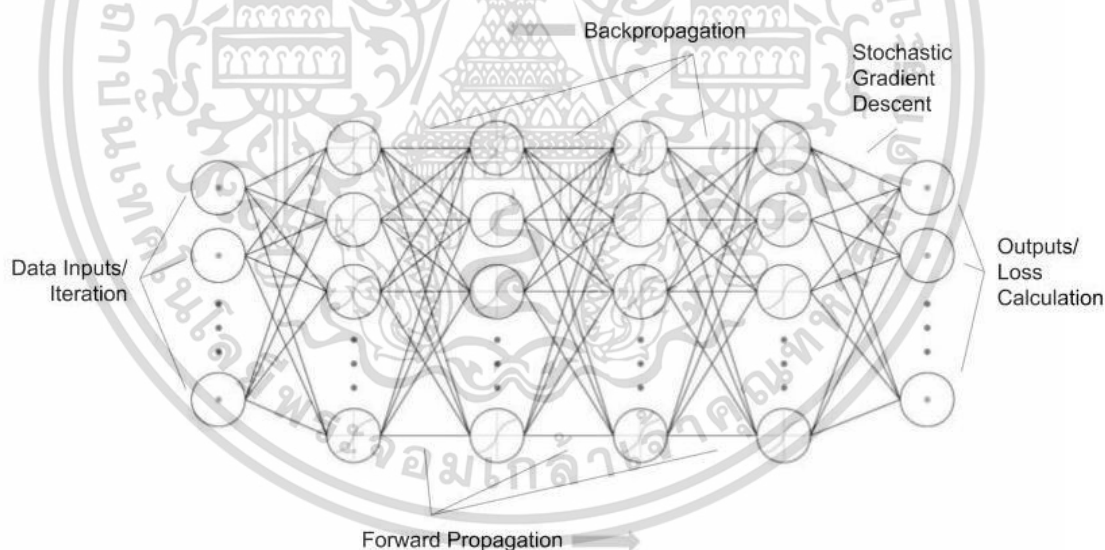
รูปที่ 2.3 การทำงานของ Deep Learning กับข้อมูลรูปภาพ [1]

โครงข่ายประสาทเทียมการเรียนรู้เชิงลึก (Deep Learning Neural Network) ประกอบด้วย โหนด (Node) ที่เชื่อมต่อกันหลายชั้น โดยแต่ละชั้นถูกสร้างจากชั้นก่อนหน้า มีวัตถุประสงค์เพื่อทำการปรับแต่ง (Refine) และช่วยเพิ่มประสิทธิภาพ (Optimization) ในการทำนายหรือการจำแนก มีการประมวลผลข้อมูลไปข้างหน้าผ่านโครงข่าย ซึ่งเรียกว่าการแบบแพร่กระจายไปข้างหน้า (Forward Propagation) โครงข่ายประสาทเทียมการเรียนรู้เชิงลึกมีโครงสร้างประกอบด้วย 3 ชั้นหลัก เริ่มจาก ชั้นป้อนข้อมูลเข้า (Input Layer) ชั้นกลางหรือชั้นซ่อน (Hidden Layer) และชั้นแสดงผลลัพธ์ (Output Layer)

การทำงานของตัวแบบ เริ่มต้นจากขั้นตอนการฝึก (Training) โดยทำการป้อนชุดข้อมูลที่มีคำตอบอยู่แล้วสำหรับการฝึก (Labeled Training Datasets) ที่ชั้นป้อนข้อมูลเข้า ข้อมูลจะถูกย่อยให้มีขนาดเล็กส่งผ่านไปยังแต่ละโหนดเพื่อใช้ในการประมวลผลในชั้นต่อไป ต่อมาที่ชั้นซ่อน ตัวแบบจะเริ่มค้นหาจุดเด่นของข้อมูล (Pattern Recognition) และสุดท้ายที่ชั้นแสดงผลลัพธ์ จะทำการรวบรวมผลที่ได้จากชั้นก่อนหน้าแล้วแสดงผลลัพธ์ซึ่งเป็นผลจากการทำนายหรือการจำแนกที่เกิดขึ้นจากตัวแบบ ผลลัพธ์ที่ได้จากการประมวลผลข้อมูลด้วยวิธีการแพร่กระจายไปข้างหน้าในรอบแรก (1st Epoch) มักจะมีความถูกต้องต่ำ (Low Accuracy) และมีค่าการสูญเสียสูง (High Loss) โครงข่ายประสาทเทียมการเรียนรู้เชิงลึกจึงได้มีการประยุกต์ใช้วิธีการแพร่กระจายแบบย้อนกลับ (Backpropagation) เพิ่มเข้ามา เช่น อัลกอริทึมเกรเดียนต์เดสเซนท์ (Gradient Descent)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Algorithm) ในการคำนวณค่าความผิดพลาดที่เกิดขึ้นจากการทำนายผล แล้วทำการปรับแก้ค่าน้ำหนักและค่าความเอนเองของแต่ละนิวรอน (Neuron) และส่งค่าพารามิเตอร์ (Parameters) ที่ได้ถูกปรับปรุงแล้วย้อนกลับไปยังโครงข่ายเพื่อลดค่าความผิดพลาดให้ต่ำที่สุด (Lowest Error) ตัวแบบจะทำการประมวลผลทั้งการแพร่กระจายไปข้างหน้าและการแพร่กระจายแบบย้อนกลับอย่างต่อเนื่องจนกระทั่งครบตามรอบการประมวลผลที่ผู้ใช้งานกำหนดไว้ในช่วงเริ่มต้นการฝึก ด้วยวิธีนี้โครงข่ายประสาทเทียมการเรียนรู้เชิงลึกจะค่อยๆทำการปรับปรุงประสิทธิภาพการเรียนรู้ในการทำนายผลให้มีความถูกต้องสูงขึ้น เมื่อเสร็จจากขั้นตอนการฝึกแล้ว ตัวแบบจะเข้าสู่ขั้นตอนการประเมินผล (Model Evaluation) โดยป้อนชุดข้อมูลสำหรับการประเมินผล (Test Dataset) ซึ่งเป็นข้อมูลที่ถูกแยกไว้โดยตัวแบบจะไม่เคยเห็นข้อมูลชุดนี้มาก่อน เพื่อใช้ประเมินประสิทธิภาพการเรียนรู้ของตัวแบบ ปัจจุบันการเรียนรู้เชิงลึกเป็นที่นิยมนำมาประยุกต์ใช้งานกับข้อมูลที่มีความซับซ้อนสูง เช่น การจำแนกภาพและวัตถุ (Image and Object Recognition) การเรียนรู้ภาษาธรรมชาติ (Natural Language Processing – NLP) การจำแนกและสังเคราะห์เสียงพูด (Speech Recognition) รถยนต์ขับเคลื่อนอัตโนมัติ (Autonomous Vehicles) การวินิจฉัยโรคเบื้องต้น (Healthcare Diagnostic) ระบบให้คำแนะนำ (Recommender Systems) และการตรวจจับความผิดปกติ (Fraud Detection) เป็นต้น



รูปที่ 2.4 การทำงานโครงข่ายประสาทเทียมการเรียนรู้เชิงลึก [3]

2.2 อัลกอริทึมเกรเดียนต์เดสเซนท์ (Gradient Descent Algorithm)

เป็นอัลกอริทึมการเพิ่มประสิทธิภาพ (Optimization) ที่ใช้กันทั่วไปในการเรียนรู้ของเครื่องและการเรียนรู้เชิงลึกเพื่อลดการสูญเสียของฟังก์ชันการสูญเสีย (Loss Function) ของตัวแบบ อัลกอริทึมนี้ทำงานแบบวนซ้ำเพื่อปรับพารามิเตอร์ (Parameters) ของตัวแบบในทิศทางที่ลาดชันเอียงลง เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่สุดของฟังก์ชันการสูญเสีย เป้าหมายของการทำเกรเดียนต์เดสเซนท์ (Gradient Descent) คือการ ค้นหาชุดพารามิเตอร์ของตัวแบบที่ช่วยลดความแตกต่างระหว่างผลลัพธ์จากการทำนายของตัวแบบ เปรียบเทียบกับผลลัพธ์จริง เป็นการพยายามหาจุดที่เหมาะสมที่สุด (Best Fit) ของตัวแบบกับข้อมูลที่ กำหนดให้

เกรเดียนต์เดสเซนท์ทำการหาค่าของฟังก์ชันการสูญเสีย (Loss Function) หรือมุมเทตา (θ) ที่ต่ำที่สุด อธิบายได้ดังสมการต่อไปนี้

$$\theta^1 = \theta^0 - \alpha \nabla J(\theta)$$

เมื่อ θ^1 คือ ตำแหน่งถัดไป θ^0 คือ ตำแหน่งปัจจุบัน (กำหนดค่าสุ่ม) – คือ ทิศทางตรงกันข้าม α คือ อัตราการเรียนรู้ ∇ คือ เกรเดียนต์ และ $J(\theta)$ คือ ฟังก์ชันการสูญเสีย โดยเริ่มการคำนวณที่ θ^0

ค่าความผิดพลาด

$$Error = Y' - Y$$

เมื่อ Y' คือ ค่าทำนายที่ได้จากตัวแบบ และ Y คือ ค่าจริง

ฟังก์ชันการสูญเสียของข้อมูล N ตัว [4]

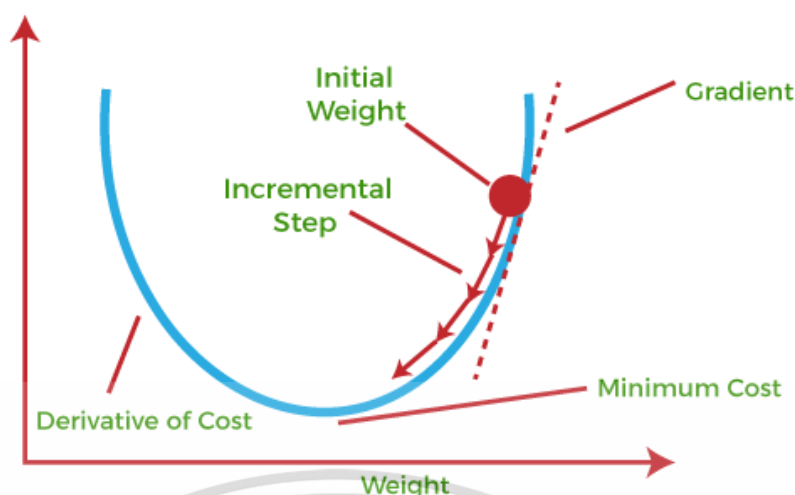
$$Loss Function = \frac{1}{N} \sum_{i=1}^N (Y' - Y)^2$$

จะได้

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N (Error_i)^2$$

เมื่อ N คือ จำนวนข้อมูล (Data Points)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.5 การทำงานของ Gradient Descent [5]

2.2.1 สรุปขั้นตอนการทำงานของเกรเดียนต์เดสเซนส์ที่ได้ดังนี้

2.2.1.1 ขั้นเริ่มต้น กำหนดค่าแบบสุ่มให้กับพารามิเตอร์ของตัวแบบทั้งค่าน้ำหนักและค่าความเอนเอียง

2.2.1.2 ขั้นการคำนวณการสูญเสีย ตัวแบบทำการทำนายด้วยข้อมูลการฝึก และฟังก์ชันการสูญเสียทำการคำนวณเพื่อเปรียบเทียบค่าความผิดพลาดระหว่างผลลัพธ์ทำนายและผลลัพธ์จริง

2.2.1.3 ขั้นการคำนวณเกรเดียนต์ คำนวณเกรเดียนต์ของฟังก์ชันการสูญเสียที่เกี่ยวข้องกับพารามิเตอร์ของตัวแบบ ค่าของเกรเดียนต์แสดงถึงอัตราการเปลี่ยนแปลงของฟังก์ชันการสูญเสียเกี่ยวกับพารามิเตอร์แต่ละตัว

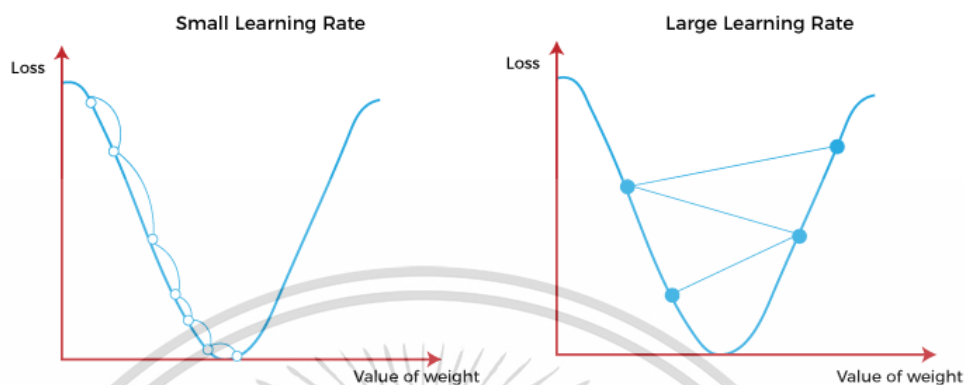
2.2.1.4 ขั้นการปรับปรุงพารามิเตอร์ ปรับปรุงพารามิเตอร์ด้วยขั้นเล็กๆ ในทิศทางตรงกันข้ามกับเกรเดียนต์ โดยขนาดของขั้นเรียกว่าอัตราการเรียนรู้ (Learning Rate) ซึ่งใช้ควบคุมขนาดในการปรับปรุงพารามิเตอร์

2.2.1.5 ขั้นการวนซ้ำ ทำซ้ำขั้นตอนที่ 2.2.1.2 ถึง 2.2.1.4 ซ้ำๆ จนกว่าจะถึงเงื่อนไขการหยุด เงื่อนไขนี้สามารถเป็นจำนวนสูงสุดของการวนซ้ำ การบรรลุระดับความแม่นยำที่ต้องการ หรือถึงเกณฑ์การปรับปรุงที่กำหนดไว้ล่วงหน้าในฟังก์ชันการสูญเสีย

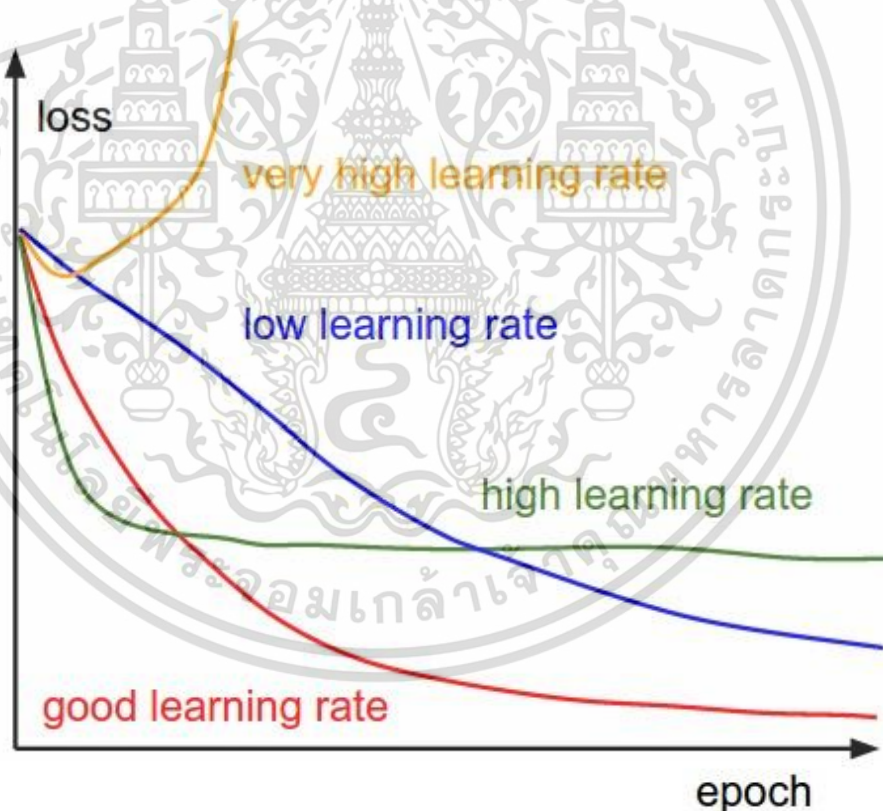
จากการปรับปรุงพารามิเตอร์ตามค่าความลาดเอียงที่เปลี่ยนไปเรื่อยๆ ค่าของเกรเดียนต์เดสเซนส์จะค่อยๆ เข้าสู่ค่าของชุดพารามิเตอร์ที่ทำให้ฟังก์ชันการสูญเสียมีค่าการสูญเสียต่ำที่สุด (Error เข้าใกล้ศูนย์) โดยอัตราการเรียนรู้ (Learning Rate) จะเป็นตัวกำหนดขนาด (Step Size) ในขั้นตอนของการวนซ้ำแต่ละครั้ง ผู้ใช้งานจำเป็นต้องเลือกจำนวนรอบการฝึก (Epoch) และอัตราการเรียนรู้ (Learning Rate) ที่เหมาะสมเพื่อให้ตัวแบบเกิดการลู่เข้าสู่ค่าต่ำสุด การเลือกจำนวนรอบที่ต่ำเกินไปอาจทำให้การฝึกไม่สมบูรณ์ หรือการเลือกจำนวนรอบที่สูงเกินไปอาจทำให้เกิดการโอเวอร์ฟิต (Overfitting) ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เกินไปและอัตราการเรียนรู้ที่สูงเกินไปอาจจะทำให้ตัวแบบค้นหาค่าพารามิเตอร์ที่ต่ำที่สุดของฟังก์ชันการสูญเสียไม่ได้ ซึ่งเกิดจากการที่ตัวแบบก้าวข้ามจุดต่ำสุดในท้องถิ่น (Local Minima) ไปแล้ว



รูปที่ 2.6 ความแตกต่างของ Learning Rate [5]



รูปที่ 2.7 ผลกระทบต่อการสูญเสียที่ค่า Learning Rate ต่างกัน [6]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.2 ประเภทของเกรเดียนต์เดสเซนท์

สามารถแบ่งออกได้เป็น Batch Gradient Descent Stochastic Gradient Descent และ Mini-Batch Gradient Descent

2.2.2.1 เกรเดียนต์เดสเซนท์แบบแบทช์ (Batch Gradient Descent - BGD)

ใช้เพื่อค้นหาข้อผิดพลาดสำหรับแต่ละจุดในชุดข้อมูลการฝึกและปรับปรุงพารามิเตอร์หลังจากทำการประเมินตัวอย่างการฝึกทั้งหมด เป็นวิธีการแบบโลภ (Greedy Approach) ที่ต้องรวมผลของตัวอย่างทั้งหมดสำหรับการปรับปรุงพารามิเตอร์ในแต่ละครั้ง

ข้อดีของเกรเดียนต์เดสเซนท์แบบแบทช์

- สร้างสัญญาณรบกวนน้อยกว่าเมื่อเปรียบเทียบกับวิธีเกรเดียนต์เดสเซนท์แบบอื่น
- เกรเดียนต์เดสเซนท์มีการลู่เข้าที่เสถียร
- มีประสิทธิภาพในการคำนวณเนื่องจากใช้ทรัพยากรทั้งหมดสำหรับข้อมูลตัวอย่างการฝึกทั้งหมด

2.2.2.2 เกรเดียนต์เดสเซนท์แบบสุ่ม (Stochastic Gradient Descent - SGD)

ใช้หนึ่งตัวอย่างการฝึกต่อการรอบการวนซ้ำ โดยระบบจะประมวลผลช่วงเวลาการฝึกสำหรับแต่ละตัวอย่างภายในชุดข้อมูล และปรับปรุงพารามิเตอร์ของตัวอย่างการฝึกทีละรายการ เนื่องจากต้องใช้เพียงหนึ่งตัวอย่างการฝึกในแต่ละครั้งและเป็นการง่ายที่จะจัดเก็บค่าไว้ในหน่วยความจำที่จัดสรรไว้ แต่ต้องสูญเสียประสิทธิภาพการคำนวณบางส่วนเมื่อเปรียบเทียบกับเกรเดียนต์เดสเซนท์แบบแบทช์ เนื่องจากมีการปรับปรุงค่าพารามิเตอร์บ่อยครั้ง ซึ่งต้องการรายละเอียดและความเร็วที่มากกว่า และจากการปรับปรุงค่าพารามิเตอร์บ่อยครั้งยังทำให้เกิดสัญญาณรบกวนเกรเดียนต์ แต่บางครั้ง SGD ก็มีประโยชน์สำหรับใช้ค้นหาจุดต่ำสุดสัมบูรณ์ (Global Minimum) และหลีกเลี่ยงจุดต่ำสุดสัมพัทธ์ (Local Minima)

ข้อดีของเกรเดียนต์เดสเซนท์แบบสุ่ม

- ง่ายต่อการจัดสรรหน่วยความจำที่ต้องการ
- การคำนวณค่อนข้างเร็วกว่าเกรเดียนต์เดสเซนท์แบบแบทช์

- มีประสิทธิภาพมากกว่าสำหรับชุดข้อมูลขนาดใหญ่

2.2.2.3 เกรเดียนต์เดสเซนท์แบบมินิแบทช์ (MiniBatch Gradient Descent)

คือ การรวมกันของทั้งเกรเดียนต์เดสเซนท์แบบแบทช์และเกรเดียนต์เดสเซนท์แบบสุ่ม โดยจะแบ่งชุดข้อมูลการฝึกออกเป็นชุดย่อยๆ จากนั้นทำการปรับปรุงพารามิเตอร์ของชุดข้อมูลเหล่านั้นแยกจาก การแยกชุดข้อมูลการฝึกออกเป็นกลุ่มเล็กๆ เป็นการสร้างสมดุลเพื่อรักษาประสิทธิภาพในการคำนวณของเกรเดียนต์เดสเซนท์แบบแบทช์และรักษาความเร็วของเกรเดียนต์เดสเซนท์แบบสุ่ม ทำให้มีประสิทธิภาพการคำนวณที่สูงขึ้นและมีสัญญาณรบกวนน้อยลง

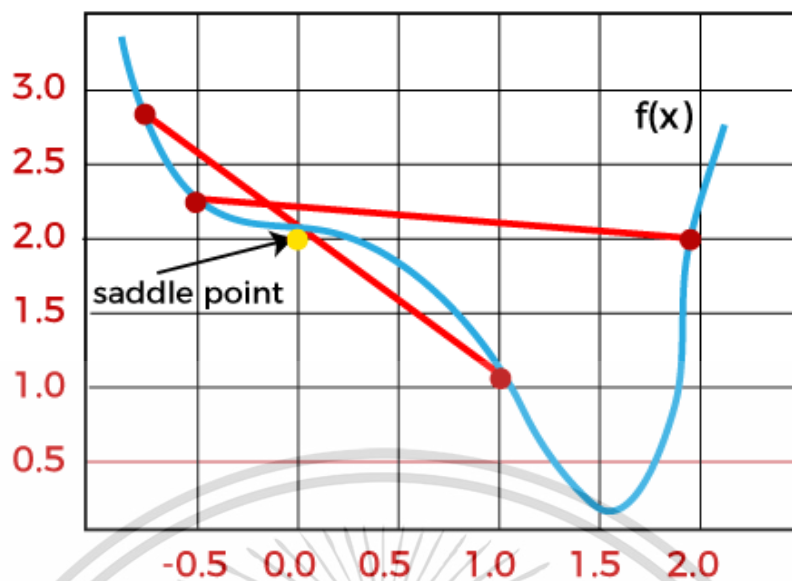
ข้อดีของเกรเดียนต์เดสเซนท์แบบมินิแบทช์

- ง่ายต่อการจัดการหน่วยความจำที่ต้องการให้เหมาะสม
- มีประสิทธิภาพในการคำนวณที่ดี
- เกรเดียนต์เดสเซนท์มีการลู่อื่นที่เสถียร

2.2.3 ปัญหาของเกรเดียนต์เดสเซนท์

2.2.3.1 จุดต่ำสุดสัมพัทธ์ (Local Minima) และจุดอานม้า (Saddle Point)

เกรเดียนต์เดสเซนท์สามารถหาค่าต่ำสุดสัมบูรณ์ (Global Minimum) ได้ง่ายกว่าสำหรับปัญหาแบบจุดโค้งเว้าเดียว (Convex) แต่ในบางครั้งการหาค่าต่ำสุดสัมบูรณ์ (Global Minimum) ที่ทำให้ตัวแบบการเรียนรู้เครื่องจักรมีผลลัพธ์ที่ดีที่สุด อาจจะทำได้ยากกว่าสำหรับปัญหาแบบหลายจุดโค้งเว้า (Non Convex) ตัวแบบจะหยุดการเรียนรู้เพิ่มเติมเมื่อความชันของฟังก์ชันการสูญเสียมีค่าเข้าใกล้หรือกับศูนย์ นอกจากการหยุดเรียนรู้ที่จุดที่มีค่าต่ำสุดสัมบูรณ์ (Global Minimum) แล้ว อาจจะมีบางกรณีที่ค่าความชันเข้าใกล้ศูนย์ได้ที่จุดอานม้า (Saddle Point) ซึ่งเป็นจุดที่มีค่าต่ำสุดสัมพัทธ์ (Local Minima) เมื่อดูจากกราฟฟังก์ชันการสูญเสียจะสังเกตได้ว่าจุดที่มีค่าต่ำสุดสัมพัทธ์ (Local Minima) มีรูปร่างคล้ายกับจุดที่มีค่าต่ำสุดสัมบูรณ์ (Global Minimum) ซึ่งให้ค่าความชันเข้าใกล้ศูนย์เช่นเดียวกัน แต่ที่จุดที่มีค่าต่ำสุดสัมบูรณ์ (Global Minimum) จะมีค่าความชันของฟังก์ชันการสูญเสียเพิ่มขึ้นทั้งสองด้านจากจุดนั้น



รูปที่ 2.8 จุดต่ำสุดสัมพัทธ์ (Local Minima) และจุดอานม้า (Saddle Point) [5]

2.2.3.2 การหายไปและการระเบิดของเกรเดียนต์ (Vanishing and Exploding Gradient)

ในการเรียนรู้ของโครงข่ายประสาทเทียมการเรียนรู้เชิงลึกใช้การฝึกตัวแบบด้วยวิธีเกรเดียนต์เดสเซนท์และการกระจายย้อนกลับ ซึ่งอาจทำให้เกิดปัญหาอีก 2 แบบคือ

- การหายไปของเกรเดียนต์ (Vanishing Gradient) เกิดขึ้นเมื่อเกรเดียนต์มีขนาดเล็กกว่าที่คาดไว้ในระหว่างการกระจายย้อนกลับ การที่เกรเดียนต์มีขนาดเล็กลงจากอนุพันธ์ของฟังก์ชันกระตุ้น (Activation Function) จะทำให้อัตราการเรียนรู้ของชั้นก่อนหน้ามีขนาดลดลงกว่าชั้นในลำดับถัดไปของโครงข่าย ซึ่งพารามิเตอร์น้ำหนักถูกปรับปรุงค่าจนไม่นัยสำคัญ สามารถแก้ไขได้โดยเปลี่ยนชนิดของฟังก์ชันกระตุ้นในโครงข่าย [7]
- การระเบิดของเกรเดียนต์ (Exploding Gradient) เกิดขึ้นเมื่อเกรเดียนต์มีขนาดใหญ่เกินไปและตัวแบบเกิดเสถียรภาพ โดยค่าน้ำหนักของตัวแบบมีค่าเพิ่มขึ้นและถูกแสดงเป็นค่า NaN สามารถแก้ไขได้โดยใช้เทคนิคการลดมิติ (Dimensionality Reduction Techniques) เข้ามาช่วยในการลดความซับซ้อนของตัวแบบ

2.3 ตัวประมาณค่าโมเมนต์ชนิดปรับปรุงตัวเองได้ (Adaptive Moment Estimator) หรือชื่อย่อ อัดมอฟติไมเซอร์ (Adam Optimizer)

เป็นอัลกอริธึมช่วยเพิ่มประสิทธิภาพ (Optimization) รูปแบบหนึ่งสำหรับวิธีเกรเดียนต์เดสเซนส์ชนิดสุ่ม (Stochastic Gradient Descent - SGD) ซึ่งทำการประมาณค่าโมเมนต์ (Moment) ลำดับที่หนึ่งและสองของเกรเดียนต์เพื่อปรับอัตราการเรียนรู้สำหรับแต่ละค่าน้ำหนักในโครงข่ายประสาทเทียม อัดม (Adam) ได้รับการยอมรับว่าเป็นตัวเพิ่มประสิทธิภาพสุ่ม (Stochastic Optimizer) ที่มีประสิทธิภาพมากที่สุด ซึ่งต้องการเพียงเกรเดียนต์ลำดับที่หนึ่งเท่านั้น โดยมีความต้องการใช้หน่วยความจำที่ต่ำมาก ทำให้มันมีประสิทธิภาพที่ดีกว่าในด้านการเรียนรู้ทั่วไป (Generalizing Performance) นอกจากนี้ อัดมมีไฮเปอร์พารามิเตอร์ที่เข้าใจได้ง่ายและต้องการการปรับแต่งน้อยลง อัดมทำงานแตกต่างจาก RMSProp ซึ่งทำการปรับอัตราการเรียนรู้ตามค่าเฉลี่ยของโมเมนต์เดียว แต่มีการใช้ค่าเฉลี่ยลำดับที่สองเกรเดียนต์เพิ่มเข้ามา โดยอัลกอริธึมนี้จะคำนวณค่าเฉลี่ยเคลื่อนที่แบบทวีคูณ (Exponential Moving Average) ของเกรเดียนต์และเกรเดียนต์กำลังสอง (Square Gradients) และใช้พารามิเตอร์ β_1 และ β_2 เพื่อควบคุมอัตราการสลายตัว (Decay) ของค่าเฉลี่ยเคลื่อนที่เหล่านี้ เป็นการผสมผสานระหว่างวิธีเกรเดียนต์เดสเซนส์ทั้งสองวิธีเข้าด้วยกัน ประกอบด้วย Momentum และ RMSProp [8]

2.3.1 โมเมนตัม (Momentum)

เป็นอัลกอริธึมการเพิ่มประสิทธิภาพที่นำค่าเฉลี่ยถ่วงน้ำหนักแบบทวีคูณ (Exponential Weighted Average) และการเร่งความเร็วเกรเดียนต์เดสเซนส์เข้ามา ซึ่งเป็นส่วนขยายของอัลกอริธึมการเพิ่มประสิทธิภาพเกรเดียนต์เดสเซนส์ อัลกอริธึมโมเมนตัมถูกอธิบายได้ด้วยสมการ 2 ส่วน ส่วนแรกคือการคำนวณการเปลี่ยนแปลงตำแหน่ง และส่วนที่สองคือการปรับตำแหน่งเก่า

ตำแหน่งที่มีการเปลี่ยนแปลงกำหนดโดย

$$update = \alpha * m_t$$

ตำแหน่งหรือค่าน้ำหนักใหม่ในช่วงเวลา t กำหนดโดย

$$w_t + 1 = w_t - update$$

จากสมการด้านบน α (step size) เป็นไฮเปอร์พารามิเตอร์ซึ่งควบคุมการเคลื่อนไหวกว้างในพื้นที่ย่อยค้นหา ซึ่งเรียกอีกอย่างว่าอัตราการเรียนรู้ (Learning Rate) และ $f'(x)$ คือฟังก์ชันอนุพันธ์หรือการรวมของเกรเดียนต์ที่เวลา t

เมื่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$m_t = \beta_1 * m_t + (1 - \beta_1) * \left(\frac{\delta L}{\delta w_t}\right)$$

โดย m_t และ $m_t - 1$ คือผลรวมของเกรเดียนต์ในช่วงเวลา t และ $t - 1$ ตามลำดับ

โมเมนตัมมีผลในการลดการเปลี่ยนแปลง (Dampening) ของเกรเดียนต์และขนาดของขั้น (Step Size) ตามลำดับสำหรับจุดใหม่แต่ละจุดในพื้นที่การค้นหา

2.3.2 ค่าเฉลี่ยกำลังสองของการแพร่กระจาย (Root Mean Square Propagation – RMSP)

เป็นอัลกอริทึมเพิ่มประสิทธิภาพแบบปรับได้ ซึ่งเป็นเวอร์ชันปรับปรุงของ [9] AdaGrad เพื่อจัดการปัญหาของโมเมนตัมและทำงานได้ดีในการตั้งค่าออนไลน์ (Online Settings) RMSP ใช้วิธีค่าเฉลี่ยทวีคูณ (Exponential Average) เขียนเป็นสมการได้ดังนี้

$$w_t + 1 = w_t - \left(\frac{\alpha_t}{\sqrt{v_t}} + e \right) * \left(\frac{\delta L}{\delta w_t} \right)$$

เมื่อ

$$v_t = \beta * v_t + (1 - \beta) * \left(\frac{\delta L}{\delta w_t} \right)^2$$

โดย m_t และ $m_t - 1$ คือผลรวมของเกรเดียนต์ในช่วงเวลา t และ $t - 1$ ตามลำดับ

w_t และ $w_t + 1$ คือค่าน้ำหนักในช่วงเวลา t และ $t + 1$ ตามลำดับ

α_t คือ อัตราการเรียนรู้ ซึ่งเป็นไฮเปอร์พารามิเตอร์

δL คือ อนุพันธ์ของฟังก์ชันการสูญเสีย

δw_t คือ อนุพันธ์ของน้ำหนักที่เวลา t

β คือ พารามิเตอร์เฉลี่ย (Average Parameter)

e คือ ค่าคงที่

จากการศึกษาพบว่าเครื่องมือเพิ่มประสิทธิภาพทั้ง Momentum และ RMSP มีปัญหาบางอย่าง เช่น ประสิทธิภาพการเรียนรู้โดยทั่วไป [10] กล่าวว่า Adam รับช่วงแอตทริบิวต์ต่อจากตัวเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพิ่มประสิทธิภาพสองตัวข้างต้นและนำไปปรับปรุงต่อเพื่อให้เกรเดียนต์เดสเซนท์มีประสิทธิภาพที่ดียิ่งขึ้น

อัลกอริธึมอดัม สามารถอธิบายได้โดยนำสมการที่ใช้ในเครื่องมือเพิ่มประสิทธิภาพสองตัวข้างต้น

$$m_t = \beta_1 * m_t + (1 - \beta_1) * \left(\frac{\delta L}{\delta w_t}\right)$$

และ

$$v_t = \beta_2 * v_t + (1 - \beta_2) * \left(\frac{\delta L}{\delta w_t}\right)^2$$

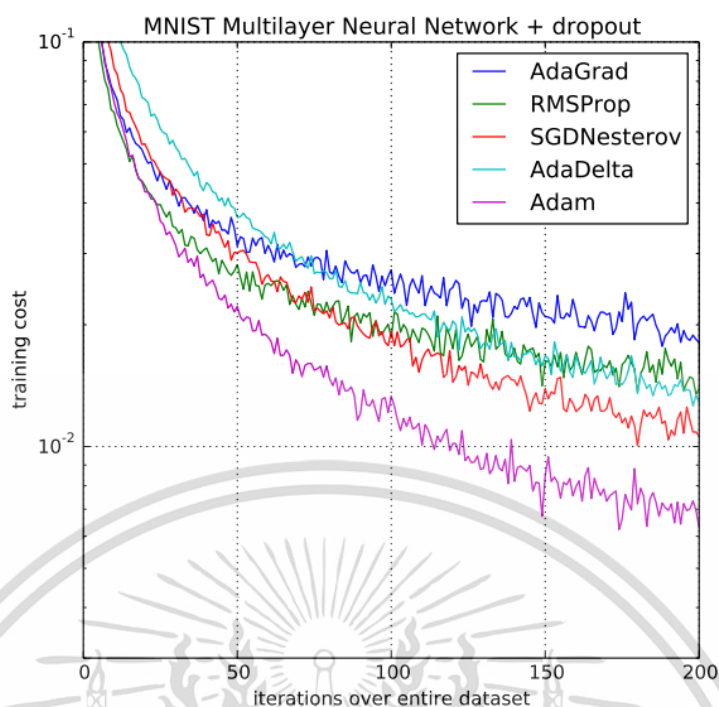
ในขั้นเริ่มต้น ทั้ง m_t และ v_t ถูกตั้งค่าเป็น 0 ทั้งคู่มีแนวโน้มที่จะมีค่าเอนเอียงออกไปจาก 0 เมื่อ β_1 และ β_2 มีค่าเท่ากับ 1 จากการคำนวณโดยทำการแก้ไขค่าความเอนเอียงให้ถูกต้อง (Bias-Corrected) $\hat{m}_t$ และ $\hat{v}_t$ ซึ่งปัญหานี้ถูกแก้ไขได้ด้วยเครื่องมือเพิ่มประสิทธิภาพอดัม โดยมีสมการดังนี้

$$\hat{m}_t = m_t \div (1 - \beta_1^t)$$

$$\hat{v}_t = v_t \div (1 - \beta_2^t)$$

ตัวแบบจะทำเกรเดียนต์เดสเซนท์ทุกรอบการวนซ้ำ จึงทำควบคุมการปรับค่าและความเอนเอียงไว้ได้ และเมื่อแทนที่พารามิเตอร์ใหม่ด้วยพารามิเตอร์เก่าจะได้สมการ

$$w_t = w_{(t-1)} - \alpha * \left(\frac{\hat{m}_t}{\sqrt{(\hat{v}_t)}} + e\right)$$



รูปที่ 2.9 การเปรียบเทียบตัวเพิ่มประสิทธิภาพที่ใช้สำหรับการฝึกเพื่อปรับปรุงประสิทธิภาพของโครงข่ายประสาทเทียมหลายชั้นด้วยข้อมูลภาพ MNIST [8]

2.4 ปัญหาโอเวอร์ฟิตติงและอันเดอร์ฟิตติง (Overfitting and Underfitting)

เป็นปัญหาพื้นฐานในการเรียนรู้ของเครื่องซึ่งอาจส่งผลเสียต่อประสิทธิภาพที่แท้จริงของตัวแบบ ทำให้ตัวแบบทำการทำนายข้อมูลใหม่ที่ไม่เคยเห็นมาก่อนได้ไม่มีประสิทธิภาพและเกิดความผิดพลาดสูง

2.4.1 ปัญหาโอเวอร์ฟิตติง (Overfitting)

เป็นปัญหาของตัวแบบที่มีค่าความผิดพลาดในการฝึกต่ำ (Low Training Error) และมีค่าความผิดพลาดในการประเมินผลหรือการทดสอบสูง (High Validation/Test Error) เกิดขึ้นเมื่อตัวแบบพยายามปรับการเรียนรู้ให้พอดีกับข้อมูลการฝึกมากเกินไป มีการเรียนรู้สัญญาณรบกวน (Noise) รูปแบบเฉพาะของข้อมูลฝึกมากเกินไป ส่งผลให้ตัวแบบมีซับซ้อนสูงและเริ่มจดจำตัวอย่างการฝึกแทนที่จะเรียนรู้จากรูปแบบโดยทั่วไป ตัวแบบจะทำงานได้ดีเฉพาะกับข้อมูลการฝึก แต่ไม่สามารถทำนายข้อมูลใหม่ที่ไม่เคยเห็นได้อย่างมีประสิทธิภาพ โดยมีสาเหตุได้ดังนี้

- ตัวแบบมีความซับซ้อนสูงเกินไป

- ตัวแบบมีพารามิเตอร์มากเกินไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- มีข้อมูลสำหรับใช้ฝึกไม่เพียงพอ
- ใช้จำนวนรอบการฝึกสูงเกินไป

สามารถแก้ไขได้โดยใช้เทคนิคดังนี้

- วิธีเรกูลาไรเซชันแบบ L1 หรือ L2 (L1 or L2 Regularization) เพิ่มการลงโทษ (Penalty) เพื่อลดความซับซ้อนของฟังก์ชันการสูญเสีย
- ใช้ข้อมูลการฝึกที่หลากหลายและครอบคลุมรูปแบบของข้อมูล
- ใช้ประเมินแบบครอสวาเลชัน (Cross-Validation) เพื่อประเมินประสิทธิภาพของตัวแบบด้วยข้อมูลย่อยหลายชุด
- ลดความซับซ้อนของตัวแบบโดยใช้สถาปัตยกรรมที่เรียบง่ายขึ้นหรือใช้เทคนิคในการลดขนาดมิติ (Dimensionality Reduction Techniques)

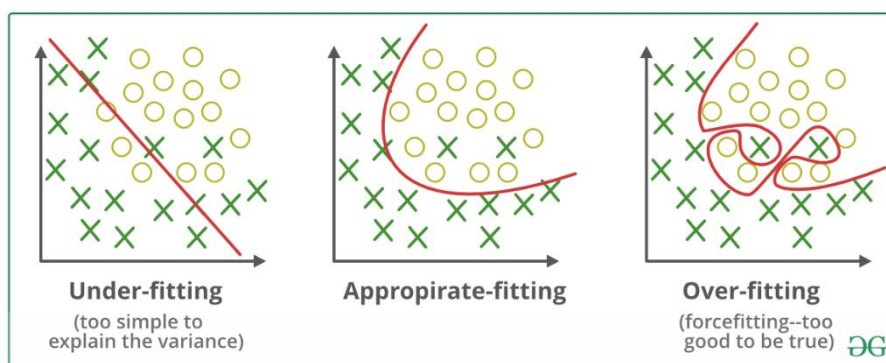
2.4.2 ปัญหาอันเดอร์ฟิตติง (Underfitting)

เป็นปัญหาของตัวแบบที่มีค่าความผิดพลาดจากการฝึกสูง (High Training Error) และมีค่าความผิดพลาดในการประเมินผลหรือการทดสอบสูง (High Validation/Test Error) ซึ่งอาจมีค่าเท่าหรือใกล้เคียงกัน เกิดขึ้นเมื่อตัวแบบเรียบง่ายเกินไปหรือขาดความสามารถในการตรวจจับรูปแบบพื้นฐานของข้อมูล ขาดความสามารถในการตรวจจับความซับซ้อนและความแตกต่างของปัญหา ส่งผลให้ตัวแบบมีประสิทธิภาพการทำงานที่ไม่ดีกับข้อมูลการฝึกและข้อมูลใหม่ โดยมีสาเหตุได้ดังนี้

- ตัวแบบมีความเรียบง่ายเกินไป
- มีพารามิเตอร์น้อยเกินไป
- ข้อมูลการฝึกไม่เพียงพอ หรือขาดความหลากหลายในข้อมูล
- ตัวแบบมีการฝึกไม่เพียงพอ หรือ ไม่เกิดการลู่เข้า (Convergence)

สามารถแก้ไขได้โดยใช้เทคนิคดังนี้

- เพิ่มความซับซ้อนของตัวแบบโดยการเพิ่มชั้น (Layer) เซลล์ประสาท (Neuron) หรือพารามิเตอร์ (Parameter)
- ใช้ข้อมูลการฝึกที่หลากหลายและครอบคลุมรูปแบบของข้อมูล
- เพิ่มจำนวนรอบการฝึก
- เลือกใช้ตัวแบบที่ซับซ้อนมากขึ้น หรือเลือกใช้อัลกอริทึมที่แตกต่างไป



รูปที่ 2.10 ตัวอย่างของปัญหาอันเดอร์ฟิตติงและโอเวอร์ฟิตติง [11]

2.4.3 เทคนิคการแก้ปัญหาโอเวอร์ฟิตติงสำหรับโครงข่ายประสาทเทียมการเรียนรู้เชิงลึก

2.4.3.1 เรกูลาร์ไรเซชัน (Regularization)

เป็นเทคนิคที่ใช้เพื่อปรับปรุงความสามารถทั่วไปของโครงข่ายประสาทเทียมการเรียนรู้เชิงลึกและแก้ปัญหาโอเวอร์ฟิตติง (Overfitting) เมื่อโครงข่ายประสาทเทียมทำงานได้ดีกับข้อมูลการฝึก แต่ไม่สามารถทำนายเป็นข้อมูลใหม่ที่ไม่เคยเห็นได้ดี โดยสามารถเลือกใช้เทคนิคในการแก้ปัญหาด้วยเรกูลาร์ไรเซชัน L1 และ L2 (Regularization L1 and L2) โดยทำการเพิ่มเทอมเรกูลาร์ไรเซชัน (Regularization Term) ให้กับฟังก์ชันการสูญเสียในระหว่างการฝึก เรกูลาร์ไรเซชัน L1 จะทำการเพิ่มเทอมค่าสัมบูรณ์ของน้ำหนักให้กับฟังก์ชันการสูญเสีย ในขณะที่เรกูลาร์ไรเซชัน L2 หรือการปรับลดน้ำหนัก (Weight Decay) จะทำการเพิ่มเทอมของน้ำหนักกำลังสอง ทั้ง L1 และ L2 เป็นการเพิ่มเทอมการลงโทษเพื่อลดค่าน้ำหนักที่มีขนาดใหญ่ลงเพื่อช่วยป้องกันการเกิดปัญหาโอเวอร์ฟิตติง

เรกูลาร์ไรเซชัน L1 หรือการถดถอยแบบลาสโซ (Lasso Regression – Least Absolute Shrinkage and Selection Operator) สามารถแสดงสมการได้ดังนี้

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^N |w_i|$$

โดย λ คือ ไฮเปอร์พารามิเตอร์สำหรับการปรับแต่ง และ $\sum_{i=1}^N |w_i|$ คือ เทอมการลงโทษ L1

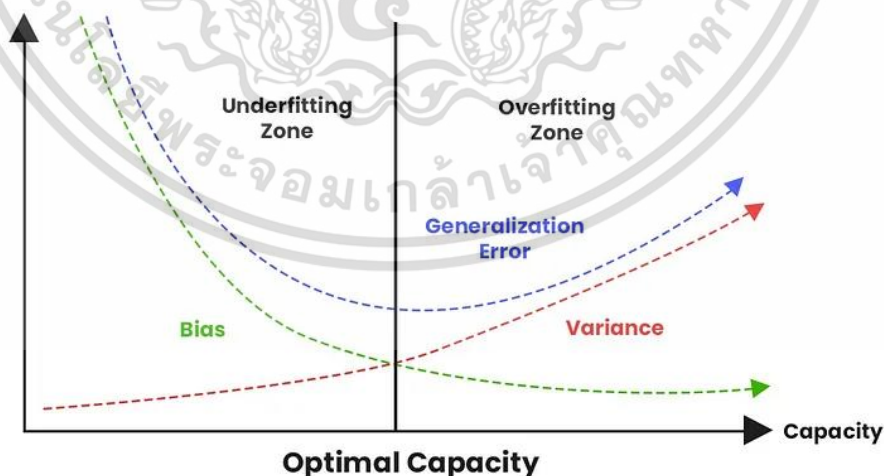
เรกูลาร์ไรเซชัน L2 หรือการถดถอยแบบบริดจ์ (Ridge Regression) สามารถแสดงเป็นสมการได้ดังนี้

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^N w_i^2$$

โดย $\sum_{i=1}^N w_i^2$ คือ เทอมการลงโทษ L2

ตารางที่ 2.1 ตารางเปรียบเทียบเพื่อในการเลือกใช้เรกูลาร์ไรเซชัน L1 และ L2 [12]

เรกูลาร์ไรเซชัน L1	เรกูลาร์ไรเซชัน L2
ผลรวมค่าสัมบูรณ์ของน้ำหนัก	ผลรวมของค่าน้ำหนักกำลังสอง
ได้คำตอบแบบสปาร์ส (Sparse)	ได้คำตอบที่แยกออกมา (Non-sparse)
มีความเป็นไปได้ที่จะมีคำตอบมากกว่า 1 คำตอบ	มีคำตอบเดียว
มีกลไกการคัดเลือกคุณสมบัติในเทอมการลงโทษ	ไม่มีกลไกการคัดเลือกคุณสมบัติในเทอมการลงโทษ
มีความแข็งแกร่งต่อค่าผิดปกติ (Outlier)	ไม่มีความแข็งแกร่งต่อค่าผิดปกติ (Outlier) เนื่องจากเทอมยกกำลังสอง
ถูกใช้กับชุดข้อมูลที่มีคุณสมบัติทางสปาร์ส (Sparse)	ถูกใช้กับชุดข้อมูลที่มีคุณสมบัติซับซ้อน

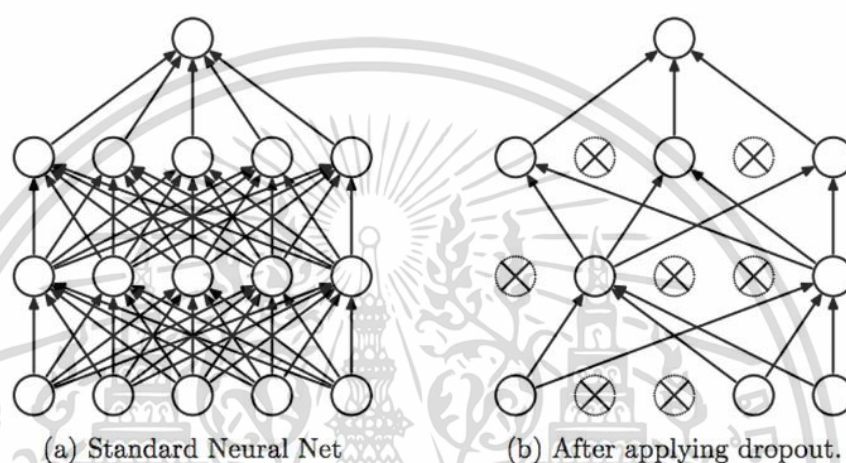


รูปที่ 2.11 ผลการการปรับ Regularization ใน Deep Learning [12]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.3.2 การละทิ้ง (Dropout)

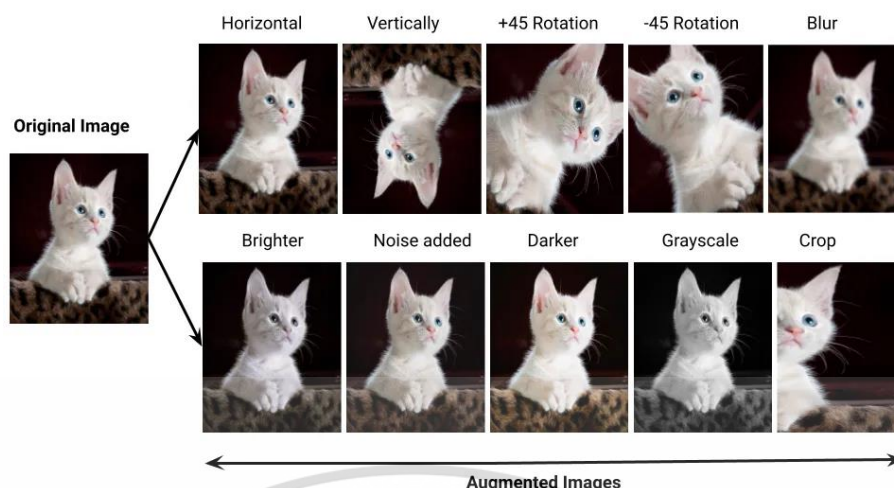
เป็นเทคนิคเรกูลาร์ไรเซชันที่ใช้กันอย่างแพร่หลาย ซึ่งจะทำให้การสุ่มตั้งค่าเฉพาะส่วนของข้อมูลขาเข้าหรือที่เซลล์ประสาทให้มีค่าเป็นศูนย์ในระหว่างแต่ละรอบของการฝึก โดยการทำเช่นนี้โครงข่ายจะถูกบังคับให้ต้องพึ่งพาเซลล์ประสาทอื่นทดแทนส่วนที่ถูกละทิ้งไป ช่วยป้องกันการปรับตัวร่วมกันของเซลล์ประสาทและปรับปรุงการเรียนรู้ลักษณะโดยทั่วไป



รูปที่ 2.12 การทำ Dropout ใน Deep Learning [13]

2.4.3.3 การเพิ่มข้อมูล (Data Augmentation)

ทำการเพิ่มข้อมูลที่เกี่ยวข้องโดยใช้เทคนิคการแปลงแบบสุ่มด้วยการสร้างข้อมูลใหม่เพิ่มจากชุดข้อมูลการฝึกที่มีอยู่ เช่น การหมุน (Rotation) การเลื่อน (Translation) การพลิก (Flip) การตัดรูป (Crop) การขยายรูปเข้าออก (Zoom In/Out) การเพิ่มสัญญาณรบกวน (Noise) และการปรับสี ทำให้มีข้อมูลการฝึกที่หลากหลายเพิ่มมากขึ้น ช่วยลดปัญหาโอเวอร์ฟิตติงจากการเพิ่มขนาดชุดข้อมูลการฝึกอย่างมีประสิทธิภาพ



รูปที่ 2.13 การทำ Data Augmentation ใน Deep Learning [14]

2.4.3.4 การหยุดก่อนกำหนด (Early Stopping)

ผู้ใช้อำหนดให้ตัวแบบเฝ้าดูผลการประเมินประสิทธิภาพตัวแบบด้วยชุดข้อมูลประเมินผล (Validation Dataset) ในระหว่างการฝึก และจะทำการหยุดกระบวนการฝึกอบรมเมื่อผลของประสิทธิภาพชุดข้อมูลประเมินผลเริ่มมีความถูกต้องลดลง ช่วยลดปัญหาโอเวอร์ฟิตติงโดยการหาจุดที่ตัวแบบมีประสิทธิภาพการเรียนรู้ลักษณะทั่วไปที่ดีที่สุด

2.4.3.5 การปรับให้เป็นมาตรฐานแบบแบทช์ (Batch Normalization)

เป็นเทคนิคการปรับชิฟ (Shift) และสเกล (Scale) ให้กับการกระตุ้น (Activation) ที่ซ่อนภายในมินิแบทช์ให้เป็นค่ามาตรฐานและมีขนาดที่เหมาะสมก่อนส่งต่อไปยังโหนดถัดไปโดยเทียบกับค่าเฉลี่ย (Mean) และค่ามาตรฐานเบี่ยงเบน (Standard Deviation) ช่วยแก้ปัญหการเปลี่ยนแปลงของโควาเรียตภายใน (Internal Covariate Shift) และสามารถทำหน้าที่เป็นตัวกำหนดมาตรฐาน (Regularizer) โดยการเพิ่มสัญญาณรบกวนในโครงข่ายระหว่างการฝึก [15]

Input: Values of x over a mini – batch: B
 $= \{x_1 \dots x_m\};$

Parameters to be learned: γ, β

Output: $\{y_i = BN_{\gamma, \beta}(x_i)\}$

$$\mu_B \leftarrow \frac{1}{m} \sum_{i=1}^m x_i ; \text{ mini – batch mean}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\sigma_B^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2 ; \text{mini - batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} ; \text{normalize}$$

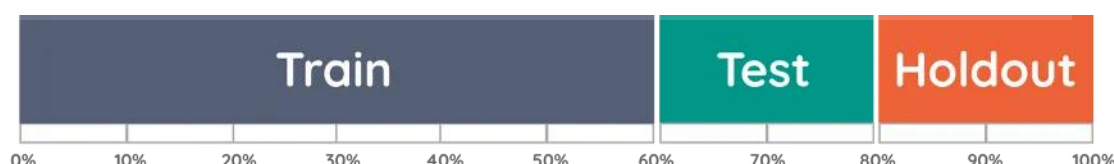
$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv BN_{\gamma, \beta}(x_i) ; \text{scale and shift}$$

2.5 การตรวจสอบความถูกต้องของตัวแบบในระหว่างการฝึก (Model Validation)

เป็นขั้นตอนในการประเมินผลประสิทธิภาพตัวแบบที่ผ่านการฝึกในแต่ละรอบว่ามีความถูกต้องและมีประสิทธิภาพตามที่คาดการณ์ไว้หรือไม่ เพื่อนำผลที่ได้ไปปรับปรุงตัวแบบในรอบการฝึกถัดไปให้ดียิ่งขึ้น โดยมีเทคนิคที่เป็นที่นิยมดังนี้

2.5.1 การแบ่งชุดข้อมูลฝึกและทดสอบตามสัดส่วน (Train/Test Split หรือ Hold-Out Validation)

เป็นเทคนิคการแบ่งชุดข้อมูลออกเป็นชุดข้อมูลฝึก (Train Dataset) ชุดข้อมูลประเมิน (Validation Dataset) และชุดข้อมูลทดสอบ (Test Dataset) ตามสัดส่วนที่ผู้ใช้งานกำหนด เช่น 80% 10% 10% ตามลำดับ โดยตัวแบบจะใช้ชุดข้อมูลฝึกในการฝึก ทำการประเมินค่าผลความถูกต้อง (Accuracy) และค่าความสูญเสีย (Loss) เปรียบเทียบกับผลจากชุดข้อมูลประเมินเพื่อทำการปรับปรุงค่าพารามิเตอร์ของตัวแบบ จากนั้นใช้ชุดข้อมูลทดสอบกับตัวแบบเพื่อประเมินว่าตัวแบบนั้นทำงานได้มีประสิทธิภาพดีเพียงใดกับชุดข้อมูลที่ไม่เคยเห็นมาก่อน เป็นเทคนิคที่ง่ายและตรงไปตรงมาที่สุด แต่มีข้อเสียคือไม่เหมาะกับข้อมูลขนาดเล็ก เพราะจะทำให้ตัวแบบมีความแปรปรวนสูง (High Variance)

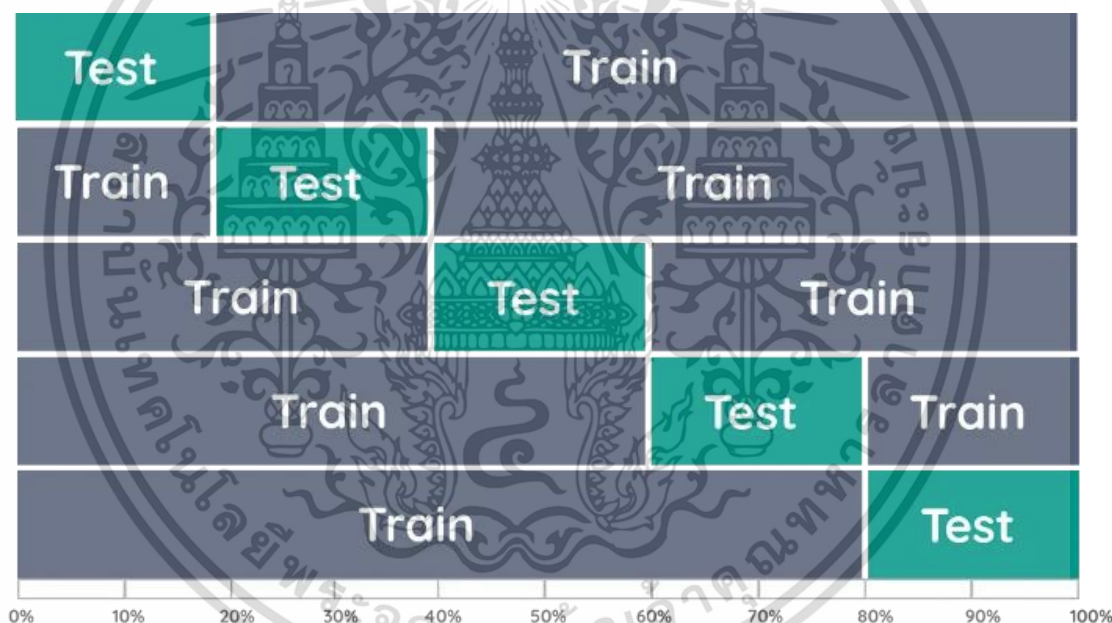


รูปที่ 2.14 การแบ่งชุดข้อมูลแบบ Train/Test Split หรือ Hold-Out Validation [16]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.2 การตรวจสอบแบบข้าม K-Fold (K-Fold Cross-Validation)

เป็นเทคนิคแบ่งชุดข้อมูลซึ่งจะถูกสุ่มแบ่งออกเป็นจำนวน K กลุ่มตามที่ผู้ใช้งานกำหนด โดยในแต่ละ K กลุ่ม ข้อมูลส่วนหนึ่งจะถูกแบ่งไว้เป็นชุดข้อมูลทดสอบและส่วนที่เหลือจะถูกใช้เพื่อเป็นชุดข้อมูลการฝึก เพื่อใช้ประเมินประสิทธิภาพเปรียบเทียบกัน ตัวแบบจะทำการประเมินผลโดยหาค่าเฉลี่ยในแต่ละรอบของการฝึกและการทดสอบ เป็นเทคนิคที่ช่วยให้ตัวแบบได้รับฝึกและทดสอบด้วยชุดข้อมูลมีการกระจายตัวครอบคลุมชุดข้อมูลทั้งหมดและมีการแจกแจงความน่าจะเป็นแบบสม่ำเสมอ (Uniform Probability Distribution) แต่เนื่องจากวิธีนี้ใช้การสุ่มเลือกข้อมูลในแต่ละ K กลุ่มจึงไม่เหมาะกับข้อมูลที่ไม่สมดุล (Imbalanced Datasets) เนื่องจากในบาง Fold อาจจะมีจำนวนข้อมูลบางคลาสมากกว่าอีกคลาสนึงทำให้เกิดปัญหาตัวแบบทำนายผลถูกต้องเฉพาะคลาสข้อมูลที่มีจำนวนมากกว่า (Majority Class)

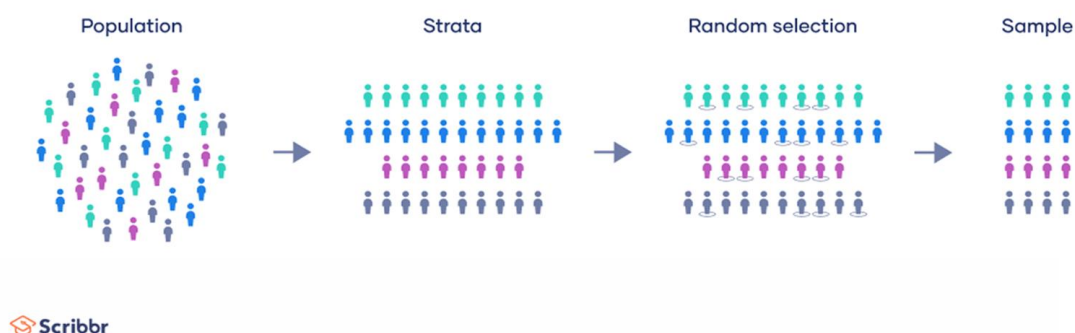


รูปที่ 2.15 การแบ่งชุดข้อมูลแบบ K-Fold Cross-Validation [16]

2.5.3 การตรวจสอบแบบข้าม K-Fold แบบมีการแบ่งชั้น (Stratified K-Fold Cross-Validation)

เป็นเทคนิคเช่นเดียวกับการตรวจสอบแบบข้าม K-Fold แต่เพิ่มเทคนิคการแบ่งชั้น (Strata) เข้ามาช่วยเลือกข้อมูลเป็นตัวแทนที่ครอบคลุมสำหรับแต่ละชุดข้อมูล เพื่อแก้ไขปัญหข้อมูลไม่สมดุล (Imbalanced Datasets)

Stratified sampling



Scribbr

รูปที่ 2.16 การสุ่มแบบ Stratified [17]

2.5.4 การตรวจสอบข้ามแบบเว้นครั้งเดียว (Leave-One-Out Validation)

เป็นเทคนิคเช่นเดียวกับการตรวจสอบแบบข้าม K-Fold แต่ทำในการวนซ้ำจะมีการระบุเวลา โดยชุดข้อมูลจะถูกแบ่งออกเป็นจำนวน $n-1$ ชุด และส่วนของชุดข้อมูลที่เลือกออกไปจะถูกนำไปใช้เป็นข้อมูลทดสอบ มีการประเมินประสิทธิภาพในลักษณะเดียวกับการตรวจสอบแบบข้าม K-Fold เทคนิคนี้เหมาะสำหรับข้อมูลที่มีขนาดเล็ก

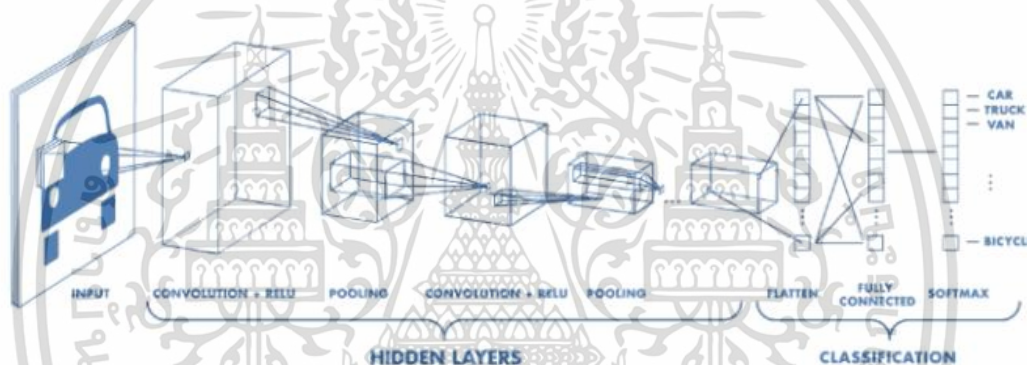


รูปที่ 2.17 การแบ่งชุดข้อมูลแบบ Leave-One-Out Validation [16]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6 โครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (Convolutional Neural Network)

เป็นอัลกอริทึมการเรียนรู้เชิงลึกที่ถูกออกแบบมาเพื่อใช้ในการวิเคราะห์ข้อมูลภาพ เช่น รูปภาพหรือวิดีโอ มีประสิทธิภาพดีในการสกัดคุณลักษณะที่โดดเด่น (Features) มักจะนำไปใช้ประโยชน์ในจำแนกความแตกต่างของรูปภาพ (Image Classification) โดยเฉพาะในงานด้านที่เกี่ยวข้องกับการมองเห็นของเครื่องจักรหรือคอมพิวเตอร์ (Machine/Computer Vision) จากการป้อนข้อมูลรูปภาพเข้าผ่านโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล โดยมีสถาปัตยกรรมโดยทั่วไปประกอบด้วย 3 ชั้น ได้แก่ชั้นคอนโวลูชันนอล (Convolution Layer) ชั้นพูลลิง (Pooling Layer) และชั้นฟูลลี่ย้อนเชื่อม (Fully Connected Layer) ดังที่แสดงไว้ในรูปที่ 2.18



รูปที่ 2.18 สถาปัตยกรรม CNN [18]

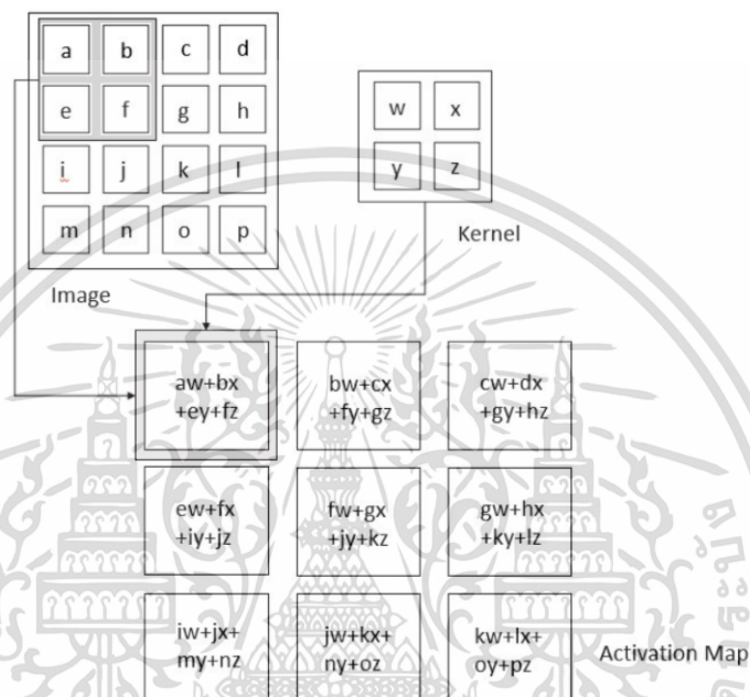
2.6.1 ชั้นคอนโวลูชันนอล (Convolutional Layer)

เป็นชั้นแรกและเป็นส่วนประกอบสำคัญของโครงข่ายประสาทเทียม CNN ทำหน้าที่ในการสกัดคุณลักษณะของข้อมูลรูปภาพที่ถูกส่งเข้ามา โดยใช้ตัวกรอง (Kernel) ในการกรอง (Filter) ข้อมูลเข้าและสกัดข้อมูลที่เป็นประโยชน์ (Features Extraction) เช่น ขอบของวัตถุ สี ความเข้ม ความสว่าง ความกว้าง ความสูง และความลึก เป็นต้น ตัวกรองที่อยู่ในรูปของเมทริกซ์จะเคลื่อนผ่านเมทริกซ์ของรูปภาพขาเข้าและทำการ Dot Product กัน เพื่อสร้างฟังก์ชันลักษณะ (Feature/Activation Map)

กำหนดให้ I เป็นเมทริกซ์รูปภาพ K เป็นเมทริกซ์ตัวกรอง (Kernel) ขนาด $2k + 1 \times 2k + 1$ และ G คือผลลัพธ์จากการทำ Dot Product

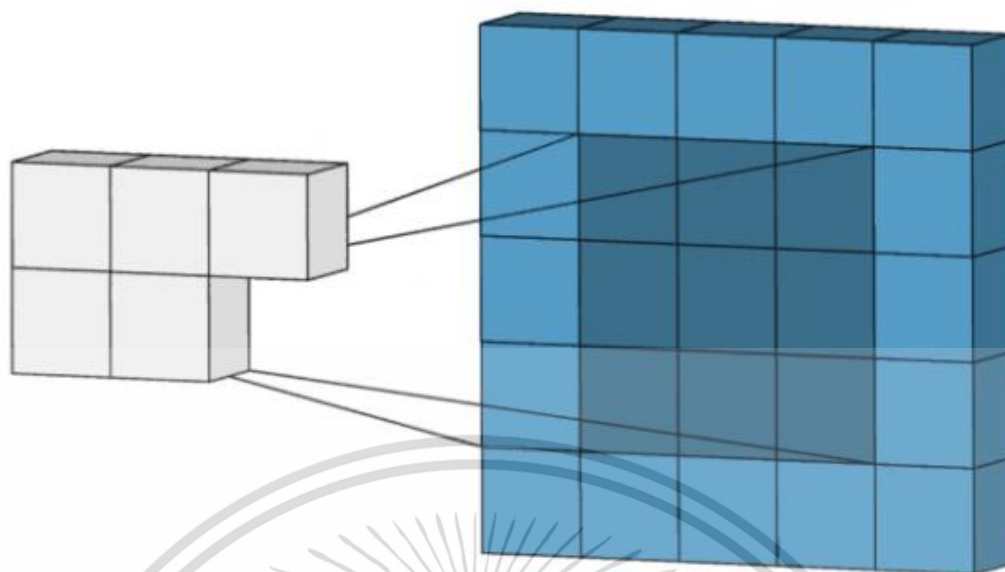
$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k K[u, v] I[i - u, j - v]$$

เรียกสมการ $G = K \otimes I$ ว่า Convolutional Operation

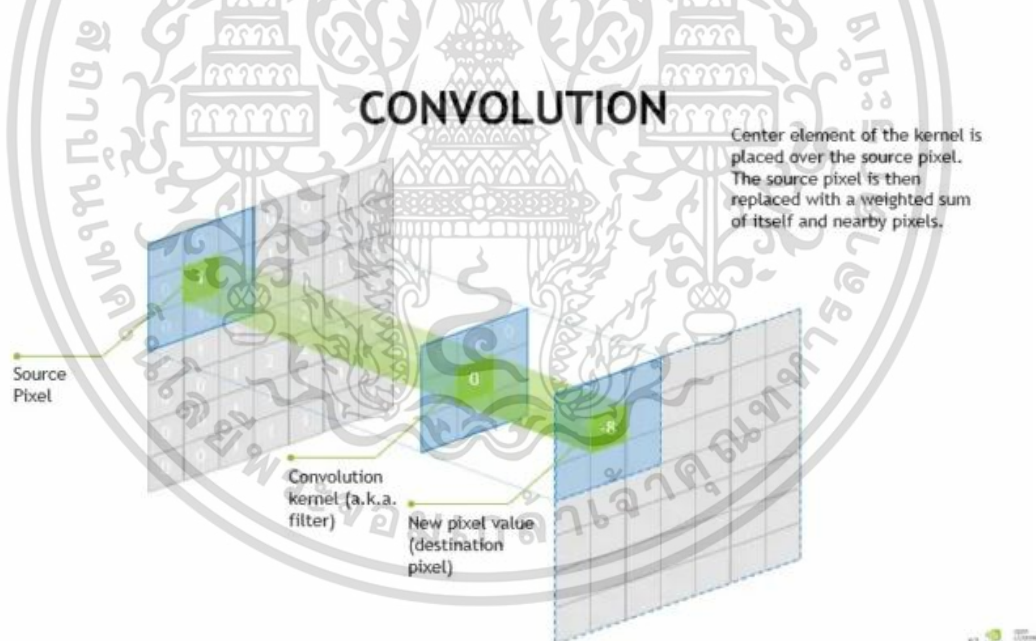


รูปที่ 2.19 การทำDot Product ของเมทริกซ์รูปภาพและตัวกรองเพื่อสร้างฟังก์ชันลักษณะ [19]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.20 การทำ Convolution เพื่อสร้างเมทริกซ์ Activation Map รูปเมทริกซ์สีน้ำเงินคือรูปภาพขาเข้า และเมทริกซ์สีเทาคือ Activation Map [19]



รูปที่ 2.21 ภาพรวมการทำคอนโวลูชันนอล โดยตัวกรองจะเลื่อนไปตามเมทริกซ์ของรูปภาพขาเข้าจนครบทุกด้าน เพื่อสร้างเมทริกซ์ฝังคุณลักษณะขึ้นมา [20]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

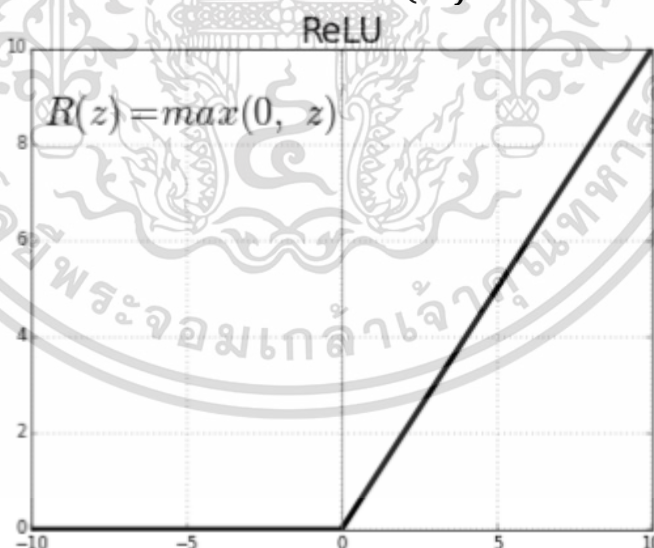
กำหนดให้รูปภาพขาเข้าเป็นเมทริกซ์ขนาด $W \times W \times D$ และ D_{out} เป็นจำนวนของตัวกรอง (Kernel) ที่มีขนาด F มีการขยับ (Stride) ครั้งละ S และมีปริมาณ Padding คือ P จะสามารถคำนวณหาปริมาตรของเมทริกซ์ผลลัพธ์ได้จากสูตร

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

โดยปริมาตรของเมทริกซ์ผลลัพธ์มีขนาดเป็น $W_{out} \times W_{out} \times D_{out}$

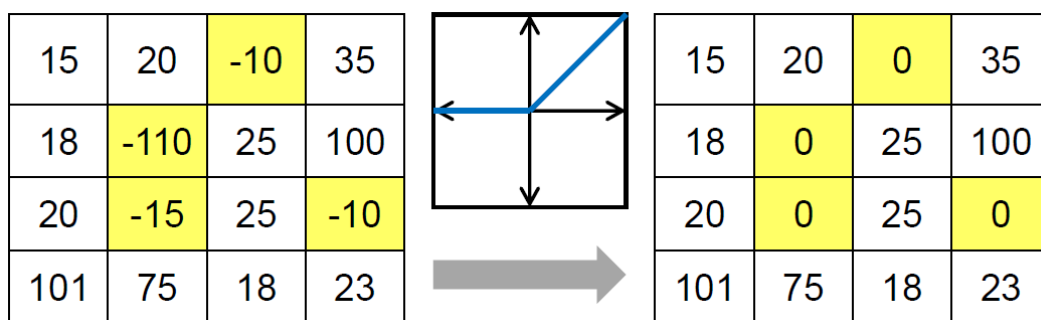
หลังจากการทำ Convolutional Operation แล้วมักจะตามด้วย Non-Linear Operation ในแต่ละชั้น จึงมีการนำฟังก์ชันกระตุ้น (Activation Function) ที่เป็นที่นิยม เช่น Rectified Linear Unit (ReLU) มาประยุกต์ใช้กับฟังก์ชันลักษณะ (Feature/Activation Map) ที่ได้จากแต่ละชั้น Convolution เพื่อทำให้คุณลักษณะที่สำคัญของฟังก์ชันลักษณะถูกดึงขึ้นมาให้โดดเด่นและตัดคุณลักษณะที่ไม่โดดเด่นออกไป ก่อนส่งต่อไปในชั้นถัดไป สามารถอธิบายสมการของ ReLU ได้ดังนี้

$$f(z) = \max(0, z) = \begin{cases} 0 & \text{for } z \leq 0 \\ z & \text{for } z \geq 0 \end{cases}$$



รูปที่ 2.22 กราฟฟังก์ชันกระตุ้น ReLU [21]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

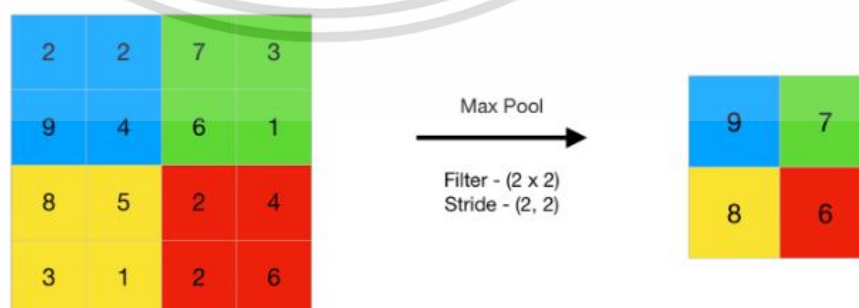


รูปที่ 2.23 การทำงานของฟังก์ชันกระตุ้น ReLU [22]

2.6.2 ชั้นการรวมกัน (Pooling Layer)

อยู่ถัดจากชั้นคอนโวลูชันนอล มีวัตถุประสงค์เพื่อลดขนาดของฟังก์ชันลักษณะ (Feature/Activation Map) ให้มีขนาดเล็กลง ช่วยลดการใช้หน่วยความจำและลดต้นทุนในการประมวลผลของคอมพิวเตอร์ โดยใช้ตัวกรองการรวม (Pooling Filter) ซึ่งมีขนาดเล็กกว่าขนาดของฟังก์ชันลักษณะ ในการทำ Pooling Operation จะได้ผลลัพธ์เป็นเมทริกซ์ฟังก์ชันลักษณะที่มีขนาดเล็กลงกว่าเดิมจากการเลือกเก็บเฉพาะคุณลักษณะที่มีความสำคัญโดดเด่นเฉพาะซึ่งเป็นตัวแทนของฟังก์ชันลักษณะเดิม โดยเป็นที่นิยมใช้กันอยู่ 2 วิธี

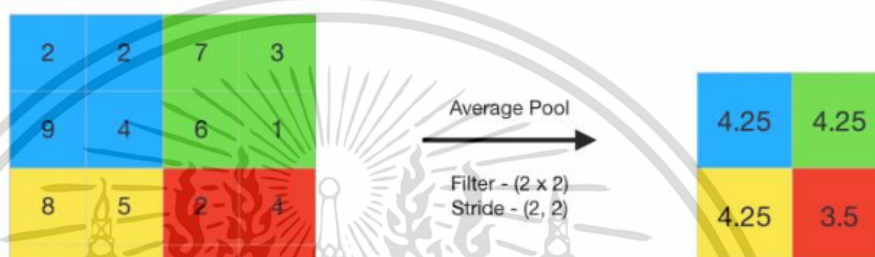
- วิธีการรวมค่าสูงสุด (Max Pooling) เป็นการดึงค่าสูงที่เป็นตัวแทนของฟังก์ชันลักษณะออกมารวมไว้เป็นเมทริกซ์ฟังก์ชันลักษณะใหม่ที่มีขนาดเล็กลงกว่าเดิม โดยมีตัวอย่างขั้นตอนการทำงานดังนี้ กำหนดเมทริกซ์ฟังก์ชันลักษณะขนาด 4x4 และตัวกรองขนาด 2x2 โดยมีการขยับทีละ 2x2 พิกเซล ตัวกรองจะทำการเลือกค่าสูงสุดที่เป็นตัวแทนของแต่ละพิกเซลบนเมทริกซ์ฟังก์ชันลักษณะเดิม ได้ผลลัพธ์เป็นเมทริกซ์ฟังก์ชันลักษณะใหม่ที่มีขนาดลดลงเป็น 2x2



รูปที่ 2.24 ขั้นตอนการทำ Max Pooling [23]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

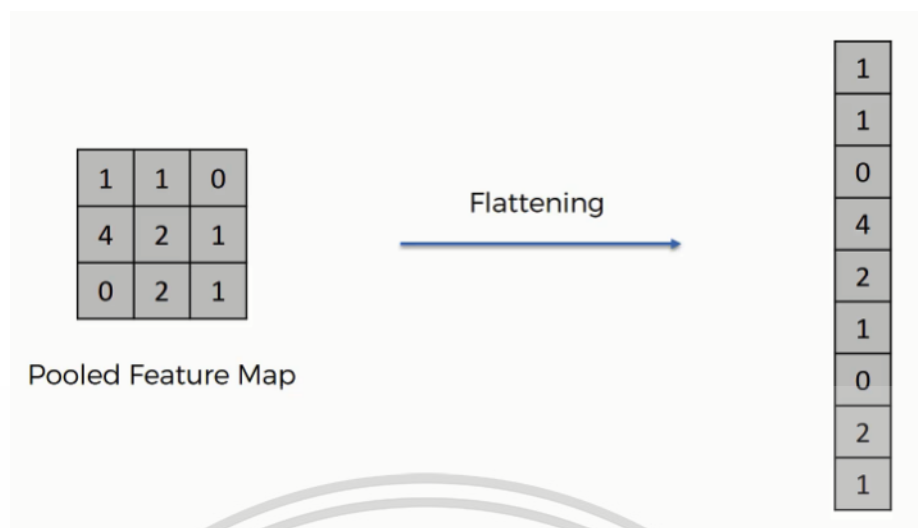
- วิธีการรวมค่าเฉลี่ย (Average Pooling) เป็นการดึงค่าเฉลี่ยที่เป็นตัวแทนของฟังก์ชันลักษณะออกมารวมไว้เป็นเมทริกซ์ฟังก์ชันลักษณะใหม่ที่มีขนาดเล็กกว่าเดิม โดยมีตัวอย่างขั้นตอนการทำงานดังนี้ กำหนดเมทริกซ์ฟังก์ชันลักษณะขนาด 4×4 และตัวกรองขนาด 2×2 โดยมีการขยับทีละ 2×2 พิกเซล ตัวกรองจะทำการเลือกค่าเฉลี่ยที่เป็นตัวแทนของแต่ละพิกเซลบนเมทริกซ์ฟังก์ชันลักษณะเดิม ได้ผลลัพธ์เป็นเมทริกซ์ฟังก์ชันลักษณะใหม่ที่มีขนาดลดลงเป็น 2×2



รูปที่ 2.25 ขั้นตอนการทำ Average Pooling [23]

2.6.3 ชั้นการเชื่อมต่อสมบูรณ์ (Fully Connected Layer)

รับข้อมูลเมทริกซ์ฟังก์ชันลักษณะซึ่งถูกลดขนาดลงแล้วต่อจากชั้นการรวมกัน โดยเมทริกซ์ฟังก์ชันลักษณะจากชั้นการรวมกันจะถูกทำการปรับลดมิติลง (Flattening) ให้เป็นเวกเตอร์เดี่ยว (Single Vector) ที่ชั้นนี้เป็นชั้นที่มีเซลล์ประสาทเทียมของแต่ละชั้นเชื่อมต่อกันทั้งหมดเช่นเดียวกับโครงข่ายประสาทเทียม (Artificial Neural Network) เพื่อวิเคราะห์ค้นหาคุณลักษณะสำคัญของข้อมูล จากนั้นในขั้นสุดท้ายของชั้นนี้จะทำการจำแนกคลาสข้อมูล (Classification) ด้วยฟังก์ชันกระตุ้น Softmax เพื่อแจกแจงความน่าจะเป็นของผลลัพธ์แต่ละคลาสที่ตัวแบบทำนาย ในชั้นการเชื่อมต่อสมบูรณ์ อาจจะมีการพิจารณาใช้เทคนิคการละทิ้ง (Dropout) โดยการสุ่มลดจำนวนเซลล์ประสาทเทียมลงจำนวนหนึ่งในกรณีที่ตัวแบบเกิดปัญหาโอเวอร์ฟิตติง



รูปที่ 2.26 การปรับลดมิติของฟังก์ชันลักษณะลง (Flattening) ให้เป็นเวกเตอร์เดี่ยว (Single Vector)

[24]

สมการฟังก์ชันกระตุ้น Softmax เขียนได้ดังนี้

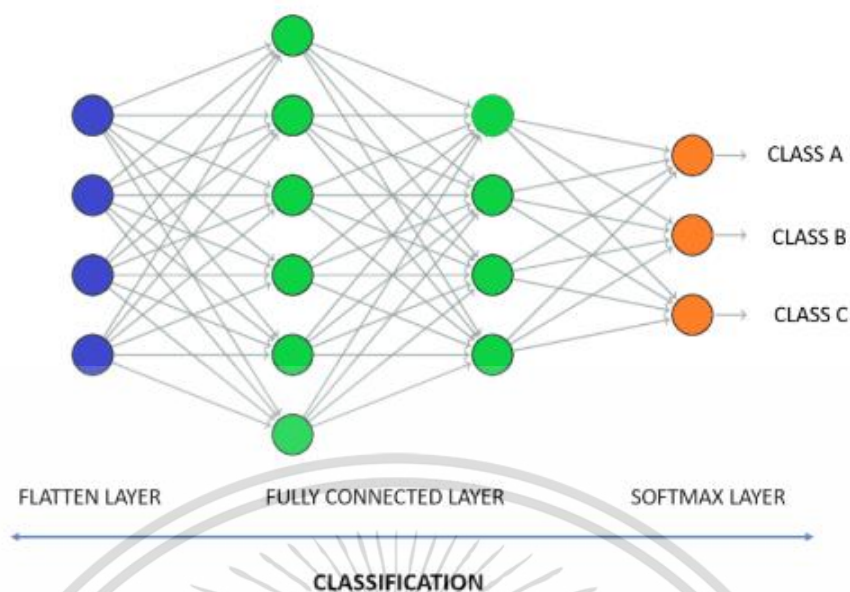
$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

โดยกำหนดให้

$\vec{z}$ คือ เวกเตอร์ขาเข้าของฟังก์ชัน Softmax ประกอบด้วย $(z_0, \dots, z_K)$

z_i คือ ส่วนประกอบของเวกเตอร์ขาเข้าที่ฟังก์ชัน Softmax เป็นจำนวนจริง

K คือ จำนวนคลาสในการจำแนก



รูปที่ 2.27 โครงสร้างชั้น Fully Connected และฟังก์ชันกระตุ้น Softmax สำหรับใช้จำแนกข้อมูล [25]

2.7 การสูญเสียเอนโทรปีข้ามหมวดหมู่ (Categorical Cross-Entropy Loss) หรือ การสูญเสีย Softmax (Softmax Loss)

เป็นฟังก์ชันการสูญเสียพื้นฐานที่ใช้ในการเรียนรู้ของเครื่องสำหรับการจำแนกปัญหาแบบหลายคลาส (Multi-Class Classification) ฟังก์ชันจะทำการประเมินความแตกต่างระหว่างความน่าจะเป็นของคลาสจากการทำนายกับคลาสจริง เพื่อลดความผิดพลาด (Error) ของการจำแนกให้ต่ำที่สุดในระหว่างขั้นตอนการฝึกเป็นการช่วยปรับปรุงความแม่นยำของตัวแบบในการทำนายคลาสที่ถูกต้องให้ดียิ่งขึ้น [26] [27]

Multi-class Classification

Class			
Class#	1	2	3
Class Label	0	1	2
One-hot encoding	[1, 0, 0]	[0, 1, 0]	[0, 0, 1]

รูปที่ 2.28 การจำแนกหลายคลาส [28]



รูปที่ 2.29 Categorical Cross Entropy Loss หรือ Softmax Loss [29]

สมการฟังก์ชันกระตุ้น Softmax คือ

$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}}$$

s_i คือ เวกเตอร์คะแนนของคลาสที่สนใจ โดยขึ้นตรงกับทุกองค์ประกอบของ S s_j คือ เวกเตอร์คะแนนรวมของแต่ละคลาส และ C คือ จำนวนคลาส

สมการการสูญเสียเอนโทรปี คือ

$$CE = - \sum_i^C t_i \log(s_i)$$

เมื่อ t_i และ s_i คือ เวกเตอร์คำตอบจริง (Target/Ground Truth Vector) ของแต่ละคลาส C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนื่องจากคำตอบจริงเป็นแบบ One-Hot ทำให้มีเฉพาะคลาสที่เป็นบวก $C = C_p$ ถูกเก็บไว้ในเทอมการสูญเสีย และเวกเตอร์ตอบจริงมีองค์ประกอบเพียงแค่หน่วยเดียวซึ่งไม่เป็นศูนย์ $t_i = t_p$ จึงสามารถละส่วนของผลรวมคลาสจริงมีค่าเป็นศูนย์ไว้ได้ และสามารถเขียนสมการการสูญเสียเอนโทรปีข้ามหมวดหมู่ได้เป็น

$$CE = -\log \left(\frac{e^{s_p}}{\sum_j^C e^{s_j}} \right)$$

เมื่อ s_p คือ คะแนนของคลาสที่เป็นบวก

โดยเกรเดียนต์ของคลาสที่เป็นบวก คือ

$$\frac{\partial}{\partial s_p} \left(-\log \left(\frac{e^{s_p}}{\sum_j^C e^{s_j}} \right) \right) = \left(\frac{e^{s_p}}{\sum_j^C e^{s_j}} - 1 \right)$$

โดยเกรเดียนต์ของคลาสที่เป็นลบ คือ

$$\frac{\partial}{\partial s_n} \left(-\log \left(\frac{e^{s_p}}{\sum_j^C e^{s_j}} \right) \right) = \left(\frac{e^{s_n}}{\sum_j^C e^{s_j}} \right)$$

เมื่อ s_n คือ คะแนนของคลาสที่เป็นลบใน C ซึ่งไม่อยู่ใน C_p

2.8 การถ่ายโอนการเรียนรู้ (Transfer Learning)

เป็นเทคนิคการเรียนรู้ของเครื่องที่เกี่ยวข้องกับการใช้ประโยชน์จากความรู้ที่ได้รับจากการแก้ปัญหาแบบหนึ่งและนำไปประยุกต์ใช้กับปัญหาอื่นที่มีความเกี่ยวข้องกัน โดยไม่ต้องเริ่มกระบวนการเรียนรู้ใหม่ตั้งแต่เริ่มต้น การถ่ายโอนการเรียนรู้ช่วยให้ตัวแบบได้รับประโยชน์จากการเรียนรู้คุณลักษณะทั่วไปหรือการเป็นตัวแทนข้อมูลก่อนหน้านี้

ในแนวทางการเรียนรู้ของเครื่องแบบดั้งเดิม ตัวแบบจะได้รับการฝึกด้วยชุดข้อมูลเฉพาะสำหรับปัญหาแต่ละประเภท เช่น การจำแนกภาพหรือการประมวลผลภาษาธรรมชาติ โดยตัวแบบจะเรียนรู้การสกัดคุณลักษณะสำคัญของข้อมูลนำเข้าและทำการทำนายผลลัพธ์ตามคุณลักษณะเหล่านั้น แต่การฝึกตัวแบบการเรียนรู้เชิงลึกตั้งแต่เริ่มต้นด้วยชุดข้อมูลขนาดใหญ่มีค่าใช้จ่ายในการคำนวณสูงและจำเป็นต้องใช้ข้อมูลที่ถูกระบุคำตอบไว้แล้วเป็นจำนวนมาก การถ่ายโอนการเรียนรู้เข้ามามีส่วนช่วยแก้ปัญหานี้ด้วยการประยุกต์ใช้ตัวแบบที่ผ่านการฝึกมาแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ด้วยชุดข้อมูลขนาดใหญ่ โดยตัวแบบที่ผ่านการฝึกมาแล้วล่วงหน้าได้ผ่านการเรียนรู้คุณลักษณะทั่วไปและการเป็นตัวแทนข้อมูลที่เป็นประโยชน์สำหรับงานด้านอื่นที่เกี่ยวข้องได้

การถ่ายโอนการเรียนรู้มีขั้นตอนดังต่อไปนี้:

2.8.1 การฝึกล่วงหน้า (Pre-Training)

ตัวแบบจะถูกฝึกด้วยชุดข้อมูลขนาดใหญ่ในงานที่เกี่ยวข้อง เช่น การฝึกโครงข่ายประสาทเทียมการเรียนรู้เชิงลึกล่วงหน้าด้วยชุดข้อมูลรูปภาพจำนวนมากสำหรับใช้จำแนกชนิดรูปภาพ ทำให้ตัวแบบสามารถเรียนรู้คุณลักษณะและรูปแบบทั่วไปของข้อมูลได้

2.8.2 การสกัดคุณสมบัติ (Feature Extraction)

หลังจากทำการฝึกล่วงหน้าเสร็จแล้ว ชั้นของตัวแบบจะถูกหยุดไว้ (Frozen) โดยคุณลักษณะที่ถูกเรียนรู้จะถูกแยกออกจากตัวแบบที่ผ่านการฝึกล่วงหน้า ซึ่งคุณลักษณะเหล่านี้จัดเก็บข้อมูลระดับสูงเกี่ยวกับข้อมูลที่เข้าเอาไว้เพื่อนำไปใช้สรุปงานต่างๆ ต่อไป

2.8.3 การปรับแต่งอย่างละเอียด (Fine-Tuning)

คุณลักษณะที่ถูกสกัดออกมาแล้วจะถูกใช้เป็นข้อมูลขาเข้าสำหรับตัวแบบใหม่ ซึ่งมักจะมีขนาดเล็กกว่าและเป็นงานที่มีความเฉพาะตัว ตัวแบบใหม่นี้จะได้รับการฝึกด้วยชุดข้อมูลที่มีขนาดเล็กลงและมีความเฉพาะเจาะจงสำหรับปัญหานั้น พารามิเตอร์ของตัวแบบที่ผ่านการฝึกล่วงหน้าอาจถูกปรับแต่งเพิ่มเติมในระหว่างขั้นตอนนี้เพื่อปรับให้เหมาะสมกับปัญหาใหม่

จากประโยชน์ของการถ่ายโอนการเรียนรู้ ตัวแบบใหม่จะนำความสามารถของตัวแบบที่ได้รับการฝึกล่วงหน้ามาใช้ในการสกัดคุณลักษณะที่เป็นประโยชน์ ส่งผลให้เกิดการลู่เข้า (Convergence) ที่เร็วขึ้นและประสิทธิภาพที่ดีขึ้น แม้ว่าจะมีข้อมูลที่มีค่าตอบแล้วในจำนวนจำกัดก็ตาม การถ่ายโอนการเรียนรู้ยังช่วยให้ตัวแบบสามารถสรุปข้อมูลใหม่ที่ไม่เคยเห็นมาก่อนได้ดีขึ้น เป็นผลจากการที่ตัวแบบได้รับการฝึกล่วงหน้าได้เรียนรู้การเป็นตัวแทนที่มีประสิทธิภาพจากชุดข้อมูลขนาดใหญ่และมีความหลากหลาย

2.9 สถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันนอล

ในการศึกษาประสิทธิภาพการเรียนรู้เชิงลึกในการจำแนกรูปภาพภัยพิบัตินี้ ผู้วิจัยทำได้เลือกใช้ตัวแบบการถ่ายโอนการเรียนรู้จาก TensorFlow Hub ประกอบด้วย EfficientNet V2 ที่ผ่านการฝึกด้วยข้อมูลรูปภาพ ImageNet-21K และ Inception V3 กับ MobileNet V3 ที่ผ่านการฝึกด้วยข้อมูลรูปภาพ ImageNet (ILSVRC-2012-CLS)

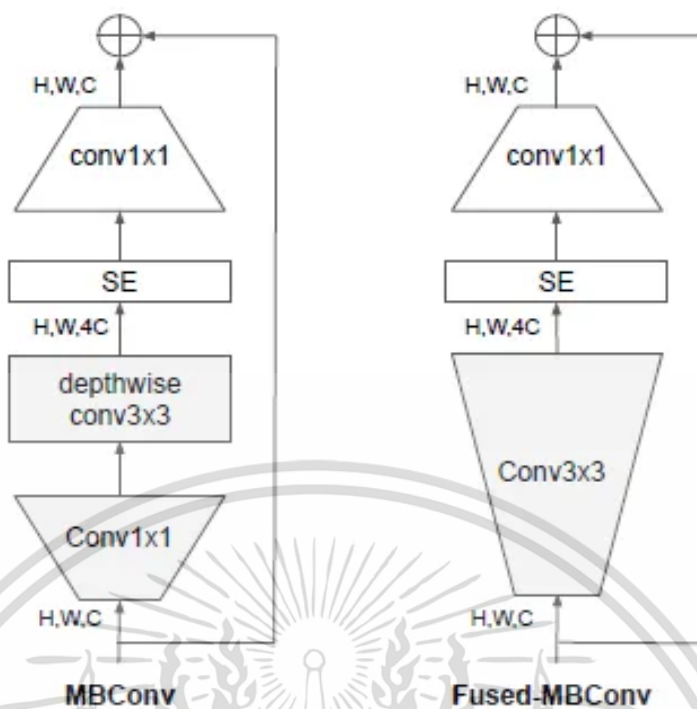
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.9.1 สถาปัตยกรรม EfficientNet V2

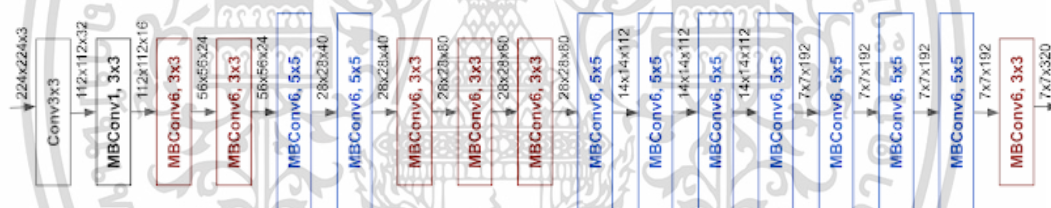
เป็นสถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันนอล (CNN) เปิดตัวในปี 2564 คิดค้นโดย Mingxing Tan และ Quoc V. Le [30] ได้รับการออกแบบให้มีประสิทธิภาพมากกว่าสถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันนอลรุ่นก่อนหน้าในขณะที่ยังคงให้ความแม่นยำสูง สถาปัตยกรรมของ EfficientNetV2 ถูกสร้างขึ้นบนพื้นฐานของ EfficientNet โดยมีคุณสมบัติใหม่ที่สำคัญ เช่น

- Fused-MBConv เป็นการดำเนินการคอนโวลูชันนอลแบบใหม่ที่รวมข้อดีของทั้งการทำคอนโวลูชันนอลเชิงลึก (Depth-Wise) และการทำคอนโวลูชันนอลตามจุด (Pointwise)
- การเรียนรู้แบบก้าวหน้า (Progressive Learning) เป็นเทคนิคในการค่อยๆ เพิ่มขนาดและความซับซ้อนของตัวแบบในระหว่างการฝึก
- เรกูลาไรเซชันแบบปรับได้ (Adaptive Regularization) เป็นเทคนิคสำหรับการปรับปริมาณการใช้เรกูลาไรเซชันโดยอัตโนมัติในระหว่างการฝึก
- ใช้เทคนิคการผสมผสานระหว่างการค้นหาสถาปัตยกรรมประสาทและการปรับขนาด (Neural Architecture Search and Scaling)

ตัวอย่างเช่น EfficientNetV2-B0 มีประสิทธิภาพความถูกต้อง 84.4% สูงสุด 5 อันดับแรกในชุดข้อมูล ImageNet ในขณะที่มีพารามิเตอร์เพียง 3.4M และ 1.5B FLOPS ซึ่งมีขนาดเล็กกว่าและเร็วกว่ารุ่นล้ำสมัยก่อนหน้าอย่างมากเมื่อเปรียบเทียบกับสถาปัตยกรรมรุ่นก่อน เช่น ResNet50 และ InceptionV3 โดย EfficientNetV2 ได้รับการพิสูจน์แล้วว่ามีประสิทธิภาพสำหรับงานด้านต่างๆ โดยเฉพาะงานด้านการจำแนกภาพ การตรวจจับวัตถุ และการแบ่งส่วนความหมาย (Semantic Segmentation)

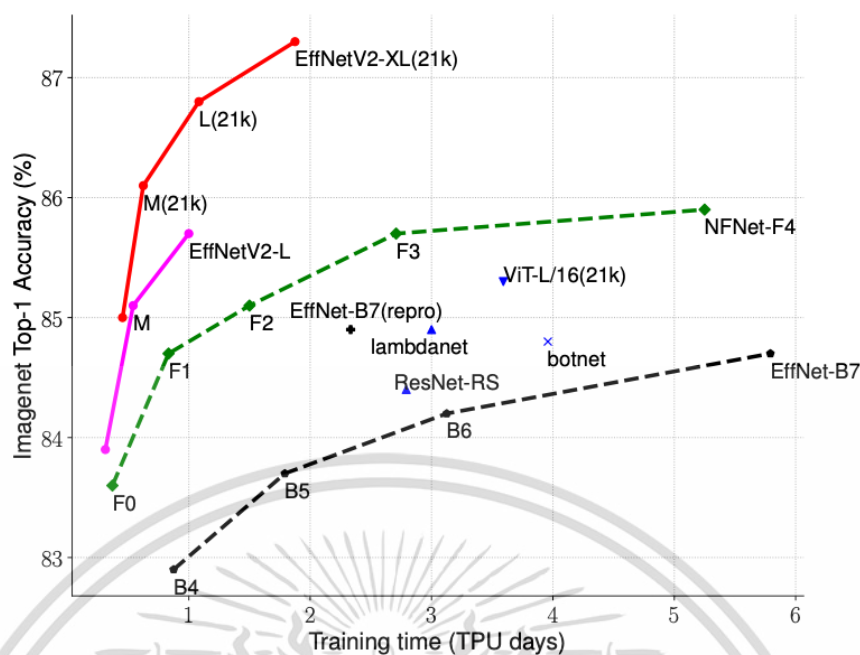


รูปที่ 2.30 โครงสร้าง Fused-MBConv ใน EfficientNet V2 [31]



รูปที่ 2.31 สถาปัตยกรรมพื้นฐานของโครงข่าย EfficientNet-B0 [30]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(a) Training efficiency.

	EfficientNet (2019)	ResNet-RS (2021)	DeiT/ViT (2021)	EfficientNetV2 (ours)
Top-1 Acc.	84.3%	84.0%	83.1%	83.9%
Parameters	43M	164M	86M	24M

(b) Parameter efficiency.

รูปที่ 2.32 ประสิทธิภาพความถูกต้องของ EfficientNet V2 [32]

2.9.2 สถาปัตยกรรม Inception v3

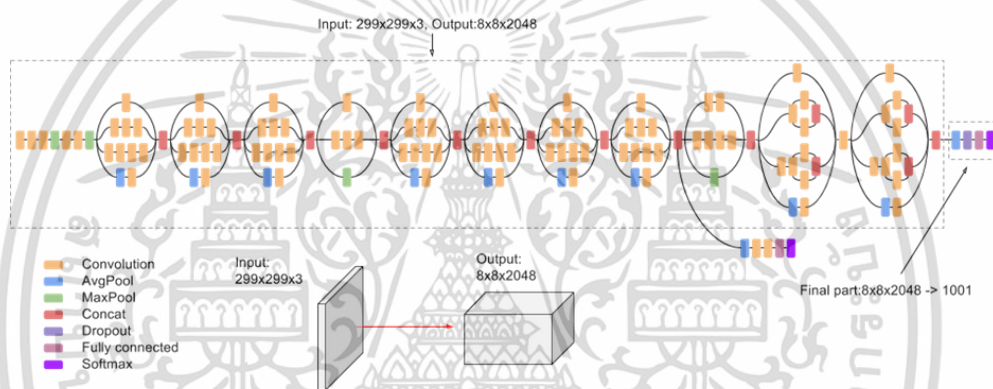
เป็นสถาปัตยกรรมโครงข่ายประสาทเทียมคอนโวลูชันนอล (CNN) เปิดตัวในปี 2558 ถูกพัฒนาโดย Christian Szegedy Sergey Ioffe Vincent Vanhoucke และ Alex Alemi [33] จากการดัดแปลงสถาปัตยกรรม Inception v2 รุ่นก่อนหน้า เพื่อปรับปรุงความถูกต้องและประสิทธิภาพให้ดีขึ้น โดยมีคุณสมบัติใหม่ที่สำคัญ เช่น

- โมดูล Inception (Inception Modules) เป็นโมดูลเฉพาะทางซึ่งรวมการปฏิบัติการด้าน Convolution Pooling และ Nonlinear ประเภทต่างๆไว้ด้วยกัน เพื่อเรียนรู้คุณลักษณะข้อมูลให้มีประสิทธิภาพมากขึ้น
- ตัวแยกประเภทเสริม (Auxiliary Classifiers) เป็นตัวแยกประเภทขนาดเล็กที่ถูกเพิ่มเข้ากับโครงข่ายที่ชั้นกลางช่วยฝึกโครงข่ายในการเรียนรู้การจำแนกคุณลักษณะได้มากขึ้น เพื่อปรับปรุงความถูกต้องให้ดีขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การปรับข้อมูลคำตอบให้เรียบ (Label Smoothing) เป็นเทคนิคในการลดผลกระทบจากปัญหาโอเวอร์ฟิตติงโดยการเพิ่มสัญญาณรบกวนที่ข้อมูลคำตอบระหว่างการฝึก

Inception v3 ได้รับการพิสูจน์แล้วว่ามีประสิทธิภาพสำหรับงานต่างๆ โดยเฉพาะงานด้านการจำแนกภาพ การตรวจจับวัตถุ และการแบ่งส่วนความหมาย ตัวแบบมีความถูกต้องสูงถึง 78.1% สูงสุด 5 อันดับแรกในชุดข้อมูล ImageNet มีประสิทธิภาพโดยมีเพียง 24 ล้านพารามิเตอร์และ 13 พันล้าน FLOPS แต่มีข้อจำกัดบางอย่าง เช่น การฝึกอาจทำได้ยากเนื่องจากต้องใช้ชุดข้อมูลขนาดใหญ่และใช้การประมวลผล GPU สูง และไม่ยืดหยุ่นเท่าสถาปัตยกรรมอื่นๆ เช่น ResNet



รูปที่ 2.33 ภาพรวมโครงข่ายสถาปัตยกรรม Inception V3 [34]

2.9.3 สถาปัตยกรรม MobileNetV3

เป็นสถาปัตยกรรมเครือข่ายประสาทเทียมคอนโวลูชันนอล (CNN) ถูกออกแบบมาเฉพาะสำหรับการประมวลผลบนโทรศัพท์มือถือ โดย Andrew Howard และทีมพัฒนา [35] ในปี 2562 เป็นโครงข่ายที่มีพื้นฐานในการผสมผสานระหว่างการค้นหาสถาปัตยกรรมโครงข่ายที่คำนึงถึงทรัพยากรฮาร์ดแวร์ (Hardware-Aware Network Architecture Search - NAS) และสถาปัตยกรรมใหม่ที่ถูกพัฒนาขึ้น NAS มีขั้นตอนการทำงานโดยใช้พื้นที่การค้นหา MnasNet ซึ่งเป็นชุดสถาปัตยกรรมคอนโวลูชันนอลที่มีขนาดใหญ่และมีความเป็นไปได้ที่หลากหลาย ทำการค้นหาบนชุดข้อมูล ImageNet จากนั้นตัวแบบจะเลือกสถาปัตยกรรมที่มีประสิทธิภาพดีที่สุด การพัฒนาสถาปัตยกรรมใหม่ของ MobileNetV3 ประกอบกับการใช้ฟังก์ชันกระตุ้นแบบ Hard Swish และโมดูล Squeeze-and-Excitation (SE) ซึ่ง Hard Swish เป็นฟังก์ชันกระตุ้นแบบ Non-Linear ที่มีประสิทธิภาพมากกว่าฟังก์ชันกระตุ้น ReLU และโมดูล SE ทำการปรับปรุง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประสิทธิภาพของโครงข่ายโดยการปรับตั้งค่าความสำคัญของช่องคุณลักษณะต่างๆ MobileNetV3 ได้รับการพิสูจน์แล้วว่ามีความถูกต้องที่ล้ำหน้ากว่าสถาปัตยกรรมอื่นๆ ในงานด้านการมองเห็นบนโทรศัพท์มือถือที่หลากหลาย โดยเฉพาะงานด้านการจำแนกภาพ การตรวจจับวัตถุ และการแบ่งส่วนความหมาย มีประสิทธิภาพที่ดีในการทำงานด้วยอัตราเฟรมสูงบนอุปกรณ์พกพา

Size	Depth multiplier	Top1 accuracy (%)	Pixel 1 (ms)	Pixel 2 (ms)	Pixel 3 (ms)
Large	1.0	75.2	51.2	61	44
Large	0.75	73.3	39.8	48	32
Small	1.0	67.5	15.8	19.4	14.4
Small	0.75	65.4	12.8	15.9	11.6

รูปที่ 2.34 ประสิทธิภาพความถูกต้องของ MobileNet V3 [36]

2.10 การประเมินผลประสิทธิภาพของตัวแบบ (Performance Metrics)

งานวิจัยนี้ใช้เมทริกซ์ความสับสน (Confusion Matrix) ตัวชี้วัดประสิทธิภาพค่าการเรียกคืน (Recall) และค่าความถูกต้อง (Accuracy) ในการประเมินผลประสิทธิภาพของตัวแบบ

2.10.1 เมทริกซ์ความสับสน (Confusion Matrix)

ในการประเมินผลประสิทธิภาพของตัวแบบโครงข่ายประสาทเทียมการเรียนรู้เชิงลึกคอนโวลูชัน นอลแบบมีการถ่ายโอนการเรียนรู้ ทำการคำนวณเปรียบเทียบประสิทธิภาพเพิ่มเติมด้วยตัวชี้วัดดังนี้

		ACTUAL VALUES	
		Positive	Negative
PREDICTED VALUES	Positive	TP	FP
	Negative	FN	TN

รูปที่ 2.35 เมทริกซ์ความสับสน (Confusion Matrix) [37]

- ผลบวกจริง (True Positive – TP) คือ กรณีที่ตัวแบบทำนายผลเป็นจริงและข้อมูลมีค่าเป็นจริง เช่น ผู้ป่วยติดเชื้อโควิด 19 และตัวแบบทำนายเป็นติดเชื้อโควิด 19
- ผลบวกเท็จ (False Positive – FP) หรือ ข้อผิดพลาด Type-1 (Type-1 Error) คือ กรณีที่ตัวแบบทำนายผลเป็นจริงแต่ข้อมูลมีค่าเป็นเท็จ เช่น ผู้ป่วยไม่ได้ติดเชื้อโควิด 19 แต่ตัวแบบทำนายเป็นติดเชื้อโควิด 19
- ผลลบจริง (True Negative – TN) คือ กรณีที่ตัวแบบทำนายผลเป็นเท็จและข้อมูลมีค่าเป็นเท็จ เช่น ผู้ป่วยไม่ได้ติดเชื้อโควิด 19 และตัวแบบทำนายเป็นไม่ติดเชื้อโควิด 19
- ผลลบเท็จ (False Negative – FN) หรือ ข้อผิดพลาด Type-2 (Type-2 Error) คือ กรณีที่ตัวแบบทำนายผลเป็นเท็จแต่ข้อมูลมีค่าเป็นจริง เช่น ผู้ป่วยติดเชื้อโควิด 19 แต่ตัวแบบทำนายเป็นไม่ติดเชื้อโควิด 19

2.10.2 ตัวชี้วัดประสิทธิภาพ

สามารถนำค่า TP FP TN และ FP ทั้ง 4 ตัวนี้ไปคำนวณตัวชี้วัดประสิทธิภาพประกอบด้วย

- ค่าความถูกต้อง (Accuracy) คือ ผลลัพธ์ที่ตัวแบบทำนายเป็นบวกจริงเปรียบเทียบกับการทำนายทั้งหมด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

การใช้ค่าความถูกต้องในการประเมินประสิทธิภาพตัวแบบ มีข้อควรระวังสำหรับชุดข้อมูลทดสอบที่ไม่สมดุล (Imbalanced Test Dataset) เมื่อประเมินตัวแบบด้วยค่าความถูกต้องเพียงอย่างเดียวอาจจะทำให้ผู้ใช้แปลความหมายประสิทธิภาพของตัวแบบผิดไปจากความเป็นจริง ซึ่งผู้ใช้จำเป็นต้องดูค่าตัวชี้วัดประสิทธิภาพอื่นๆประกอบในการประเมินผลตัวแบบด้วย

- ค่าความแม่นยำ (Precision) คือ ผลลัพธ์ที่ตัวแบบทำนายเป็นบวกจริงเปรียบเทียบกับค่าที่ตัวแบบทำนายเป็นบวกทั้งหมด

$$Precision = \frac{TP}{TP + FP}$$

- ค่าเรียกคืน (Recall) หรือค่าความไว (Sensitivity) คือ ผลลัพธ์ที่ตัวแบบทำนายได้ถูกต้องเป็นค่าบวกเปรียบเทียบกับค่าบวกจริงทั้งหมด

$$Recall = \frac{TP}{TP + FN}$$

เนื่องจากประเภทย่อยของภัยพิบัติส่งผลกระทบต่อชีวิตและทรัพย์สินสูง ทางผู้วิจัยต้องการให้เกิดความผิดพลาดในการจำแนกประเภทรูปภาพภัยพิบัติต่ำที่สุด จึงใช้เลือกค่าการเรียกคืนในการประเมินตัวแบบร่วมกับค่าความถูกต้อง โดยการเลือกตัวแบบที่ให้ผลบวกจริง (TP) สูงที่สุด ในขณะที่ให้ผลลบเท็จ (FN) ต่ำที่สุด

- คะแนน F1 (F1 Score) คือ ค่าฮาร์โมนิกเฉลี่ยของค่าความแม่นยำกับค่าเรียกคืน

$$F1\ Score = 2x \frac{(Precision \times Recall)}{(Precision + Recall)}$$

2.11 งานวิจัยที่เกี่ยวข้อง

ศึกษางานวิจัยที่เกี่ยวข้องกับการจำแนกรูปภาพภัยพิบัติ ด้วยโครงข่ายประสาทเทียมคอนโวลูชันนอลแบบมีการถ่ายโอนการเรียนรู้ จากงานวิจัยที่มีวิธีการและผลลัพธ์ที่มีแนวโน้มจะสามารถนำมาปรับใช้กับข้อมูลของการค้นคว้าอิสระนี้

2.11.1 EmergencyNet: Efficient Aerial Image Classification for Drone-Based Emergency Monitoring Using Atrous Convolutional Feature Fusion โดย C. Kyrkou [38]

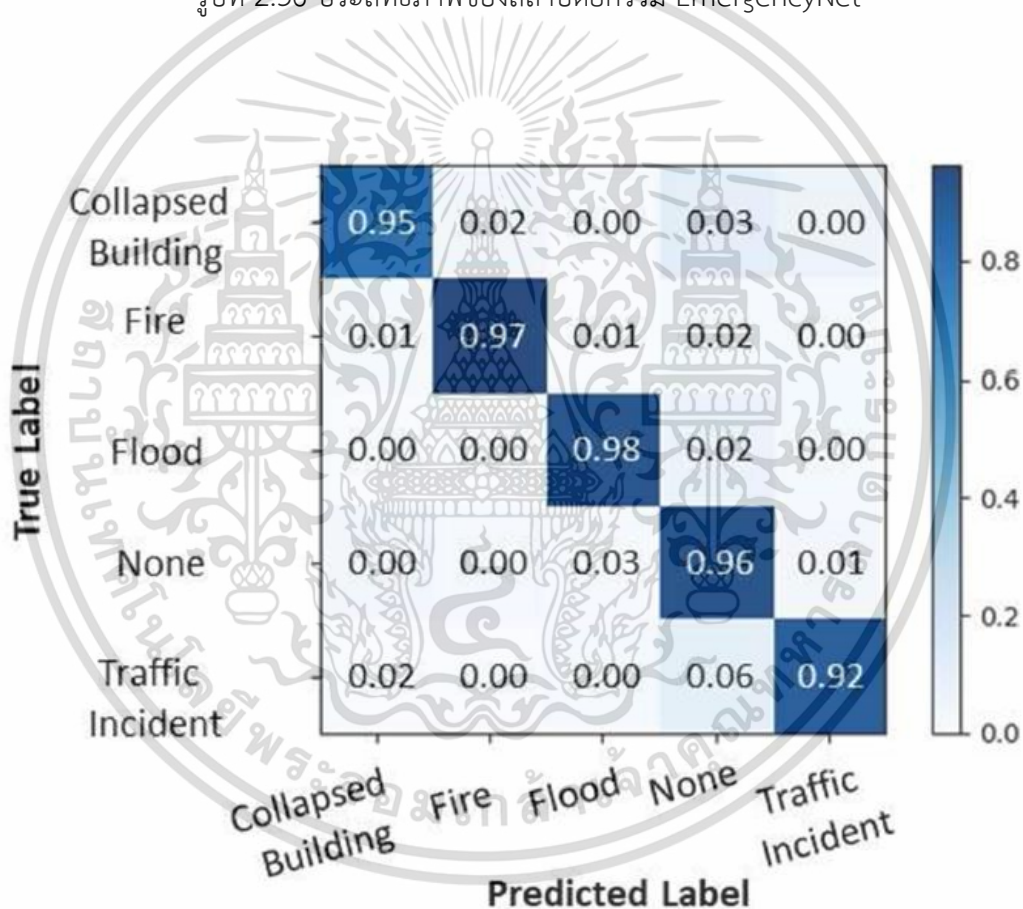
EmergencyNet เป็นสถาปัตยกรรมเครือข่ายประสาทเทียมคอนโวลูชันนอล (CNN) ที่มีขนาดเล็ก ถูกออกแบบมาสำหรับการจำแนกภาพถ่ายทางอากาศที่มีประสิทธิภาพ โดยทำงานร่วมกับแพลตฟอร์มอากาศยานไร้คนขับ (UAV) สำหรับใช้ในงานตอบสนองต่อเหตุฉุกเฉิน มีพื้นฐานจากการทำคอนโวลูชันนอลแบบกว้าง (Atrous/Dilated) ในการประมวลผลคุณลักษณะที่มีความละเอียดหลากหลายและสามารถทำงานได้อย่างมีประสิทธิภาพบนแพลตฟอร์มฝังตัวที่ใช้พลังงานต่ำ EmergencyNet มีประสิทธิภาพที่สูงขึ้นประมาณ 20 เท่า เมื่อเทียบกับตัวแบบอื่นที่เป็นที่นิยมในปัจจุบัน มีความต้องการใช้หน่วยความจำต่ำและมีความถูกต้องลดลงต่ำกว่า 1% มีการประยุกต์ต้องค์ประกอบหลักสำคัญเพิ่มเติม 2 ส่วน ประกอบด้วย

- คอนโวลูชันนอลแบบกว้าง (Atrous Convolutions) เป็นคอนโวลูชันนอลประเภทหนึ่งที่สามารถจำแนกคุณลักษณะที่มีความละเอียดหลากหลายออกจากรูปภาพได้ โดยการเพิ่มเลขศูนย์ระหว่างพิกเซลขาเข้า ซึ่งช่วยลดระยะก้าว (Stride) ของการทำคอนโวลูชันนอลได้อย่างมีประสิทธิภาพ ทำให้โครงข่ายสามารถเรียนรู้คุณลักษณะในระดับต่างๆที่แตกต่างกัน
- การรวมคุณลักษณะ (Feature Fusion) เป็นเทคนิคที่รวมคุณลักษณะจากชั้นคอนโวลูชันนอลต่างๆ โดยการรวบรวมหรือเฉลี่ยคุณสมบัติจากชั้นต่างๆ ช่วยให้โครงข่ายเรียนรู้คุณลักษณะซับซ้อนได้มากขึ้น

สถาปัตยกรรม EmergencyNet ใช้ชุดข้อมูลรูปภาพ Aerial Image Dataset for Emergency Response Applications (AIDER) ซึ่งเป็นชุดข้อมูลภาพถ่ายทางอากาศขนาดใหญ่สำหรับใช้ในงานตอบสนองเหตุฉุกเฉิน จากผลการวิจัยพบว่า EmergencyNet มีค่าคะแนน F1 (F1 Score) สูงที่ 97.5% ซึ่งใกล้เคียงกับการใช้โครงข่ายที่ผ่านการฝึกล่วงหน้ามาแล้ว โดยใช้ FLOPS และพารามิเตอร์น้อยกว่าสถาปัตยกรรมอื่น มีความต้องการใช้หน่วยความจำที่ต่ำประมาณ 360 KB และตัวแบบมีขนาดเล็กกว่า MobileNet V3 ถึง 30 เท่า

Model	Parameters	Memory (MB)	F1 Score (%)	FLOPS ($\times 10^6$)
<i>EmergencyNet</i>	90,892	0.368	95.7	57
<i>ResNet50</i>	24,113,541	96.4	96.1	4,533
<i>VGG16</i>	14,849,349	59.39	96.4	17,620
<i>Xception</i>	21,387,309	85.549	95.3	419
<i>MobileNet V1</i>	3,492,549	13.9	95	550
<i>MobileNet V2</i>	2,587,205	10.3	95.2	279
<i>MobileNet V3</i>	3,046,037	12.1	95.3	60
<i>SqueezeNet</i>	698,917	2.7	91.5	833
<i>ShuffleNet</i>	4,282,425	17.1	91.1	524
<i>EfficientNet (B0)</i>	4,378,785	17.5	96.0	420
<i>Fire_Net</i>	5,235,860	5.2	90.5	1577

รูปที่ 2.36 ประสิทธิภาพของสถาปัตยกรรม EmergencyNet



รูปที่ 2.37 Confusion Matrix จากการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม EmergencyNet

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11.2 Exploring The Role Of Deep Neural Networks For Post-Disaster Decision Support โดย N. Chaudhuri et al [39]

ศึกษาการเปรียบเทียบประสิทธิภาพโครงข่ายประสาทเทียมคอนโวลูชันนอล (CNN) ResNet-50 Inception V3 และ AlexNet แบบมีการถ่ายโอนการเรียนรู้ (Transfer Learning) กับตัวแบบการเรียนรู้ของเครื่อง (Machine Learning) SVM และ ANN ในการจำแนกประเภทรูปภาพภัยพิบัติ โครงข่ายประสาทเทียมการเรียนรู้เชิงลึก (Deep Neural Network) เป็นอัลกอริธึมการเรียนรู้ของเครื่องประเภทหนึ่งที่สามารถใช้เพื่อแยกคุณสมบัติที่ซับซ้อนออกจากข้อมูลได้ดี มีประโยชน์สำหรับใช้ประกอบการตัดสินใจหลังเกิดภัยพิบัติต่างๆ เช่น

- การประเมินความเสียหาย ใช้โครงข่ายประสาทเทียมการเรียนรู้เชิงลึกเพื่อประเมินความเสียหายที่เกิดจากภัยพิบัติ เช่น ขอบเขตความเสียหายของโครงสร้าง จำนวนอาคารที่เสียหาย และจำนวนผู้พลัดถิ่น เป็นข้อมูลเพื่อใช้จัดลำดับความสำคัญของทรัพยากรและจัดสรรความช่วยเหลือไปยังพื้นที่ที่ได้รับผลกระทบมากที่สุด
- การวางแผนการอพยพ ใช้โครงข่ายประสาทเทียมการเรียนรู้เชิงลึกเพื่อคาดการณ์ว่าผู้คนมีแนวโน้มที่จะอพยพไปที่ใดในกรณีเกิดภัยพิบัติ เป็นข้อมูลเพื่อวางแผนเส้นทางอพยพและตรวจสอบให้แน่ใจว่าผู้คนสามารถเข้าถึงที่พักพิงที่ปลอดภัยได้
- การจัดสรรทรัพยากร ใช้โครงข่ายประสาทเทียมการเรียนรู้เชิงลึกเพื่อจัดสรรทรัพยากร เช่น อาหาร น้ำ และเวชภัณฑ์ ไปยังพื้นที่ที่ต้องการมากที่สุด เป็นข้อมูลเพื่อให้แน่ใจว่าผู้คนสามารถเข้าถึงทรัพยากรที่จำเป็นในการกู้คืนจากภัยพิบัติ
- ผู้วิจัยพบว่าข้อมูลรูปภาพบางส่วนได้ที่จาก Facebook และ Twitter จากเหตุการณ์แผ่นดินไหวที่เมืองโตโฮคุ ประเทศญี่ปุ่น ปี 2011 และเหตุการณ์แผ่นดินไหวที่เมืองโบโลญา ประเทศอิตาลี ปี 2012 ที่นำมาใช้ มีรูปมนุษย์หรือโครงร่างของมนุษย์อยู่ในรูปด้วย ซึ่งอาจจะทำให้ตัวแบบมีความถูกต้องลดลง จึงแก้ปัญหานี้ขึ้นการจัดเตรียมข้อมูล โดยแบ่งข้อมูลและทำคำตอบแยกเป็น 2 กลุ่ม คลาส I มีรูปภาพมนุษย์ และคลาส II ไม่มีรูปภาพมนุษย์ ผลการทดลองพบว่าโครงข่ายประสาทเทียมการเรียนรู้เชิงลึกแบบมีการถ่ายโอนการเรียนรู้มีศักยภาพสูงกว่าการเรียนรู้ของเครื่อง โดยมีผลลัพธ์ดังแสดงในรูปที่

2.39

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Performance scores for ML and DL models on test data.

Metrics	DL			ML	
	ResNet-50	Inception V3	AlexNet	SVM	ANN
Accuracy	0.9216	0.8784	0.8054	0.7595	0.7432
Sensitivity	0.9333	0.8778	0.8056	0.7486	0.7322
Specificity	0.9105	0.8789	0.8053	0.7701	0.7540
PPV	0.9081	0.8729	0.7967	0.7611	0.7444
F1 score	0.9205	0.8753	0.8011	0.7548	0.7383
NPV	0.9351	0.8836	0.8138	0.7579	0.7421
FOR	0.0649	0.1164	0.1862	0.2421	0.2579
GM	0.9219	0.8784	0.8054	0.7593	0.7430
ROC-AUC	0.9814	0.9283	0.9148	0.8212	0.7919

รูปที่ 2.38 ประสิทธิภาพของ Deep Learning และ Machine Learning ในการจำแนกรูปภาพภัยพิบัติ

Per-class performance scores and ROC-AUC score for the DL models.

Metrics	ResNet-50		Inception-V3		AlexNet	
	Class I	Class II	Class I	Class II	Class I	Class II
Accuracy per class	0.9333	0.9105	0.8778	0.8789	0.8056	0.8053
PPV per class	0.9081	0.9351	0.8729	0.8836	0.7967	0.8138
Sensitivity per class	0.9333	0.9105	0.8778	0.8789	0.8056	0.8053
F1 score per class	0.9205	0.9227	0.8753	0.8813	0.8011	0.8095

รูปที่ 2.39 ประสิทธิภาพของ Deep Learning กับชุดข้อมูล Class I และ Class II

2.11.3 Multi-Modality Helps In Crisis Management: An Attention-Based Deep Learning Approach Of Leveraging Text For Image Classification โดย Z. Ahmad et al [40]

งานวิจัยนี้กล่าวถึงการใช้ชุดข้อมูลแบบผสมทั้งข้อความและรูปภาพในสถานการณ์ภัยพิบัติ โดยใช้ตัวแบบการเรียนรู้เชิงลึกร่วมกับข้อมูลข้อความและข้อมูลรูปภาพจากแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ (Twitter) เพื่อใช้จำแนกรูปภาพที่เกี่ยวข้องกับภัยพิบัติ ซึ่งการรวมคุณลักษณะของข้อความเข้ามามีส่วนช่วยปรับปรุงประสิทธิภาพการจำแนกประเภทได้ดียิ่งขึ้นเมื่อเทียบกับการใช้รูปภาพเพียงอย่างเดียว พบว่าตัวแบบการเรียนรู้เชิงลึกแบบให้ความสนใจ (Attention-Based) ทำงานได้ดีที่สุดสำหรับการจำแนกรูปภาพในโดเมนภัยพิบัติ ข้อมูลที่เลือกใช้ประกอบด้วย 7 ประเภทที่สำคัญ คือ ความเสียหายของโครงสร้างพื้นฐานและสาธารณูปโภค ความเสียหายของยานพาหนะ เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิจกรรมอาสาสมัครกู้ภัยหรือการบริจาค ผู้บาดเจ็บหรือเสียชีวิต บุคคลที่ได้รับผลกระทบ ผู้สูญหายหรือถูกค้นพบ และข้อมูลอื่นๆที่เกี่ยวข้อง จากความหลากหลายของข้อมูลซึ่งข้อความและรูปภาพมีคำตอบที่ระบุไว้มันตรงกัน ส่วนของรูปภาพอาจมีวัตถุปรากฏอยู่หลายชนิด และส่วนของข้อความอาจมีหลายหัวข้อที่สื่อถึง ซึ่งสร้างความสับสนในการจำแนกประเภท โดยกลไกให้ความสนใจช่วยให้ตัวแบบสกัดคุณลักษณะสำคัญเกี่ยวข้องจากข้อความและรูปภาพได้ดี ด้วยการให้ความสนใจในคำหรือภาพรวมโดยเฉพาะทั้งในข้อความและรูปภาพ ทำให้ตัวแบบเข้าใจบริบทได้ดีขึ้นและทำการทำนายได้ถูกต้องยิ่งขึ้น จากการวิจัยพบว่าการรวมคุณสมบัติที่เป็นข้อความในขณะที่จำแนกรูปภาพที่สอดคล้องกันนั้นช่วยปรับปรุงประสิทธิภาพการจำแนกประเภทได้ดีกว่าเมื่อเทียบกับการใช้รูปภาพเพียงอย่างเดียว โดยมีผลลัพธ์ของตัวแบบการเรียนรู้เชิงลึกแบบให้ความสนใจบนชุดข้อมูล CrisisMMD ซึ่งมีการระบุคำตอบไว้แล้ว ทำให้ตัวแบบมีประสิทธิภาพความถูกต้องดีขึ้นจากการใช้รูปภาพอย่างเดียวที่ 66% เป็น 73% เมื่อมีการใช้ข้อความร่วมกับรูปภาพ และเป็น 75% เมื่อใช้กลไกการให้ความสนใจร่วมด้วย

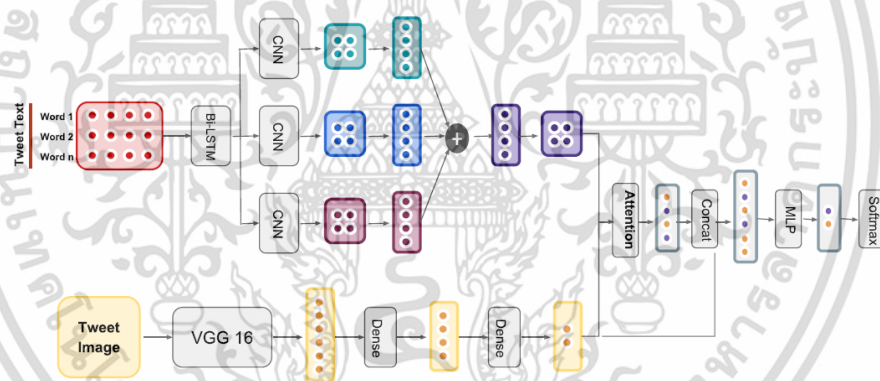


Fig. 5. Overall architecture of the proposed multi-modal (MA) attention model.

รูปที่ 2.40 สถาปัตยกรรม Attention-Based Deep Learning สำหรับชุดข้อมูล Multi-Modality

Table 2

Class-wise, macro-averaged, micro-averaged and weighted averaged F1-score for ‘feature based text classifier’, ‘text baseline’ (TB), ‘image baseline’ (IB), ‘multi-modal without attention’ (MNA), ‘InceptionV3 multi-modal’ (IM) and ‘multi-modal attention’ (MA) model: 5-Fold cross-validated.

Classes	FBT	TB	IB	MNA	IM	MA
<i>Other relevant information</i>	0.71	0.58	0.8	0.84	0.81	0.85
<i>Rescue volunteering or donation effort</i>	0.56	0.47	0.6	0.6	0.61	0.68
<i>Infrastructure and utility damage</i>	0.13	0.57	0.7	0.78	0.77	0.81
<i>Affected individuals</i>	0.03	0.08	0.12	0.29	0.33	0.31
<i>Injured or dead people</i>	0.29	0.20	0	0.24	0.06	0.34
<i>Vehicle damage</i>	0	0.06	0.008	0.45	0.45	0.45
<i>Missing or found people</i>	0	0	0	0	0	0
Macro-averaged	0.25	0.27	0.34	0.46	0.45	0.51
Micro-averaged	0.60	0.52	0.66	0.73	0.71	0.75
Weighted average	0.54	0.50	0.65	0.73	0.70	0.74

รูปที่ 2.41 ประสิทธิภาพของ Deep Learning แบบใช้และไม่ใช้ Attention-Based

2.11.4 Transfer Learning-Based Automatic Hurricane Damage Detection Using Satellite Images โดย S.Kaur et al [41]

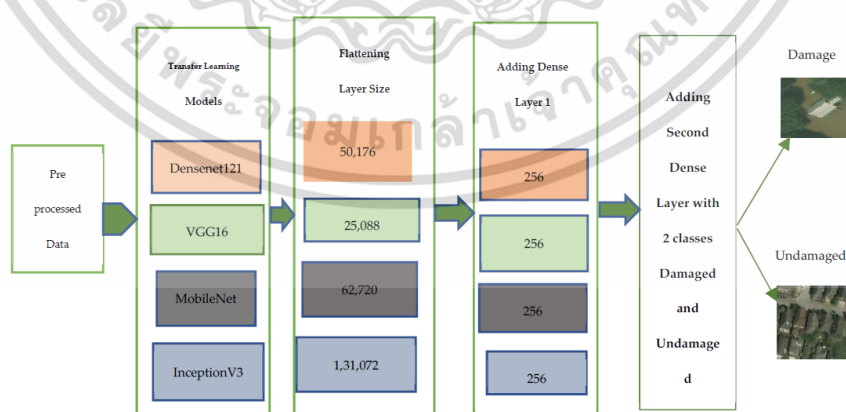
งานวิจัยนี้กล่าวถึงแนวทางการถ่ายโอนการเรียนรู้สำหรับการตรวจจับความเสียหายจากพายุเฮอริเคนโดยอัตโนมัติโดยใช้ภาพถ่ายดาวเทียม ใช้เทคนิคการจัดเตรียมข้อมูล (Preprocessing) เข้ามาช่วยปรับปรุงตัวแบบ เช่น การทำให้เป็นมาตรฐาน (Normalization) และการเพิ่มข้อมูล (Data Augmentation) เพื่อทำให้ตัวแบบทำงานได้ครอบคลุม (Generalization) และใช้เทคนิคในการปรับประสิทธิภาพตัวแบบ เช่น การปรับค่าไฮเปอร์พารามิเตอร์ (Hyper-Parameters Tuning) โดยใช้รอบการเรียนรู้ที่ 40 รอบ ขนาดแบทช์ที่ 100 อัตราการเรียนรู้ที่ 0.0001 การเพิ่มชั้นใหม่ (New Layer Adding) การเลือกตัวปรับปรุงประสิทธิภาพ (Optimizers) เช่น SGD Adadelta Adam และ RMSprop โดยใช้แบบตัวแบบการเรียนรู้เชิงลึกคอนโวลูชันนอล (CNN) ที่ผ่านการฝึกมาแล้ว DenseNet121 MobileNetV2 InceptionV3 และ VGG16 ช่วยให้ตัวแบบมีประสิทธิภาพดียิ่งขึ้นในการจำแนกประเภทความเสียหายจากข้อมูลภาพถ่ายดาวเทียม ผู้วิจัยใช้ข้อมูลภาพถ่ายดาวเทียมจากเหตุการณ์พายุเฮอริเคนฮาร์วีย์ (Hurricane Harvey) ที่เกิดขึ้นในปี 2017 มีแหล่งข้อมูลจาก FEMA และ TOMNOD โดยพบปัญหาของข้อมูลดังนี้

- คุณภาพของภาพถ่ายดาวเทียมของพื้นที่ที่ได้รับผลกระทบจากพายุเฮอริเคนอาจมีเมฆปกคลุมบางส่วนหรือทั้งหมด ทำให้ภาพเสียหายและมีสัญญาณรบกวนยากต่อการระบุและจำแนกพื้นที่ที่เสียหายอย่างถูกต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ความแปรปรวนของรูปแบบความเสียหาย ความเสียหายจากพายุเฮอริเคนอาจมีความแตกต่างกันในแง่ของความรุนแรงและประเภทของการทำลายล้าง ควรพัฒนาตัวแบบที่สามารถตรวจจับและจำแนกความเสียหายประเภทต่างๆได้อย่างมีประสิทธิภาพมีครอบคลุมความเสียหายทางโครงสร้างและรวมถึงการสะสมของเศษซากที่เกิดขึ้นจากความเสียหาย
- ข้อมูลมีการระบุคำตอบที่จำกัด เนื่องจากชุดข้อมูลภาพถ่ายดาวเทียมมีขนาดใหญ่และมีความหลากหลายแต่มีการระบุคำตอบไว้จำกัด อาจส่งผลต่อประสิทธิภาพด้านถูกต้องของตัวแบบในการสรุปและตรวจจับความเสียหายในสถานการณ์ที่ยังไม่เคยเห็นมาก่อนได้

ผลจากงานวิจัยพบว่า DenseNet121 มีค่าความถูกต้องในการฝึกสูงสุด 0.9727 และการเรียกคืนในการฝึก 0.9735 เมื่อใช้รอบการเรียนรู้ 40 รอบและอัตราการเรียนรู้ 0.0001 InceptionV3 มีค่าการตรวจสอบความถูกต้องสูงสุด 0.9670 และการเรียกคืนการตรวจสอบความถูกต้อง 0.9658 เมื่อใช้รอบการเรียนรู้ 40 รอบ DenseNet121 มีค่าการสูญเสียระหว่างการฝึกต่ำสุด 0.0666 และการตรวจสอบความสูญเสียระหว่างการฝึก 0.0956 เมื่อใช้รอบการเรียนรู้ 40 รอบ และเมื่อเปรียบเทียบกับพารามิเตอร์ของการจำแนกประเภทพบว่า VGG16 มีประสิทธิภาพโดดเด่นกว่าตัวแบบอื่นๆ โดยมีค่าความถูกต้อง 0.75 คะแนน F1 0.83 และการเรียกคืน 0.95 และเมื่อทำการปรับปรุงตัวเพิ่มประสิทธิภาพเป็น RMSprop พบว่า VGG16 ทำงานได้ดีขึ้นโดยมีความถูกต้อง 0.78



รูปที่ 2.42 สถาปัตยกรรม CNN ที่มีการเพิ่มขึ้น Dense Layer ชั้นที่ 2

Model Name	Original Transfer Learning Model				Modified Transfer Learning Model			
	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score	Accuracy
DenseNet121	0.82	0.08	0.15	0.40	0.92	0.10	0.17	0.40
MobileNetV2	0.69	0.41	0.52	0.50	0.64	0.63	0.64	0.53
InceptionV3	0.65	0.95	0.77	0.64	0.65	1.00	0.79	0.65
VGG16	0.75	0.85	0.80	0.72	0.74	0.95	0.83	0.75

รูปที่ 2.43 ประสิทธิภาพของ DenseNet121 MobileNet V2 Inception V3 และ VGG16 แบบไม่มี
และมีการเพิ่มชั้น Dense Layer ชั้นที่ 2



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

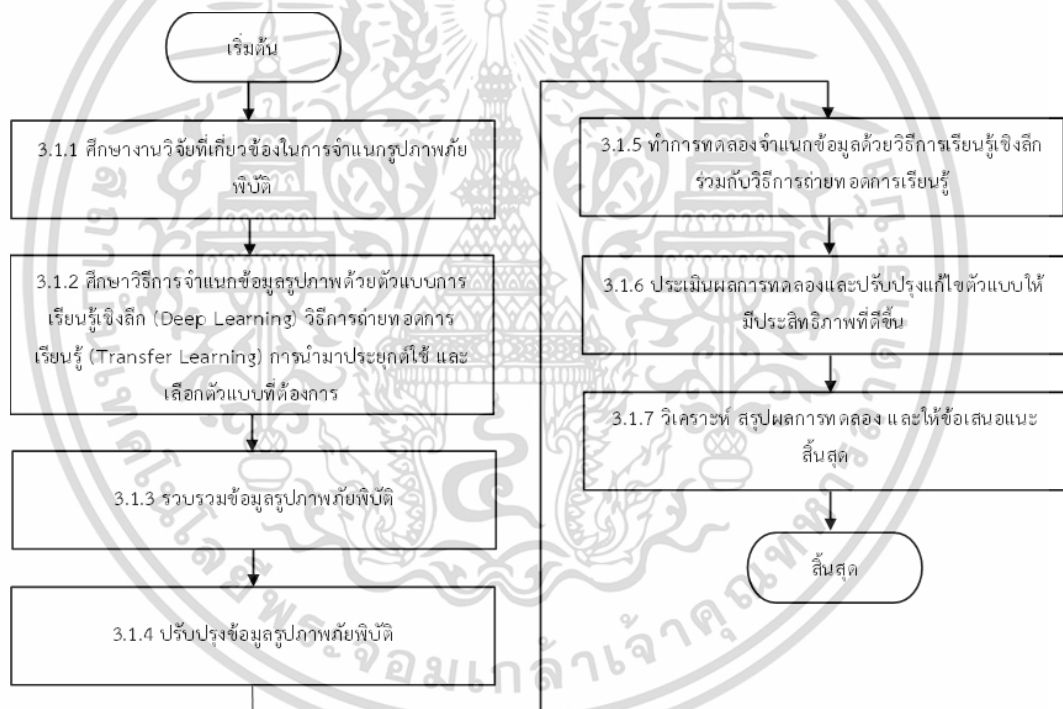
บทที่ 3

วิธีการดำเนินงานวิจัย

ในบทนี้จะกล่าวถึงระเบียบวิธีวิจัยซึ่งเป็นขั้นตอนการปฏิบัติงานทั้งหมดตั้งแต่เริ่มงานวิจัย โดยประกอบไปด้วยระเบียบวิธีวิจัย ขั้นตอนการเตรียมข้อมูล ขั้นตอนการสร้างตัวแบบจำแนกรูปภาพภัยพิบัติ การประเมินและเปรียบเทียบประสิทธิภาพตัวแบบโดยละเอียด

3.1 ระเบียบวิธีวิจัย

ขั้นตอนการศึกษาค้นคว้า การทดลอง และการประเมินผลมีลำดับการดำเนินงาน ดังแสดงในรูปที่ 3.1



รูปที่ 3.1 ระเบียบวิธีวิจัยการจำแนกรูปภาพภัยพิบัติ

3.1.1 ศึกษางานวิจัยที่เกี่ยวข้องในการจำแนกรูปภาพภัยพิบัติ

ศึกษาค้นคว้าที่เกี่ยวข้องประกอบด้วย วิธีการทำงานของโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (CNN) วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) วิธีการใช้งานตัวแบบที่ผ่านการฝึกมาแล้ว (Pre-trained Model) วิธีการจัดเตรียมข้อมูลรูปภาพ ลักษณะสำคัญของรูปภาพภัยพิบัติแต่ละประเภท วิธีการปรับปรุงข้อมูลรูปภาพ วิธีการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เลือกตัวแบบให้เหมาะสมกับประเภทข้อมูล วิธีการปรับปรุงตัวแบบ และวิธีการประเมินผลตัวแบบที่เหมาะสม

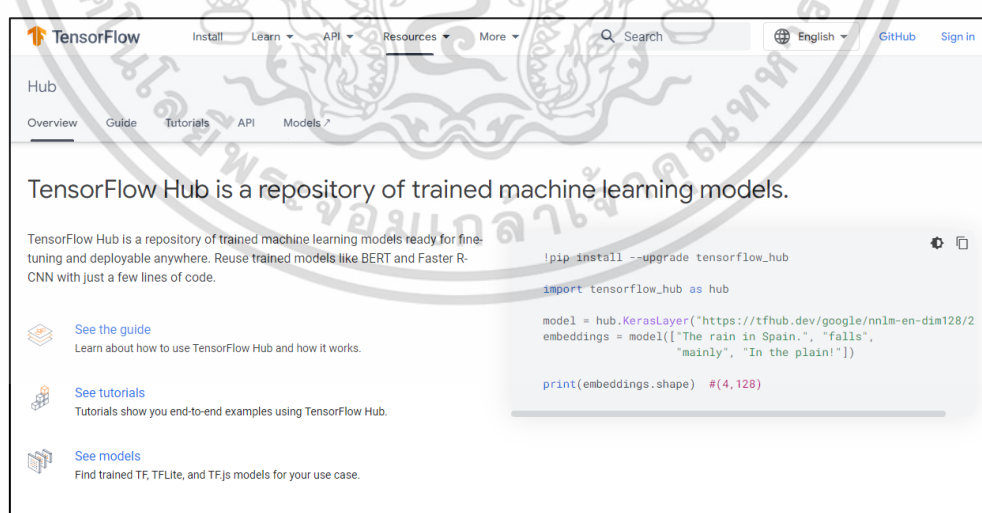
3.1.2 ศึกษาวิธีการจำแนกข้อมูลรูปภาพด้วยตัวแบบการเรียนรู้เชิงลึก (Deep Learning) วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) การนำมาประยุกต์ใช้ และเลือกตัวแบบที่ต้องการ

ศึกษาการทำงานและการประยุกต์ใช้เครื่องมือจาก TensorFlow Hub ซึ่งเป็นแหล่งเก็บรวบรวมตัวแบบการเรียนรู้ของเครื่องที่ผ่านการฝึกมาแล้ว โดยเลือกใช้ตัวแบบล่าสุดที่มีประสิทธิภาพที่โดดเด่น และเหมาะสมกับปัญหาด้านการจำแนกประเภทรูปภาพ ในงานวิจัยนี้มีการเลือกใช้ตัวแบบดังนี้

3.1.2.1 EfficientNet V2 นำเวกเตอร์คุณสมบัติของภาพจาก EfficientNet V2 กำหนดขนาดข้อมูลขาเข้าที่ 384×384 พิกเซล ตัวแบบผ่านการฝึกด้วยชุดข้อมูลรูปภาพ ImageNet-21K (Full ImageNet, Fall 2011 release) และผ่านการปรับปรุง (Fine-Tuned) ด้วยชุดข้อมูลรูปภาพ ImageNet1K

3.1.2.2 Inception V3 นำเวกเตอร์คุณสมบัติของภาพจาก Inception V3 กำหนดขนาดข้อมูลขาเข้าที่ 299×299 พิกเซล ตัวแบบผ่านการฝึกด้วยชุดข้อมูลรูปภาพ ImageNet (ILSVRC-2012-CLS)

3.1.2.3 MobileNet V3 นำเวกเตอร์คุณสมบัติของภาพจาก MobileNet V3 large กำหนดขนาดข้อมูลขาเข้าที่ 224×224 พิกเซล ตัวแบบผ่านการฝึกด้วยชุดข้อมูลรูปภาพ ImageNet (ILSVRC-2012-CLS)



รูปที่ 3.2 TensorFlow Hub

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

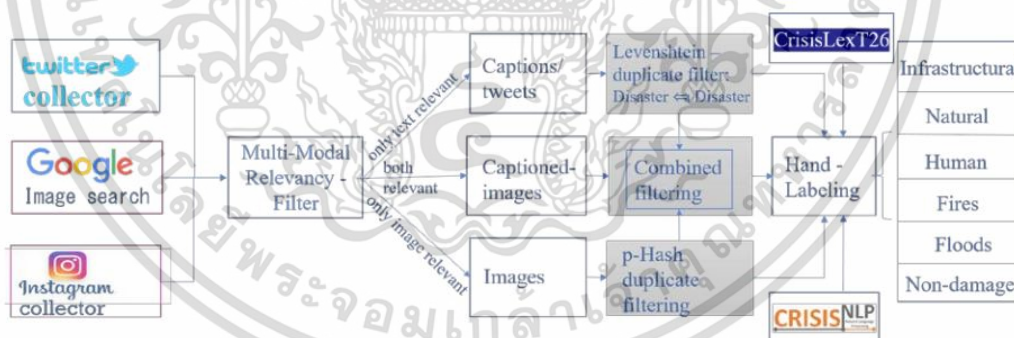
3.1.3 รวบรวมข้อมูลรูปภาพภัยพิบัติ

เก็บรวบรวมรูปภาพภัยพิบัติจากฐานข้อมูลเกี่ยวกับภัยพิบัติจาก 2 แหล่ง ประกอบด้วย

3.1.3.1 Crisis Vision Benchmark Dataset (CVB) [42]

ชุดข้อมูลเกณฑ์มาตรฐานภาพวิกฤต (Crisis Image Benchmark Dataset) ประกอบด้วยข้อมูลจากแหล่งข้อมูลหลายแหล่ง เช่น CrisisMMD AIDR และชุดข้อมูลความเสียหายแบบผสม (Damage Multimodal Dataset - DMD) โดยมีจุดประสงค์ในการรวมชุดข้อมูลเข้าด้วยกันสำหรับสร้างชุดข้อมูลสำหรับการฝึก (Train) การพัฒนา (Develop) และการทดสอบ (Test) ที่ไม่ซ้อนทับกัน เพื่อใช้เป็นเกณฑ์มาตรฐานอ้างอิงสำหรับผู้สนใจ ชุดข้อมูลนี้เหมาะสำหรับการใช้ในการตรวจจับประเภทภัยพิบัติ มีข้อมูลสำหรับการจำแนกประเภทที่เป็นประโยชน์ และมีการประเมินความรุนแรงของความเสียหายที่เกิดขึ้น โดยชุดข้อมูลทั้งหมดได้ถูกทำการระบุคำตอบของแต่ละคลาสไว้เรียบร้อยแล้ว

โดยทำการเลือกใช้เฉพาะชุดรูปภาพจากชุดข้อมูลแบบผสมรูปภาพและข้อความจากงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning [43] ซึ่งมีการจัดเก็บข้อมูลและระบุคำตอบของชุดข้อมูลตามรูปที่ 3.3



รูปที่ 3.3 ขั้นตอนการจัดเก็บข้อมูลในงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning

Table 1. Dataset Statistics

Label	Captioned Images	Text	Images without captions	Total
Fires	349	1819	856	3024
Floods	385	3685	1236	5306
Natural landscape	515	1876	882	3273
Infrastructural	1418	3547	3002	7967
Human	240	2910	1008	4158
Non-damage	2972	5194	3891	12057
Total	5879	19031	10875	35785

รูปที่ 3.4 ประเภทและจำนวนรูปภาพจากชุดข้อมูลในงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning

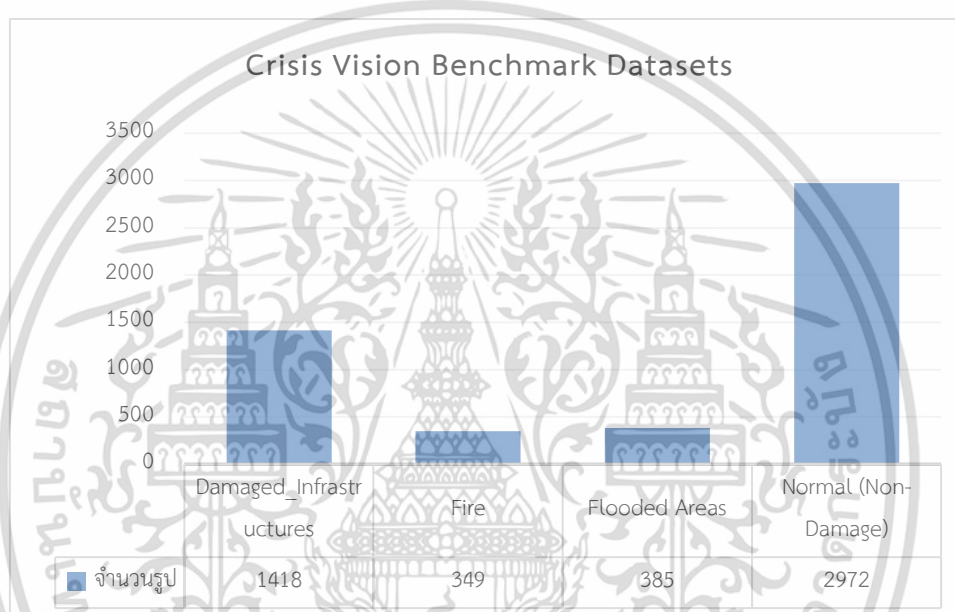


รูปที่ 3.5 ตัวอย่างรูปภาพแต่ละประเภทจากชุดข้อมูลในงานวิจัย Damage Identification in Social Media Posts using Multimodal Deep Learning

แต่เนื่องจากต้องการเปรียบเทียบประสิทธิภาพของตัวแบบกับชุดข้อมูลรูปภาพภัยพิบัติทางอากาศจาก AIDER ทางผู้วิจัยจึงพิจารณาเลือกประเภทของความเสียหายดังที่แสดงในตารางที่ 3.1

ตารางที่ 3.1 ประเภทและจำนวนรูปภาพจากชุดข้อมูล CVB ที่นำมาใช้ในงานวิจัยนี้

ประเภทความเสียหาย	คำตอบ	จำนวนรูป
สิ่งปลูกสร้าง	Damaged_Infrastructures	1,418
ไฟไหม้	Fire	349
น้ำท่วม	Flooded Areas	385
ไม่มีความเสียหาย	Normal (Non-Damage)	2,972



รูปที่ 3.6 การกระจายตัวของชุดข้อมูล CVB

3.1.3.2 Aerial Image Dataset for Emergency Response Applications (AIDER) [44]

เป็นชุดข้อมูลรูปภาพเหตุการณ์ภัยพิบัติทางอากาศจากหลายแหล่งข้อมูล เช่น Google Images Bing Images Youtube เว็บไซต์สำนักข่าว และบางส่วนเป็นภาพจากแพลตฟอร์มอากาศยานไร้คนขับ (UVA) ประกอบด้วยรูปภาพอาคารถล่ม/ซากปรักหักพัง ไฟไหม้/ควันไฟ น้ำท่วม อุบัติเหตุจราจร และเหตุการณ์ปกติ ในระหว่างการเก็บรวบรวมข้อมูลรูปภาพภัยพิบัติจะถูกเก็บบันทึกภายใต้ความละเอียดและเงื่อนไขที่แตกต่างกันทั้งด้านความสว่างและมุมมอง



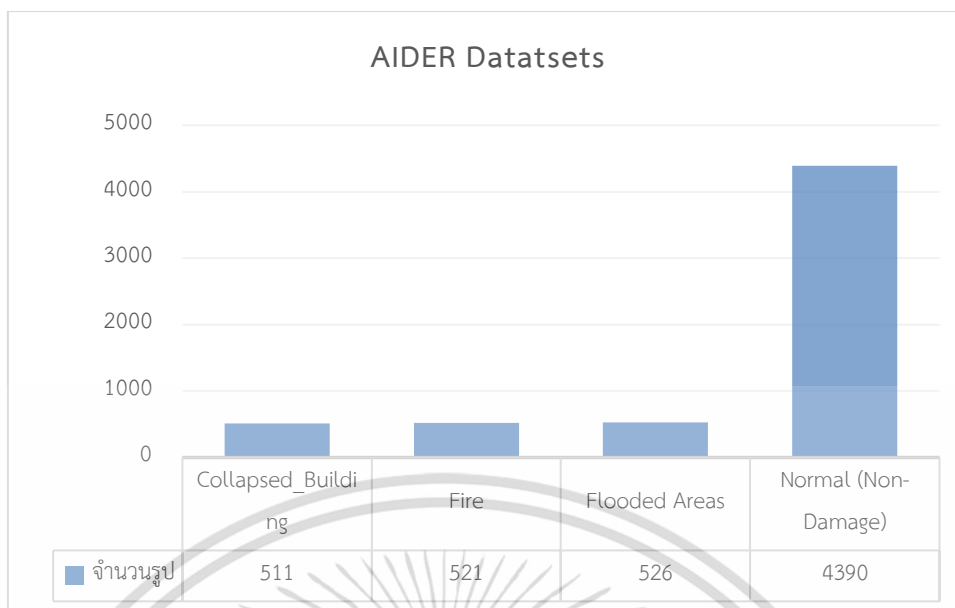
รูปที่ 3.7 ตัวอย่างรูปภาพแต่ละประเภทจากชุดข้อมูล AIDER

แต่เนื่องจากต้องการเปรียบเทียบประสิทธิภาพของตัวแบบกับชุดข้อมูลรูปภาพจาก CVB ทางผู้วิจัยจึงพิจารณาเลือกประเภทของความเสียหายดังที่แสดงในตารางที่ 3.2

ตารางที่ 3.2 ประเภทและจำนวนรูปภาพจากชุดข้อมูล AIDER ที่นำมาใช้ในงานวิจัยนี้

ประเภทความเสียหาย	คำตอบ	จำนวนรูป
สิ่งปลูกสร้าง	Collapsed_Building	511
ไฟไหม้	Fire	521
น้ำท่วม	Flooded Areas	526
ไม่มีความเสียหาย	Normal (Non-Damage)	4,390

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.8 การกระจายตัวของชุดข้อมูล AIDER

3.1.4 ปรับปรุงข้อมูลรูปภาพภัยพิบัติ (Data Preprocessing)

เนื่องจากชุดข้อมูลรูปภาพทั้ง 2 ชุดมีปัญหาความไม่สมดุล (Imbalanced Dataset) ดังแสดงในตารางที่ 3.1 และ 3.2 จึงทำการปรับปรุงข้อมูลด้วยการปรับลดจำนวนข้อมูลลงมาให้เท่ากับคลาสที่มีจำนวนน้อยที่สุดของแต่ละชุดข้อมูล เพื่อช่วยลดปัญหาความถูกต้องปฏิทรรศน์ (Accuracy Paradox) [45] โดยใช้วิธีการสุ่มเลือกแบบ Random Sampling Without Replacement [46] เพื่อย้ายไฟล์รูปภาพด้วยวิธีการสุ่มเลือกข้อมูลตามจำนวนที่ผู้ใช้กำหนดไปยังโฟลเดอร์ที่ Google Drive ก่อนนำเข้าตัวแบบ โดยใช้คำสั่ง Python บน Colab ดังนี้

3.1.4.1 นำเข้าไลบรารีที่เกี่ยวข้อง และเชื่อมต่อ Google Drive เข้ากับ Colab

```

1 import itertools
2 import os
3 import random
4 import shutil
5
6 import matplotlib.pyplot as plt
7 import numpy as np

[ ] 1 from google.colab import drive
     2 drive.mount('/content/drive')
     3 datapath = '/content/drive/MyDrive/IS_Datasets/Crisis_Vision/'
  
```

3.1.4.2 ตรวจสอบโครงสร้างของโฟลเดอร์ต้นทาง

```

3 def get_sub_folders(path):
4     all_sub_directories_list = os.popen(f"find {path} -type d").read().splitlines()
5     return all_sub_directories_list

1 #list all sub-folder
2 sub_folders_path = []
3 sub_folders_path = get_sub_folders(datapath)
4 sub_folders_path

['/content/drive/MyDrive/IS_Datasets/Crisis_Vision/',
'/content/drive/MyDrive/IS_Datasets/Crisis_Vision/normal',
'/content/drive/MyDrive/IS_Datasets/Crisis_Vision/flooded_areas',
'/content/drive/MyDrive/IS_Datasets/Crisis_Vision/fire',
'/content/drive/MyDrive/IS_Datasets/Crisis_Vision/damaged_infrastructures']

```

3.1.4.3 ทำการย้ายไฟล์จากโฟลเดอร์ต้นทางไปยังโฟลเดอร์ปลายทางที่กำหนดด้วยวิธีการสุ่มเลือกแบบ Random Sampling Without Replacement โดยระบุจำนวนไฟล์ที่ต้องการ กำหนดชื่อโฟลเดอร์ปลายทางที่ต้องการ ซึ่งโปรแกรมจะสร้างโฟลเดอร์ปลายทางขึ้นมาให้ถ้าไม่มีโฟลเดอร์นั้นอยู่ก่อนหน้า

```

3 files_list = []
4
5 for root, dirs, files in os.walk("/content/drive/MyDrive/IS_Datasets/Crisis_Vision/damaged_infrastructures"):
6     for file in files:
7         #all
8         if file.endswith(".jpg") or file.endswith(".png") or file.endswith(".jpeg"):
9             files_list.append(os.path.join(root, file))
10
11 #print images
12 #lets me count and print the amount of jpeg,jpg,png
13 file_count = len(files_list)
14 print(file_count)
15
16 # # print files_list
17 filesToCopy = random.sample(files_list, 349) #prints 500 random files from list
18
19 destPath = "/content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/damaged_infrastructures"
20
21 # # if destination dir does not exists, create it
22 if os.path.isdir(destPath) == False:
23     os.makedirs(destPath)
24
25 # # iterate over all random files and move them
26 for file in filesToCopy:
27     shutil.move(file, destPath)
28

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงสร้างโฟลเดอร์ CVB หลังจากทำการปรับลดจำนวนรูปภาพให้สมดุล มีจำนวนรูป
 คลาสละ 349 รูป

My Drive/IS_Datasets/Crisis_Vision/random_selected

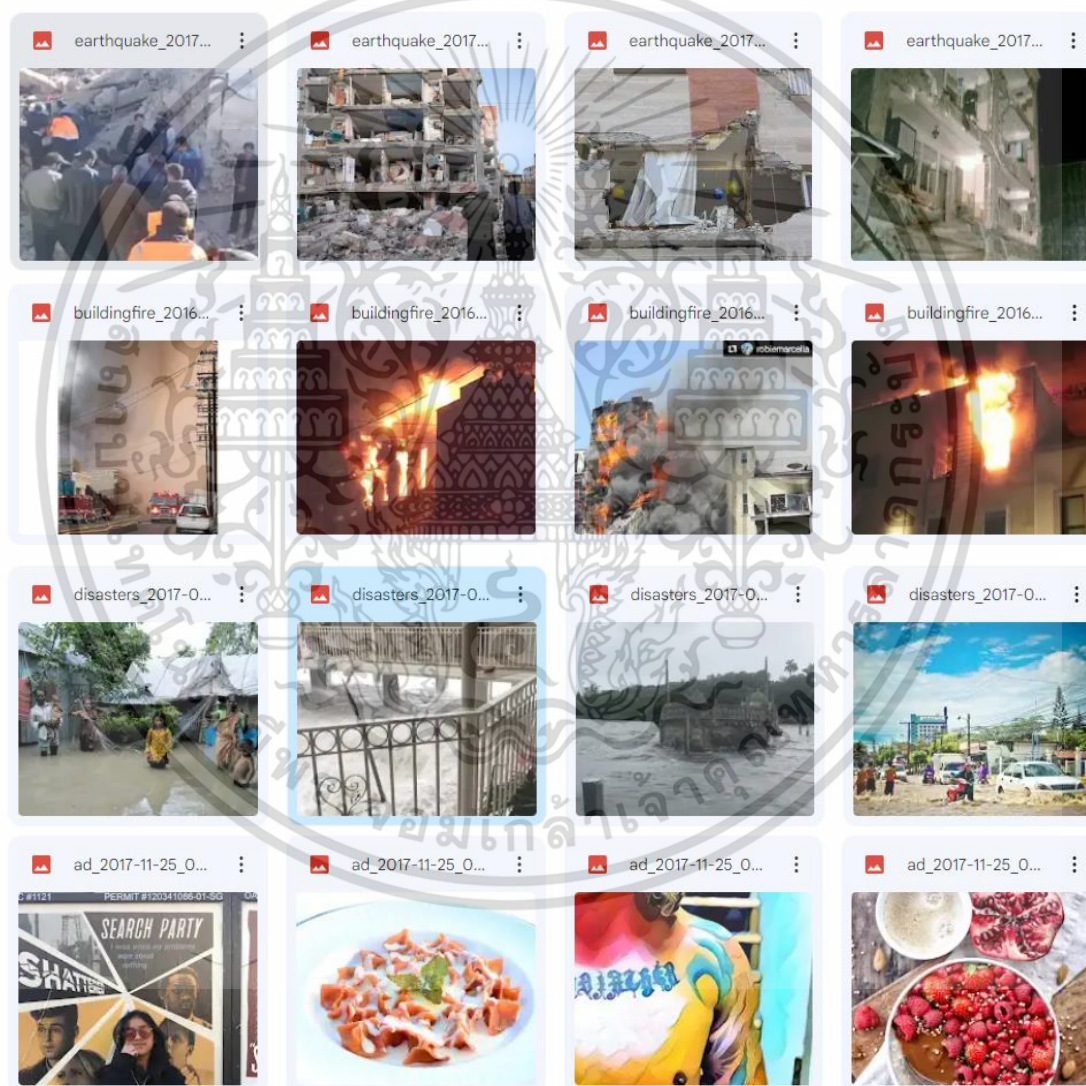
My

Drive/IS_Datasets/Crisis_Vision/random_selected/damaged_infrastructures

My Drive/IS_Datasets/Crisis_Vision/random_selected/fire

My Drive/IS_Datasets/Crisis_Vision/random_selected/flooded_areas

My Drive/IS_Datasets/Crisis_Vision/random_selected/normal



รูปที่ 3.9 ตัวอย่างภาพที่ถูกสุ่มเลือกจากชุดข้อมูล CVB

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงสร้างโพลเดอร์ AIDER หลังจากทำการปรับลดจำนวนรูปภาพให้สมดุล มีจำนวนรูป
 คลาสละ 500 รูป

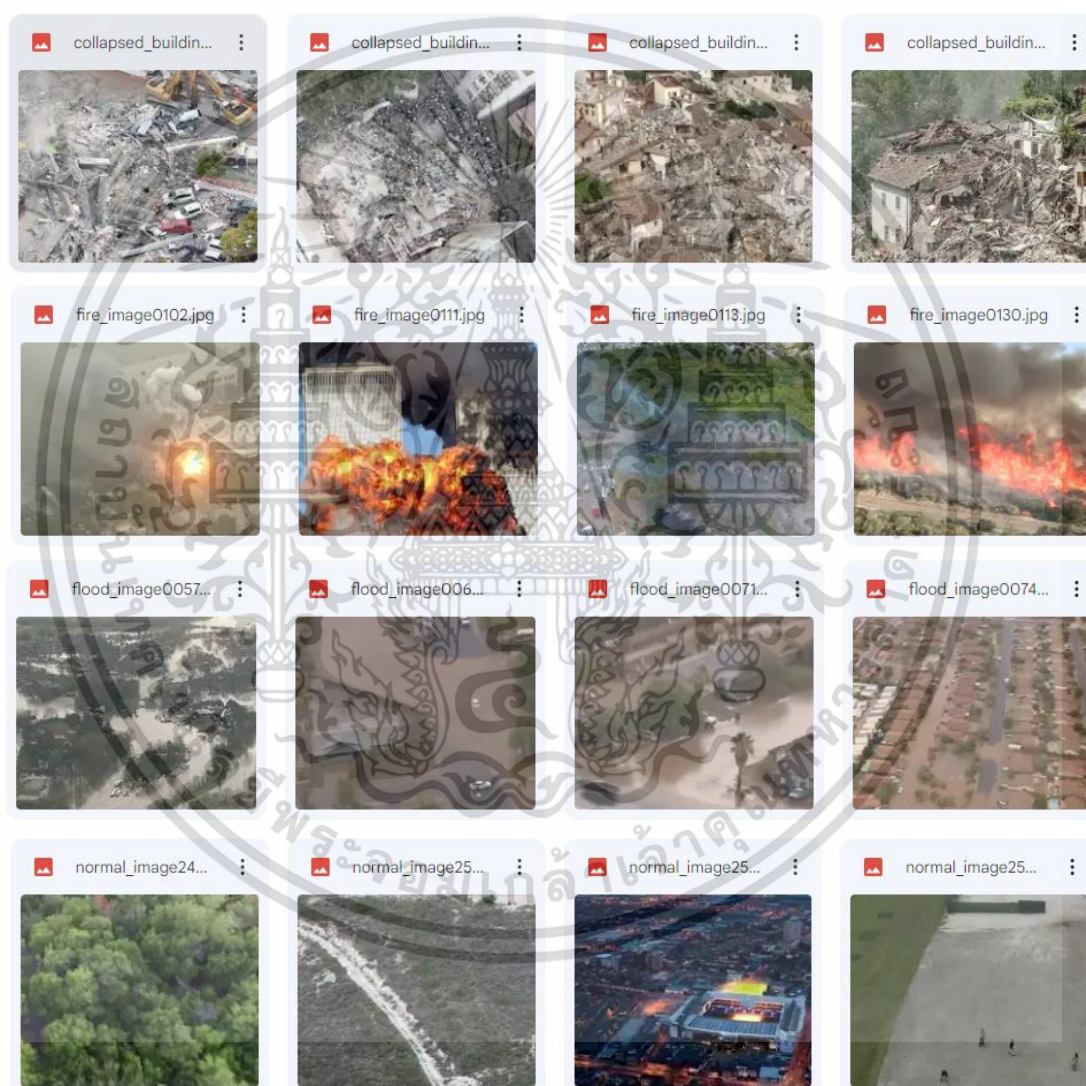
My Drive/IS_Datasets/AIDER/random_selected

My Drive/IS_Datasets/AIDER/random_selected/damaged_infrastructures

My Drive/IS_Datasets/AIDER/random_selected/fire

My Drive/IS_Datasets/AIDER/random_selected/flooded_areas

My Drive/IS_Datasets/AIDER/random_selected/normal



รูปที่ 3.10 ตัวอย่างภาพที่ถูกสุ่มเลือกจากชุดข้อมูล AIDER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5 การทดลองจำแนกข้อมูลด้วยวิธีการเรียนรู้เชิงลึกร่วมกับวิธีการถ่ายโอนการเรียนรู้

ทำการฝึกสอนตัวแบบทั้ง 3 ตัว EfficientNet V2, Inception V3 และ MobileNet V3 จาก Tensorflow Hub ด้วยชุดข้อมูลรูปภาพทั้ง 2 ชุดที่ถูกปรับความสมดุลแล้ว เพื่อประเมินประสิทธิภาพตัวแบบในขั้นการฝึกสอน โดยใช้คำสั่ง Python บน Colab ดังนี้

3.1.5.1 นำเข้าไลบรารีที่เกี่ยวข้อง และตรวจสอบสถานะการเปิดใช้งาน GPU ที่ Colab

```
1 import itertools
2 import os
3
4 import matplotlib.pyplot as plt
5 import numpy as np
6
7 import tensorflow as tf
8 import tensorflow_hub as hub
9
10 print("TF version:", tf.__version__)
11 print("Hub version:", hub.__version__)
12 print("GPU is", "available" if tf.config.list_physical_devices('GPU') else "NOT AVAILABLE")
```

TF version: 2.12.0
Hub version: 0.13.0
GPU is available

3.1.5.2 เลือกตัวแบบที่ต้องการใช้งาน model_name:

```
1 #Select the TF2 SavedModel module to use https://www.tensorflow.org
2 model_name = "efficientnetv2-s-21k-ft1k" # @param ['efficientnetv2-
3
4 model_handle_map = {
5     "efficientnetv2-s": "https://tfhub.dev/google/imagenet/efficientn
6     "efficientnetv2-m": "https://tfhub.dev/google/imagenet/efficientn
7     "efficientnetv2-l": "https://tfhub.dev/google/imagenet/efficientn
8     "efficientnetv2-s-21k": "https://tfhub.dev/google/imagenet/effici
9     "efficientnetv2-m-21k": "https://tfhub.dev/google/imagenet/effici
10    "efficientnetv2-l-21k": "https://tfhub.dev/google/imagenet/effici
11    "efficientnetv2-x1-21k": "https://tfhub.dev/google/imagenet/effic
12    "efficientnetv2-b0-21k": "https://tfhub.dev/google/imagenet/effic
13    "efficientnetv2-b1-21k": "https://tfhub.dev/google/imagenet/effic
14    "efficientnetv2-b2-21k": "https://tfhub.dev/google/imagenet/effic
15    "efficientnetv2-b3-21k": "https://tfhub.dev/google/imagenet/effic
16    "efficientnetv2-s-21k-ft1k": "https://tfhub.dev/google/imagenet/e
17    "efficientnetv2-m-21k-ft1k": "https://tfhub.dev/google/imagenet/e
18    "efficientnetv2-l-21k-ft1k": "https://tfhub.dev/google/imagenet/e
19    "efficientnetv2-x1-21k-ft1k": "https://tfhub.dev/google/imagenet/
20    "efficientnetv2-b0-21k-ft1k": "https://tfhub.dev/google/imagenet/
```

model_name: efficientnetv2-s-21k-ft1k

โดยแต่ละตัวแบบจะถูกกำหนดขนาดของข้อมูลรูปภาพเข้าไว้โดยอัตโนมัติ model_image_size_map เมื่อผู้ใช้เลือก model_name

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

57 model_image_size_map = {
58     "efficientnetv2-s": 384,
59     "efficientnetv2-m": 480,
60     "efficientnetv2-l": 480,
61     "efficientnetv2-b0": 224,
62     "efficientnetv2-b1": 240,
63     "efficientnetv2-b2": 260,
64     "efficientnetv2-b3": 300,
65     "efficientnetv2-s-21k": 384,
66     "efficientnetv2-m-21k": 480,
67     "efficientnetv2-l-21k": 480,
68     "efficientnetv2-xl-21k": 512,
69     "efficientnetv2-b0-21k": 224,
70     "efficientnetv2-b1-21k": 240,
71     "efficientnetv2-b2-21k": 260,
72     "efficientnetv2-b3-21k": 300,
73     "efficientnetv2-s-21k-ft1k": 384,
74     "efficientnetv2-m-21k-ft1k": 480,
75     "efficientnetv2-l-21k-ft1k": 480,
76     "efficientnetv2-xl-21k-ft1k": 512,
77     "efficientnetv2-b0-21k-ft1k": 224,
78     "efficientnetv2-b1-21k-ft1k": 240,
79     "efficientnetv2-b2-21k-ft1k": 260,
80     "efficientnetv2-b3-21k-ft1k": 300,
81     "efficientnet_b0": 224,
82     "efficientnet_b1": 240,
83     "efficientnet_b2": 260,
84     "efficientnet_b3": 300,
85     "efficientnet_b4": 380,
86     "efficientnet_b5": 456,
87     "efficientnet_b6": 528,
88     "efficientnet_b7": 600,
89     "inception_v3": 299,
90     "inception_resnet_v2": 299,
91     "nasnet_large": 331,
92     "pnasnet_large": 331,
93 }
94

```

3.1.5.3 เชื่อมต่อ Google Drive เข้ากับ Colab

```

[ ] 1 from google.colab import drive
2 drive.mount('/content/drive')
3
4 train_datapath = '/content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/train/'
5 val_datapath = '/content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/validation/'
6 test_datapath = '/content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/test/'
7 eval_datapath = '/content/drive/MyDrive/IS_Datasets/Evaluation_Datasets/random_selected/'

```

3.1.5.4 นำข้อมูลจาก Google Drive เข้าสู่ตัวแบบ

โดยแบ่งชุดข้อมูลออกเป็นข้อมูลฝึก (Train Dataset) 70% ข้อมูลประเมินตัวแบบ (Validation Dataset) 10% ข้อมูลทดสอบ (Test Dataset) 20% และข้อมูลทดสอบจากภายนอก (Evaluation Dataset) กำหนดให้ใช้คำตอบของชุดข้อมูล (Label) ตามชื่อโฟลเดอร์เก็บข้อมูล แปลงคำตอบของข้อมูลให้เป็นแบบ One Hot และให้โปรแกรมสับข้อมูล (Shuffle) เฉพาะชุดข้อมูลฝึก ข้อมูลประเมินตัวแบบ และข้อมูลทดสอบจากภายนอก ในขณะที่นำเข้าตัวแบบ

ตารางที่ 3.3 และ 3.4 แสดงจำนวนรูปของชุดข้อมูล CVB และ AIDER หลังทำการแก้ไขปัญหาข้อมูลไม่สมดุลและทำการแบ่งชุดข้อมูลเป็นข้อมูลฝึก 70% ข้อมูลประเมินตัวแบบ 10% และข้อมูลทดสอบ 20%

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.3 จำนวนรูปของชุดข้อมูล CVB หลังทำการแก้ไขปัญหาข้อมูลไม่สมดุลและทำการแบ่งชุดข้อมูล

ประเภทความเสียหาย	คำตอบ	จำนวนรูป	ข้อมูลฝึก (Train 70%)	ข้อมูลประเมินผล (Validation 10%)	ข้อมูลทำนาย (Test 20%)
สิ่งปลูกสร้าง	Damaged_Infrastructures	349	244	35	70
ไฟไหม้	Fire	349	244	35	70
น้ำท่วม	Flooded Areas	349	244	35	70
ไม่มีความเสียหาย	Normal (Non-Damage)	349	244	35	70

ตารางที่ 3.4 จำนวนรูปของชุดข้อมูล AIDER หลังทำการแก้ไขปัญหาข้อมูลไม่สมดุลและทำการแบ่งชุดข้อมูล

ประเภทความเสียหาย	คำตอบ	จำนวนรูป	ข้อมูลฝึก (Train 70%)	ข้อมูลประเมินผล (Validation 10%)	ข้อมูลทำนาย (Test 20%)
สิ่งปลูกสร้าง	Collapsed_Building	500	350	50	100
ไฟไหม้	Fire	500	350	50	100
น้ำท่วม	Flooded Areas	500	350	50	100
ไม่มีความเสียหาย	Normal (Non-Damage)	500	350	50	100

3.1.5.4.1 คำสั่งนำเข้าข้อมูลฝึก

```

6 seed_train_validation = 456 # Must be same for train_ds and val_ds
7 # val_split = 0.3
8
9 train_ds = tf.keras.utils.image_dataset_from_directory(
10     train_datapath,
11     labels='inferred',
12     label_mode='categorical', #int vs categorical !! need to update model loss to match
13     #class_names=None,
14     #color_mode='rgb',
15     batch_size=BATCH_SIZE,
16     image_size=IMAGE_SIZE,
17     shuffle=True,
18     seed=seed_train_validation,
19     validation_split=None,
20     subset=None,
21     #interpolation='bilinear',
22     #follow_links=False,
23     #crop_to_aspect_ratio=False,
24 )

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5.4.2 คำสั่งนำเข้าข้อมูลประเมินตัวแบบ

```
[ ] 1 val_ds = tf.keras.utils.image_dataset_from_directory(
2     val_datapath,
3     labels='inferred',
4     label_mode='categorical',
5     #class_names=None,
6     #color_mode='rgb',
7     batch_size=BATCH_SIZE,
8     image_size=IMAGE_SIZE,
9     shuffle=True,
10    seed=seed_train_validation,
11    validation_split=None,
12    subset=None,
13    #interpolation='bilinear',
14    #follow_links=False,
15    #crop_to_aspect_ratio=False,
16 )
```

3.1.5.4.3 คำสั่งนำเข้าข้อมูลทดสอบ โดยไม่มีการสับข้อมูล (Shuffle)

```
3 test_ds = tf.keras.utils.image_dataset_from_directory(
4     test_datapath,
5     labels='inferred',
6     label_mode='categorical',
7     #class_names=None,
8     #color_mode='rgb',
9     batch_size=BATCH_SIZE,
10    image_size=IMAGE_SIZE,
11    shuffle=False,
12    seed=None,
13    validation_split=None,
14    subset=None,
15    #interpolation='bilinear',
16    #follow_links=False,
17    #crop_to_aspect_ratio=False,
18 )
```

เนื่องจากการทดลองทำนายผลข้อมูลด้วยตัวแบบในรอบแรกๆ พบว่าข้อมูลที่แบ่งไว้ใช้ทดสอบประสิทธิภาพการทำนาย (Test Dataset) เกิดการสับค่าเอง (Shuffle) ตามการกำหนดการสับค่าของคำสั่งนำเข้าข้อมูลรูปภาพจากไลบรารี `tf.keras.utils.image_dataset_from_directory()` เนื่องจากตัวแปร `test_ds` จะถูกแปลงเป็นชนิดเทนเซอร์ (Tensor) และเก็บการตั้งค่าการสับข้อมูลเอาไว้ตั้งแต่ตอนที่นำข้อมูลเข้า เมื่อมีการนำเทนเซอร์ไปใช้ในการวนรอบของการทำนายทำให้ค่าจากการทำนายถูกสับไปด้วย แต่ค่าคำตอบจริง `y_true_class` ที่ถูกแยกออกมาจาก `test_ds` เป็นข้อมูลชนิดอาร์เรย์ (Array) ที่ไม่ได้ถูกสับตามการวนรอบการทำนายไปด้วย ส่งผลให้ตัวแบบมีปัญหาประสิทธิภาพความถูกต้องในการทำนายต่ำกว่า 30% ดังรูปที่ 3.11 เมื่อทำการเปรียบเทียบค่า `y_true_class` กับค่าทำนาย `y_predicted` ดังรูปที่ 3.12 ทั้งๆที่ตัวแบบมีผลของประสิทธิภาพความถูกต้องที่ดีในขั้นตอนการฝึกและการประเมินตัวแบบที่ดีดังรูปที่ 3.13

```
**** Classification Report ****
```

	precision	recall	f1-score	support
damaged_infrastructures	0.31	0.28	0.29	72
fire	0.28	0.29	0.28	72
flooded_areas	0.18	0.25	0.21	60
normal	0.30	0.22	0.26	76
accuracy			0.26	280
macro avg	0.27	0.26	0.26	280
weighted avg	0.27	0.26	0.26	280

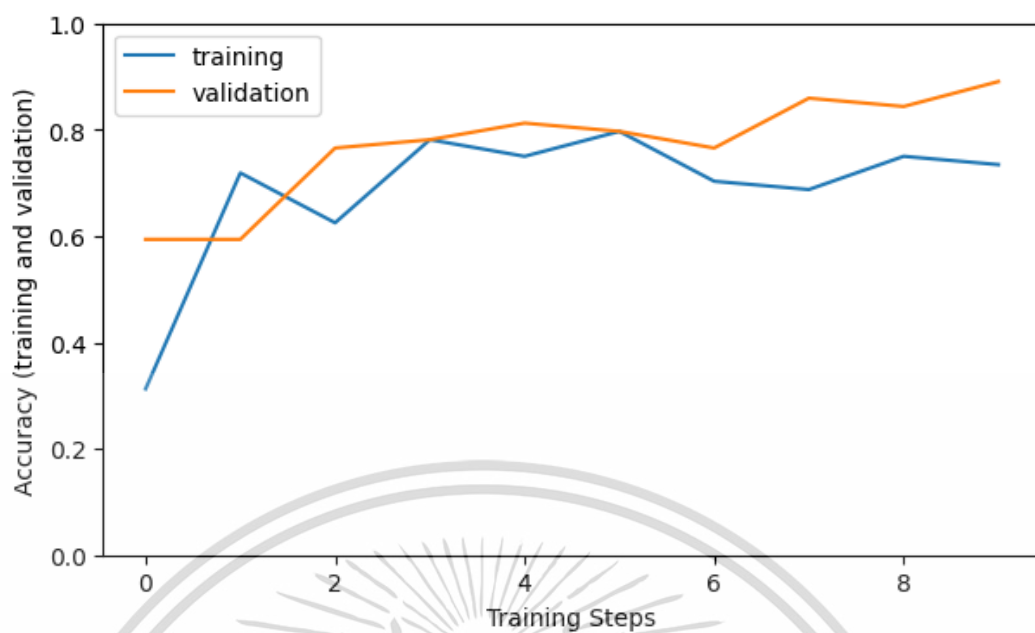
รูปที่ 3.11 รายงานประสิทธิภาพการทำนายจากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)

```
[ ] 1 y_true_class
array([2, 3, 3, 1, 3, 3, 2, 2, 3, 3, 1, 3, 0, 3, 1, 0, 3, 2, 1, 3, 3, 3,
       0, 2, 1, 0, 1, 0, 2, 3, 2, 0, 2, 3, 1, 3, 1, 0, 3, 1, 3, 0, 1, 3,
       3, 2, 3, 1, 0, 2, 0, 1, 3, 3, 1, 2, 2, 0, 1, 0, 0, 3, 3, 1, 3, 2,
       1, 1, 1, 0, 2, 3, 1, 1, 3, 2, 0, 2, 1, 3, 0, 0, 0, 2, 0, 3, 3, 1,
       1, 2, 0, 2, 2, 0, 0, 1, 2, 1, 3, 3, 1, 1, 0, 0, 1, 0, 2, 1, 2, 1,
       0, 1, 0, 3, 0, 0, 2, 0, 0, 3, 1, 3, 3, 1, 3, 3, 0, 1, 1, 2, 1, 1,
       2, 3, 0, 2, 1, 2, 0, 0, 1, 0, 1, 2, 0, 2, 2, 1, 1, 3, 3, 1, 2, 3,
       3, 1, 2, 3, 3, 3, 3, 3, 1, 1, 3, 2, 0, 0, 1, 3, 2, 0, 3, 2, 1, 0,
       1, 0, 3, 3, 2, 1, 3, 1, 1, 0, 3, 2, 0, 3, 3, 2, 1, 0, 1, 2, 0, 2,
       3, 0, 0, 0, 3, 0, 2, 3, 3, 0, 2, 2, 1, 0, 0, 3, 2, 3, 2, 2, 0, 2,
       0, 2, 1, 3, 1, 1, 1, 0, 0, 0, 2, 1, 3, 1, 2, 0, 0, 2, 0, 2, 3, 2,
       2, 3, 3, 0, 2, 0, 3, 3, 0, 1, 2, 1, 1, 1, 0, 0, 0, 3, 3, 1, 0, 0,
       3, 1, 1, 0, 3, 2, 1, 2, 0, 3, 1, 3, 1, 0, 1, 2])

[ ] 1 y_predicted
array([3, 0, 2, 3, 0, 3, 3, 0, 3, 3, 0, 0, 1, 3, 1, 0, 1, 1, 1, 2, 2, 3,
       0, 1, 0, 2, 1, 2, 3, 3, 3, 2, 1, 2, 3, 1, 2, 2, 2, 1, 0, 2, 3, 2,
       3, 1, 0, 3, 1, 0, 3, 1, 1, 1, 3, 3, 2, 2, 2, 1, 2, 1, 1, 3, 3, 1,
       2, 3, 2, 0, 2, 2, 1, 3, 2, 1, 1, 0, 1, 3, 3, 1, 0, 3, 1, 2, 0, 2,
       2, 3, 1, 1, 2, 2, 0, 1, 2, 3, 0, 2, 2, 0, 0, 0, 2, 3, 2, 1, 1, 3,
       0, 0, 2, 3, 1, 2, 0, 0, 3, 1, 0, 1, 0, 3, 1, 2, 1, 2, 1, 1, 2, 2,
       0, 1, 1, 0, 1, 1, 1, 3, 3, 1, 1, 1, 3, 0, 0, 2, 1, 3, 0, 0, 1, 3,
       0, 1, 2, 2, 3, 0, 0, 3, 1, 2, 2, 0, 0, 1, 2, 2, 0, 2, 1, 2, 2, 2,
       0, 0, 0, 1, 2, 1, 2, 3, 2, 0, 3, 3, 2, 2, 0, 2, 3, 2, 2, 2, 0, 2,
       2, 3, 0, 0, 2, 1, 2, 2, 2, 2, 0, 3, 2, 0, 1, 2, 2, 1, 0, 0, 0, 3,
       1, 3, 2, 1, 1, 3, 0, 0, 1, 0, 2, 2, 2, 1, 1, 3, 0, 2, 2, 1, 0, 1,
       0, 0, 2, 1, 3, 1, 2, 3, 1, 2, 1, 1, 1, 2, 3, 1, 1, 2, 0, 3, 2, 1,
       0, 0, 0, 2, 0, 1, 3, 3, 3, 2, 0, 3, 2, 2, 1, 1])
```

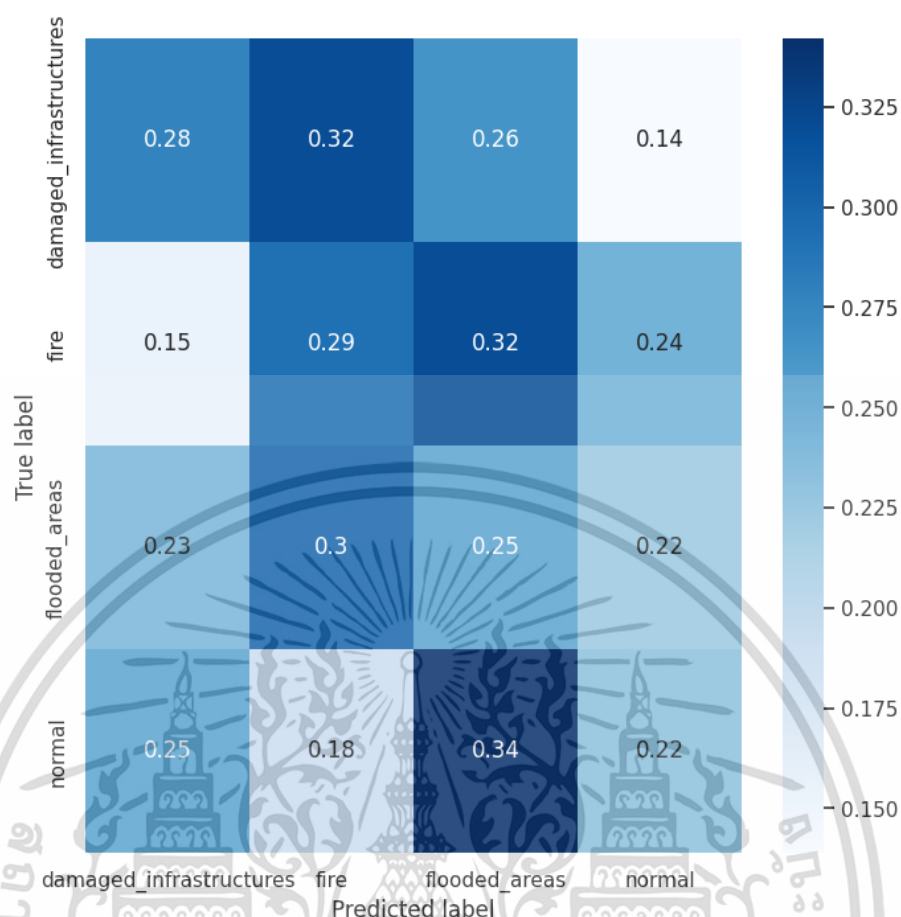
รูปที่ 3.12 ค่าคำตอบจริงเปรียบเทียบกับคำตอบจากการทำนายของตัวแบบจากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.13 ความแม่นยำของตัวแบบในขั้นการฝึกและประเมินตัวแบบจากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.14 เมทริกซ์ความสับสน (Confusion Matrix) จากปัญหาการสับข้อมูลทำนาย (Test Dataset Tensor Shuffled)

3.1.5.4.4 คำสั่งนำเข้าข้อมูลทดสอบจากภายนอก

```
3 eval_ds = tf.keras.utils.image_dataset_from_directory(
4     eval_datapath,
5     labels='inferred',
6     label_mode='categorical',
7     #class_names=None,
8     #color_mode='rgb',
9     batch_size=BATCH_SIZE,
10    image_size=IMAGE_SIZE,
11    shuffle=True,
12    seed=None,
13    validation_split=None,
14    subset=None,
15    #interpolation='bilinear',
16    #follow_links=False,
17    #crop_to_aspect_ratio=False,
18 )
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5.5 ตรวจสอบข้อมูลที่นำเข้ามา

3.1.5.5.1 ตรวจสอบคลาสของชุดข้อมูล ประกอบด้วย 4 คลาส

```
1 # class_names = train_ds.class_names
2 class_names = tuple(train_ds.class_names) #debug 04.13.2023
3 print(class_names)

('damaged_infrastructures', 'fire', 'flooded_areas', 'normal')
```

3.1.5.5.2 ตรวจสอบขนาดชุดข้อมูลฝึกที่ถูกนำเข้ามา โดยขนาดของข้อมูลขาเข้าจะมีขนาดตามค่าที่ตัวแบบกำหนดไว้ และขนาดของคำตอบซึ่งได้ถูกแปลงเป็นข้อมูลแบบ One Hot จากขั้นตอนที่ 3.1.1.12

```
[ ] 1 for image_batch, labels_batch in train_ds:
2     print(image_batch.shape)
3     print(labels_batch.shape)
4     break

(4, 384, 384, 3)
(4, 4)
```

ตารางที่ 3.5 รูปแบบ One Hot ของข้อมูลคำตอบ

คำตอบ	รูปแบบ One Hot	รูปแบบ Integer
Damaged_Infrastructures/Collapsed_Building	[1.,0.,0.,0.]	0
Fire	[0.,1.,0.,0.]	1
Flooded Areas	[0.,0.,1.,0.]	2
Normal (Non-Damage)	[0.,0.,0.,1.]	3

3.1.5.6 จัดเตรียมข้อมูล (Data Preprocessing) ก่อนนำเข้าตัวแบบ เนื่องจากชุดข้อมูลมีขนาดเล็กผู้ใช้งานจึงกำหนดขนาดของ BATCH_SIZE = 4

3.1.5.6.1 ทำการจัดเตรียมเทนเซอร์ของข้อมูลฝึกตามขนาดของ BATCH_SIZE ที่กำหนดด้วยคำสั่ง dataset.unbatch().batch(BATCH_SIZE)

3.1.5.6.2 ทำการลดความซับซ้อนของข้อมูลรูปภาพด้วยวิธี Normalization ให้อยู่ในช่วง [0, 1] ด้วยคำสั่ง tf.keras.layers.Rescaling(1. / 255) เพื่อปรับปรุงประสิทธิภาพในการประมวลผล

3.1.5.6.3 6ทำการจับคู่ (Map) ข้อมูลรูปภาพที่ผ่านการทำ Normalization แล้วเข้ากับชุดคำตอบ เพื่อทดแทนข้อมูลชุดเดิมด้วยคำสั่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
dataset.map(lambda images, labels:(normalization_layer(images),
labels))
```

```
1 train_size = train_ds.cardinality().numpy()
2 train_ds = train_ds.unbatch().batch(BATCH_SIZE)
3 train_ds = train_ds.repeat()
4
5 normalization_layer = tf.keras.layers.Rescaling(1. / 255)
6 preprocessing_model = tf.keras.Sequential([normalization_layer])
7
8 do_data_augmentation = False #former False
9 if do_data_augmentation:
10     preprocessing_model.add(
11         tf.keras.layers.RandomRotation(40))
12     preprocessing_model.add(
13         tf.keras.layers.RandomTranslation(0, 0.2))
14     preprocessing_model.add(
15         tf.keras.layers.RandomTranslation(0.2, 0))
16     # Like the old tf.keras.preprocessing.image.ImageDataGenerator(),
17     # image sizes are fixed when reading, and then a random zoom is applied.
18     # If all training inputs are larger than image_size, one could also use
19     # RandomCrop with a batch size of 1 and rebatch later.
20     preprocessing_model.add(
21         tf.keras.layers.RandomZoom(0.2, 0.2))
22     preprocessing_model.add(
23         tf.keras.layers.RandomFlip(mode="horizontal"))
24 train_ds = train_ds.map(lambda images, labels:
25                         (preprocessing_model(images), labels))
26
27 valid_size = val_ds.cardinality().numpy()
28 val_ds = val_ds.unbatch().batch(BATCH_SIZE)
29 val_ds = val_ds.map(lambda images, labels:
30                     (normalization_layer(images), labels))
31
32
[ ] 1 test_size = test_ds.cardinality().numpy()
2 test_ds = test_ds.unbatch().batch(BATCH_SIZE)
3 test_ds = test_ds.map(lambda images, labels:
4                       (normalization_layer(images), labels))
5
[ ] 1 # 05.11.2023 Added the evaluation dataset from real-world
2
3 eval_size = eval_ds.cardinality().numpy()
4 eval_ds = eval_ds.unbatch().batch(BATCH_SIZE)
5 eval_ds = eval_ds.map(lambda images, labels:
6                       (normalization_layer(images), labels))
```

3.1.5.6.4. โปรแกรมมีความสามารถในการเพิ่มข้อมูล (Data Augmentation) สำหรับการฝึก เมื่อกำหนดค่า `do_data_augmentation = True` โปรแกรมจะทำการเพิ่มขึ้น Preprocessing ในขั้นตอนการฝึกที่ตัวแบบดังนี้

- `tf.keras.layers.RandomRotation(40)` ทำการหมุนรูปภาพทวนเข็มนาฬิกาและตามเข็มนาฬิกาแบบสุ่มที่ $40 \times 2\pi$
- `tf.keras.layers.RandomTranslation(0, 0.2)` ทำการเลื่อนรูปภาพไปทางซ้ายและขวาแบบสุ่มที่ 20%
- `tf.keras.layers.RandomTranslation(0.2, 0)` ทำการเลื่อนรูปภาพขึ้นและลงแบบสุ่มที่ 20%

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- `tf.keras.layers.RandomZoom(0.2, 0.2)` ทำการขยายภาพออกทั้งแนวตั้งและแนวนอนแบบสุ่มที่ 20%
- `tf.keras.layers.RandomFlip(mode="horizontal")` ทำการกลับภาพตามแนวตั้งจากซ้ายไปขวา

3.1.5.6.5 ทำการเตรียมชุดข้อมูลทำนาย โดยแยกข้อมูลคุณลักษณะ (Feature) เป็น `test_ds_x` และข้อมูลคำตอบ (Label) เป็น `test_ds_y`

```
4 test_ds_x = np.concatenate([x for x, y in test_ds], axis=0)
5 test_ds_y = np.concatenate([y for x, y in test_ds], axis=0)
```

และทำการแปลงข้อมูลคำตอบ `test_ds_y` จาก One Hot กลับเป็นคลาสในรูปแบบ Integer [0, 3]

```
1 test_ds_y_true = np.argmax(test_ds_y, axis=1)

[ ] 1 test_ds_y_true

array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3])
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5.7 ทำ Data Visualization ของชุดข้อมูลฝึก เพื่อตรวจสอบความถูกต้องของข้อมูลคุณลักษณะกับข้อมูลคำตอบ

```

3 import matplotlib.pyplot as plt
4
5 plt.figure(figsize=(8, 8))
6 for images, labels in train_ds.take(1):
7     for i in range(4):
8         ax = plt.subplot(3, 3, i + 1)
9         plt.imshow(images[i].numpy())
10        plt.title(np.argmax(labels[i]))
11        plt.axis("off")
12
13 # collapsed_building [1.,0.,0.,0.] = 0, fire [0.,1.,0.,0.] = 1, flooded_area [0.,0.,1.,0.]
14 # test_ds_take is shuffled from util import setting

```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5.8 สร้างตัวแบบจากการรวมชั้นต่างๆเข้าด้วยกัน

```

3 do_fine_tuning = False #Former = False
4
5 print("Building model with", model_handle)
6 model = tf.keras.Sequential([
7     # Explicitly define the input shape so the model can be properly
8     # loaded by the TFLiteConverter
9     tf.keras.layers.InputLayer(input_shape=IMAGE_SIZE + (3,)),
10    hub.KerasLayer(model_handle, trainable=do_fine_tuning),
11    tf.keras.layers.Dropout(rate=0.2),
12    tf.keras.layers.Dense(len(class_names), kernel_regularizer=tf.keras.regularizers.l2(0.0001))
13 ])
14 model.build((None,)+IMAGE_SIZE+(3,))
15 model.name = model_name #rename the model for debugging https://github.com/keras-team/keras/issues/6194
16 model.summary()

```

Building model with https://tfhub.dev/google/imagenet/efficientnet_v2_imagenet21k_ft1k_s/feature_vector/2
Model: "efficientnetv2-s-21k-ft1k"

Layer (type)	Output Shape	Param #
keras_layer (KerasLayer)	(None, 1280)	20331360
dropout (Dropout)	(None, 1280)	0
dense (Dense)	(None, 4)	5124

=====
Total params: 20,336,484
Trainable params: 5,124
Non-trainable params: 20,331,360

3.1.5.8.1 ชั้น Input (Input Layer) ใช้คำสั่ง `tf.keras.layers.InputLayer` (`input_shape=IMAGE_SIZE + (3,)`) โดยกำหนดขนาดของ `input_shape` = ขนาด `IMAGE_SIZE` ซึ่งถูกกำหนดไว้จากขั้นตอนที่ 2 เมื่อผู้ใช้เลือกตัวแบบ และ `+ (3,)` คือค่าช่องสัญญาณสี RGB

3.1.5.8.2 ชั้นตัวแบบที่ผ่านการฝึก (Pre-trained Model Layer) .ใช้สถาปัตยกรรมตัวแบบที่ผ่านการฝึกแล้วจาก Tensorflow Hub ใช้คำสั่ง `hub.KerasLayer(model_handle, trainable=do_fine_tuning)` ซึ่งตัวแบบ `model_handle` ที่นำมาใช้ถูกเลือกไว้จากขั้นตอนที่ 3.1.5.2 แล้ว โดยโปรแกรมมีความสามารถในการฝึกตัวแบบด้วยชุดข้อมูลจากผู้ใช้งาน เมื่อกำหนดค่า `do_fine_tune = True`

3.1.5.8.3 ชั้นการละทิ้ง (Dropout Layer) ใช้คำสั่ง `tf.keras.layers.Dropout(rate=0.2)` เพื่อทำการละทิ้งข้อมูลขาเข้าแบบสุ่มด้วยการทำให้มีค่าเป็น 0 ที่ 20%

3.1.5.8.4 ชั้น Dense (Dense Layer) ใช้คำสั่ง `tf.keras.layers.Dense` (`len(class_names), kernel_regularizer=tf.keras.regularizers.l2(0.0001)`) โดยกำหนดคลาสตาม `class_names` และ ใช้เทอมการลงโทษเรกูลาไรเซอร์ L2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่ 0.0001 ซึ่ง Tensorflow ใช้ค่านี้ในการคำนวณฟังก์ชันการสูญเสียด้วยสมการ

$$loss = l2 \times reduce_sum(square(x))$$

3.1.5.8.5 สร้างตัวแบบใช้คำสั่ง `model.build((None,)+IMAGE_SIZE+(3,))`

เพื่อรวมชั้นต่างๆเข้าด้วยกันตามลำดับ ค่าพารามิเตอร์ (None) คือ ไม่ตั้งชื่อตัวแบบ และขนาดข้อมูลขาเข้า +IMAGE_SIZE+(3,) เช่นเดียวกับข้อ 3.1.5.8.1

3.1.5.8.6 ทำการสรุปและแสดงโครงสร้างของตัวแบบที่ถูกสร้างขึ้นแบบ Sequential ใช้คำสั่ง `model.summary()`

3.1.5.9 การฝึกตัวแบบ

```
[ ] 1 from time import time

1 #Training the model
2 model.compile(
3     optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
4     loss=tf.keras.losses.CategoricalCrossentropy(from_logits=True, label_smoothing=0.1),
5     metrics=[tf.keras.metrics.CategoricalAccuracy()]
6 )
7
8 steps_per_epoch = train_size // BATCH_SIZE
9 validation_steps = valid_size // BATCH_SIZE
10
11
12 start = time()
13
14 callback = tf.keras.callbacks.EarlyStopping(monitor='categorical_accuracy', patience=3, restore_best_weights=True)
15 checkpoint_path = "/content/drive/MyDrive/IS Datasets/Cmis Vision/random_selected/efficientnetv2/cp.ckpt"
16 checkpoint_dir = os.path.dirname(checkpoint_path)
17 cp_callback = tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
18                                                  save_weights_only=True,
19                                                  verbose=1)
20
21 hist = model.fit(
22     train_ds,
23     epochs=10, #former 10
24     steps_per_epoch=steps_per_epoch,
25     validation_data=val_ds,
26     validation_steps=validation_steps,
27     callbacks=[callback, cp_callback]).history
28
29 print("training time is ", time()-start)
```

3.1.5.9.1 เนื่องจากผู้วิจัยต้องการประเมินระยะเวลาที่ในการฝึก จึงใช้ไลบรารี time และคำสั่ง time() เข้ามาช่วยในการจับเวลาการฝึกของแต่ละตัวแบบ โดยคำนวณเวลาการฝึกจากคำสั่ง time()-start

3.1.5.9.2 รวบรวมตัวแบบและกำหนดค่าไฮเปอร์พารามิเตอร์ ใช้คำสั่ง `model.compile()` เลือกใช้ตัวปรับปรุงประสิทธิภาพ Adam อัตราการเรียนรู้ 0.001 ฟังก์ชันการสูญเสียแบบ CategoricalCrossentropy โดยเลือกการแปลงค่าหน้าจะเป็น [0, 1] ไปเป็นค่าจำนวนจริง R (-inf, +inf) โดยกำหนดค่า `from_logits = True` อธิบายด้วยสมการ $L = \ln \frac{p}{1-p}$ และ `label_smoothing=0.1` เพื่อช่วยให้ปรับค่าความมั่นใจของคำตอบให้กว้างขึ้น (The Confidence On Label Values Are Relaxed) ตัวอย่างเช่น เมื่อเลือก `label_smoothing=0.1` จะใช้ค่า $0.1/\text{num_classes} = 0.1/4 = 0.025$ สำหรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Non-Target Labels และใช้ค่า $(0.9+0.1)/\text{num_classes} = 1/4 = 0.25$ สำหรับ Target Labels

3.1.5.9.3 การหยุดฝึกเมื่อค่า categorical_accuracy ไม่มีการเปลี่ยนแปลงไปจากเดิม ใช้คำสั่ง `tf.keras.callbacks.EarlyStopping(monitor = 'categorical_accuracy', patience = 3, restore_best_weights = True)`

3.1.5.9.4 สร้างจุด Check Point เพื่อบันทึกค่าน้ำหนักของการฝึกในแต่ละรอบ ใช้คำสั่ง `tf.keras.callbacks.ModelCheckpoint(filepath = checkpoint_path, save_weights_only = True, verbose = 1)`

3.1.5.9.5 เริ่มการฝึกตัวแบบ ใช้คำสั่ง `model.fit(train_ds, epochs = 10, steps_per_epoch = steps_per_epoch, validation_data = val_ds, validation_steps = validation_steps, callbacks = [callback, cp_callback]).history` กำหนดรอบการฝึก epochs = 10 จำนวนข้อมูลฝึกต่อรอบการฝึก steps_per_epoch = train_size // BATCH_SIZE และจำนวนข้อมูลประเมินตัวแบบต่อรอบการฝึก validation_steps = valid_size // BATCH_SIZE

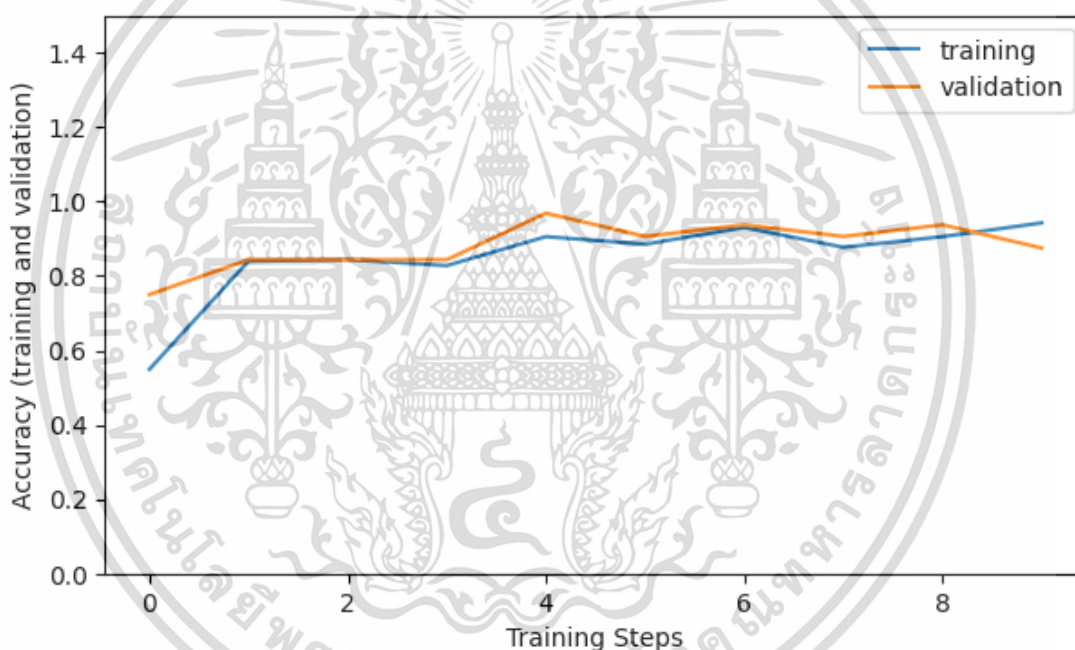
```
Epoch 1/10
61/61 [=====] - ETA: 0s - loss: 1.2614 - categorical_accuracy: 0.5492
Epoch 1: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 97s 15/step - loss: 1.2614 - categorical_accuracy: 0.5492 - val_loss: 0.8534 - val_categorical_accuracy: 0.7500
Epoch 2/10
61/61 [=====] - ETA: 0s - loss: 0.7669 - categorical_accuracy: 0.8402
Epoch 2: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 51s 848ms/step - loss: 0.7669 - categorical_accuracy: 0.8402 - val_loss: 0.7088 - val_categorical_accuracy: 0.8438
Epoch 3/10
61/61 [=====] - ETA: 0s - loss: 0.7220 - categorical_accuracy: 0.8443
Epoch 3: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 49s 819ms/step - loss: 0.7220 - categorical_accuracy: 0.8443 - val_loss: 0.7160 - val_categorical_accuracy: 0.8438
Epoch 4/10
61/61 [=====] - ETA: 0s - loss: 0.7430 - categorical_accuracy: 0.8279
Epoch 4: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 44s 729ms/step - loss: 0.7430 - categorical_accuracy: 0.8279 - val_loss: 0.7044 - val_categorical_accuracy: 0.8438
Epoch 5/10
61/61 [=====] - ETA: 0s - loss: 0.6126 - categorical_accuracy: 0.9057
Epoch 5: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 75 113ms/step - loss: 0.6126 - categorical_accuracy: 0.9057 - val_loss: 0.4936 - val_categorical_accuracy: 0.9688
Epoch 6/10
61/61 [=====] - ETA: 0s - loss: 0.6099 - categorical_accuracy: 0.8852
Epoch 6: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 9s 141ms/step - loss: 0.6099 - categorical_accuracy: 0.8852 - val_loss: 0.6451 - val_categorical_accuracy: 0.9062
Epoch 7/10
61/61 [=====] - ETA: 0s - loss: 0.5896 - categorical_accuracy: 0.9303
Epoch 7: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 7s 123ms/step - loss: 0.5896 - categorical_accuracy: 0.9303 - val_loss: 0.6505 - val_categorical_accuracy: 0.9375
Epoch 8/10
61/61 [=====] - ETA: 0s - loss: 0.6255 - categorical_accuracy: 0.8770
Epoch 8: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 8s 125ms/step - loss: 0.6255 - categorical_accuracy: 0.8770 - val_loss: 0.6643 - val_categorical_accuracy: 0.9062
Epoch 9/10
61/61 [=====] - ETA: 0s - loss: 0.5777 - categorical_accuracy: 0.9057
Epoch 9: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 8s 129ms/step - loss: 0.5777 - categorical_accuracy: 0.9057 - val_loss: 0.5267 - val_categorical_accuracy: 0.9375
Epoch 10/10
61/61 [=====] - ETA: 0s - loss: 0.5506 - categorical_accuracy: 0.9426
Epoch 10: saving model to /content/drive/MyDrive/IS_Datasets/Crisis_Vision/random_selected/efficientnetv2/cp.ckpt
61/61 [=====] - 7s 116ms/step - loss: 0.5506 - categorical_accuracy: 0.9426 - val_loss: 0.7438 - val_categorical_accuracy: 0.8750
training time is 285.6790884838867
```

รูปที่ 3.15 ผลการฝึกตัวแบบ EfficientNet V2 ด้วยชุดข้อมูล CVB ที่ Epoch = 10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

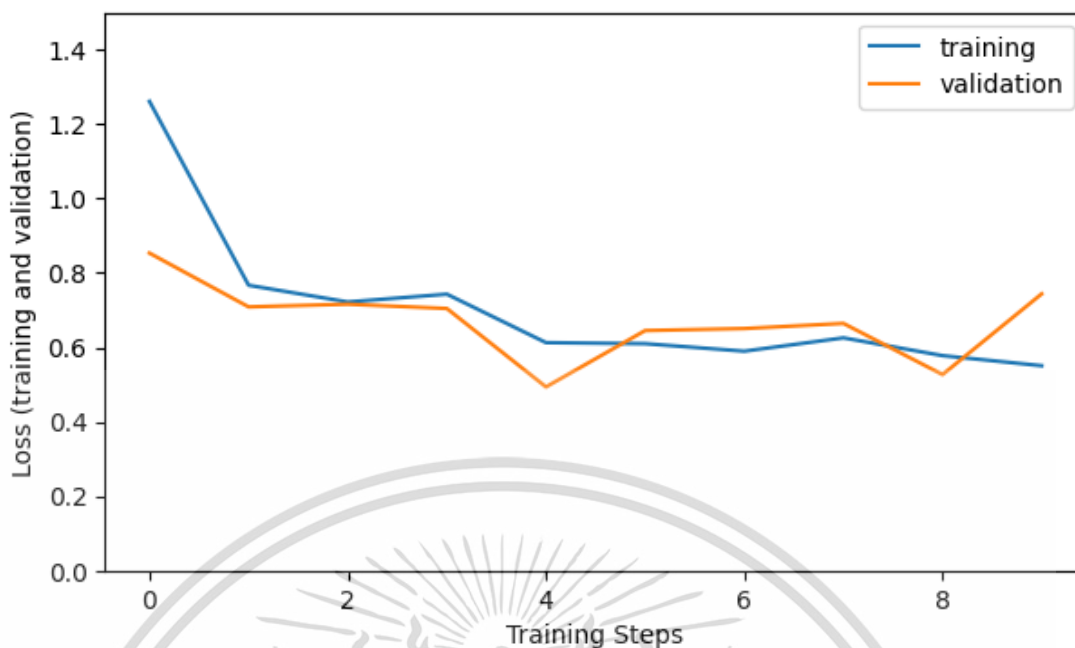
3.1.5.10 ทำการสร้างกราฟของค่าความถูกต้องและค่าการสูญเสียในการฝึกตัวแบบด้วยชุดข้อมูล train_ds และ val_ds โดยใช้คำสั่ง plt.plot(hist[“ค่า accuracy หรือ loss จาก model.fit”]) เพื่อเปรียบเทียบค่าความถูกต้องและค่าการสูญเสียของตัวแบบ

```
1 plt.figure(figsize=(7, 4))
2 plt.ylabel("Loss (training and validation)")
3 plt.xlabel("Training Steps")
4 plt.ylim([0,1.5])
5 plt.plot(hist["loss"])
6 plt.plot(hist["val_loss"])
7 plt.legend(["training", "validation"])
8
9 plt.figure(figsize=(7, 4))
10 plt.ylabel("Accuracy (training and validation)")
11 plt.xlabel("Training Steps")
12 plt.ylim([0,1.5])
13 plt.plot(hist["categorical_accuracy"])
14 plt.plot(hist["val_categorical_accuracy"])
15 plt.legend(["training", "validation"])
```



รูปที่ 3.16 ตัวอย่างกราฟค่าความถูกต้อง Training และ Validation ของตัวแบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.17 ตัวอย่างกราฟค่าการสูญเสีย Training และ Validation ของตัวแบบ

3.1.5.11 ใช้ตัวแบบทำนายผลด้วยชุดข้อมูล `test_ds` ด้วยคำสั่ง `model.predict(test_ds_x)` โดยให้ผลการทำนายที่ตัวแปร `predictions` ทำการแปลงผลจากค่าความน่าจะเป็นกลับไปเป็นค่าคลาส `[0, 3]` ซึ่งเป็นจำนวนเต็มด้วยคำสั่ง `np.argmax(predictions, axis=1)`

```
[ ] 1 print("Generate predictions for test_ds")
2 predictions = model.predict(test_ds_x)
3 print("predictions shape:", predictions.shape)

Generate predictions for test_ds
9/9 [=====] - 8s 500ms/step
predictions shape: (280, 4)

[ ] 1 predictions

array([[ 3.7912016, -1.7847437, -3.1749275, -1.7608896 ],
       [ 2.5987573, -3.4056306, -0.7626961, -4.449318 ],
       [ 1.1617513, -2.162711, -0.25753406, -1.9660352 ],
       ...,
       [-1.2189505, -1.8749455,  0.197976,  0.9107158 ],
       [ 0.23585676, -0.37106812,  1.0228629,  0.27591676],
       [-0.22371282, -0.67787546, -1.841388, -1.8770503 ]],
      dtype=float32)

[ ] 1 y_predicted = np.argmax(predictions, axis=1)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

3 from sklearn.metrics import confusion_matrix, classification_report
4 from matplotlib import pyplot as plt
5 import seaborn as sns
6
7 def plot_confusion_matrix(y_test, y_scores, classNames):
8     # y_test=np.argmax(y_test, axis=1)
9     # y_scores=np.argmax(y_scores, axis=1)
10    classes = len(classNames)
11    cm = confusion_matrix(y_test, y_scores)
12    print("**** Confusion Matrix ****")
13    print(cm)
14    print("**** Classification Report ****")
15    print(classification_report(y_test, y_scores, target_names=classNames))
16    con = np.zeros((classes, classes))
17    for x in range(classes):
18        for y in range(classes):
19            con[x,y] = cm[x,y]/np.sum(cm[x,:])
20
21    plt.figure(figsize=(8,8))
22    sns.set(font_scale=1.0) # for label size
23    df = sns.heatmap(con, annot=True, fmt='.2', cmap='Blues', xticklabels= classNames, yticklabels= classNames)
24    df.figure.savefig("image2.png")
25    plt.ylabel('True label')
26    plt.xlabel('Predicted label')
27
28 # classNames = ['collapsed_building', 'fire', 'flooded_areas', 'normal']
29 classNames = class_names
30 plot_confusion_matrix(test_ds_y_true, y_predicted, classNames)
31 #y_test is your ground truth
32 #y_scores is your predicted probabilities

```

3.1.5.13.1 สร้างรายงานการจำแนกประเภทรูปภาพภัยพิบัติของตัวแบบ EfficientNet V2 ด้วยชุดข้อมูล CVB

```

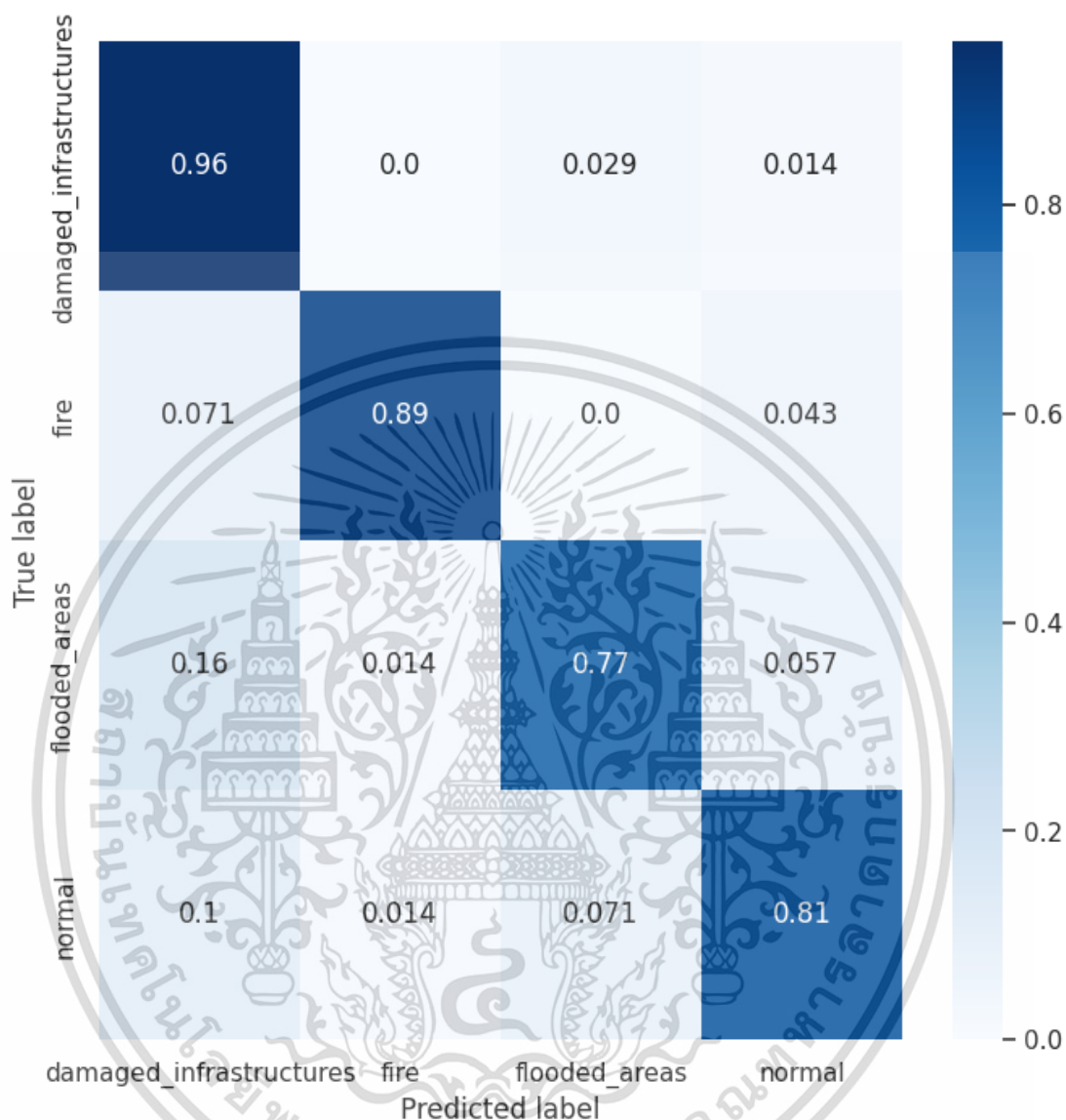
**** Classification Report ****

```

	precision	recall	f1-score	support
damaged_infrastructures	0.74	0.96	0.84	70
fire	0.97	0.89	0.93	70
flooded_areas	0.89	0.77	0.82	70
normal	0.88	0.81	0.84	70
accuracy			0.86	280
macro avg	0.87	0.86	0.86	280
weighted avg	0.87	0.86	0.86	280

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.5.13.2 สร้างเมทริกซ์ความสับสนของการจำแนกประเภทรูปภาพภัยพิบัติของตัวแบบ EfficientNet V2 ด้วยชุดข้อมูล CVB



3.1.5.14 ทดสอบประสิทธิภาพตัวแบบด้วยชุดข้อมูล eval_ds จาก CVB ซึ่งได้จากแพลตฟอร์มทวิตเตอร์ (Twitter) และเป็นชุดข้อมูลที่ตัวแบบไม่เคยเห็นมาก่อน ใช้คำสั่ง `model.evaluation(eval_ds)`

```
1 # #Model evaluation on eval_ds
2 results = model.evaluate(eval_ds)

50/50 [=====] - 52s 847ms/step - loss: 0.6425 - categorical_accuracy: 0.8750
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

... > Evaluation_Datasets > random_selected ▾

Name	↑	Owner
 damaged_infrastructures		 me
 fire		 me
 flooded_areas		 me
 normal		 me

รูปที่ 3.18 โครงสร้างโฟลเดอร์ของชุดข้อมูล eval_ds

ตารางที่ 3.6 ประเภทและจำนวนรูปของชุดข้อมูล eval_ds

ประเภทความเสียหาย	คำตอบ	จำนวน รูป
สิ่งปลูกสร้าง	Damaged Infrastructures	50
ไฟไหม้	Fire	50
น้ำท่วม	Flooded Areas	50
ไม่มีความเสียหาย	Normal (Non-Damage)	50

3.1.6 ประเมินผลการทดลองและปรับปรุงแก้ไขตัวแบบให้มีประสิทธิภาพที่ดีขึ้น

ทดสอบตัวแบบ EfficientNet V2, Inception V3 และ MobileNet V3 ด้วยชุดข้อมูลฝึก (Train Dataset) 70% ข้อมูลประเมินตัวแบบ (Validation Dataset) 10% ข้อมูลทดสอบ (Test Dataset) 20% และข้อมูลทดสอบจากภายนอก (Evaluation Dataset) ที่กำหนดไว้ จากนั้นทดลองปรับปรุงตัวแบบด้วยวิธีการเพิ่มข้อมูล (Data Augmentation) และการปรับปรุง (Fine Tuning) ทำการเปรียบเทียบประสิทธิภาพตัวแบบทั้งหมดด้วยตัวชี้วัด ความแม่นยำ (Precision) ค่าเรียกคืน (Recall) คะแนน F1 (F1 Score) และค่าความถูกต้อง (Accuracy) ประกอบกับระยะเวลาในการฝึกสอนตัวแบบ

3.1.7 วิเคราะห์ สรุปผลการทดลอง และให้ข้อเสนอแนะ

นำผลลัพธ์ของตัวชี้วัดประสิทธิภาพและระยะเวลาในการฝึกอบรมจากผลการทดลองทั้งหมดใส่ในตารางเพื่อเปรียบเทียบประสิทธิภาพ ทำการวิเคราะห์และสรุปผล พร้อมให้ข้อเสนอแนะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ผลการทดลองและการอภิปรายผล

ในบทนี้จะกล่าวถึงผลการทดลองและการอภิปรายผลการทดลองในการจำแนกประเภทรูปภาพภัยพิบัติประกอบด้วย อาคารถล่ม/ซากปรักหักพัง ไฟไหม้/ควันไฟ น้ำท่วม และเหตุการณ์ปกติ จากชุดข้อมูล Crisis Vision Benchmark (CVB) เป็นชุดข้อมูลรูปภาพจากแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ และ Aerial Image Database for Emergency Response Applications (AIDER) เป็นชุดข้อมูลรูปภาพจากอากาศยานไร้คนขับ ด้วยตัวแบบโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (CNN) แบบมีการถ่ายโอนการเรียนรู้ (Transfer Learning) ใช้สถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 จาก Tensorflow Hub แบบมาตรฐาน (Standard - Std) โดยเพิ่มการปรับปรุงตัวแบบด้วยวิธีการเพิ่มข้อมูล (Data Augmentation - DA) วิธีการฝึกตัวแบบด้วยชุดข้อมูลปัจจุบัน (Fine Tuning - FT) และการปรับปรุงด้วยทั้ง 2 วิธีร่วมกัน (Data Augmentation And Fine Tuning - DA & FT) ซึ่งในบทที่ 4 จะเป็นการรายงานผลการเปรียบเทียบประสิทธิภาพตัวแบบที่ขึ้นการฝึก ขึ้นการทดสอบ และขึ้นการทดสอบด้วยข้อมูลที่ไม่เคยเห็นมาก่อน ด้วยตัวชี้วัด ค่าเรียกคืน (Recall) ค่าความถูกต้อง (Accuracy) ระยะเวลาในการฝึกตัวแบบ และระบุถึงปัญหาที่เกิดขึ้นจากการทดลอง

4.1 ผลการทดลองการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม EfficientNet V2 กับชุดข้อมูล CVB และชุดข้อมูล AIDER

ใช้ตัวแบบโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (CNN) แบบมีการถ่ายโอนการเรียนรู้ (Transfer Learning) สถาปัตยกรรม EfficientNet V2 จาก Tensorflow Hub ซึ่งผ่านการฝึกมาแล้วด้วยชุดข้อมูลรูปภาพ ImageNet-21K (Full ImageNet, Fall 2011 release) และผ่านการปรับปรุง (Fine-Tuned) ด้วยชุดข้อมูลรูปภาพ ImageNet1K โดยมีผลลัพธ์ดังนี้

4.1.1 ผลลัพธ์จากการฝึกตัวแบบ EfficientNet V2

การฝึกตัวแบบ EfficientNet V2 มาตรฐาน ด้วยชุดข้อมูล CVB มีค่าความถูกต้องที่ 94% และค่าการสูญเสียที่ 55% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล AIDER มีค่าความถูกต้องที่ 93% และค่าการสูญเสียที่ 53% เพียงเล็กน้อย แต่ใช้ระยะเวลาในการฝึกนานกว่าประมาณ 49% โดยการฝึกด้วยชุดข้อมูล AIDER หยุดการฝึกที่ 7 Epochs

ตารางที่ 4.1 ประสิทธิภาพการฝึกตัวแบบ EfficientNet V2

Dataset	Model Tuning	Accuracy	Loss	Time (ms)	Early Stopping
CVB	Std	0.94	0.55	285.68	N/A
	DA	0.82	0.77	327.43	N/A
	FT	0.77	0.87	364.16	N/A
	DA & FT	0.57	1.08	662.23	N/A
AIDER	Std	0.93	0.53	191.30	7 Epochs
	DA	0.88	0.63	394.55	N/A
	FT	0.82	0.71	423.47	N/A
	DA & FT	0.71	0.92	748.53	N/A

4.1.2 ผลลัพธ์จากการทดสอบตัวแบบ EfficientNet V2

ผลการทดสอบตัวแบบ EfficientNet V2 มาตรฐาน ที่ฝึกด้วยชุดข้อมูล AIDER มีค่าการเรียกคืนของคลาสความเสียหายจากน้ำท่วมสูงสุดที่ 97% มีค่าการเรียกคืนเฉลี่ยของทุกคลาสสูงสุดที่ 95% และมีค่าความถูกต้องสูงสุดที่ 94% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีค่าการเรียกคืนของคลาสความเสียหายที่สิ่งปลูกสร้างสูงสุดที่ 96% มีค่าการเรียกคืนเฉลี่ยของทุกคลาสที่ 86% และค่าความถูกต้องที่ 86%

ตารางที่ 4.2 ประสิทธิภาพการทดสอบตัวแบบ EfficientNet V2

Dataset	Class	Std		DA		FT		DA & FT	
		Recall	Accu- racy	Recall	Accu- racy	Recall	Accu- racy	Recall	Accu- racy
CVB	damaged_infrastruct ures/collapsed_build ing	0.96	0.86	0.87	0.84	0.69	0.73	0.23	0.57
	fire	0.89		0.87		0.83		0.76	
	flooded_areas	0.77		0.77		0.79		0.5	
	normal	0.81		0.84		0.6		0.81	
	All Class Average	0.86	N/A	0.84	N/A	0.73	N/A	0.58	N/A
AIDER	damaged_infrastruct ures/collapsed_build ing	0.96	0.94	0.91	0.94	0.84	0.9	0.45	0.79
	fire	0.96		0.97		0.91		0.85	
	flooded_areas	0.97		0.95		0.96		0.89	
	normal	0.89		0.93		0.89		0.96	
	All Class Average	0.95	N/A	0.94	N/A	0.90	N/A	0.79	N/A

4.1.3 ผลลัพธ์จากการทดสอบตัวแบบ EfficientNet V2 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

ผลการทดสอบตัวแบบ EfficientNet V2 มาตรฐาน ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อนจากแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ ตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีความถูกต้องสูงสุดที่ 88% และมีค่าการสูญเสียต่ำสุดที่ 64% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล AIDER ที่มีค่าความถูกต้องที่ 67% และมีค่าการสูญเสียที่ 104%

ตารางที่ 4.3 ประสิทธิภาพการทดสอบตัวแบบ EfficientNet V2 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

Model Tuning	CVB		AIDER	
	Accuracy	Loss	Accuracy	Loss
Std	0.88	0.64	0.67	1.04
DA	0.83	0.72	0.58	1.20
FT	0.76	0.86	0.44	1.53
DA & FT	0.57	1.17	0.38	2.08

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 ผลการทดลองการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม Inception V3 กับชุดข้อมูล CVB และชุดข้อมูล AIDER

ใช้ตัวแบบโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (CNN) แบบมีการถ่ายโอนการเรียนรู้ (Transfer Learning) สถาปัตยกรรม Inception V3 จาก Tensorflow Hub ซึ่งผ่านการฝึกมาแล้วด้วยชุดข้อมูลรูปภาพ ImageNet (ILSVRC-2012-CLS) โดยมีผลลัพธ์ดังนี้

4.2.1 ผลลัพธ์จากการฝึกตัวแบบ Inception V3

การฝึกตัวแบบ Inception V3 มาตรฐาน ด้วยชุดข้อมูล AIDER มีค่าความถูกต้องที่ 94% และค่าการสูญเสียที่ 0.52% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีค่าความถูกต้องที่ 89% และค่าการสูญเสียที่ 0.62% แต่ใช้ระยะเวลาในการฝึกนานกว่าประมาณ 26% โดยการฝึกด้วยชุดข้อมูล CVB หยุดการฝึกที่ 9 Epochs

ตารางที่ 4.4 ประสิทธิภาพการฝึกตัวแบบ Inception V3

Dataset	Model Tuning	Accuracy	Loss	Time (ms)	Early Stopping
CVB	Std	0.89	0.62	171.96	9 Epochs
	DA	0.72	0.89	332.52	8 Epochs
	FT	0.34	1.63	386.11	7 Epochs
	DA & FT	0.30	1.63	235.35	4 Epochs
AIDER	Std	0.94	0.52	216.07	N/A
	DA	0.82	0.73	504.89	N/A
	FT	0.57	1.32	258.62	N/A
	DA & FT	0.44	1.52	261.43	7 Epochs

4.2.2 ผลลัพธ์จากการทดสอบตัวแบบ Inception V3

ผลการทดสอบตัวแบบ Inception V3 มาตรฐาน ที่ฝึกด้วยข้อมูล AIDER มีค่าการเรียกคืนของคลาสความเสียหายจากน้ำท่วมสูงสุดที่ 96% ค่าการเรียกคืนเฉลี่ยของทุกคลาสสูงสุดที่ 92% และมีค่าความถูกต้องสูงสุดที่ 92% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยข้อมูล CVB มีค่าการเรียกคืนของคลาสความเสียหายจากไฟไหม้สูงสุดที่ 93% ค่าการเรียกคืนเฉลี่ยของทุกคลาสที่ 86% และค่าความถูกต้องที่ 86%

ตารางที่ 4.5 ประสิทธิภาพการทดสอบตัวแบบ Inception V3

Dataset	Class	Std		DA		FT		DA & FT	
		Recall	Accu- racy	Recall	Accu- racy	Recall	Accu- racy	Recall	Accu- racy
CVB	damaged_infrastructures/collapsed_building	0.84	0.86	0.51	0.73	Error	Error	Error	Error
	fire	0.93		0.76		Error		Error	
	flooded_areas	0.83		0.86		Error		Error	
	normal	0.84		0.77		Error		Error	
	All Class Average	0.86	N/A	0.73	N/A	Error	Error	Error	Error
AIDER	damaged_infrastructures/collapsed_building	0.85	0.92	0.86	0.89	0.81	0.46	Error	Error
	fire	0.92		0.87		0.51		Error	
	flooded_areas	0.96		0.87		0.4		Error	
	normal	0.94		0.97		0.13		Error	
	All Class Average	0.92	N/A	0.89	N/A	0.46	N/A	Error	Error

4.2.3 ผลลัพธ์จากการทดสอบตัวแบบ Inception V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

ผลการทดสอบตัวแบบ Inception V3 มาตรฐาน ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน จากแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ ตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีค่าความถูกต้องสูงสุดที่ 86% และมีค่าการสูญเสียต่ำสุดที่ 67% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล AIDER มีค่าความถูกต้องที่ 59% และมีค่าการสูญเสียที่ 129%

ตารางที่ 4.6 ประสิทธิภาพการทดสอบตัวแบบ Inception V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

Model Tuning	CVB		AIDER	
	Accuracy	Loss	Accuracy	Loss
Std	0.86	0.67	0.59	1.29
DA	0.68	0.86	0.50	1.34
FT	Error	Error	0.25	25.38
DA & FT	Error	Error	Error	Error

4.3 ผลการทดลองการจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม MobileNet V3 กับชุดข้อมูล CVB และชุดข้อมูล AIDER

ใช้ตัวแบบโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (CNN) แบบมีการถ่ายโอนการเรียนรู้ (Transfer Learning) ด้วยสถาปัตยกรรม MobileNet V3 จาก Tensorflow Hub ซึ่งผ่านการฝึกมาแล้วด้วยชุดข้อมูลรูปภาพ ImageNet (ILSVRC-2012-CLS)

4.3.1 ผลลัพธ์จากการฝึกตัวแบบ MobileNet V3

การฝึกตัวแบบ MobileNet V3 มาตรฐาน ด้วยชุดข้อมูล AIDER มีค่าความถูกต้องที่ 95% และค่าการสูญเสียที่ 48% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีค่าความถูกต้องที่ 95% และค่าการสูญเสียที่ 52% เพียงเล็กน้อย แต่ใช้ระยะเวลาในการฝึกนานกว่าประมาณ 39%

ตารางที่ 4.7 ประสิทธิภาพการฝึกตัวแบบ MobileNet V3

Dataset	Model Tuning	Accuracy	Loss	Time (ms)	Early Stopping
CVB	Std	0.95	0.52	157.65	N/A
	DA	0.77	0.84	307.68	9 Epochs
	FT	0.81	1.00	386.09	N/A
	DA & FT	0.57	1.37	199.42	7 Epochs
AIDER	Std	0.95	0.48	219.00	N/A
	DA	0.90	0.62	276.52	N/A
	FT	0.86	0.90	233.02	N/A
	DA & FT	0.75	1.08	236.83	7 Epochs

4.3.2 ผลลัพธ์จากการทดสอบตัวแบบ MobileNet V3

ผลจากการทดสอบตัวแบบ MobileNet V3 มาตรฐาน ที่ฝึกด้วยชุดข้อมูล AIDER มีค่าการเรียกคืนของคลาสความเสียหายจากน้ำท่วมสูงสุดที่ 96% มีค่าการเรียกคืนเฉลี่ยของทุกคลาสสูงสุดที่ 93% และมีค่าความถูกต้องสูงสุดที่ 93% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีค่าการเรียกคืนของคลาสความเสียหายจากไฟไหม้สูงสุดที่ 93% มีค่าการเรียกคืนเฉลี่ยของทุกคลาสที่ 85% และค่าความถูกต้องที่ 85%

ตารางที่ 4.8 ประสิทธิภาพการทดสอบตัวแบบ MobileNet V3

Dataset	Class	Std		DA		FT		DA & FT	
		Recall	Accu- racy	Recall	Accu- racy	Recall	Accu- racy	Recall	Accu- racy
CVB	damaged_infrastructures/collapsed_building	0.87	0.85	0.86	0.75	0.66	0.7	0.76	0.52
	fire	0.93		0.91		0.69		0.09	
	flooded_areas	0.80		0.66		0.71		0.57	
	normal	0.79		0.59		0.74		0.67	
	All Class Average	0.85	N/A	0.76	N/A	0.70	N/A	0.52	N/A
AIDER	damaged_infrastructures/collapsed_building	0.90	0.93	0.89	0.92	0.86	0.85	0.75	0.69
	fire	0.94		0.91		0.95		0.98	
	flooded_areas	0.96		0.94		0.72		0.28	
	normal	0.93		0.94		0.89		0.73	
	All Class Average	0.93	N/A	0.92	N/A	0.86	N/A	0.69	N/A

4.3.3 ผลลัพธ์จากการทดสอบตัวแบบ MobileNet V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

ผลจากการทดสอบตัวแบบ MobileNet V3 มาตรฐาน ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อนจากแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ ตัวแบบที่ฝึกด้วยชุดข้อมูล CVB มีความถูกต้องสูงสุดที่ 87% และมีค่าการสูญเสียต่ำสุดที่ 66% มีประสิทธิภาพดีกว่าตัวแบบที่ฝึกด้วยชุดข้อมูล AIDER มีค่าความถูกต้องที่ 55% และมีค่าการสูญเสียที่ 122%

ตารางที่ 4.9 ประสิทธิภาพการทดสอบตัวแบบ MobileNet V3 ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

Model Tuning	CVB		AIDER	
	Accuracy	Loss	Accuracy	Loss
Std	0.87	0.66	0.55	1.22
DA	0.80	0.79	0.51	1.34
FT	0.68	1.25	0.46	2.35
DA & FT	0.49	1.73	0.39	9.47

4.4 ปัญหาที่พบในการจำแนกประเภทรูปภาพภัยพิบัติ

หัวข้อนี้จะอธิบายถึงปัญหาที่เกิดขึ้นระหว่างการทดลองจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 แบบใช้วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) และแนวทางในการแก้ไขปัญหา

4.4.1 ปัญหาการใช้ข้อมูลที่เหมาะสมในการฝึกตัวแบบ

ข้อมูลประเภทรูปภาพภัยพิบัติจากแหล่ง Open Source มีจำนวนน้อย มีความหลากหลายและความไม่สมดุลสูง เนื่องจากเหตุการณ์ภัยพิบัติแต่ละประเภทมีอาจความแตกต่างกันมากเมื่อเกิดขึ้นในแต่ละพื้นที่ ทั้งในด้านความรุนแรงและมุมมองของรูปภาพที่ถูกบันทึกไว้ แม้ว่าข้อมูลส่วนมากจะถูกระบุคำตอบไว้แล้ว แต่เป็นมักจะเป็นการทำคำตอบด้วยมนุษย์ซึ่งมีโอกาสเกิดความผิดพลาดสูง ผู้ใช้งานจึงจำเป็นต้องใช้เวลาในการตรวจสอบและปรับปรุงชุดข้อมูลก่อนนำไปใช้งานจริง รวมทั้งการเลือกข้อมูลให้เหมาะสมกับโดเมนที่ผู้ใช้งานสนใจ เพื่อช่วยทำให้การฝึกตัวแบบมีความถูกต้องดีขึ้น

4.4.2 ปัญหาจากการใช้งาน Tensorflow

Tensorflow เป็นระบบ (Framework) ที่มีประโยชน์ในการช่วยพัฒนาตัวแบบและการนำไปประยุกต์ใช้ในการสร้างแอปพลิเคชันที่สนใจได้อย่างรวดเร็ว แต่ระบบมีลักษณะการใช้ที่เป็นคำสั่งสำเร็จรูปคล้ายกับกล่องดำ (Black Box) มีระบบการจัดการเอกสารนี้เป็นเอกสารที่สับสนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลที่มีความซับซ้อนสูง ผู้ใช้งานจำเป็นต้องใช้เวลาในการศึกษารายละเอียดและทำความเข้าใจการทำงานของระบบให้ดี

4.4.3 ปัญหาด้านทรัพยากรการประมวลผลที่จำกัด

เนื่องจากข้อมูลรูปภาพในการทดลองมีจำนวนมาก จึงจำเป็นต้องใช้การประมวลผลด้วย GPU ที่มีประสิทธิภาพและมีหน่วยความจำที่เหมาะสม ระบบควรทำงานได้ต่อเนื่องยาวนานหรือทำงานแบบคู่ขนานกันได้เพื่อประหยัดเวลาในการทดสอบ แต่การใช้งาน Google Colab Free มีข้อจำกัดในส่วนนี้ ทางผู้วิจัยจึงเลือกใช้ Google Colab Pro ซึ่งมีค่าใช้จ่ายในการประมวลผลแบบ Compute Unit หรือ Subscription เพื่อช่วยเพิ่มประสิทธิภาพการประมวลผลและลดระยะเวลาของการทดสอบลง

4.5 การอภิปรายผล

หัวข้อนี้จะกล่าวถึงองค์ความรู้ที่ได้จากการทดลองจำแนกรูปภาพภัยพิบัติด้วยสถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 แบบใช้วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning)

4.5.1 องค์ความรู้จากการทดลองสถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 แบบใช้วิธีการถ่ายโอนการเรียนรู้ (Transfer Learning) จาก Tensorflow Hub

การประเมินผลการทดลองด้วยค่าเรียกคืน (Recall) และค่าความถูกต้อง (Accuracy) พบว่าตัวแบบ EfficientNet V2, Inception V3 และ MobileNet V3 แบบมาตรฐาน ให้ผลค่าเรียกคืนเฉลี่ยอยู่ในช่วง 85-95% ค่าความถูกต้องอยู่ในช่วง 85-94% เมื่อทำการปรับปรุงตัวแบบด้วยวิธีเพิ่มข้อมูล ให้ผลเฉลี่ยค่าเฉลี่ยค่าเรียกคืนอยู่ในช่วง 73-94% ค่าความถูกต้องอยู่ในช่วง 73-94% เมื่อทำการปรับปรุงตัวแบบด้วยวิธีการฝึกด้วยข้อมูลปัจจุบัน ให้ผลเฉลี่ยค่าเฉลี่ยค่าเรียกคืนอยู่ในช่วง 46-90% ค่าความถูกต้องอยู่ในช่วง 46-90% เมื่อทำการปรับปรุงตัวแบบด้วยวิธีเพิ่มข้อมูลและวิธีการฝึกด้วยข้อมูลปัจจุบัน ให้ผลเฉลี่ยค่าเฉลี่ยค่าเรียกคืนอยู่ในช่วง 52-79% ค่าความถูกต้องอยู่ในช่วง 52-79% แสดงให้เห็นว่าตัวแบบการถ่ายโอนความรู้แบบมาตรฐานมีประสิทธิภาพที่ดีเพียงพอสำหรับการใช้งานจำแนกประเภทรูปภาพภัยพิบัติโดยไม่จำเป็นต้องทำการปรับปรุงตัวแบบเพิ่มเติม ซึ่งทั้ง 3 ตัวแบบมาตรฐานได้ผ่านการฝึกล่วงหน้าด้วยชุดข้อมูล ImageNet มาก่อนหน้านี้แล้ว

ตารางที่ 4.10 เปรียบเทียบประสิทธิภาพการทดลองตัวแบบ EfficientNet V2, Inception V3 และ MobileNet V3

Dataset	Model	Class	Std		DA		FT		DA & FT	
			Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy
CVB	EfficientNet V2	All Class Average	0.86	0.86	0.84	0.84	0.73	0.73	0.58	0.57
	Inception V3	All Class Average	0.86	0.86	0.73	0.73	Error	Error	Error	Error
	MobileNet V3	All Class Average	0.85	0.85	0.76	0.75	0.70	0.70	0.52	0.52
AIDER	EfficientNet V2	All Class Average	0.95	0.94	0.94	0.94	0.90	0.90	0.79	0.79
	Inception V3	All Class Average	0.92	0.92	0.89	0.89	0.46	0.46	Error	Error
	MobileNet V3	All Class Average	0.93	0.93	0.92	0.92	0.86	0.85	0.69	0.69

4.5.2 องค์ความรู้จากการเลือกใช้ชุดข้อมูลที่เหมาะสมในการฝึกตัวแบบ

การกำหนดวัตถุประสงค์ของโดเมนที่สนใจและการเลือกใช้ข้อมูลจากโดเมนที่เหมาะสมในการพัฒนาตัวแบบส่งผลโดยตรงต่อการทดสอบประสิทธิภาพตัวแบบด้วยชุดข้อมูลจริงที่ไม่เคยเห็นมาก่อน ตัวแบบที่ถูกฝึกด้วยชุดข้อมูล CVB ซึ่งเป็นชุดข้อมูลที่ได้มาจากอินเทอร์เน็ตรวมกับแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ มีประสิทธิภาพความถูกต้องและค่าการสูญเสียที่ดีกว่าในการทดสอบด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อนจากอินเทอร์เน็ตหรือสื่อสังคมออนไลน์ เมื่อเปรียบเทียบกับตัวแบบที่ถูกฝึกด้วยชุดข้อมูล AIDER ซึ่งเป็นชุดข้อมูลที่ได้มาจากอากาศยานไร้คนขับ

ตารางที่ 4.11 เปรียบเทียบประสิทธิภาพการทดสอบตัวแบบ EfficientNet V2, Inception V3 และ MobileNet V3 มาตรฐาน ด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน

Model	CVB		AIDER	
	Accuracy	Loss	Accuracy	Loss
EfficientNet V2 Std	0.88	0.64	0.67	1.04
Inception V3 Std	0.86	0.67	0.59	1.29
MobileNet V3 Std	0.87	0.66	0.55	1.22



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปผลการดำเนินงานและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงบทสรุปของงานค้นคว้าอิสระนี้ และข้อเสนอแนะเพิ่มเติมสำหรับการนำงานค้นคว้าอิสระนี้ไปพัฒนาต่อในอนาคต

5.1 สรุปผลการดำเนินงาน

การค้นคว้าอิสระนี้ได้นำเสนอตัวแบบโครงข่ายประสาทเทียมแบบคอนโวลูชันนอล (CNN) แบบมีการถ่ายโอนการเรียนรู้ (Transfer Learning) ใช้สถาปัตยกรรม EfficientNet V2, Inception V3 และ MobileNet V3 จาก Tensorflow Hub ในการจำแนกประเภทรูปภาพภัยพิบัติประกอบด้วย อาคารถล่ม/ซากปรักหักพัง ไฟไหม้/ควันไฟ น้ำท่วม และเหตุการณ์ปกติ ใช้ชุดข้อมูลจาก Crisis Vision Benchmark (CVB) เป็นชุดข้อมูลรูปภาพจากแพลตฟอร์มสื่อสังคมออนไลน์ทวิตเตอร์ และ Aerial Image Database for Emergency Response Applications (AIDER) เป็นชุดข้อมูลรูปภาพจากอากาศยานไร้คนขับ ทำการแก้ไขปัญหาคือข้อมูลไม่สมดุลด้วยเทคนิคการปรับจำนวนข้อมูลของแต่ละคลาสของชุดข้อมูลลงให้เท่ากันก่อน โดยการสุ่มเลือกข้อมูลแบบ Random Sampling Without Replacement เพื่อช่วยลดปัญหาความถูกต้องปฏิทรรศน์ (Accuracy Paradox) ทำการฝึกตัวแบบโดยแบ่งการทดลองเป็น 4 ประเภทหลัก คือ ใช้ตัวแบบมาตรฐานจาก Tensorflow Hub เพิ่มการปรับปรุงตัวแบบด้วยวิธีการเพิ่มข้อมูล (Data Augmentation) เพิ่มการปรับปรุงตัวแบบด้วยวิธีการฝึกตัวแบบด้วยชุดข้อมูลปัจจุบัน (Fine Tuning) และเพิ่มการปรับปรุงตัวแบบด้วยทั้ง 2 วิธีร่วมกัน (Data Augmentation And Fine Tuning)

ในขั้นตอนการฝึก ตัวแบบมาตรฐาน MobileNet V3 ที่ฝึกด้วยชุดข้อมูล AIDER มีค่าความถูกต้อง (Accuracy) สูงสุดที่ 95% มีค่าการสูญเสีย (Loss) ต่ำสุดที่ 48% ใช้ระยะเวลาในการฝึกที่ 219 ms.

ในขั้นตอนการทดสอบประสิทธิภาพการของตัวแบบด้วยชุดข้อมูลปัจจุบัน ตัวแบบมาตรฐาน EfficientNet V2, Inception V3 และ MobileNet V3 ที่ฝึกด้วยชุดข้อมูล AIDER มีค่าการเรียกคืน (Recall) สูงสุดในคลาสน้ำท่วมที่ 97% 96% และ 96% ตามลำดับ มีค่าเฉลี่ยการเรียกคืน (Mean Recall) ที่ 95% 92% และ 93% ตามลำดับ และมีค่าความถูกต้อง (Accuracy) ที่ 94% 92% และ 93% ตามลำดับ ซึ่งมีประสิทธิภาพที่ดีกว่าตัวแบบมาตรฐาน EfficientNet V2, Inception V3 และ MobileNet V3 ที่ฝึกด้วยชุดข้อมูล CVB มีค่าเฉลี่ยการเรียกคืนที่ 86% 86% และ 85% ตามลำดับ และมีค่าความถูกต้องที่ 86% 86% และ 85% ตามลำดับ และเมื่อเปรียบเทียบประสิทธิภาพกับงานวิจัยที่เกี่ยวข้องในข้อ 2.11.2 Exploring The Role Of Deep Neural Networks For Post-

Disaster Decision Support โดย N. Chaudhuri et al [39] ซึ่งใช้ตัวแบบ ResNet-50 กับชนิดชุดเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลรูปภาพอย่างเดียว ให้ค่าการเรียกคืนและค่าความถูกต้องสูงสุดที่ 93.33% และ 92.16% ตามลำดับ พบว่าตัวแบบมาตรฐาน EfficientNet V2 จากงานวิจัยนี้ ที่ฝึกด้วยชุดข้อมูล AIDER มีค่าเฉลี่ยการเรียกคืนที่ 95% และมีค่าความถูกต้องที่ 94% มีประสิทธิภาพสูงกว่าเล็กน้อย

จากการที่ตัวแบบมาตรฐาน EfficientNet V2, Inception V3 และ MobileNet V3 ที่ฝึกด้วยชุดข้อมูล AIDER มีค่าการเรียกคืนสูงสุดในคลาสความเสียหายจากน้ำท่วมที่ 96-97% เป็นผลมาจากชุดข้อมูลที่ได้มาจากอากาศยานไร้คนขับ ซึ่งเป็นรูปถ่ายจากมุมบน (Top View) แสดงให้เห็นความชัดเจนของสีพื้นน้ำที่ปกคลุมในบริเวณภาพโดยรอบและมักจะครอบคลุมพื้นที่ความเสียหายเป็นบริเวณกว้าง ทำให้ตัวแบบเรียนรู้ลักษณะสำคัญของภาพและทำการทำนายได้มีประสิทธิภาพดีกว่าคลาสอื่น

ในการทดสอบประสิทธิภาพของตัวแบบด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อน ตัวแบบมาตรฐาน EfficientNet V2 ที่ฝึกด้วยชุดข้อมูล CVB มีค่าความถูกต้องสูงสุดที่ 88% และค่าการสูญเสียต่ำสุดที่ 64%

เนื่องจากข้อมูลในโดเมนจริงที่ต้องการนำตัวแบบไปใช้งานมีความคล้ายคลึงกับข้อมูลชุด CVB มากกว่าข้อมูลชุด AIDER จึงอาจจะพิจารณาเลือกใช้ตัวแบบมาตรฐาน EfficientNet V2 ที่ฝึกด้วยข้อมูลชุด CVB ในขั้นตอนการฝึกมีค่าความถูกต้องที่ 94% ค่าการสูญเสียที่ 55% ดีที่สุดเป็นอันดับ 2 ในขั้นตอนการทดสอบด้วยชุดข้อมูลปัจจุบันมีค่าเฉลี่ยการเรียกคืนที่ 86% ค่าความถูกต้องที่ 86% ดีที่สุดเป็นอันดับ 1 เท่ากับตัวแบบมาตรฐาน Inception V3 ในขั้นตอนการทดสอบด้วยชุดข้อมูลที่ไม่เคยเห็นมาก่อนมีค่าความถูกต้องสูงสุดที่ 88% และค่าการสูญเสียต่ำสุดที่ 64% ดีที่สุดเป็นอันดับ 1

5.2 ข้อเสนอแนะ

เนื่องจากข้อจำกัดด้านระยะเวลาในการดำเนินงาน และเพื่อเป็นแนวทางให้กับผู้สนใจที่จะนำการค้นคว้าอิสระนี้ไปปรับปรุงและพัฒนาต่อไปในอนาคต ทางผู้วิจัยจึงขอเสนอข้อเสนอแนะเพิ่มเติมดังนี้

5.2.1 ทดลองสร้างตัวแบบด้วยชุดข้อมูลผสมระหว่าง CVB และ AIDER

จากการทดลองพบว่าตัวแบบที่ฝึกด้วยชุดข้อมูลที่แตกต่างกัน 2 ชุดนี้ ส่งผลโดยตรงต่อประสิทธิภาพของตัวแบบ เมื่อนำไปใช้ทดสอบกับข้อมูลจริงที่ไม่เคยเห็นมาก่อนได้มาจากอินเทอร์เน็ตและแพลตฟอร์มสื่อสังคมออนไลน์ ทางผู้วิจัยจึงขอเสนอให้ทดลองสร้างตัวแบบใหม่ขึ้นมาจากการรวมข้อมูล 2 ชุดนี้เข้าด้วยกัน จะทำให้ตัวแบบมีประโยชน์ในการจำแนกข้อมูลภัยพิบัติประเภทรูปภาพที่ครอบคลุมมากขึ้น

5.2.2 ทดลองสร้างตัวแบบด้วยชุดข้อมูลสมรรถภาพและข้อความ (Multimodal Dataset)

เนื่องจากผู้ใช้งานอินเทอร์เน็ตและแพลตฟอร์มสื่อสังคมออนไลน์ไม่ได้แบ่งปันข้อมูลเฉพาะรูปภาพเพียงอย่างเดียว แต่การโพสต์ข้อมูลโดยส่วนใหญ่ของผู้ใช้งานหรือการรายงานข่าวของสำนักข่าวมักจะเป็นข้อมูลที่ประกอบด้วยรูปภาพและข้อความ ซึ่งตัวแบบจำเป็นต้องมีความสามารถเพิ่มเติมในการประมวลผลด้านภาษา (NLP) ด้วย จะทำให้ตัวแบบมีประโยชน์ในการจำแนกข้อมูลภัยพิบัติแบบผสมได้ครอบคลุมมากขึ้น

5.2.3 สร้างระบบ Social Listening Platform

เนื่องจากมีภัยพิบัติเกิดขึ้นตลอดเวลาและมีความถี่ในการอุบัติมากขึ้น การสร้างระบบ Social Listening Platform ทำหน้าที่เฝ้าระวังและตรวจสอบภัยพิบัติที่เกิดขึ้นจากข้อมูลของผู้ใช้งานอินเทอร์เน็ตและสื่อสังคมออนไลน์ตลอดเวลา และทำการรายงานเหตุการณ์ให้กับหน่วยงานที่เกี่ยวข้อง จะเป็นประโยชน์ให้กับหน่วยงานที่เกี่ยวข้องในการตอบสนองต่อเหตุการณ์หลังเหตุภัยพิบัติได้ดีและรวดเร็วยิ่งขึ้น

เอกสารอ้างอิง

- [1] "Deep Learning," [Online]. Available: https://en.wikipedia.org/wiki/Deep_learning.
- [2] R. Nagyfi, "The differences between Artificial and Biological Neural Networks," [Online]. Available: <https://towardsdatascience.com/the-differences-between-artificial-and-biological-neural-networks-a8b46db828b7>.
- [3] D. Cowan, "backpropagation," [Online]. Available: <https://www.ml-science.com/backpropagation>.
- [4] P. Pandey, "Understanding the Mathematics behind Gradient Descent.," [Online]. Available: <https://towardsdatascience.com/understanding-the-mathematics-behind-gradient-descent-dde5dc9be06e#:~:text=In%20the%20equation%2C%20y%20%3D%20mX,%3Db%2D%CE%B4b%2C%20respectively..>
- [5] "Gradient Descent in Machine Learning," [Online]. Available: <https://www.javatpoint.com/gradient-descent-in-machine-learning>.
- [6] "CS231n Convolutional Neural Networks for Visual Recognition," [Online]. Available: <https://cs231n.github.io/neural-networks-3/>.
- [7] "KDNuggets," [Online]. Available: <https://www.kdnuggets.com/2022/02/vanishing-gradient-problem.html#:~:text=The%20vanishing%20gradient%20problem%20is,activation%20function%20such%20as%20ReLU..>
- [8] A. Ajagekar, "Adam," [Online]. Available: <https://optimization.cbe.cornell.edu/index.php?title=Adam#:~:text=Adam%20optimizer%20is%20the%20extended,processing%20in%20the%20future%20years..>
- [9] Z. Zhang, "Improved Adam Optimizer for Deep Neural Networks," *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), Banff, AB, Canada, 2018*, pp. 1-2, 2018.
- [10] W. E. L. Ilboudo, T. Kobayashi and K. Sugimoto, "TAdam: A Robust Stochastic Gradient Optimizer," *arXiv*, 2020.
- [11] dewangNautiyal, "ML | Underfitting and Overfitting," [Online]. Available: <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/>.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- [12] U. Tewari, "Regularization — Understanding L1 and L2 regularization for Deep Learning," 10 November 2021. [Online]. Available: <https://medium.com/analytics-vidhya/regularization-understanding-l1-and-l2-regularization-for-deep-learning-a7b9e4a409bf>.
- [13] A. Budhiraja, "Dropout in (Deep) Machine learning," 16 December 2016. [Online]. Available: <https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5>.
- [14] TagX, "Data Augmentation for Computer Vision," 4 February 2023. [Online]. Available: <https://medium.com/@tagxdata/data-augmentation-for-computer-vision-b97cbe0bae9e>.
- [15] C. S. Sergey Ioffe, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv*, 2015.
- [16] B. Zhu, "Model Validation," 20 April 2021. [Online]. Available: <https://medium.com/unpackai/overview-of-model-validation-d2fc1f1b1c7e>.
- [17] L. Thomas, "Stratified Sampling | Definition, Guide & Examples," 19 June 2019. [Online]. Available: <https://www.scribbr.com/methodology/stratified-sampling/>.
- [18] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way," 15 December 2018. [Online]. Available: <https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/>.
- [19] M. Mishra, "Convolutional Neural Networks, Explained," 27 August 2020. [Online]. Available: <https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939#:~:text=A%20CNN%20typically%20has%20three,and%20a%20full%20connected%20layer..>
- [20] S. MARTIN, "What's the Difference Between a CNN and an RNN?," 5 September 2018. [Online]. Available: <https://blogs.nvidia.com/blog/2018/09/05/whats-the-difference-between-a-cnn-and-an-rnn/>.
- [21] S. SHARMA, "Activation Functions in Neural Networks," 6 September 2017. [Online]. Available: <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- [22] S. Jaiyen, *Convolutional Neural Networks*.
- [23] savyakhosla, "CNN | Introduction to Pooling Layer," [Online]. Available: <https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/#:~:text=Max%20pooling%20is%20a%20pooling,of%20the%20previous%20feature%20map..>
- [24] S. Team, "Convolutional Neural Networks (CNN): Step 3 - Flattening," 18 August 2018. [Online]. Available: <https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening>.
- [25] I. T. Warrior, "Fully Connected Layers in Convolutional Neural Networks," [Online]. Available: <https://indiantechwarrior.com/fully-connected-layers-in-convolutional-neural-networks/>.
- [26] K. E. Koech, "Cross-Entropy Loss Function," 3 October 2020. [Online]. Available: <https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e>.
- [27] neuralthreads, "Categorical cross-entropy loss — The most important loss function," 2 December 2021. [Online]. Available: <https://neuralthreads.medium.com/categorical-cross-entropy-loss-the-most-important-loss-function-d3792151d05b>.
- [28] D. Shah, "Cross Entropy Loss: Intro, Applications, Code," 26 January 2023. [Online]. Available: <https://www.v7labs.com/blog/cross-entropy-loss-guide#:~:text=Categorical%20Cross%20Entropy%20is%20also,N%20classes%20for%20each%20image>.
- [29] R. Gómez, "Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names," 23 May 2018. [Online]. Available: https://gombru.github.io/2018/05/23/cross_entropy_loss/.
- [30] Q. V. L. Mingxing Tan, "EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling," 29 May 2019. [Online]. Available: <https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html>.

- [31] S.-H. Tsang, "Review — EfficientNetV2: Smaller Models and Faster Training," 14 February 2022. [Online]. Available: <https://medium.com/aiguys/review-efficientnetv2-smaller-models-and-faster-training-47d4215dcdfb>.
- [32] Google, "Feature vectors of images with EfficientNet V2 with input size 384x384, trained on imagenet-21k (Full ImageNet, Fall 2011 release). Fine-tuned on ImageNet1K," [Online]. Available: https://tfhub.dev/google/imagenet/efficientnet_v2_imagenet21k_ft1k_s/feature_vector/2.
- [33] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016*, pp. 2818-2826, 2016.
- [34] Google, "Advanced Guide to Inception v3," 19 June 2023. [Online]. Available: <https://cloud.google.com/tpu/docs/inception-v3-advanced>.
- [35] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le and H. Adam, "Searching for MobileNetV3," *arXiv*, 2019.
- [36] Google, "Feature vectors of images with MobileNet V3 large(depth multiplier 1.00) trained on ImageNet (ILSVRC-2012-CLS)," [Online]. Available: https://tfhub.dev/google/imagenet/mobilenet_v3_large_100_224/feature_vector/5.
- [37] A. Suresh, "What is a confusion matrix?," 17 November 2020. [Online]. Available: <https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5>.
- [38] T. T. Christos Kyrkou, "EmergencyNet: Efficient Aerial Image Classification for Drone-Based Emergency Monitoring Using Atrous Convolutional Feature Fusion," *arXiv*, 2021.
- [39] N. Chaudhuri and I. Bose, "Exploring the role of deep neural networks for post-disaster decision support," *SCOPUS, Decision Support Systems, Elsevier B.V.*, 2020.
- [40] Z. Ahmad, R. Jindal, M. N.S., A. Ekbal and P. Bhattacharyya, "Multi-modality helps in crisis management: An attention-based deep learning approach of

leveraging text for image classification," *Science Direct, Expert Systems with Applications*, vol. 195, 2022.

- [41] S. Kaur, S. Gupta, S. Singh, V. T. Hoang, S. Almakdi, T. Alelyani and A. Shaikh, "Transfer Learning-Based Automatic Hurricane Damage Detection Using Satellite Images," *MDPI, Electronics*, 2022.
- [42] F. Alam, F. Ofli, M. Imran, T. Alam and U. Qazi, "Deep Learning Benchmarks and Datasets for Social Media Image Classification for Disaster Response," *arXiv*, 2020.
- [43] H. Mouzannar, Y. Rizk and M. Awad, "Damage Identification in Social Media Posts using Multimodal Deep Learning," *ISCRAM 2018 Conference Proceedings – 15th International Conference on Information Systems for Crisis Response and Management*, pp. 529-543, 2018.
- [44] C. Kyrkou and T. Theodoridis, "AIDER: Aerial Image Database for Emergency Response Applications," 28 December 2020. [Online]. Available: <https://github.com/ckyrkou/AIDER>.
- [45] Wikipedia, "Accuracy Paradox," 18 March 2023. [Online]. Available: https://en.wikipedia.org/wiki/Accuracy_paradox.
- [46] P. E. C. o. Science, "34.1 - Random Sampling without Replacement," [Online]. Available: <https://online.stat.psu.edu/stat482/lesson/34/34.1#:~:text=sampling%20without%20replacement%2C%20in%20which,be%20selected%20more%20than%20once..>