การลดโครงข่ายประสาทเทียมคอนโวลูชั่นโดยใช้เทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับเกรเดียนต์เชิงพื้นที่และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น

FILTER PRUNING IN CONVOLUTIONAL NEURAL NETWORKS BASED ON LOCAL GRADIENT ACTIVATION MAPPING AND CONVOLUTIONAL APPROXIMATION SMALL MODEL FRAMEWORK

มณฑล อินทรประสิทธิ์

MONTHON INTRARAPRASIT

วิทยานิพนธ์นี้สำหรับการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2566

KMITL-2023-EN-D-018-175

FILTER PRUNING IN CONVOLUTIONAL NEURAL NETWORKS BASED ON LOCAL GRADIENT ACTIVATION MAPPING AND CONVOLUTIONAL APPROXIMATION SMALL MODEL FRAMEWORK

MONTHON INTRARAPRASIT

A THESIS SUBMITTED IN FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

DOCTOR OF ENGINEERING IN ELECTRICAL ENGINEERING

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2023

KMITL-2023-EN-D-018-175

หัวข้อวิทยานิพนธ์	การลดโครงข่ายประสาทเทียมคอนโวลูชั่นโดยใช้เทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับเกรเดียนต์เชิงพื้นที่และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น
นักศึกษา	นายมณฑล อินทรประสิทธิ์
รหัสนักศึกษา	61601175
ปริญญา	วิศวกรรมศาสตรดุษฎีบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้า
พ.ศ.	2566
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ.ดร.อรฉัตร จิตต์โสภักตร์

บทคัดย่อ

วิทยานิพนธ์ฉบับนี้ได้นำเสนอเทคนิคการวิเคราะห์ตัวกรองด้วยเทคนิคการจับคู่แผนผังคุณลักษณะกับเกรเดียนต์เชิงพื้นที่และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น เทคนิคนี้ถูกนำมาใช้ในการลดขนาดของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น ซึ่งเป็นหนึ่งในรูปแบบโครงข่ายประสาทเทียมเชิงลึกที่นิยมใช้ในงานด้านการประมวลผลภาพและคอมพิวเตอร์วิทัศน์ การวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับเกรเดียนต์เชิงพื้นที่ช่วยในการคัดเลือกตัวกรองที่อยู่ภายในแต่ละเลเยอร์ของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นว่าสำคัญหรือไม่สำคัญ ก่อนทำการนำตัวกรองที่ถูกประเมินแล้วว่าไม่สำคัญกับโครงสร้างออกจากโครงข่ายประสาทเทียมแบบคอนโวลูชั่น หลังจากนำตัวกรองในแต่ละเลเยอร์ออก ประสิทธิภาพของโครงข่ายประสาทเทียมจะลดลง การใช้เฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่นจะช่วยเรียกคืนประสิทธิภาพกลับมาได้ ทำให้การลดขนาดของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นยังคงประสิทธิภาพของแบบจำลองไว้ได้ใกล้เคียงกับโครงข่ายต้นแบบ ผลการทดลองแสดงให้เห็นว่าเทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับเกรเดียนต์เชิงพื้นที่สามารถคัดเลือกตัวกรองได้อย่างมีประสิทธิภาพ และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่นสามารถช่วยเรียกคืนประสิทธิภาพได้อย่างดี ส่งผลให้โครงข่ายประสาทเทียมแบบคอนโวโลชั่นมีขนาดเล็กลง ใช้เวลาในการประมวลผลลดลง อีกทั้งยังมีประสิทธภาพในการทำนายผลใกล้เคียงกับโครงข่ายประสาทเทียมแบบคอนโวลูชั่นต้นฉบับ

Thesis	Filter Pruning in Convolutional Neural Networks based on Local Gradient Activation Mapping and Convolutional Approximation Small Model Framework
Student	Mr. Monthon Intraraprasit
Student ID.	61601175
Degree	Doctor of Engineering
Program	Electrical Engineering
Year	2023
Thesis Advisor	Assoc. Prof. Dr. Orachat Chitsobhuk

ABSTRACT

This thesis proposes Localized Gradient Activation heatmaP technique (LGAP) and Convolutional Approximation Small Model (CASM) Framework. This technique is used to reduce the size of convolutional neural network (CNN), which is widely used in image processing and computer vision applications. Localized Gradient Activation heatmaP is a filter selection technique that helps in analyzing and selecting the filters within each layer of a convolutional neural network, by determining their significance. After filter selection process, the insignificant filters are pruned from each CNN layer. The pruned CNN model suffers from performance deterioration. The CASM Framework can be adopted to recover the performance of pruned model, allowing for the reduction in the size of the convolutional neural network while maintaining the performance of the model close to the original network. From experiments, the results illustrate that LGAP technique efficiently selects weak filters to be pruned, and using CASM Framework can effectively restore the model performance. This leads to a reduction in the size of the CNN model, a decrease in processing time, all while retaining predictive performance similar to the original convolutional neural network.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สามารถบรรลุวัตถุประสงค์อย่างเป็นอย่างดี อันเนื่องด้วยความช่วยเหลือและความกรุณาของอาจารย์ที่ปรึกษารองศาสตรจารย์ ดร.อรฉัตร จิตต์โสภักตร์ ที่ให้ความช่วยเหลือในด้านความรู้ทางวิชาการ ทักษะการทำงาน ทักษะการใช้ชีวิต เพื่อประยุกต์ใช้ในการทำงานและการใช้ในชีวิตประจำวัน ให้กำลังใจนักศึกษา เมื่อเจอหนทางที่ยากลำบากในการแก้ปัญหา อีกทั้งช่วยจัดสรรทรัพยากรในการทำงานวิจัย เพื่อให้นักศึกษาสามารถทำงานได้อย่างราบรื่นและมีประสิทธิภาพ ข้าพเจ้าขอขอบคุณและขอแสดงความนับถือเป็นอย่างยิ่ง

ขอขอบคุณตัวข้าพเจ้าเองที่สามารถพัฒนาตนเองทั้งในด้านความรู้ทางวิชาการ จิตใจ กระบวนการคิด จนสามารถแก้ไขปัญหาต่าง ๆ และทำงานวิจัยได้สำเร็จ

ขอขอบคุณ รุ่นพี่ บุคลากร และสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่ช่วยอำนวยความสะดวกในการจัดการเอกสารและสถานที่ในการทำงานวิจัยของข้าพเจ้า

ขอขอบคุณคณะกรรมการทุกท่านที่ช่วยให้คำชี้แนะต่าง ๆ ในงานวิจัย เพื่อพัฒนาและแก้ไขงานวิจัยในวิทยานิพนธ์เล่มนี้

ท้ายที่สุดขอขอบคุณครอบครัวของข้าพเจ้าที่อบรม สั่งสอน เลี้ยงดู และสนับสนุนการเรียนของข้าพเจ้าในทุก ๆ ด้าน

มณฑล อินทรประสิทธิ์

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญตาราง (ต่อ)

สารบัญรูป

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

โครงข่ายประสาทเทียมแบบคอนโวลูชั่นเป็นโครงข่ายชนิดหนึ่งในโครงข่ายประสาทเทียมเชิงลึกที่ถูกใช้อย่างกว้างขวางในปัจจุบัน ทั้งในภาคอุตสาหกรรมและภาควิชาการ การใช้งานโดยส่วนใหญ่จะใช้ในสายงานคอมพิวเตอร์วิทัศน์ โดยในงานของคอมพิวเตอร์วิทัศน์ ได้แก่ การจำแนกประเภทรูปภาพ, การตรวจจับวัตถุ, การรู้จำวัตถุ, การจำแนกประเภทโดยการตัดส่วน เป็นต้น ซึ่งโครงสร้างของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นถูกสร้างขึ้นมาจากการตัวกรองหลาย ๆ ตัวขึ้นเป็นเลเยอร์และหลาย ๆ เลเยอร์ประกอบกันขึ้นเป็นโครงข่ายประสาทเทียมขนาดใหญ่ที่มีโครงสร้างซับซ้อน โครงสร้างที่ถูกสร้างขึ้นมานั้นสามารถใช้ในการตรวจจับรูปแบบของข้อมูลผ่านการใช้ตัวกรองที่อยู่ภายในแต่ละเยอร์ได้ โดยปกติแล้วโครงสร้างที่ถูกสร้างขึ้น เลเยอร์ช่วงแรกจะใช้ในการตรวจจับขอบและสี เลเยอร์ที่ลึกขึ้นสามารถตรวจจับรายละเอียดที่ซับซ้อนขึ้นได้จากการใช้ข้อมูลเลเยอร์ช่วงต้น เช่น กริด ลวดลาย เป็นต้น สิ่งเหล่านี้ส่งผลให้ตัวกรองประกอบไปด้วยจำนวนพารามิเตอร์ขนาดใหญ่และต้องใช้พื้นที่ในการเก็บข้อมูลที่มีขนาดใหญ่ ยกตัวอย่าง แบบจำลอง ResNet-50 ประกอบไปด้วยพารามิเตอร์จำนวน 25.6 ล้านพารามิเตอร์และใช้พื้นที่ในการเก็บแบบจำลอง 98.2 เมกะไบต์ จำนวนพารามิเตอร์ที่มีขนาดใหญ่ส่งผลให้เวลาที่ใช้ในการสอนข้อมูลและการทำนายผลสูงขึ้นมากไปกว่านั้นการ์ดจอที่ใช้ในการประมวลผลก็ต้องรุ่นใหญ่เพื่อให้สามารถประมวลผลแบบจำลองได้ ทำให้ไม่สามารถใช้โครงข่ายประสาทเทียมแบบคอนโวลูชั่นบนอุปกรณ์ที่มีทรัพยากรแบบจำกัดได้ เช่น ไมโครคอนโทรลเลอร์ สมาร์ทโฟน เป็นต้น ตัวอย่างของทรัพยากรแบบจำกัด เช่น ข้อจำกัดทางด้านแบตเตอรี่, ข้อจำกัดด้านหน่วยประมวลผล และข้อกำจัดในด้านพื้นที่เก็บข้อมูล เป็นต้น การลดขนาดแบบจำลองเป็นกลยุทธ์สำคัญสำหรับการทำงานของโครงข่ายแบบคอนโวลูชั่นให้เกิดประสิทธิภาพสูงในสภาพแวดล้อมที่ถูกจำกัดในด้านทรัพยากรข้างต้น

ดังนั้นเพื่อทำให้การนำโครงข่ายประสาทเทียมแบบคอนโวลูชั่นไปใช้ได้อย่างมีประสิทธิภาพมากขึ้น ในงานวิจัยนี้จึงนำเสนอการลดขนาดโครงข่ายประสาทเทียมแบบคอนโวลูชั่นโดยการยุบตัวกรองผ่านการใช้เทคนิค การจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่และการเรียกคืนประสิทธิภาพแบบจำลองด้วยเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น เทคนิคการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่จะเป็นเทคนิคที่ใช้ในการตัวกรองของเลเยอร์ในโครงข่าย เพื่อทำการคัดเลือกตัวกรองที่ไม่มีประสิทธิภาพและข้อมูลการคัดเลือกจะถูกนำไปในการยุบตัวกรองออกจากโครงข่าย หลังจากยุบตัวกรองออกจากโครงข่ายประสิทธิภาพของแบบจำลองจะลดลง เฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่นจะเป็นตัวช่วยในการเรียกคืนประสิทธิภาพของแบบจำลอง ทำให้รักษาประสิทธิภาพของแบบจำลองที่ถูกยุบตัวกรองไว้ได้ ภายหลัง

การทำการยุบตัวกรองในโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจะสามารถช่วยลดเวลาที่ใช้ในการประมวล พื้นที่เก็บข้อมูลของแบบจำลอง สามารถทำให้นำโครงข่ายประสาทเทียมแบบคอนโวลูชั่นไปใช้ในอุปกรณ์ที่มีหน่วยประมวลผลขนาดเล็กลงและพื้นที่เก็บข้อมูลที่มีขนาดเล็ก

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

1. ศึกษากระบวนการยุบโครงสร้างโดยการยุบตัวกรองของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น
2. พัฒนาเทคนิคการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ที่ใช้ในการวิเคราะห์ตัวกรอง เพื่อยุบตัวกรองออกจากโครงสร้าง ซึ่งเป็นการวิเคราะห์ตัวกรองด้วยความสัมพันธ์เชิงพื้นที่ ความสัมพันธ์ของตัวกรองในเลเยอร์ และความสัมพันธ์ของตัวกรองกับผลลัพธ์จากการทำนาย
3. พัฒนาเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น เพื่อใช้ในการเรียกคืนประสิทธิภาพของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นที่ถูกยุบตัวกรองออกจากโครงสร้าง
4. ประเมินประสิทธิภาพของเทคนิคที่ใช้ในการวิเคราะห์สำหรับยุบตัวกรอง และเทคนิคที่ใช้ในการเรียกคืนประสิทธิภาพ

1.3 สมมติฐานของการศึกษา

1. กระบวนการยุบโครงสร้างโดยการยุบตัวกรองของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น จะมีกระบวนการที่ใช้ในการวิเคราะห์ตัวกรอง ซึ่งในงานวิจัยนี้ คือ การจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ วิธีการนี้สามารถที่จะใช้ในการวิเคราะห์ตัวกรอง เพื่อยุบโครงสร้างได้
2. การเรียกคืนประสิทธิภาพของแบบจำลองหลังการยุบตัวกรองของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น ด้วยเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น สามารถใช้งานได้

1.4 ทฤษฎีหรือแนวความคิดที่ใช้ในงานวิจัย

การยุบโครงข่ายประสาทเทียมคอนโวลูชั่นโดยใช้เทคนิคการการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น ใช้ทฤษฎีและหลักการดังต่อไปนี้

1. การยุบโครงข่ายประสาทเทียมคอนโวลูชั่นด้วยกระบวนการยุบโครงสร้างแบบตัวกรองถูกใช้ในกระบวนการยุบโครงสร้าง

2. gradient เชิงพื้นที่ ถูกนำมาใช้ในการวิเคราะห์ตัวกรอง เพื่อระบุถึงความสามารถของตัวกรองว่าจำเป็นต่อโครงข่ายหรือไม่
3. Cosine similarity ถูกนำมาใช้ทั้งในส่วนของการวิเคราะห์ความเหมือนของชุดข้อมูล ถูกนำมาใช้ในกระบวนการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ และในเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น
4. การปรับปรุงประสิทธิภาพของแบบจำลอง (fine-tuning) ถูกนำมาใช้ในการเรียกคืนประสิทธิภาพของแบบจำลองที่ถูกยุบโครงสร้าง

1.5 ขอบเขตของงานวิจัย

วิทยานิพนธ์ฉบับนี้ได้ทำการศึกษา วิจัยการยุบโครงข่ายประสาทเทียมคอนโวลูชั่นโดยใช้เทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น ซึ่งมีขอบเขตของการวิจัยดังต่อไปนี้

1. การยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่นโดยการวิเคราะห์ตัวกรองผ่านเทคนิคการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ เพื่อลดขนาดของโครงข่ายและตรวจสอบประสิทธิภาพของโครงข่ายหลังทำการเรียกคืนประสิทธิภาพ
2. การเรียกคืนประสิทธิภาพของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจากวิธีการทางพื้นฐานและวิธีแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น เพื่อปรับปรุงประสิทธิภาพของโครงข่ายที่ถูกยุบภายหลังการเรียกคืนประสิทธิภาพ
3. ชุดข้อมูลที่ถูกใช้ที่ในงานวิจัย คือ CIFAR-10 [1] และ ImageNet [2]
4. โครงข่ายประสาทเทียมแบบคอนโวลูชั่นที่ถูกใช้ในงานวิจัย เป็นโครงข่ายที่ได้รับการสอนข้อมูลมาแล้ว คือ VGG-16 [3] และ ResNet-50 [4]

1.6 ขั้นตอนการศึกษา

1. ศึกษาหัวข้อวิจัยจากงานวิจัยก่อนหน้า เพื่อให้ทราบวิธีการยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่น
2. วิเคราะห์องค์ความรู้ที่ได้จากการอ่านงานวิจัย เพื่อคิดเทคนิคในการยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่นและเทคนิคการเรียกคืนประสิทธิภาพ
3. ตั้งสมมติฐาน กำหนดวัตถุประสงค์และขอบเขตของงานวิจัย
4. วางแผนการทำงานวิจัย
5. พัฒนาเฟรมเวิร์คสำหรับการยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่น รวมถึงเทคนิคที่ใช้ในการวิเคราะห์ตัวกรองของโครงข่ายประสาทเทียมและเทคนิคการเรียกคืนประสิทธิภาพ

6. ทดลองเฟรมเวิร์คและเทคนิคทั้งหมดด้วยโครงข่ายประสาทเทียมแบบคอนโวลูชั่นและชุดข้อมูล
7. ตรวจสอบผลการทดลองในด้านของโครงข่ายประสาทเทียมหลังจากทำการยุบโครงสร้าง
8. วิเคราะห์ผลและสรุปผลการทดลองจากการใช้เทคนิคการวิเคราะห์ตัวกรองของโครงข่ายประสาทเทียมและเทคนิคการเรียกคืนประสิทธิภาพ

1.7 เครื่องมือและอุปกรณ์ที่ใช้ในงานวิจัย

1. เครื่องคอมพิวเตอร์ส่วนบุคคลที่มีหน่วยประมวลผล Intel Core i5, การ์ดจอ RTX 3060 จำนวน 1 เครื่อง
2. การ์ดจอ GTX 1070, GTX 1080, RTX 2060 และ RTX 3090 จำนวนอย่างละ 1 ชิ้น
3. ระบบปฏิบัติการ windows 10 64 bits
4. ชุดคำสั่ง Keras
5. โปรแกรม Visual Studio Code
6. Jupyter Notebook: web application
7. ชุดคำสั่งภาษา python
8. โปรแกรม Microsoft office 2016

1.8 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ถูกแบ่งออกเป็น 7 บท โดยแต่ละบทประกอบไปด้วยเนื้อหาดังต่อไปนี้

บทที่ 1 กล่าวถึงความเป็นมาและความสำคัญของปัญหา ความมุ่งหมายและวัตถุประสงค์ของการศึกษา สมมติฐานของการศึกษา ทฤษฎีหรือแนวความคิดที่ใช้ในงานวิจัย ขอบเขตของงานวิจัย ขั้นตอนการศึกษา และเครื่องมือและอุปกรณ์ที่ใช้ในงานวิจัย

บทที่ 2 กล่าวถึงงานวิจัยที่เกี่ยวกับการยุบโครงสร้างของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น

บทที่ 3 กล่าวถึงความรู้พื้นฐานที่เกี่ยวข้องกับงานวิจัยนี้นำเสนอเพื่อใช้ในการยุบโครงข่ายประสาทเทียมคอนโวลูชั่น ได้แก่ การยุบโครงข่ายประสาทเทียมคอนโวลูชั่นด้วยกระบวนการยุบโครงสร้างแบบตัวกรอง, Cosine similarity และการปรับปรุงประสิทธิภาพของแบบจำลอง

บทที่ 4 กล่าวถึงงานวิจัยที่นำเสนอ โดยมีเนื้อหาเกี่ยวกับเทคนิคการวิเคราะห์ตัวกรองด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น

บทที่ 5 กล่าวถึงการทดลองของงานวิจัย ซึ่งมีดังต่อไปนี้ การเตรียมข้อมูลและโครงข่ายประสาทเทียมแบบคอนโวลูชั่น, วิธีการในการจำลองการยุบตัวกรอง และการวัดผลของแบบจำลองที่ถูกยุบตัวกรอง

บทที่ 6 ผลการทดลองและการวิเคราะห์ผลการทดลอง

บทที่ 7 สรุปผลการทดลองและแนวทางการพัฒนาสำหรับนำไปใช้ต่อในอนาคต

บทที่ 2
งานวิจัยที่เกี่ยวข้อง

การยุบโครงข่ายเป็นเทคนิคที่ใช้แก้ปัญหาโครงข่ายประสาทเทียมคอนโวลูชั่นที่มีขนาดใหญ่ โดยทำการลดขนาดของแบบจำลอง ซึ่งจะส่งผลช่วยลดเวลาที่ใช้ในการประมวลผล เทคนิคแรกเริ่มที่มีการวิจัย คือ เทคนิคการลดสัมประสิทธิ์แบบจำลอง โดยใช้เทคนิคเช่นเดียวกับการเพิ่มประสิทธิภาพสมองที่เกิดความเสียหาย โดยการเลือกพารามิเตอร์ เพื่อให้เกิดการสูญเสียที่น้อยที่สุดสำหรับกระบวนการยุบโครงสร้างโดยการใช้อนุกรม Taylor อันดับ 2 หลังจากกระบวนการยุบโครงสร้างจะมีกระบวนการปรับประสิทธิภาพของโครงข่าย ซึ่งต้องใช้วิธีเมทริกซ์ Hessian ในวิธีการนี้จำเป็นจะต้องใช้หน่วยความจำขนาดใหญ่และการประมวลผลอย่างมหาศาล ต่อมาได้มีการพัฒนาเทคนิคใหม่โดยการรวมการยุบโครงสร้างโดยพิจารณาการเกาะกลุ่มของค่าสัมประสิทธิ์ตัวกรอง และนำการวิเคราะห์ตัวกรองมาไว้ในระหว่างกระบวนการสอนข้อมูล [5] โดยวิธีการนี้ทำให้เกิดการสูญเสียที่น้อยที่สุดจากหลักการการวิเคราะห์กระจายของกลุ่มค่าสัมประสิทธิในตัวกรองได้ เนื่องจากค่าน้ำหนักส่วนใหญ่ของกลุ่มมีค่าเข้าใกล้ 0 ตัวกรองเหล่านี้สามารถนำออกจากโครงสร้างได้ งานวิจัยต่อมาใช้เทคนิค Structured Sparsity Learning (SSL) [6] เทคนิคนี้สามารถใช้ในการยุบโครงสร้างได้หลายทาง ทั้งการใช้ตัวกรอง, ขนาดของตัวกรอง, channel และจำนวนชั้นของ layer แม้ว่า SSL จะสามารถยุบโครงสร้างได้อย่างมีประสิทธิภาพ แต่โครงสร้างของโครงข่ายจะมีการสูญเสียไป เป็นผลให้ไม่สามารถใช้งานได้กับชุดคำสั่งพื้นฐานได้ ในภายหลังมีการพัฒนาการยุบโครงสร้างโดยการยุบผ่านตัวกรอง ทำให้โครงสร้างของโครงข่ายประสาทเทียมแบบคอนโวลูชันยังใช้งานได้กับชุดคำสั่งพื้นฐาน เช่น การยุบตัวกรองโดยเทคนิคผลรวมของค่าน้ำหนัก [7], การยุบตัวกรองโดยเทคนิคค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์ [8], ค่าเฉลี่ยของ gradient [9] เป็นต้น โดยพื้นฐานเทคนิคการยุบโครงสร้างของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจะประกอบไปด้วย 3 กระบวนการคือ 1.การเลือกตัวกรอง เป็นกระบวนการวิเคราะห์ตัวกรอง โดยแตกต่างกันออกไปในแต่ละเทคนิค เพื่อดูว่าตัวกรองมีความสามารถสูงหรือต่ำ 2.การยุบตัวกรองออกจากโครงสร้างในแต่ละเลเยอร์ภายในโครงข่ายประสาทเทียมแบบคอนโวลูชั่น 3.การเรียกคืนประสิทธิภาพ เป็นกระบวนการสอนข้อมูลซ้ำ เพื่อทำการเรียกคืนประสิทธิภาพของแบบจำลองหลังจากที่ตัวกรองในเลเยอร์ถูกนำออกการโครงสร้าง

2.1 เทคนิคการยุบตัวกรองด้วยผลรวมของค่าน้ำหนัก [7]

ในงานวิจัยนี้เป็นการยุบโครงสร้างของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น โดยการยุบตัวกรองออกจากโครงสร้างโดยไม่ได้รับผลกระทบจากการกระจายของรูปแบบการเชื่อมโครงสร้าง (sparse connectivity patterns) ในส่วนของเทคนิคของการวิเคราะห์ตัวกรองในกระบวนการเลือกตัวกรอง จะวิเคราะห์ผ่านค่าน้ำหนักของตัวกรอง [7] ซึ่งสามารถคำนวณได้จากสมการที่ 2.1

$$S_i^{Weighted\ sum} = \sum |W(:,:,:,i)| \quad (2.1)$$

โดย $S_i^{Weighted\ sum}$ คือ คะแนนผลรวมของค่าน้ำหนักของตัวกรอง, W คือ ค่าน้ำหนักตัวกรอง และ i คือ ตัวกรองใด ๆ

คะแนนของตัวกรองในแต่ละตัวจะถูกคำนวณออกมาในรูปแบบของผลรวมของค่าสมบูรณ์ของค่าน้ำหนัก ตัวกรองที่มีคะแนนสูงจะถือเป็นตัวกรองที่มีความสามารถ และตัวกรองที่มีคะแนนต่ำจะถือเป็นตัวกรองที่ไม่มีความสามารถต้องนำออกจากโครงสร้าง โดยวิธีการจะวิเคราะห์ตัวกรองผ่านเพียงการใช้ค่าน้ำหนัก

2.2 เทคนิคการยุบตัวกรองโดยค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์ [8]

ในงานวิจัยนี้เป็นการยุบโครงสร้างโดยการวิเคราะห์ข้อมูลผ่านการส่งข้อมูลรูปภาพเข้าสู่โครงข่ายประสาทเทียมแบบคอนโวลูชั่นและทำการวิเคราะห์แผนผังคุณลักษณะที่ตำแหน่งของเลเยอร์ activation (หลังจากเลเยอร์คอนโวลูชั่นที่ต้องการยุบตัวกรอง) เพื่อวิเคราะห์แผงผังคุณลักษณะที่ออกมาจากคอนโวลูชั่นจากข้อมูลที่ส่งเข้าไป เพิ่มตรวจสอบว่ามีตัวกรองไหนบ้างที่ไม่มีการตอบรับข้อมูล (พื้นที่ของแผงผังคุณลักษณะมีการตอบรับเท่ากับ 0) ในปริมาณมาก โดยใช้สมการที่ 2.2 ในการวิเคราะห์

$$S_i^{APoZ} = \frac{1}{NxHxW} \sum_{n=1}^{N} (f(O_k(:,:,i)=0)) \quad (2.2)$$

โดย S_i^{APoZ} คือ คะแนนค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์ของตัวกรอง, N คือ จำนวนรูปภาพที่ใช้ในการวิเคราะห์, HxW คือ ขนาดของแผนผังคุณลักษณะ $O_k \in R^{HxWxD}$, $f(.)$ มีค่าเท่ากับ 1 ถ้า $(.)$ เป็นจริง, และ $f(.)$ มีค่าเท่ากับ 0 ถ้า $(.)$ เป็นเท็จ

ตัวกรองที่มีคะแนนสูงถือว่าเป็นตัวกรองที่มีประสิทธิภาพต่ำต้องนำออกจากโครงข่ายประสาทเทียม เพราะการตอบรับข้อมูลที่ส่งเข้ามามีพื้นที่ 0 ในปริมาณมาก และตัวกรองที่มีคะแนนต่ำถือว่าเป็นตัวกรองที่มีประสิทธิภาพสูง เพราะการตอบรับข้อมูลที่ส่งเข้ามามีพื้นที่ 0 ในปริมาณน้อย เทคนิคนี้จะวิเคราะห์ตัวกรองผ่านการใช้การส่งข้อมูลเข้าไปในโครงข่าย ซึ่งจะแตกต่างจากเทคนิคก่อนหน้าที่จะวิเคราะห์ค่าน้ำหนักในโครงข่ายประสาทเทียม

2.3 เทคนิคการยุบตัวกรองโดยค่าเฉลี่ยของ gradient [9]

ในงานวิจัยนี้เป็นการยุบโครงสร้างโดยการวิเคราะห์ตัวกรองผ่านการวิเคราะห์ด้วยการคำนวณค่าเฉลี่ยของ gradient ซึ่งสามารถคำนวณได้จากสมการที่ 2.3

$$S_i^{Mean\ gradient} = \frac{1}{N}\sum_{n=1}^{N} |Mean\left(G(:,:,i)\right)| \quad (2.3)$$

โดย $S_i^{Mean\ gradient}$ คือ คะแนนค่าเฉลี่ยของ gradient ของตัวกรอง, N คือ จำนวนรูปภาพที่ใช้ในการวิเคราะห์, G คือ gradient คำนวณได้ออกมาแต่ละตัวกรอง

ตัวกรองที่มีคะแนนสูงจะแสดงถึงว่ามีประสิทธิภาพสูง ส่วนตัวกรองที่มีคะแนนต่ำจะแสดงถึงมีประสิทธิภาพต่ำ ต้องนำออกจากแบบจำลอง

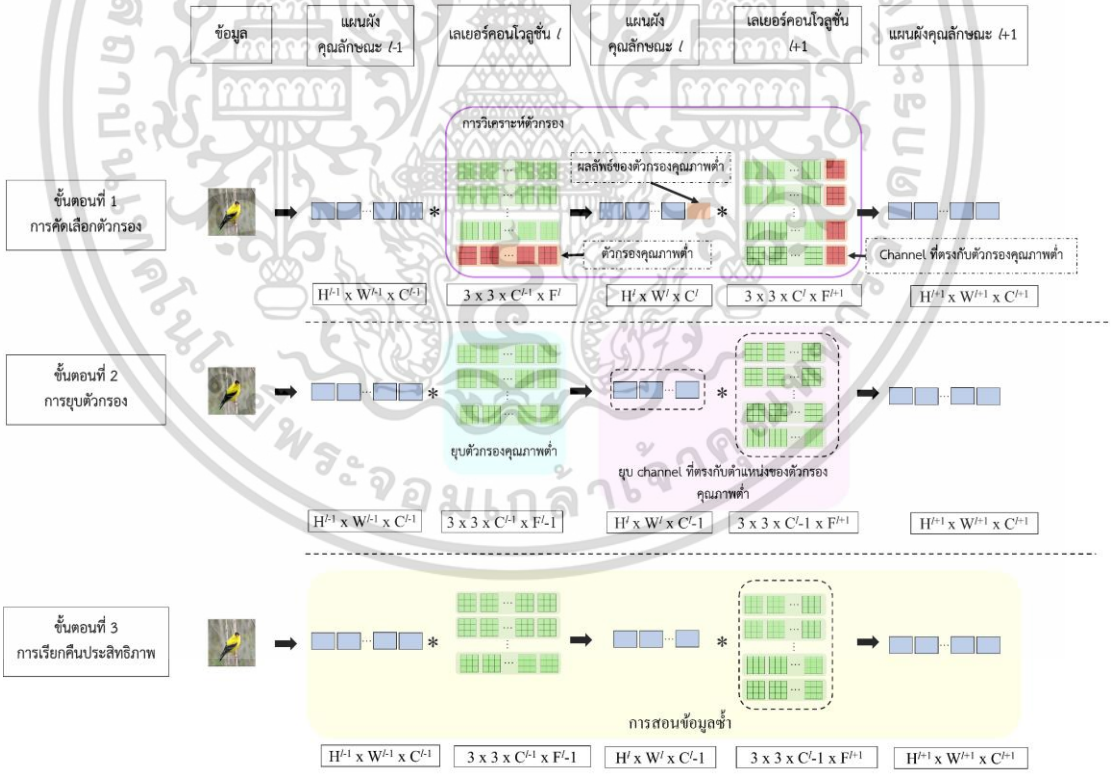
งานวิจัยข้างต้นได้แสดงถึงการยุบตัวกรองในโครงข่ายประสาทเทียมที่ใช้เทคนิคแตกต่างกันออกไป โดยเทคนิคการยุบตัวกรองด้วยผลรวมของค่าน้ำหนัก จะใช้เพียงค่าน้ำหนักในการวิเคราะห์เพื่อดูความสามารถของตัวกรอง ในขณะที่เทคนิคการวิเคราะห์โดยค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์และค่าเฉลี่ยของ gradient มีการใช้ข้อมูลส่งเข้าไปในโครงข่ายประสาทเทียมเพื่อวิเคราะห์การทำงานของตัวกรอง และประสิทธิภาพของแบบจำลองที่หลังกระบวนการเรียกคืนประสิทธิภาพแสดงออกมาว่าดีกว่าการใช้ค่าน้ำหนักในการวิเคราะห์ การใช้ข้อมูลส่งเข้าไปในโครงข่าย เพื่อทำการวิเคราะห์ตัวกรองจึงเป็นวิธีการที่น่าสนใจในการวิเคราะห์ตัวกรองก่อนทำการยุบตัวกรองออกจากโครงข่ายประสาทเทียม จากการวิเคราะห์วิธีค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์และค่าเฉลี่ยของ gradient พบว่ายังสามารถเพิ่มประสิทธิภาพในการวิเคราะห์ข้อมูลได้เพิ่มขึ้น หากวิเคราะห์ข้อมูลที่ส่งเข้ามาในโครงสร้างด้วยความสัมพันธ์เชิงพื้นที่น่าจะมีส่วนช่วยให้ประสิทธิภาพของแบบจำลองที่ทำการยุบดีขึ้น นอกจากนี้แล้วกระบวนการเรียกคืนประสิทธิภาพในงานวิจัยก่อนหน้าจะใช้วิธีการสอนข้อมูลซ้ำหลังจากยุบตัวกรอง ซึ่งในการยุบตัวกรองในแต่ละเลเยอร์ในโครงข่ายประสาทเทียมมีความอ่อนไหวทางด้านประสิทธิภาพไม่เท่ากัน [7] ทำให้จำเป็นต้องสอนข้อมูลซ้ำในแต่ละเลเยอร์มากกว่า 1 รอบ เพื่อช่วยให้กระบวนการเรียกคืนประสิทธิภาพดีขึ้นและลดข้อจำกัดในการเรียกคืนประสิทธิภาพโครงข่าย การประยุกต์ใช้การประมาณกลับค่าน้ำหนักในโครงข่ายประสาทเทียม ช่วยให้สามารถลดเวลาในการเรียกคืนประสิทธิภาพ [10] ดังนั้นในงานวิจัยจะนำเสนอเทคนิคการยุบตัวกรองด้วยการวิเคราะห์การจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่สำหรับการยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่น และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น เพื่อเรียกประสิทธิภาพของแบบจำลองที่ผ่านการยุบตัวกรองในโครงข่ายประสาทเทียม

บทที่ 3
ความรู้พื้นฐานในเรื่องที่เกี่ยวข้องกับงานวิจัย

การยุบตัวกรองของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจำเป็นต้องมีความรู้พื้นฐานในกระบวนการยุบตัวกรองในโครงข่ายประสาทเทียมแบบคอนโวลูชั่น และการวัดประสิทธิภาพของแบบจำลอง เพื่อช่วยให้ทราบถึงประสิทธิภาพของแบบจำลองก่อนทำการยุบตัวกรองและหลังยุบตัวกรอง ซึ่งในบทจะกล่าวถึงทั้ง 2 หัวข้อ

3.1 กระบวนการยุบตัวกรองในโครงข่ายประสาทเทียมแบบคอนโวลูชั่น

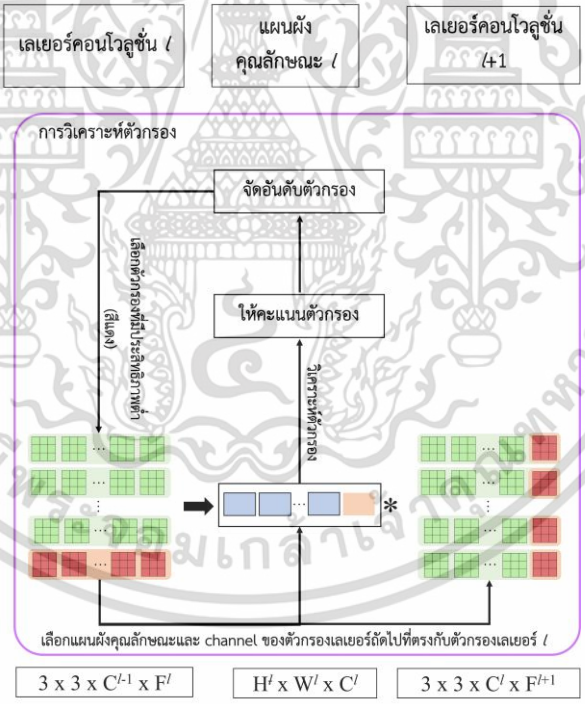
กระบวนการยุบตัวกรองในโครงข่ายประสาทเทียมเป็นกระบวนการที่จะใช้ในการลดขนาดของแบบจำลองผ่านการลดจำนวนของตัวกรองในแต่ละเลเยอร์ภายในโครงข่ายประสาทเทียมแบบคอนโวลูชั่น ประกอบไปด้วย 3 ขั้นตอน คือ 1. การคัดเลือกตัวกรอง 2. การยุบตัวกรอง 3. การเรียกคืนประสิทธิภาพ แสดงในรูปที่ 3.1



รูปที่ 3.1 ขั้นตอนการยุบตัวกรองในโครงข่ายประสาทเทียมแบบคอนโวลูชั่น, ข้อมูลรูปภาพชุดข้อมูล ImageNet [2]

3.1.1 การคัดเลือกตัวกรอง (ขั้นตอนที่ 1)

การคัดเลือกตัวกรองเป็นขั้นตอนการวิเคราะห์ตัวกรองที่อยู่ในเลยอร์ ซึ่งโครงสร้างภายในของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจะประกอบไปด้วยเลเยอร์คอนโวลูชั่นหลายเลเยอร์ ในรูปที่ 3.1 ขั้นตอนที่ 1 เป็นการวิเคราะห์ตัวกรองในเลเยอร์ l ซึ่งสามารถทำได้โดยนำข้อมูลรูปภาพในตัวอย่าง คือ รูปนก ไปทำนายผลในแบบจำลองของโครงข่ายประสาทเทียมแบบคอนโวลูชั่นที่มีการสอนข้อมูลไว้แล้ว (pre-trained model) เมื่อข้อมูลรูปภาพถูกส่งเข้าไปในโครงข่ายในแต่ละเลเยอร์จะทำกระบวนการคอนโวลูชั่น เพื่อดึงแผนผังคุณลักษณะออกมา ซึ่งในตัวอย่างคือ เลเยอร์ l ที่มีขนาดของตัวกรองเป็น $3x3xC^{l-1}xF^{l}$ (ความสูงเคอร์เนล x ความกว้างเคอร์เนล x จำนวน channel x จำนวนตัวกรอง) โดยรับข้อมูลต่อมาจากแผนผังคุณลักษณะเลเยอร์ l-1 เป็นข้อมูลขาเข้า ที่มีขนาดข้อมูลเป็น $H^{l-1}xW^{l-1}xC^{l-1}$ (สูงxกว้างxจำนวน channel ของเลเยอร์ l-1) เมื่อทำคอนโวลูชั่นกับเลเยอร์ l จะได้ผลลัพธ์ของแผนผังคุณลักษณะเลเยอร์ l ที่มีขนาด $H^{l}xW^{l}xC^{l}$ (สูงxกว้างxจำนวน channel ของเลเยอร์ l) ออกมา ข้อมูลจะถูกนำไปใช้ในการวิเคราะห์ผ่านเทคนิคการวิเคราะห์ตัวกรองตามรูปที่ 3.2 ที่ขยายรายละเอียดภายในของการวิเคราะห์ตัวกรองจากรูปที่ 3.1



รูปที่ 3.2 การวิเคราะห์ตัวกรอง

การวิเคราะห์ตัวกรองจะนำแผนผังคุณลักษณะมาวิเคราะห์เพื่อคำนวณหาคะแนนความสำคัญให้กับตัวกรอง ก่อนจะทำการจัดอันดับความสำคัญของตัวกรอง เมื่อได้ผลลัพธ์คะแนนความสำคัญของตัวกรองแล้ว ตัวกรองที่มีคะแนนความสำคัญต่ำจะถูกมองว่าไม่มีประสิทธิภาพ จำเป็นจะต้องนำออก โดยจะถูกระบุเป็นสีแดงใน 3 จุด คือ ตัวกรองของของเลเยอร์คอนโวลูชั่น l, ข้อมูลจาก

แผนผังคุณลักษณะเลเยอร์ l (ข้อมูลขาออกจากเลเยอร์คอนโวลูชั่น l) และ channel ในเลเยอร์คอนโวลูชั่น l+1 โดยในรูปที่ 3.1 และ 3.2 จำลองจากการระบุตัวกรองที่มีประสิทธิภาพต่ำเพียง 1 ตัว

3.1.2 การยุบตัวกรอง (ขั้นตอนที่ 2)

หลังจากตัวกรองที่มีประสิทธิภาพต่ำถูกระบุในเลเยอร์แล้ว (สีแดงในขั้นตอนที่ 1) ในส่วนนี้จะจำลองการยุบตัวกรองออกจากเลเยอร์คอนโวลูชั่น l จำนวน 1 ตัวกรอง ทำให้จำนวนตัวกรองที่เหลืออยู่ในเลเยอร์ l จะเหลือเพียง F^l-1 (จาก 3x3xC^{l-1}xF^l เป็น 3x3xC^{l-1}xF^l-1) ส่งผลให้ผลลัพธ์ที่ได้ออกมาจากเลเยอร์ l ในแผนผังคุณลักษณะเลเยอร์ l หายไป 1 ตัว เช่นกัน จาก H^lxW^lxC^l เป็น H^lxW^lxC^l-1 และเมื่อแผนผังคุณลักษณะหายไป จะส่งผลให้เลเยอร์คอนโวลูชั่น l+1 มีจำนวน channel หายไป 1 ตัว เช่นเดียวกัน จาก 3x3xC^lxF^{l+1} เป็น 3x3xC^l-1xF^{l+1} ดังนั้นการยุบตัวกรองในเลเยอร์คอนโวลูชั่นจะส่งผลให้จำนวน channel ของเลเยอร์คอนโวลูชั่นถัดไปหายไปด้วย แต่ไม่ทำให้จำนวนของผลลัพธ์ที่ได้ออกมาจากเลเยอร์ l+1 ลดลง (แผนผังคุณลักษณะเลเยอร์ l+1 ยังคงมีจำนวนเท่าเดิม)

3.1.3 การเรียกคืนประสิทธิภาพ (ขั้นตอนที่ 3)

ภายหลังการยุบตัวกรองที่มีประสิทธิภาพต่ำออกจากโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจะส่งผลให้ประสิทธิภาพของแบบจำลองลดลง จำเป็นจะต้องทำการเรียกคืนประสิทธิภาพโดยการการสอนข้อมูลซ้ำ (fine-tuning) การสอนข้อมูลซ้ำทำโดยการนำแบบจำลองที่ถูกยุบตัวกรองมาสอนข้อมูลซึ่งชุดข้อมูลที่ใช้จะเป็นชุดข้อมูลเดียวกับที่ใช้สร้างแบบจำลอง โดยในชุดข้อมูลจะมีข้อมูลสำหรับสอนและสำหรับทดสอบ แบบจำลองจะถูกตั้งค่าพารามิเตอร์และทำการสอนข้อมูลในเลเยอร์ที่ถูกยุบตัวกรองก่อนจะทำการยุบตัวกรองในเลเยอร์ถัดไป การเรียกคืนประสิทธิภาพจะช่วยให้ประสิทธิภาพของแบบจำลองกลับมาใกล้เคียงแบบจำลองที่ยังไม่มีการยุบตัวกรองออกจากโครงสร้างของโครงข่ายประสาทเทียม

3.2 Floating Point Operation (FLOP)

ในการวัดประสิทธิภาพด้านการประมวลผลของแบบจำลองสามารถวัดได้ในรูปแบบของ Floating Point Operation (FLOP) ซึ่งเป็นการวัดการประมวลผล เช่น บวก ลบ คูณ หาร หรืออื่น ๆ และเกี่ยวข้องกับค่า floating point ช่วยให้สามารถวัด complexity ของแบบจำลอง โดยในงานวิจัยนี้ นำมาใช้ในการวัด complexity ของเลเยอร์คอนโวลูชั่นภายในโครงข่ายประสาทเทียมทั้งหมด ซึ่งสามารถคำนวณได้จากสมการที่ 3.1

$$\text{FLOPs}_l = 2\text{x}N_l\text{x}K_l\text{x}O_l \tag{3.1}$$

FLOPs คือ complexity ของเลเยอร์คอนโวลูชั่นที่เลเยอร์ l, N_l คือ จำนวนของตัวกรองที่เลเยอร์ l, K_l คือ ขนาดของ kernel ที่เลเยอร์ l, O_l คือ ขนาดของข้อมูลขาออกที่เลเยอร์ l

FLOPs สามารถใช้เป็นค่าเปรียบเทียบ complexity ของเลเยอร์คอนโวลูชั่นที่ถูกยุบตัวกรองกับเลเยอร์คอนโวลูชั่นต้นฉบับในโครงข่ายประสาทเทียมทั้งหมด

3.3 การวัดประสิทธิภาพด้านการลดขนาดของแบบจำลองและความแม่นยำในการทำนายผล

ในการวัดประสิทธิภาพด้านการลดขนาดของแบบจำลองและความแม่นยำในการทำนายผล เป็นการวัดเพื่อเปรียบเทียบประสิทธิภาพของโครงข่ายประสาทเทียมที่มีการยุบตัวกรองกับแบบจำลองต้นฉบับ โดยการวัดประสิทธิภาพจะวัดในรูปแบบการบีบอัดแบบจำลองในแง่ของเปอร์เซ็นต์ของการบีบอัดแบบจำลองจากการลดขนาดของแบบจำลองต้นฉบับ โดยการใช้สมการที่ 3.2

$$\text{compression} = 100 - \left(\frac{\text{new size}}{\text{original size}} \text{x} 100\right) \quad (3.2)$$

compression คือ อัตราการบีบอัดแบบจำลองต้นฉบับ, new size คือ ขนาดของแบบจำลองใหม่หลังทำการบีบอัด, original size คือ ขนาดของแบบจำลองต้นฉบับ

การวัดประสิทธิภาพความแม่นยำในการทำนายของแบบจำลองในการจำแนกประเภทข้อมูลจะถูกใช้การวัดผลในด้านความแม่นยำโดยรวม (accuracy) ซึ่งถูกคำนวณจากเมทริกซ์การทำนายผลกับค่าความจริง (Confusion Matrix)

		Actual values: Positive	Actual values: Negative
Predicted values	Positive	TP	FP
	Negative	FN	TN

รูปที่ 3.3 เมทริกซ์การทำนายผลกับค่าความจริง

จากรูปที่ 3.3 True positive (TP) คือ การทำนายผลเป็นบวกและค่าความจริงเป็นบวก, False positive (FP) คือ การทำนายผลเป็นบวกและค่าความจริงเป็นลบ, False negative (FN) คือ

การทำนายผลเป็นลบและค่าความจริงเป็นบวก และ True negative (TN) คือ การทำนายผลเป็นลบและค่าความจริงเป็นลบ

ความแม่นยำโดยรวมเป็นการวัดประสิทธิภาพความแม่นยำในการทำนายผลทั้งระบบโดยจะใช้สมการที่ 3.3

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{3.3}$$

จากสมการที่ 3.3 จะเป็นการวัดประสิทธิภาพที่วัดจากผลรวมของการทำนายผลที่ถูกต้องทั้งเป็นบวกและลบหารด้วยการทำนายผลทั้งหมด

บทที่ 4
งานวิจัยที่นำเสนอ

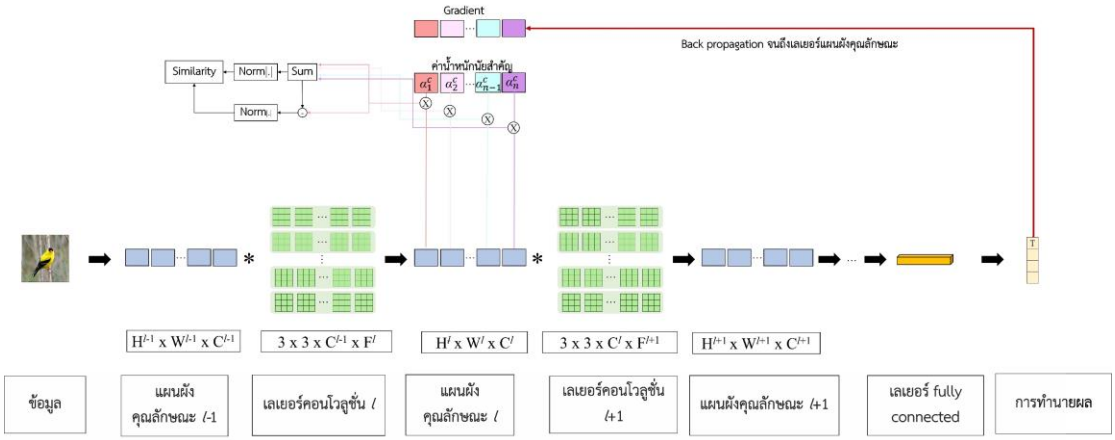
ในบทนี้จะกล่าวถึงงานวิจัยที่นำเสนอ โดยมีเนื้อหาของเทคนิคการวิเคราะห์ตัวกรองที่ใช้แผนผังคุณลักษณะกับ gradient เชิงพื้นที่สำหรับการคัดเลือกตัวกรอง ก่อนทำการยุบตัวกรองจากโครงข่ายประสาทเทียมแบบคอนโวลูชั่น และการเรียกคืนประสิทธิภาพของแบบจำลองโครงข่ายประสาทเทียมแบบคอนโวลูชั่นเพื่อประมาณค่าน้ำหนักของเลเยอร์คอนโวลูชั่นหลังจากถูกยุบตัวกรอง

4.1 เทคนิคการวิเคราะห์ตัวกรองจากค่าแผนผังคุณลักษณะกับ gradient เชิงพื้นที่

เทคนิคการวิเคราะห์ตัวกรองจากค่าแผนผังคุณลักษณะกับ gradient เชิงพื้นที่ หรือ Localized Gradient Activation heatmaP (LGAP) เป็นเทคนิคที่นำมาใช้ในการวิเคราะห์ตัวกรองภายในเลเยอร์คอนโวลูชั่นของโครงข่ายประสาทเทียมแบบคอนโวลูชั่น ซึ่งการวิเคราะห์ตัวกรองจะช่วยให้ทราบว่าตัวกรองแต่ละตัวมีประสิทธิภาพมากน้อยเพียงใด ทำให้สามารถจัดลำดับความสำคัญของตัวกรองก่อนทำการยุบออกจากเลเยอร์คอนโวลูชั่นได้

แผนผังคุณลักษณะกับ gradient เชิงพื้นที่เป็นวิธีการในการคำนวณ gradient ภายหลังจากได้ผลลัพธ์การทำนายคลาส จากการนำรูปภาพผ่านการประมวลผลของแต่ละเลเยอร์ในแบบจำลอง ค่า gradient มีความสำคัญในการระบุตำแหน่งในข้อมูลภาพที่สัมพันธ์กับคลาสผลลัพธ์ ทำให้สามารถคำนวณหาค่าน้ำหนักนัยสำคัญของแผนผังคุณลักษณะแต่ละตัวกรอง ก่อนนำค่าน้ำหนักนัยสำคัญคำนวณร่วมกับแผนผังคุณลักษณะของแต่ละตัวกรอง เพื่อทำการวิเคราะห์การทำงานของตัวกรองในเลเยอร์

ค่า gradient ถูกนำมาใช้ เพื่อนำไปใช้ในการคำนวณหาค่าน้ำหนักนัยสำคัญต่อ โดยสามารถคำนวณได้จากการส่งข้อมูลรูปภาพเข้ามาทำการทำนายผลในโครงข่ายประสาทเทียมแบบคอนโวลูชั่นจนถึงเลเยอร์การทำนายผลคำตอบที่มาค่าสูงสุด (top class prediction) ก่อนเลเยอร์ softmax เมื่อต้องการวิเคราะห์ตัวกรองจากเลเยอร์คอนโวลูชั่น l แผนผังคุณลักษณะของเลเยอร์ l จะถูกนำมาใช้ในการคำนวนค่า gradient ที่เกิดขึ้นจากการทำนายผลเทียบกับแผนผังคุณลักษณะเลเยอร์ l เป็นการทำ back propagation จากเลเยอร์การทำนายผลจนถึงแผนผังคุณลักษณะเลเยอร์ l ผลลัพธ์ที่ได้ออกมาแสดงให้เห็นว่าแผนผังคุณลักษณะของแต่ละตัวกรองมีผลกระทบแค่ไหนกับการทำนายผลลัพธ์ข้อมูลขาเข้า ซึ่งสามารถเห็นภาพรวมในรูปที่ 4.1 ในส่วนของ gradient



รูปที่ 4.1 ภาพรวมการวิเคราะห์ค่าน้ำหนักนัยสำคัญของตัวกรองจากค่าแผนผังคุณลักษณะและ gradient เชิงพื้นที่จากการผ่านรูปภาพตัวอย่างจากชุดข้อมูล ImageNet [2]

การคำนวณ gradient สามารถคำนวณผ่านสมการที่ 4.1 [11]

$$G^{l,k} = \frac{\partial y^c}{\partial A^{l,k}} \tag{4.1}$$

$G^{l,k}$ เป็นการประมาณ gradient ของตัวกรองที่ k ในเลเยอร์ l, y^c เป็นผลลัพธ์จากการทำนายผลคำตอบที่มีค่าสูงสุดของคลาส c, $A^{l,k}$ เป็นผลลัพธ์ของแผนผังคุณลักษณะของตัวกรองที่ k ในเลเยอร์ l

ค่า gradient จะถูกนำมาใช้ในการคำนวณ เพื่อหาค่าน้ำหนักนัยสำคัญของผลลัพธ์ในแต่ละตัวกรองในเลเยอร์ l ซึ่งจะเป็นค่าเฉลี่ยโดยรวมของ gradient ผลลัพธ์จากทุกตัวกรองในเลเยอร์ l สามารถคำนวณได้จากสมการที่ 4.2 [11]

$$\alpha_{l,k}^{c} = \frac{1}{H \mathrm{x} W} \sum_{i=1}^{H} \sum_{j=1}^{W} G_{i,j}^{l,k} \tag{4.2}$$

$\alpha_{l,k}^{c}$ คือ ค่าน้ำหนักนัยสำคัญของตัวกรองที่ k เลเยอร์ l จากการทำนายผลคำตอบที่มีค่าสูงสุดคลาส c, H คือ ความสูงของผลลัพธ์ของแผนผังคุณลักษณะของตัวกรอง และ W คือ ความกว้างของผลลัพธ์ของแผนผังคุณลักษณะของตัวกรองจากการทำนายผลคำตอบที่มีค่าสูงสุดคลาส c

ค่าน้ำหนักนัยสำคัญเป็นส่วนสำคัญในการให้น้ำหนักของผลลัพธ์จากการทำงานของแต่ละตัวกรอง โดยค่าน้ำหนักวิเคราะห์จากค่า gradient ของผลลัพธ์การทำนายผลเทียบกับแผนผังคุณลักษณะ แผนผังคุณลักษณะเป็นผลลัพธ์ที่ได้จากการนำข้อมูลขาเข้าทำคอนโวลูชั่นกับตัวกรองในเลเยอร์ต่าง ๆ ก่อนการทำนายผล ดังนั้นค่า gradient จึงแสดงความสัมพันธ์ระหว่างความสามารถของตัวกรองกับข้อมูลขาเข้าและผลลัพธ์ของการทำนาย เมื่อนำค่าน้ำหนักนัยสำคัญมาคำนวณร่วมกับแผงผังคุณลักษณะของแต่ละตัวกรองในเลเยอร์ l จะทำให้สามารถแสดงการความสำคัญของการ

ทำนายผลข้อมูลที่สัมพันธ์กับค่าคำตอบสูงสุดของเลเยอร์ได้ ซึ่งวิธีการที่แสดงผลออกมาได้ คือ การคำนวณค่าน้ำหนักนัยสำคัญคูณกับแผงผังคุณลักษณะและทำการหา normalization ของค่าสัมบูรณ์จากผลคูณ ในรูปที่ 4.1 คือในส่วนของ Sum และ Norm แสดงวิธีการคำนวณในสมการที่ 4.3

$$H_c = \text{norm}\,(|\sum_k \alpha_{l,k}^c A^{l,k}|) \tag{4.3}$$

H_c คือ แผนผังการตอบรับข้อมูล, |.| คือ ค่าสัมบูรณ์ และ norm คือ normalization ในช่วงค่าสูงสุดและค่าต่ำสุด

ค่าของแผนผังการตอบรับข้อมูล H_c จะแสดงการทำงานของเลเยอร์ l แสดงภาพรวมผลลัพธ์การกรองทุกตัวกรองของเลเยอร์ l แสดงในรูปของค่าสัมบูรณ์ของผลรวมค่าการให้น้ำหนักตามนัยยะสำคัญของแผนผังการตอบสนองแต่ละตัวกรอง ข้อมูลที่เป็นค่าลบเป็นค่าการตอบสนองตัวกรองในทิศตรงข้ามและมีนัยยะสำคัญในเชิงพฤติกรรมของเลเยอร์ ซึ่งถือว่าเป็นตำแหน่งสำคัญและควรรักษาไว้ [11] เพื่อให้การวิเคราะห์พฤติกรรมของตัวกรองในแต่เลเยอร์สามารถทำได้ทั้งค่าบวกและลบ โดยทำการคำนวน normalization เพื่อรักษาระดับของค่าการตอบรับข้อมูล ช่วยให้สามารถนำเป็นมาตรฐาน เมื่อต้องทำการเทียบผลกับส่วนอื่นในการวิเคราะห์ต่อไป

การวิเคราะห์ตัวกรองในเลเยอร์ l สามารถใช้ผลลัพธ์จากสมการที่ 4.3 มาใช้ต่อยอด เพื่อวิเคราะห์ผลกระทบของเลเยอร์ $H_{l,f}$ เมื่อนำผลลัพธ์จากตัวกรอง f ที่ต้องการยุบออกจากแบบจำลอง ช่วยให้ทราบว่าการทำงานของเลเยอร์เปลี่ยนแปลงไปอย่างไร เมื่อนำผลลัพธ์ของตัวกรองที่ถูกวิเคราะห์ออกจากเลเยอร์ l สามารถคำนวณได้จากสมการที่ 4.4

$$H_{l,f} = \text{norm}\,(|\left(\sum_k \alpha_{l,k}^c A^{l,k}\right) - \alpha_{l,f}^c A^{l,f}|) \tag{4.4}$$

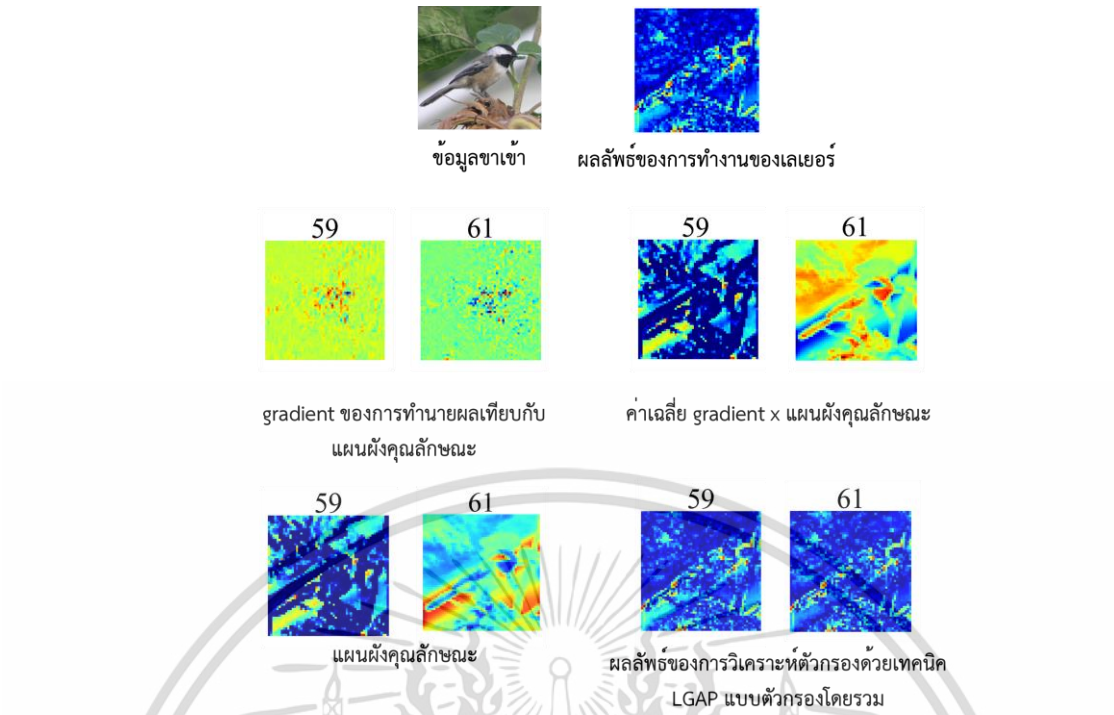
$H_{l,f}$ คือ แผนผังการตอบรับข้อมูล เมื่อยุบตัวกรองที่ถูกวิเคราะห์ออกจากเลเยอร์ และ f คือตัวกรองใด ๆ ที่ถูกวิเคราะห์ การคำนวณในสมการที่ 4.4 จะเป็นการคำนวณค่าในส่วนของ Sum, - และ Norm ในรูปที่ 4.1

การให้คะแนนตัวกรองจะต้องนำสมการที่ 4.3 และ 4.4 มาคำนวณความสามารถในการทำงานว่าแตกต่างไปต่างจากเดิมแค่ไหน โดยการเปรียบเทียบใช้การคำนวณผ่าน cosine similarity ในสมการที่ 4.5 ซึ่งเป็นการหา similarity เพื่อหาความสัมพันธ์เชิงตำแหน่งระหว่างแผนผังการตอบรับข้อมูลของทุกเลเยอร์ H_c และแผนผังการตอบรับข้อมูล เมื่อยุบตัวกรองที่ถูกวิเคราะห์ออกจากเลเยอร์ $H_{l,f}$

$$S_{c,f} = \frac{H_c \cdot H_{l,f}}{\|H_c\|\|H_{l,f}\|} \tag{4.5}$$

$S_{c,f}$ คือ คะแนน similarity ของตัวกรองที่ f โดยคำนวณจาก cosine similarity ของ H_c และ $H_{l,f}$, และ ||.|| คือ ขนาดของเวกเตอร์แบบ Euclidean

กรณีการวิเคราะห์ตัวกรองโดยอาศัยค่า gradient เฉลี่ยเพียงอย่างเดียว หาก gradient เฉลี่ยมีค่าเท่ากันหรือแตกต่างกันเล็กน้อยจะทำให้ไม่สามารถตัดสินใจความสำคัญในการเลือกตัวกรองได้ชัดเจน ตัวอย่างในรูปที่ 4.2 เมื่อส่งข้อมูลรูปภาพ ImageNet [2] เข้าสู่โครงข่ายของแบบจำลอง ResNet-50 ผลลัพธ์ของตัวกรองที่ 59 และ 61 เมื่อวิเคราะห์จากค่า gradient เฉลี่ย ซึ่งมีค่าเฉลี่ย gradient เท่ากับ 2.84E-04 และ -8.21E-05 ตามลำดับ การวิเคราะห์ตัวกรองจากข้อมูลหลายรูป หากพิจารณาโดยใช้เทคนิคค่า gradient เฉลี่ยเพียงอย่างเดียว ตัวกรองที่ 61 จะถูกนำออก (ค่าเฉลี่ย gradient ต่ำกว่า) และตัวกรองที่ 59 จะถูกเก็บ ในขณะที่เทคนิค LGAP ตัวกรองโดยรวมที่นำเสนอ ตัวกรองที่ 59 และ 61 มีค่าคะแนน LGAP ตัวกรองโดยรวมเป็น 0.9917 และ 0.7203 ตามลำดับ ซึ่ง LGAP ตัวกรองโดยรวมให้ความสำคัญกับตัวกรองที่คะแนนต่ำกว่า เนื่องจากส่งผลต่อการทำงานของเลเยอร์สูง ในการวิเคราะห์หลายรูปภาพ LGAP ตัวกรองโดยรวม เลือกที่จะเก็บตัวกรองที่ 61 และตัวกรองที่ 59 จะถูกนำออก จะเห็นว่าการใช้ข้อมูล gradient เฉลี่ยเพียงอย่างเดียวไม่สามารถใช้เป็นข้อมูลในการตัดสินใจได้ดีเพียงพอ สำหรับตัวกรองที่ตอบสนองต่อวัตถุสูงเพียงบางส่วนและกระจายตัว ทำให้ไม่มีน้ำหนักพอจะทำให้คะแนนตัวกรองที่วัดได้สะท้อนความแตกต่างของตัวกรองได้ชัดเจน หากนำค่าเฉลี่ย gradient มาช่วยให้น้ำหนักร่วมกับแผนผังคุณลักษณะ จะทำให้สามารถเน้นกลุ่มของพื้นที่ที่สัมพันธ์กับวัตถุสำหรับตัวกรองที่ตอบสนองกับวัตถุนั้นได้ดีกว่า จึงส่งผลทำให้สามารถจัดลำดับความสำคัญของตัวกรองที่มีค่า gradient ใกล้กันได้ดีกว่าการใช้ข้อมูลค่า gradient เพียงอย่างเดียว ในการวิเคราะห์ความสามารถของการทำงานของตัวกรอง ดังนั้นในงานวิจัยนี้จึงนำเสนอการใช้ข้อมูลจากแผนผังคุณลักษณะร่วมกับค่าเฉลี่ย gradient เพื่อวิเคราะห์การทำงานของตัวกรองได้มีประสิทธิภาพมากขึ้น



รูปที่ 4.2 ผลลัพธ์จากการวิเคราะห์ตัวกรองโดยการส่งข้อมูล ImageNet [2] เข้าไปในโครงข่ายของแบบจำลอง ResNet-50 ของเลเยอร์คอนโวลูชั่นที่ 1

ในการคำนวณคะแนน similarity ของตัวกรองจะหาค่าเฉลี่ยของคะแนน similarity ของทุกตัวกรองในเลเยอร์ l ในสมการที่ 4.6 โดยคำนวณจากผลของคะแนน similarity แต่ละรูปภาพที่ใช้ในการวิเคราะห์ตัวกรองจากสมการที่ 4.5

$$\text{Score}_f = \frac{1}{N}\sum_{n=1}^{N} S_{c,f} \tag{4.6}$$

Score_f คือ คะแนนเฉลี่ย similarity ของตัวกรองที่ f และ N คือ จำนวนข้อมูลรูปภาพที่ใช้ในการวิเคราะห์ตัวกรอง

คะแนนเฉลี่ยของ similarity ของตัวกรองจะถูกนำมาใช้ในจัดลำดับความสามารถของตัวกรองว่าตัวกรองใดมีประสิทธิภาพสูงหรือต่ำ โดยตัวกรองที่มีคะแนนสูงจะแสดงถึงว่ามีประสิทธิภาพต่ำ เนื่องจากการยุบตัวกรองประสิทธิภาพต่ำจะส่งผลกระทบต่อการทำงานของเลเยอร์น้อย ในขณะที่ตัวกรองที่มีคะแนนต่ำจะแสดงถึงว่ามีประสิทธิภาพสูง เนื่องจากหากยุบตัวกรองประสิทธิภาพสูงไปจะส่งผลกระทบต่อการทำงานของเลเยอร์มาก

4.2 การเรียกคืนประสิทธิภาพหลังการยุบตัวกรอง

การเรียกคืนประสิทธิภาพของแบบจำลองหลังการยุบตัวกรอง ทำได้โดยการประมาณค่าน้ำหนักของเลเยอร์คอนโวลูชั่นหรือ Convolutional Approximation Small Model (CASM) ใน

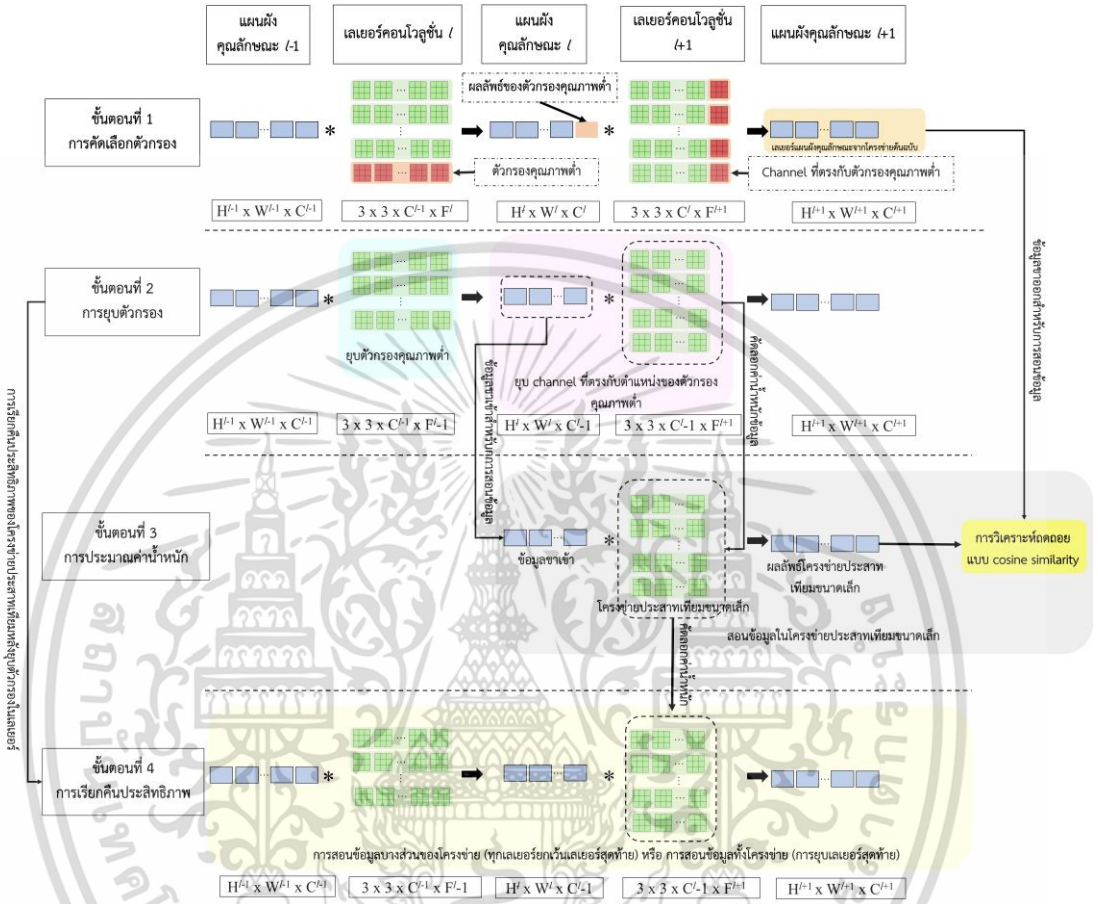
ขั้นตอนที่ 3 ในรูปที่ 4.3 ซึ่งเป็นวิธีการที่จะช่วยเรียกคืนการทำงานของเลเยอร์จากการประมาณค่าน้ำหนักโดยใช้ตัวกรองที่ยังคงเหลืออยู่หลังจากการยุบตัวกรองที่ไม่มีประสิทธิภาพออกจากเลเยอร์ ซึ่งการประมาณค่าน้ำหนักนี้จะช่วยทำให้ประสิทธิภาพของโครงข่ายประสาทเทียมกลับมาใกล้เคียงกับโครงข่ายประสาทเทียมก่อนการยุบตัวกรอง ตัวกรองที่เหลือในเลเยอร์คอนโวลูชั่น l จะให้ผลลัพธ์แผนผังคุณลักษณะที่ลดจำนวน channel ลง ดังนั้นเลเยอร์คอนโวลูชั่น $l+1$ จำนวน channel จะลดลงไปด้วย ขั้นตอนที่เพิ่มขึ้นมาจากเฟรมเวิร์คพื้นฐาน (รูปที่ 4.1) คือ ขั้นตอนการประมาณค่าน้ำหนักที่อยู่ระหว่างขั้นตอนการยุบตัวกรองและขั้นตอนการเรียกคืนประสิทธิภาพ แสดงในรูปที่ 4.3 เพื่อทำให้การทำงานของเลเยอร์กลับมาใกล้เคียงกับโครงข่ายพื้นฐาน ในขั้นตอนที่ 4 จะมีการสอนข้อมูลบางส่วนในโครงข่ายประสาทเทียมสำหรับทุกเลเยอร์ที่ทำการยุบตัวกรอง ยกเว้นการยุบตัวกรองเลเยอร์สุดท้ายที่จะใช้การสอนข้อมูลทั้งโครงข่าย ทำให้ในการยุบตัวกรองในโครงข่ายบางส่วนค่าน้ำหนักจะมีการเปลี่ยนแปลง ในขณะที่ค่าน้ำหนักตัวกรองในส่วนที่ยังไม่ยุบตัวกรองจะคงเดิม ยกเว้นการยุบตัวกรองที่เลเยอร์สุดท้าย ที่มีการสอนข้อมูลทั้งเลเยอร์ ค่าน้ำหนักจะมีการเปลี่ยนแปลงทั้งหมด ขณะที่เฟรมเวิร์คพื้นฐานจะใช้สอนข้อมูลทั้งโครงข่ายทุกเลเยอร์ ซึ่งค่าน้ำหนักของทุกเลเยอร์จะมีการเปลี่ยนแปลงตลอดการยุบตัวกรองแต่ละเลเยอร์ ในงานวิจัยนี้ได้ทำการนำการวิเคราะห์ถดถอยแบบ cosine similarity มาเป็นเครื่องมือในการทำนายผลการสอนโครงข่ายประสาทเทียมขนาดเล็กนี้ เพื่อตรวจสอบผลการทำนายของโครงข่ายประสาทเทียมขนาดเล็กนี้ว่ามีผลเปลี่ยนแปลงไปอย่างไรเมื่อเทียบกับโครงข่ายประสาทเทียมต้นฉบับ

เลเยอร์คอนโวลูชั่น $l+1$ จะถูกประมาณค่าน้ำหนักใหม่โดยการสอนข้อมูลใหม่จากการใช้ข้อมูลผลลัพธ์จากเลเยอร์แผนผังคุณลักษณะที่ $l+1$ ของโครงข่ายประสาทเทียมที่ยังไม่ได้ทำการยุบตัวกรองเป็นผลลัพธ์เป้าหมายและใช้ข้อมูลขาเข้าเป็นเลเยอร์แผนผังคุณลักษณะที่ l ของโครงข่ายประสาทเทียมที่ยุบตัวกรองแล้ว ตรงจุดนี้จะเป็นการสอนโครงข่ายประสาทเทียมขนาดเล็กเกิดขึ้นภายในโครงข่ายประสาทเทียมขนาดใหญ่ การสอนโครงข่ายประสาทเทียมขนาดเล็กถูกทำขึ้น เพื่อประมาณค่าน้ำหนักของเลเยอร์ $l+1$ ในแบบจำลองที่ถูกยุบให้เหมือนกับผลลัพธ์เป้าหมายที่ได้จากเลเยอร์ $l+1$ ของแบบจำลองต้นฉบับ สามารถใช้ optimization พื้นฐานในเชิงพื้นที่สำหรับการใช้ในฟังก์ชันวัตถุประสงค์ (objective function) ตามสมการที่ 4.7 [10] และใช้การวัดประสิทธิภาพการทำงานโดย cosine similarity

$$\text{objective function} = \frac{1}{2}\|h^{*}x - y\|^{2} \tag{4.7}$$

เมื่อ h คือ เลเยอร์คอนโวลูชั่นขนาดเล็ก, x คือ ผลลัพธ์จากเลเยอร์แผนผังคุณลักษณะ l และ y คือ ผลลัพธ์เป้าหมายจากเลเยอร์แผนผังคุณลักษณะที่ $l+1$ ของแบบจำลองต้นฉบับ

สำหรับการสอนข้อมูลในขั้นตอนที่ 4 ทุกเลเยอร์ที่ถูกยุบตัวกรอง ยกเว้นเลเยอร์สุดท้ายของแบบจำลองต้นฉบับ ทุกเลเยอร์ก่อน l+1 จนถึงเลเยอร์ l+1 และเลเยอร์ fully connected จะถูกสอนข้อมูลซ้ำ สำหรับการยุบตัวกรองที่เลเยอร์สุดท้ายจะทำการสอนข้อมูลซ้ำทุกเลเยอร์



รูปที่ 4.3 CASM เฟรมเวิร์คสำหรับการประมาณค่าน้ำหนักของเลเยอร์คอนโวลูชั่นหลังการยุบตัวกรอง

บทที่ 5
การทดลอง

บทนี้จะกล่าวถึงการทดลองยุบโครงข่ายประสาทเทียมคอนโวลูชั่นโดยใช้เทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ (LGAP) และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น (CASM) โดยเริ่มจากนำเสนอข้อมูลและโครงข่ายประสาทเทียมแบบคอนโวลูชั่นที่ใช้ในการทดลอง วิธีการในการทดลองแบบจำลองที่ถูกยุบโครงสร้างและการเรียกคืนประสิทธิภาพ การวัดผลของแบบจำลองที่ถูกยุบโครงข่ายและการเรียกคืนประสิทธิภาพ

5.1 ข้อมูลและโครงข่ายประสาทเทียมแบบคอนโวลูชั่น

เป้าหมายในการทำการยุบตัวกรองในงานวิจัยที่นำเสนอ คือ ต้องการทดสอบการยุบตัวกรองในเลเยอร์คอนโวลูชั่น ดังนั้นจึงเลือกแบบจำลองที่มีพื้นฐานเลเยอร์คอนโวลูชั่นนำมาทดสอบในงานวิจัย โดยเลือกแบบจำลองพื้นฐานที่มีเพียงเลเยอร์คอนโวลูชั่น คือ VGG-16 และเลือกแบบจำลองที่มีโครงสร้างพัฒนามาจากเลเยอร์คอนโวลูชั่นพื้นฐาน ควบคู่กับโครงสร้างพิเศษ คือ ResNet-50 เพื่อแสดงให้เห็นว่าเทคนิคการยุบตัวกรองที่งานวิจัยนี้นำเสนอ สามารถประยุกต์ใช้กับแบบจำลองใดๆ ที่มีพื้นฐานโครงสร้างของเลเยอร์คอนโวลูชั่นได้อย่างมีประสิทธิภาพ

ข้อมูลที่ใช้ในการยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่นมี 2 ชุด คือ ชุดข้อมูล CIFAR-10 [1] และ ชุดข้อมูล ImageNet (ILSVRC 2012) [2]

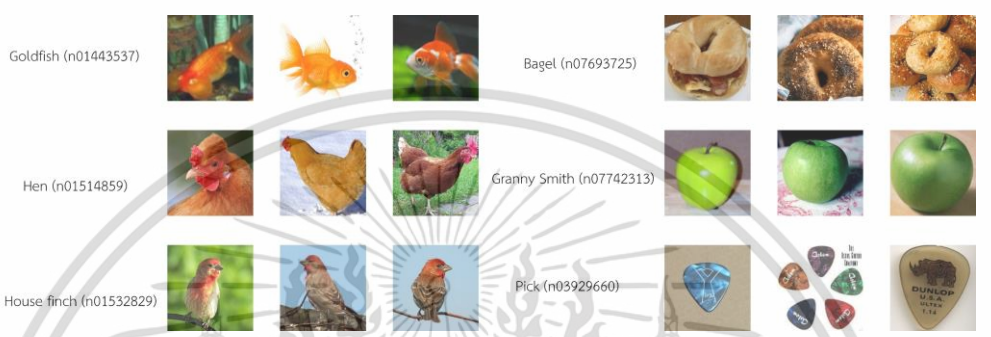
5.1.1 ชุดข้อมูล CIFAR-10

ชุดข้อมูล CIFAR-10 เป็นชุดข้อมูลที่ประกอบไปด้วยรูปภาพจำนวน 60,000 ภาพ โดยแต่ละภาพมีขนาดเป็น 32x32 พิเซล มีประเภทของข้อมูล 10 ประเภท ข้อมูลที่ใช้ในการสอนข้อมูลมี 50,000 รูป และข้อมูลสำหรับการทดสอบ 10,000 รูป

รูปที่ 5.1 ตัวอย่างชุดข้อมูล CIFAR-10 [1]

5.1.2 ชุดข้อมูล ImageNet (ILSVRC 2012)

ชุดข้อมูล ImageNet (ILSVRC 2012) เป็นชุดข้อมูลขนาดใหญ่ที่ประกอบไปด้วยข้อมูล 1,000 ประเภท แตกต่างกันออกไป มีจำนวนข้อมูลที่ใช้ในการสอนประมาณ 1.2 ล้านรูป และข้อมูลที่ใช้การตรวจสอบ 50,000 รูป การใช้งานชุดข้อมูล ImageNet ต้องเตรียมข้อมูลโดยการทำการตัดภาพจากจุดศูนย์กลางให้มีขนาด 224x224 พิเซล และต้องใช้เทคนิคการขยายข้อมูลให้มีความหลากหลาย (augmentation technique) โดยใช้วิธีการแปลงแบบพลิกในลักษณะแนวนอน (horizontal flip)



รูปที่ 5.2 ตัวอย่างชุดข้อมูลจาก ImageNet [2]

ชุดข้อมูลทั้ง 2 แบบเป็นชุดข้อมูลที่ใช้ในงานด้านการแยกประเภทข้อมูล ซึ่งในงานวิจัยนี้จะนำมาใช้ทดสอบประสิทธิภาพของแบบจำลองหลังจากที่มีการยุบโครงสร้าง

โครงข่ายประสาทเทียมที่ใช้ในการทดสอบประสิทธิภาพการยุบตัวกรองจะใช้โครงข่าย VGG-16 และ ResNet-50 โครงข่ายประสาทเทียมทั้ง 2 แบบเป็นโครงข่ายที่ถูกใช้อย่างกว้างขวางในหลากหลายงาน

5.1.3 VGG-16

VGG-16 [3] เป็นโครงข่ายประสาทเทียมแบบคอนโวลูชั่นที่ถูกออกแบบมาให้ใช้กับการแยกประเภทข้อมูลจากการใช้ข้อมูลขนาดใหญ่ในการสอนข้อมูล โครงสร้างพื้นฐานถูกออกแบบมาให้ใช้กับชุดข้อมูล ImageNet ซึ่งถูกนำมาปรับใช้กับชุดข้อมูล CIFAR-10 [12, 13, 14] และสามารถแยกประเภทข้อมูลของ CIFAR-10 ได้อย่างมีประสิทธิภาพ VGG-16 ที่ถูกประยุกต์ใช้ชุดข้อมูล CIFAR-10 มีโครงสร้างของเลเยอร์คอนโวลูชั่นทั้งหมด 13 เลเยอร์ โดยตัวกรองในเลเยอร์มีขนาดเป็น 3x3 โดยในแต่ละเลเยอร์มีองค์ประกอบตามตารางที่ 5.1 จำนวนพารามิเตอร์ของเลเยอร์คอนโวลูชั่นทั้งหมดมี 6.3E+08 และจำนวน FLOPs มีทั้งหมด 1.5E+07

โครงสร้างของ VGG-16 ถูกเปลี่ยนในส่วนของเลเยอร์ fully-connected จาก 1000 ประเภทที่ใช้กับ ImageNet เป็น 10 ประเภท ตามข้อมูล CIFAR-10 เพื่อทดสอบการยุบตัวกรองกับชุดข้อมูลขนาดเล็กและใช้ในการเปรียบเทียบประสิทธิภาพกับวิธีการวิเคราะห์ตัวกรองจากงานวิจัย [9, 10] โครงสร้าง VGG-16 ที่ใช้กับชุดข้อมูล CIFAR-10 มีพารามิเตอร์โดยรวมของโครงข่ายที่มีขนาดลดลงเป็นจำนวนมาก

5.1.4 ResNet-50

เป็นโครงข่ายประสาทเทียมแบบคอนโวลูชั่นที่ถูกพัฒนาในการแยกประเภทข้อมูลกับชุดข้อมูลขนาดใหญ่ ImageNet ซึ่งมีศักยภาพสูงกว่า VGG-16 ในการประมวลผล ResNet-50 มีโครงสร้างพิเศษที่เรียกว่า Residual block (แสดงในรูปที่ 5.3) โครงสร้างนี้ส่งผลให้ไม่สามารถยุบโครงสร้างของเลเยอร์คอนโวลูชั่นที่ 3 ได้ เนื่องจากในการยุบตัวกรองออกจะส่งให้ผลลัพธ์ที่ออกมามีจำนวนลดลง ทำให้ไม่สามารถทำการคำนวณต่อในโครงข่าย ดังนั้นการยุบตัวกรองในโครงข่ายประสาทเทียม ResNet-50 จะทำกับเลเยอร์คอนโวลูชั่นที่ 1 และ 2 โดยเลเยอร์คอนโวลูชั่นที่ 3 ในแต่ละบล๊อกจะไม่ทำการยุบตัวกรองเพื่อรักษาโครงสร้างข้อมูลในการประมวลผลใน Residual block ไว้

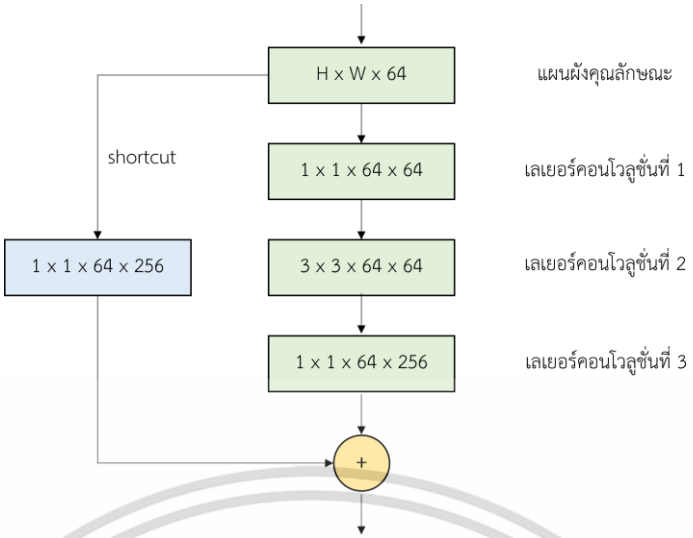
ตารางที่ 5.1 โครงสร้างเลเยอร์คอนโวลูชั่นในแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10

ชื่อเลเยอร์	ขนาดตัวกรอง (SxSxCxN)	FLOPs	จำนวนพารามิเตอร์
conv2d	3x3x3x64	3.5E+06	1.8E+03
conv2d_1	3x3x64x64	7.5E+07	3.7E+04
conv2d_2	3x3x64x128	3.8E+07	7.4E+04
conv2d_3	3x3x128x128	7.5E+07	1.5E+05
conv2d_4	3x3x128x256	3.8E+07	3.0E+05
conv2d_5	3x3x256x256	7.5E+07	5.9E+05
conv2d_6	3x3x256x256	7.5E+07	5.9E+05
conv2d_7	3x3x256x512	3.8E+07	1.2E+06
conv2d_8	3x3x512x512	7.5E+07	2.4E+06
conv2d_9	3x3x512x512	7.5E+07	2.4E+06
conv2d_10	3x3x512x512	1.9E+07	2.4E+06
conv2d_11	3x3x512x512	1.9E+07	2.4E+06
conv2d_12	3x3x512x512	1.9E+07	2.4E+06

Conv2d คือ เลเยอร์คอนโวลูชั่นที่มีขนาด 2 มิติ

SxSxCxN คือ ขนาดของตัวกรอง, SxS คือ ขนาดของหน้าต่าง, C คือ จำนวนของ channel และ N คือ จำนวนของตัวกรอง

FLOPs คือ Floating Point Operation



รูปที่ 5.3 โครงสร้าง Residual block

โครงสร้างของ ResNet-50 จะประกอบไปด้วย Residual block จำนวน 16 อัน โครงสร้างของ ResNet-50 แสดงในตารางที่ 5.2

ตารางที่ 5.2 โครงสร้างเลเยอร์คอนโวลูชั่นในแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet

ชื่อเลเยอร์	**ขนาดตัวกรอง (S x S / N)**	**จำนวน (block)**	**ขนาดของแผนผังคุณลักษณะ**
Conv2d_1	7x7/64	1	112x112
Conv2d_2_x	1x1/64 3x3/64 1x1/256	3	56x56
Conv2d_3_x	1x1/128 3x3/128 1x1/512	4	28x28
Conv2d_4_x	1x1/256 3x3/256 1x1/1024	6	14x14
Conv2d_5_x	1x1/512 3x3/512 1x1/2048	3	7x7

Conv2d คือ เลเยอร์คอนโวลูชั่นที่มีขนาด 2 มิติ

SxS/N คือ ขนาดของตัวกรอง, SxS คือ ขนาดของหน้าต่าง, และ N คือ จำนวนของตัวกรอง

จำนวนพารามิเตอร์ทั้งหมดของ ResNet-50 คือ 2.6E+07 และ FLOPs ทั้งหมดมีค่าเป็น 7.7E+09

ในการทดลองนี้จะทดลองทั้งการใช้เฟรมเวิร์คพื้นฐานและ CASM เฟรมเวิร์คของแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น โดยตารางที่ 5.3 จะแสดงพารามิเตอร์ที่ใช้ในการทำ fine-tuning สำหรับเฟรมเวิร์คพื้นฐาน พารามิเตอร์ที่ใช้ในทดลองในส่วนของ learning rate ช่วง 0.01 – 0.000001 ในการทดลองเป็นช่วงที่กว้าง เพื่อดูความครอบคลุมในงานวิจัย ซึ่งบางส่วนตรงกับงานวิจัย [10, 15] ในส่วนของ optimizer, momentum และ weight decay พารามิเตอร์ตรงกับงานวิจัย [10, 15] นอกจากนี้พารามิเตอร์ epoch ที่ใช้ทดลอง 1-2 epoch ก่อนเลเยอร์สุดท้ายใน VGG-16 ได้ถูกทดลอง เพื่อหาพารามิเตอร์ที่เหมาะสม แต่ในเลเยอร์สุดท้ายมีการทดลองในส่วนของงานวิจัย [10] สำหรับแบบจำลอง ResNet-50 มีการปรับ epoch ในช่วง 1-20 ของเลเยอร์ก่อนเลเยอร์สุดท้าย เพื่อหาพารามิเตอร์ที่เหมาะสมในการเรียกคืนประสิทธิภาพ ส่วน epoch ของเลเยอร์สุดท้าย เป็นพารามิเตอร์ที่ใช้ในงานวิจัย [10]

ตารางที่ 5.3 พารามิเตอร์ที่ดีที่สุดสำหรับ fine-tuning ในเฟรมเวิร์คพื้นฐาน

ชื่อแบบจำลอง	**ชุดข้อมูล**	**พารามิเตอร์**	**ค่า**
VGG-16	CIFAR-10	optimizer	SGD
		learning rate	0.001
		momentum	0.9
		weight decay	0.0005
ResNet-50	ImageNet	optimizer	SGD
		learning rate	0.001, 0.0001
		momentum	0.9
		weight decay	0.0005

สำหรับ Batch size ที่ใช้ในการทดลอง VGG-16 และ ResNet-50 จะใช้ 32 และ epoch ในการยุบโครงสร้างตัวกรองเลเยอร์สุดท้าย VGG-16 ใช้ 40 รอบ และ ResNet-50 ใช้ 50 รอบ และ learning rate จะใช้ 0.001-0.00001

ในตารางที่ 5.4 จะแสดงข้อมูลพารามิเตอร์ที่ใช้ในการเรียกคืนประสิทธิภาพของเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น

สำหรับ Batch size จะใช้ค่าเดียวกับเฟรมเวิร์คพื้นฐาน จำนวน epoch ในการยุบโครงสร้างตัวกรองเลเยอร์สุดท้ายเป็น 40 รอบ ทั้ง 2 แบบจำลอง และ learning rate จะใช้ 0.001-0.00001

ตารางที่ 5.4 พารามิเตอร์สำหรับการประมาณคอนโวลูชั่นของเฟรมเวิร์คแบบจำลองขนาดเล็ก

พารามิเตอร์	**ค่า**
optimizer	SGD
learning rate	0.001 – 0.0001
momentum	0.9
weight decay	0.0005
epoch สำหรับสอนข้อมูล	500

5.2 การยุบโครงสร้างตามอัตราส่วน

ในการยุบโครงสร้างตามอัตราส่วน สำหรับการยุบโครงสร้างของ VGG-16 ด้วยชุดข้อมูล CIFAR-10 จะทำการยุบตัวกรองออกทีละ 10% จนถึง 90% ของอัตราส่วน สำหรับแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet จะทำยุบตัวกรองออกเป็น 30%, 50% และ 70% ตามลำดับ เนื่องจาก ResNet-50 เป็นโครงข่ายที่มีถูกออกแบบมาด้วยชุดข้อมูลขนาดใหญ่ ดังนั้นในการทำการยุบแต่ละครั้งจะใช้เวลาในการประมวลผลสูง

5.3 การวิเคราะห์ตัวกรอง

การวิเคราะห์ตัวกรองจะวิเคราะห์ด้วยเทคนิคตามสมการที่ 4.4 และใช้สุ่มข้อมูลบางส่วนจากชุดข้อมูลสำหรับสอนมาใช้ในการวิเคราะห์ โดย VGG-16 กับชุดข้อมูล CIFAR-10 จะใช้จำนวน 10 รูปต่อประเภท รวม 100 รูป ในการวิเคราะห์ตัวรอง สำหรับแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet จะใช้ 10 รูปต่อประเภทเช่นเดียวกัน รวม 10,000 รูป หลังจากที่ได้ผลลัพธ์ออกของแต่ละรูปออกมาจากสมการที่ 4.4 จะนำผลลัพธ์มาคำนวณคะแนนโดยรวมของแต่ละตัวกรองผ่านสมการที่ 4.5

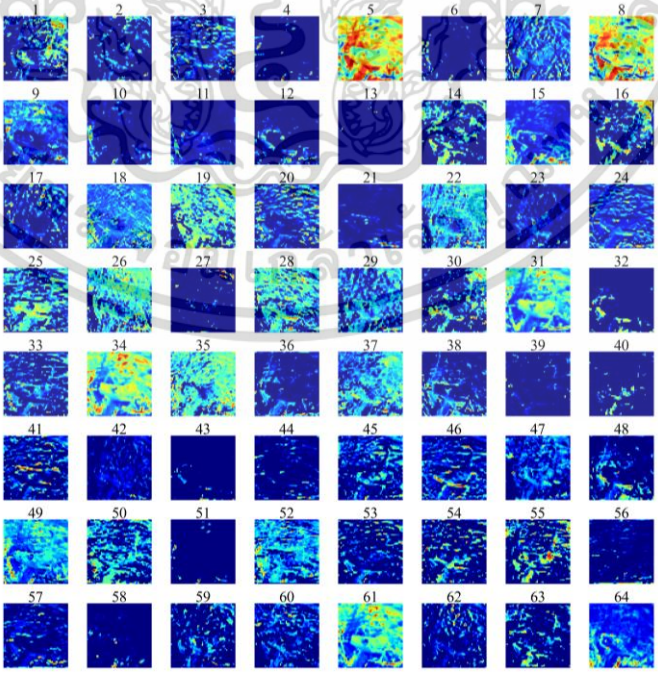
เมื่อนำข้อมูลรูปภาพจาก ImageNet ไปคำนวณตามสมการที่ 4.3 จากการส่งข้อมูลผ่านโครงข่ายประสาทเทียม ResNet-50 ที่เลเยอร์คอนโวลูชั่นที่ 1 จะได้ผลลัพธ์ที่แสดงในรูปที่ 5.4 (ด้านขวา)

แผนผังคุณลักษณะของแต่ละตัวกรองในเลเยอร์คอนโวลูชั่นที่ 1 แสดงในรูปที่ 5.5 ซึ่งเป็นการแสดงการทำงานของแต่ละตัวกรอง เมื่อส่งข้อมูลรูปภาพที่ 5.4 (ด้านซ้าย) เข้าไปประมวลผลในโครงข่ายประสาทเทียม

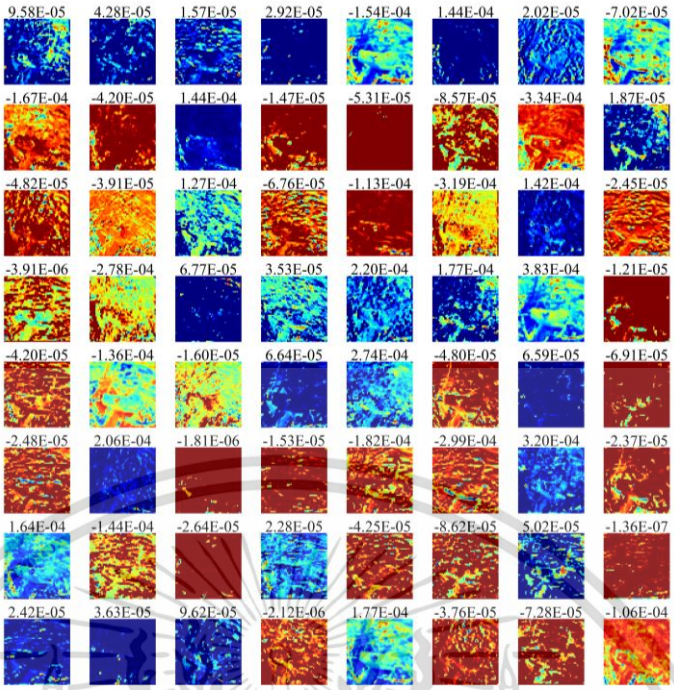
ข้อมูลรูปภาพ ผลลัพธ์จากสมการที่ 4.3

รูปที่ 5.4 ข้อมูลรูปภาพจาก ImageNet [2] ประเภทข้อมูล n01631663 (คลาส Eft) (ด้านซ้าย), ผลลัพธ์จากสมการที่ 4.3 ของแผนผังคุณลักษณะจากเลเยอร์คอนโวลูชั่นที่ 1 (ด้านขวา)

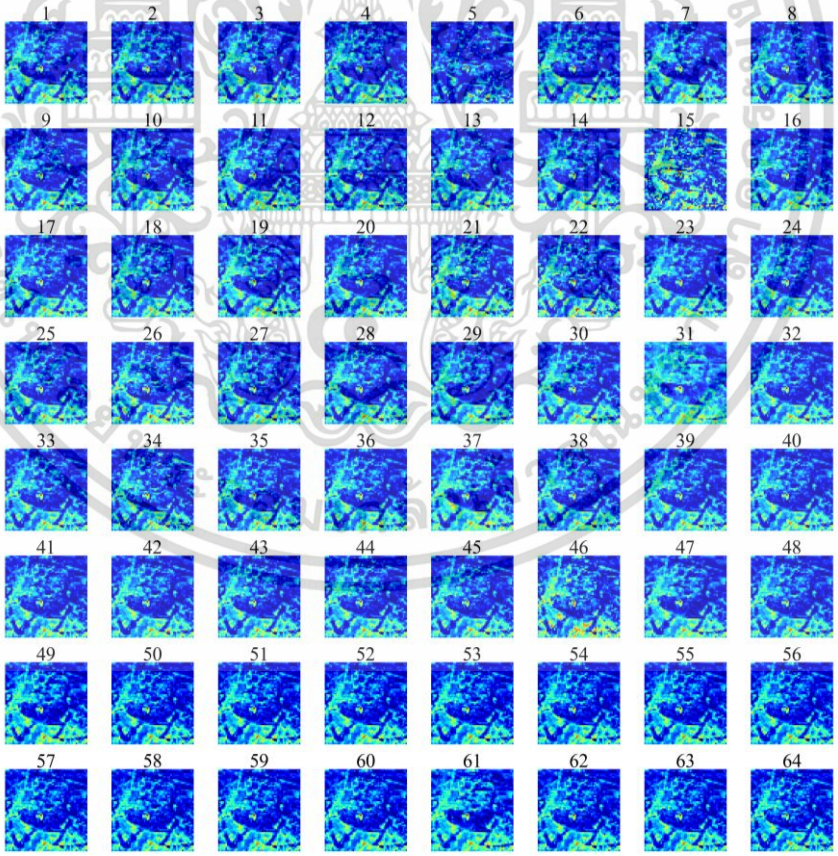
เมื่อคำนวณหาค่าน้ำหนักนัยสำคัญของแต่ละตัวกรอง (สมการที่ 4.2) และทำการคูณกับแต่ละตัวกรองจะได้ผลลัพธ์ตามในรูปที่ 5.6 ซึ่งค่าน้ำหนักนัยสำคัญถูกแสดงอยู่บนผลลัพธ์ของแต่ละตัวกรอง หลังจากได้ผลลัพธ์จากรูปที่ 5.6 เมื่อนำไปคำนวณตามสมการที่ 4.4 จะได้ผลลัพธ์ของแต่ละตัวกรองที่ถูกตัดออกจากการทำงานของเลเยอร์ แสดงในรูปที่ 5.7 ซึ่งข้อมูลนี้จะถูกนำไปใช้ต่อในการคำนวณในสมการที่ 4.5 เพื่อหาว่าตัวกรองแต่ละตัว เมื่อถูกตัดออกจากการทำงานและส่งผลให้การทำงานของเลเยอร์มีการเปลี่ยนแปลงมากน้อยเพียงใด ตัวอย่างตัวกรองที่มีผลทำให้รูปที่ 5.4 ฝั่งขวาเปลี่ยนแปลงไปมาก คือ ตัวกรองที่ 5 และ ตัวกรองที่ 15 ในรูปที่ 5.7 ซึ่งตัวกรองเหล่านี้จะถูกมองว่ามีความสามารถสูงจำเป็นต้องเก็บไว้ในเลเยอร์



รูปที่ 5.5 ข้อมูลเลเยอร์แผนผังคุณลักษณะจากเลเยอร์คอนโวลูชั่นที่ 1



รูปที่ 5.6 ข้อมูลเลเยอร์แผนผังคุณลักษณะจากเลเยอร์คอนโวลูชั่นที่ 1 ที่คูณกับค่าน้ำหนักนัยสำคัญ



รูปที่ 5.7 ข้อมูลผลลัพธ์ของเลเยอร์คอนโวลูชั่นที่ 1 เมื่อนำตัวกรองแต่ละตัวออกจากผลลัพธ์ของสมการที่ 4.3

บทที่ 6
ผลการทดลองและการวิเคราะห์ผล

ในบทนี้จะกล่าวถึงผลการทดลองจากยุบตัวกรองออกจากโครงสร้างด้วยเทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่นผ่านแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 และแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet เพื่อแสดงประสิทธิภาพของเทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่นในการเรียกคืนประสิทธิภาพ การเปรียบเทียบการใช้เทคนิคการวิเคราะห์ด้วยการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่กับเฟรมเวิร์คพื้นฐาน ร่วมทั้งการเปรียบเทียบกับการวิเคราะห์ตัวกรองด้วยเทคนิคอื่น ๆ

6.1 ผลลัพธ์ของการยุบโครงสร้าง VGG-16 กับชุดข้อมูล CIFAR-10

ในส่วนนี้จะแสดงผลลัพธ์การยุบโครงสร้างของ VGG-16 กับชุดข้อมูล CIFAR-10 โดยผลลัพธ์ที่ถูกแสดงมีดังต่อไปนี้

1. ผลประสิทธิภาพของแบบจำลอง VGG-16 ถูกยุบตัวกรองในแต่ละเลเยอร์แบบต่อเนื่องกัน
2. ผลประสิทธิภาพของแบบจำลอง VGG-16 ถูกยุบตัวกรองแบบแต่ละเลเยอร์แยกจากกัน
3. ผลความมั่นใจในการยุบตัวกรองของแต่ละเทคนิค
4. ผลประสิทธิภาพของแบบจำลอง VGG-16 ถูกยุบตัวกรองของ LGAP ในแต่ละเฟรมเวิร์ค

6.1.1 ผลประสิทธิภาพของแบบจำลอง VGG-16 ถูกยุบตัวกรองในแต่ละเลเยอร์แบบต่อเนื่องกัน

แบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 มีความแม่นยำในการทำนายผลในการแยกประเภทข้อมูลเป็น 0.9359 เลเยอร์ที่ถูกยุบจะเริ่มจาก เลเยอร์ conv2d_1 (ยุบ 1 เลเยอร์) ไปจนถึงเลเยอร์ conv2d_12 (ยุบ 12 เลเยอร์) ในตารางที่ 6.1-6.7 ในแต่ละตารางจะใช้เทคนิคการวิเคราะห์ที่แตกต่างกัน ได้แก่ เทคนิคแบบสุ่ม เทคนิคผลรวมของค่าน้ำหนัก (Weighted sum technique) [7] เทคนิคผลรวมของค่าเฉลี่ยพื้นที่ศูนย์ (APoZ: the average percentage of zeros) [8] เทคนิคผลรวมของค่าเฉลี่ย gradient (the mean gradient) [9] เทคนิค LGAP แบบตัวกรองเดี่ยว เทคนิค LGAP แบบตัวกรองโดยรวม และ เทคนิค LGAP แบบตัวกรองโดยรวมร่วมกับเฟรมเวิร์ค CASM หลังจากการยุบตัวกรองจะมีการเรียกคืนประสิทธิภาพ ซึ่งพารามิเตอร์แต่ละเลเยอร์เป็นไปตามตารางที่ 5.3 และจำนวน 2 epoch และการยุบตัวกรองเลเยอร์สุดท้ายจะเรียกคืนประสิทธิภาพโดยใช้ 40

epoch ซึ่งตารางที่ 6.1-6.6 จะใช้เฟรมเวิร์คแบบเดียวกัน คือ เฟรมเวิร์คพื้นฐาน สำหรับตารางที่ 6.7 เป็นผลจากการใช้เฟรมเวิร์ค CASM กับการสอนข้อมูลซ้ำในบางส่วนของเลเยอร์ เพื่อเรียกคืนประสิทธิภาพ (พารามิเตอร์ใช้ตามตารางที่ 5.3) หลังจากยุบตัวกรองในแต่ละเลเยอร์ ยกเว้นเลเยอร์สุดท้ายที่ทำการสอนข้อมูลทั้งเลเยอร์เช่นเดียวกับเฟรมเวิร์คพื้นฐาน

ผลการยุบตัวกรองโดยเทคนิคแบบสุ่ม แสดงในตารางที่ 6.1 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อย จาก 0.9359 เหลือ 0.9331 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้นประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.7633 จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเลเยอร์ที่ 6 ประสิทธิภาพจะลดลงต่ำสุด และค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 7 จนถึงเลเยอร์ 12 ซึ่งในเลเยอรที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืนประสิทธิภาพ

ตารางที่ 6.1 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิคแบบสุ่มร่วมกับเฟรมเวิร์คพื้นฐาน

จำนวนเลเยอร์ที่ถูกยุบ	**อัตราส่วนการยุบ**								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.7633	0.9002	0.9152	0.9161	0.9206	0.9234	0.9279	0.9304	0.9331
2	0.7871	0.8827	0.9016	0.9073	0.9165	0.9191	0.9254	0.9285	0.9332
3	0.7534	0.8563	0.8924	0.8992	0.9039	0.9118	0.9206	0.9252	0.9271
4	0.7415	0.8520	0.8759	0.8954	0.9039	0.9115	0.9198	0.9230	0.9277
5	0.7073	0.8332	0.8701	0.8865	0.8975	0.9111	0.9186	0.9242	0.9261
6	0.6704	0.8047	0.8563	0.8818	0.8935	0.9030	0.9102	0.9189	0.9256
7	0.6734	0.8015	0.8573	0.8784	0.8953	0.9050	0.9173	0.9227	0.9261
8	0.6887	0.8078	0.8613	0.8840	0.8976	0.9069	0.9154	0.9194	0.9270
9	0.6769	0.8166	0.8659	0.8860	0.8995	0.9091	0.9165	0.9212	0.9273
10	0.6896	0.8200	0.8694	0.8918	0.9034	0.9095	0.9161	0.9232	0.9275
11	0.6774	0.8132	0.8699	0.8876	0.9029	0.9094	0.9160	0.9213	0.9268
12	0.7386	0.8488	0.8925	0.9083	0.9180	0.9238	0.9271	0.9294	0.9338

ผลการยุบตัวกรองโดยเทคนิคผลรวมของค่าน้ำหนัก (Weighted Sum) แสดงในตารางที่ 6.2 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อย จาก 0.9359 เหลือ 0.9346 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้นประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.8801 จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเลเยอร์ที่ 7 ประสิทธิภาพจะลดลงต่ำสุด และค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 8 จนถึงเลเยอร์ 12 ซึ่งในเลเยอรที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืนประสิทธิภาพ

ตารางที่ 6.2 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิคผลรวมของค่าน้ำหนักร่วมกับเฟรมเวิร์คพื้นฐาน

จำนวนเลเยอร์ที่ถูกยุบ	**อัตราส่วนการยุบ**								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.8801	0.9080	0.9225	0.9239	0.9294	0.9320	0.9332	0.9340	0.9346
2	0.8487	0.8926	0.9113	0.9168	0.9232	0.9277	0.9290	0.9309	0.9327
3	0.8068	0.8704	0.8926	0.9041	0.9118	0.9175	0.9269	0.9268	0.9312
4	0.7848	0.8583	0.8839	0.9030	0.9068	0.9140	0.9244	0.9272	0.9323
5	0.7448	0.8221	0.8795	0.8917	0.9087	0.9152	0.9225	0.9284	0.9303
6	0.7227	0.8235	0.8686	0.8855	0.9083	0.9105	0.9225	0.9269	0.9298
7	0.7109	0.8221	0.8643	0.8899	0.9037	0.9081	0.9195	0.9269	0.9296
8	0.7199	0.8246	0.8697	0.8923	0.9077	0.9144	0.9226	0.9275	0.9300
9	0.7049	0.8309	0.8720	0.8963	0.9092	0.9161	0.9248	0.9283	0.9313
10	0.7110	0.8328	0.8720	0.8985	0.9112	0.9138	0.9232	0.9286	0.9290
11	0.7276	0.8331	0.8769	0.8968	0.9130	0.9182	0.9243	0.9240	0.9313
12	0.7569	0.8603	0.8970	0.9153	0.9249	0.9271	0.9336	0.9346	0.9361

ผลการยุบตัวกรองโดยเทคนิคค่าเฉลี่ยพื้นที่ศูนย์ (APoZ) แสดงในตารางที่ 6.3 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อย จาก 0.9359 เหลือ 0.9334 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้นประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.7595 ในกรณีอัตราส่วนการบีบอัด 10%-70% จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเล

เยอร์ที่ 6 และ 7 ประสิทธิภาพจะลดลงต่ำสุด และค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 7 และ 8 จนถึงเลเยอร์ 12 ซึ่งในเลเยอรที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืนประสิทธิภาพ สำหรับอัตราการบีบที่ 80%-90% พบว่าประสิทธิภาพในการทำงานของแบบจำลองลดลงเป็นอย่างมาก ซึ่งแสดงให้เห็นชัดเจนว่าการยุบตัวกรองโดยเทคนิคผลรวมของค่าเฉลี่ยพื้นที่ศูนย์ไม่สามารถรักษาประสิทธิภาพไว้ได้ โดยประสิทธิภาพตกลงจนไม่สามารถใช้งานแบบจำลองได้

ตารางที่ 6.3 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิคผลรวมของค่าเฉลี่ยพื้นที่ศูนย์ร่วมกับเฟรมเวิร์คพื้นฐาน

จำนวนเลเยอร์ที่ถูกยุบ	อัตราส่วนการยุบ								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.7595	0.8549	0.8958	0.9113	0.9204	0.9217	0.9270	0.9316	0.9334
2	0.2287	0.5798	0.8781	0.9027	0.9120	0.9166	0.9216	0.9267	0.9330
3	0.1000	0.5875	0.8514	0.8878	0.9048	0.9093	0.9184	0.9244	0.9295
4	0.1000	0.3958	0.8469	0.8891	0.8955	0.9063	0.9165	0.9226	0.9299
5	0.1000	0.3968	0.8425	0.8770	0.8975	0.9037	0.9140	0.9232	0.9278
6	0.1000	0.3952	0.8285	0.8692	0.8883	0.9025	0.9124	0.9209	0.9277
7	0.1000	0.4046	0.8263	0.8776	0.8910	0.8994	0.9100	0.9233	0.9269
8	0.1000	0.4297	0.8294	0.8764	0.8962	0.9097	0.9149	0.9255	0.9261
9	0.1000	0.4292	0.8331	0.8811	0.8953	0.9043	0.9161	0.9239	0.9315
10	0.1000	0.3978	0.8325	0.8845	0.8990	0.9068	0.9135	0.9264	0.9300
11	0.1000	0.4356	0.8357	0.8822	0.9023	0.9095	0.9150	0.9244	0.9286
12	0.1000	0.4816	0.8694	0.9081	0.9183	0.9216	0.9255	0.9345	0.9335

ผลการยุบตัวกรองโดยเทคนิคค่าเฉลี่ย gradient แสดงในตารางที่ 6.4 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อย จาก 0.9359 เหลือ 0.9343 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้นประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.8428 ในกรณีอัตราส่วนการบีบอัด 10%-80% จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเลเยอร์ที่ 6 และ 7 ประสิทธิภาพจะลดลงต่ำสุด และค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 7 และ 8 จนถึงเลเยอร์

12 ซึ่งในเลเยอรที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืนประสิทธิภาพ สำหรับอัตราการบีบที่ 90% พบว่าประสิทธิภาพในการทำงานของแบบจำลองลดลงเป็นอย่างมาก ซึ่งแสดงให้เห็นชัดเจนว่าการยุบตัวกรองโดยเทคนิคผลรวมของค่าเฉลี่ย gradient ไม่สามารถรักษาประสิทธิภาพไว้ได้ โดยประสิทธิภาพตกลงจนไม่สามารถใช้งานแบบจำลองได้ในระดับใกล้เคียงกับแบบจำลองต้นฉบับ

ตารางที่ 6.4 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิคผลรวมของค่าเฉลี่ย gradient ร่วมกับเฟรมเวิร์คพื้นฐาน

จำนวนเลเยอร์ที่ถูกยุบ	อัตราส่วนการยุบ								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.8428	0.8971	0.9139	0.9199	0.9251	0.9269	0.9304	0.9329	0.9343
2	0.5318	0.8703	0.8986	0.9061	0.9136	0.9187	0.9242	0.9281	0.9314
3	0.3937	0.8290	0.8794	0.8991	0.9100	0.9158	0.9207	0.9237	0.9286
4	0.4658	0.8222	0.8657	0.8907	0.9024	0.9118	0.9179	0.9225	0.9265
5	0.4574	0.7947	0.8609	0.8904	0.9005	0.9107	0.9128	0.9222	0.9298
6	0.4826	0.7795	0.8569	0.8729	0.8944	0.9111	0.9148	0.9222	0.9263
7	0.4850	0.7818	0.8564	0.8744	0.8985	0.9040	0.9146	0.9238	0.9270
8	0.4872	0.7892	0.8529	0.8825	0.8987	0.9111	0.9168	0.9237	0.9279
9	0.4950	0.7969	0.8571	0.8859	0.9045	0.9106	0.9192	0.9251	0.9269
10	0.5109	0.7990	0.8604	0.8843	0.9033	0.9146	0.9207	0.9212	0.9268
11	0.5089	0.8044	0.8574	0.8894	0.9048	0.9152	0.9206	0.9250	0.9299
12	0.5495	0.8394	0.8891	0.9096	0.9206	0.9251	0.9299	0.9322	0.9369

ผลการยุบตัวกรองโดยเทคนิค LGAP แบบตัวกรองเดี่ยว แสดงในตารางที่ 6.5 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อยจาก 0.9359 เหลือ 0.9328 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้นประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.6605 ในกรณีอัตราส่วนการบีบอัด 10%-70% จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเลเยอร์ที่ 6 ประสิทธิภาพจะลดลงต่ำสุด และค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 7 จนถึงเลเยอร์ 12 ซึ่งในเลเยอรที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืน

ประสิทธิภาพ สำหรับอัตราการบีบที่ 80%-90% พบว่าประสิทธิภาพในการทำงานของแบบจำลองลดลงเป็นอย่างมาก ซึ่งแสดงให้เห็นชัดเจนว่าการยุบตัวกรองโดยเทคนิค LGAP แบบตัวกรองเดี่ยว ไม่สามารถรักษาประสิทธิภาพไว้ได้ โดยประสิทธิภาพตกลงจนไม่สามารถใช้งานแบบจำลองได้

ตารางที่ 6.5 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิค LGAP แบบตัวกรองเดี่ยว ร่วมกับเฟรมเวิร์คพื้นฐาน

จำนวนเลเยอร์ที่ถูกยุบ	อัตราส่วนการยุบ								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.6605	0.8562	0.8951	0.9100	0.9220	0.9257	0.9293	0.9310	0.9328
2	0.1000	0.7576	0.8618	0.8967	0.9138	0.9159	0.9198	0.9250	0.9320
3	0.1000	0.7235	0.8298	0.8898	0.8992	0.9108	0.9121	0.9214	0.9270
4	0.1000	0.6193	0.8358	0.8896	0.8984	0.9081	0.9112	0.9211	0.9280
5	0.1000	0.6063	0.8197	0.8747	0.8944	0.9097	0.9116	0.9222	0.9290
6	0.1000	0.5944	0.8009	0.8589	0.8892	0.9009	0.9092	0.9197	0.9278
7	0.1000	0.5889	0.8065	0.8668	0.8882	0.9074	0.9113	0.9225	0.9279
8	0.1000	0.5993	0.8010	0.8679	0.8939	0.9053	0.9092	0.9236	0.9275
9	0.1000	0.6111	0.8232	0.8749	0.8988	0.9070	0.9125	0.9214	0.9305
10	0.1000	0.6070	0.8233	0.8826	0.8949	0.9107	0.9140	0.9205	0.9252
11	0.1000	0.6124	0.8378	0.8783	0.8992	0.9086	0.9171	0.9217	0.9259
12	0.1000	0.6726	0.8636	0.9054	0.9152	0.9211	0.9282	0.9315	0.9346

ผลการยุบตัวกรองโดยเทคนิค LGAP แบบตัวกรองโดยรวม แสดงในตารางที่ 6.6 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อย จาก 0.9359 เหลือ 0.9356 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้นประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.8640 จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเลเยอร์ที่ 6,7 ประสิทธิภาพจะลดลงต่ำสุดและค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 7, 8 จนถึงเลเยอร์ 12 ซึ่งในเลเยอร์ที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืนประสิทธิภาพ

ตารางที่ 6.6 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิค LGAP แบบตัวกรองโดยรวม ร่วมกับเฟรมเวิร์คพื้นฐาน

จำนวนเลเยอร์ที่ถูกยุบ	อัตราส่วนการยุบ								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.8640	0.9111	0.9226	0.9279	0.9284	0.9330	0.9327	0.9336	0.9356
2	0.8298	0.8897	0.9018	0.9168	0.9219	0.9269	0.9289	0.9324	0.9339
3	0.8076	0.8685	0.8891	0.9030	0.9121	0.9173	0.9230	0.9252	0.9329
4	0.7831	0.8554	0.8811	0.8947	0.9065	0.9149	0.9223	0.9244	0.9305
5	0.7426	0.8407	0.8732	0.8936	0.9051	0.9141	0.9188	0.9248	0.9298
6	0.7325	0.8043	0.8555	0.8850	0.8993	0.9125	0.9157	0.9223	0.9306
7	0.7212	0.8045	0.8550	0.8905	0.8977	0.9135	0.9184	0.9214	0.9312
8	0.7286	0.8200	0.8619	0.8907	0.9025	0.9147	0.9198	0.9234	0.9288
9	0.7237	0.8199	0.8629	0.8981	0.9054	0.9159	0.9229	0.9236	0.9307
10	0.7336	0.8342	0.8716	0.8935	0.9060	0.9155	0.9204	0.9263	0.9295
11	0.7378	0.8314	0.8688	0.8986	0.9099	0.9184	0.9243	0.9246	0.9320
12	0.7652	0.8612	0.8966	0.9147	0.9193	0.9272	0.9288	0.9336	0.9368

ผลการยุบตัวกรองโดยเทคนิค LGAP แบบตัวกรองโดยรวม ร่วมกับเฟรมเวิร์ค CASM แสดงในตารางที่ 6.7 เมื่อใช้อัตราส่วนการยุบ 10% ในการยุบ 1 เลเยอร์ จะส่งผลให้ประสิทธิภาพของแบบจำลองลดลงเล็กน้อย จาก 0.9359 เหลือ 0.9329 เมื่อเพิ่มอัตราส่วนการบีบอัดมากขึ้น ประสิทธิภาพความแม่นยำในการทำนายผลจะลดลง เมื่อใช้อัตราการบีบอัดถึง 90% ประสิทธิภาพจะเหลือ 0.8376 จำนวนเลเยอร์ที่ถูกยุบเพิ่มขึ้นจะลดประสิทธิภาพในการทำนายผลลงจนถึงเลเยอร์ที่ 6, 7, 8 ประสิทธิภาพจะลดลงต่ำสุด และค่อย ๆ กลับเพิ่มขึ้น เมื่อยุบเลเยอร์ที่ 7, 8, 9 จนถึงเลเยอร์ 12 ซึ่งในเลเยอร์ที่ 12 ประสิทธิภาพกลับมาสูงจากการสอนข้อมูลซ้ำหลายรอบในการเรียกคืนประสิทธิภาพ เฟรมเวิร์ค CASM สามารถเรียกประสิทธิภาพกลับมาได้สูงกว่าแบบเฟรมเวิร์คพื้นฐาน เมื่ออิงประสิทธิภาพในช่วงการยุบเลเยอร์ที่ 11

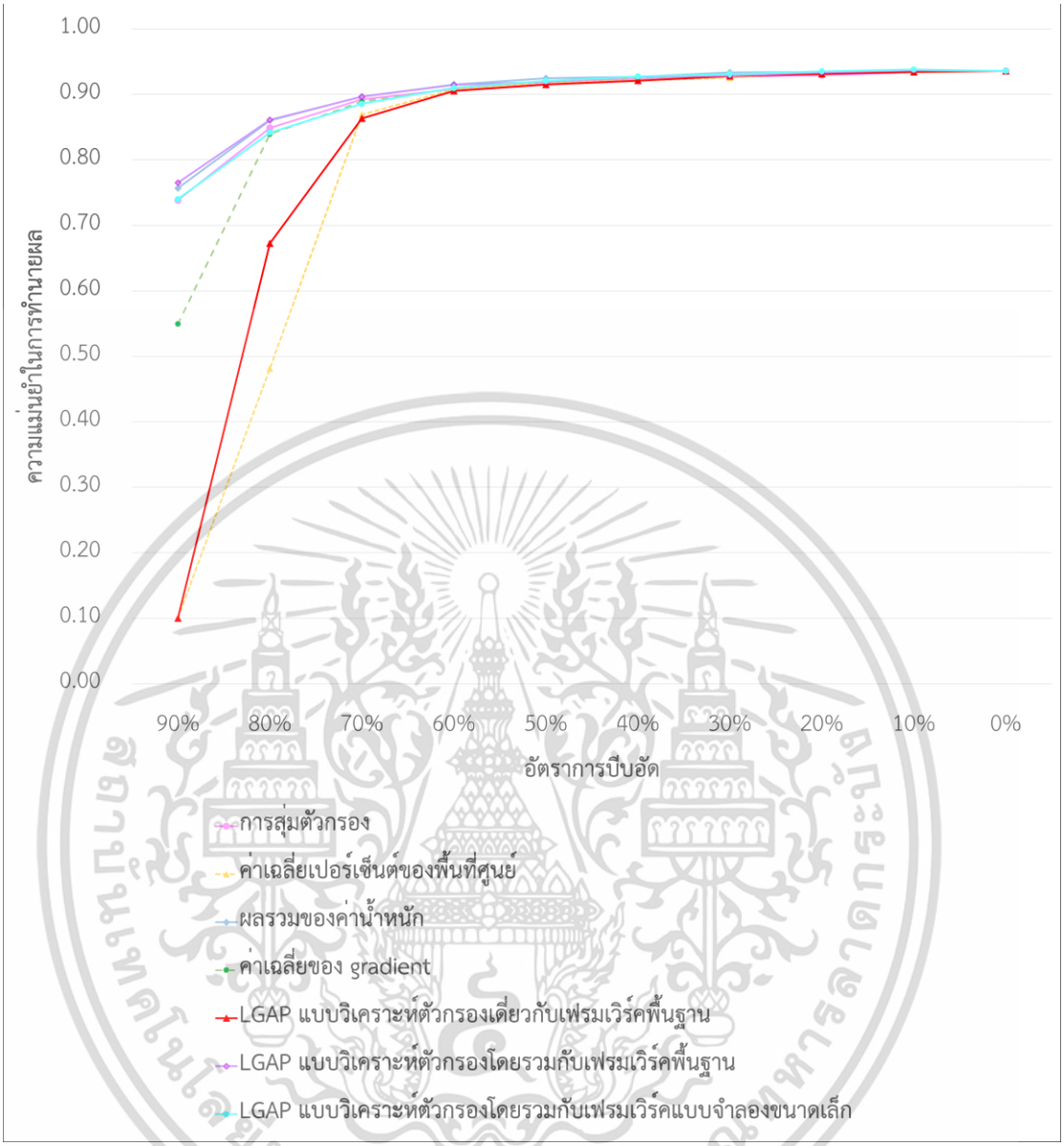
ตารางที่ 6.7 ค่าความแม่นยำในการยุบตัวกรองในแต่เลเยอร์ตามอัตราส่วนการยุบด้วยเทคนิค LGAP แบบตัวกรองโดยรวม ร่วมกับเฟรมเวิร์ค CASM

จำนวนเลเยอร์ที่ถูกยุบ	อัตราส่วนการยุบ								
	90%	80%	70%	60%	50%	40%	30%	20%	10%
1	0.8376	0.9039	0.9182	0.9227	0.9313	0.9295	0.9326	0.9313	0.9329
2	0.8088	0.8829	0.9044	0.9164	0.9131	0.9250	0.9241	0.9297	0.9272
3	0.7693	0.8530	0.8915	0.9065	0.8986	0.9179	0.9196	0.9247	0.9235
4	0.7367	0.8369	0.8748	0.8932	0.8985	0.9101	0.9138	0.9189	0.9225
5	0.6654	0.8056	0.8547	0.8877	0.8926	0.9025	0.9117	0.9148	0.9132
6	0.6662	0.7949	0.8436	0.8750	0.8773	0.9031	0.9100	0.9112	0.9112
7	0.6531	0.7949	0.8439	0.8747	0.8775	0.9038	0.9073	0.9152	0.9111
8	0.6750	0.7890	0.8489	0.8802	0.8864	0.9057	0.9099	0.9083	0.9135
9	0.6833	0.8027	0.8501	0.8810	0.8936	0.9065	0.9134	0.9176	0.9182
10	0.6776	0.8003	0.8521	0.8815	0.8916	0.9091	0.9077	0.9137	0.9143
11	0.6943	0.8050	0.8573	0.8807	0.8994	0.9041	0.9126	0.9158	0.9184
12	0.7403	0.8418	0.8859	0.9103	0.9207	0.9275	0.9312	0.9353	0.9381

จากตารางที่ 6.1–6.7 ประสิทธิภาพของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ที่ถูกยุบในแต่ละเทคนิคการวิเคราะห์ตัวกรองที่ต่างกันและให้ผลลัพธ์ออกมาแตกต่างกัน ซึ่งเป็นที่แน่ชัดว่าเทคนิคการวิเคราะห์ตัวกรองส่งผลต่อความแม่นยำของแบบจำลองที่ได้หลังผ่านกระบวนการยุบตัวกรองและเรียกคืนประสิทธิภาพ สำหรับเทคนนิคการวิเคราะห์ด้วยผลรวมของค่าเฉลี่ยพื้นที่ศูนย์, ค่าเฉลี่ย gradient และ LGAP แบบตัวกรองเดี่ยว ไม่สามารถนำมาใช้งานได้ในกรณีใช้อัตราการบีบ 80%-90% การยุบตัวกรองของทุกวิธีประสิทธิภาพจะลดลงต่ำสุด เมื่อยุบตัวกรองในช่วง 6-8 เลเยอร์ และหลังจากยุบเลเยอร์ถัดไปประสิทธิภาพกลับเพิ่มขึ้น สิ่งที่เกิดขึ้นคาดว่าเป็นผลจากตัวกรองที่หายไปในเลเยอร์ต้องทำงานร่วมกับตัวกรองที่ยังไม่ได้ถูกยุบในเลเยอร์ถัดมา ฉะนั้นประสิทธิภาพในการทำงานของตัวกรองในเลเยอร์ถัดมาจึงลดลง ซึ่งตัวกรองเหล่านี้ถูกพิจารณาว่ามีประสิทธิภาพน้อยในการให้คะแนนของแต่ละเทคนิคการวิเคราะห์ การคงอยู่ของตัวกรองเหล่านี้เลยส่งผลทำให้การทำงานโดยรวมลดลง เมื่อถูกนำออกไปประสิทธิภาพของแบบจำลองที่ถูกยุบจึงค่อย ๆ กลับขึ้นมา และจะสูงขึ้นอย่างชัดเจน เมื่อทำการยุบตัวกรองที่เลเยอร์สุดท้ายและเรียกคืนประสิทธิภาพ การยุบตัวกรองโดยใช้อัตราการบีบอัดที่ 10% ด้วยเทคนิคผลรวมของค่าน้ำหนัก, ค่าเฉลี่ย gradient, LGAP แบบตัว

กรองโดยรวมในเฟรมเวิร์คพื้นฐาน และ LGAP แบบตัวกรองโดยรวมในเฟรมเวิร์ค CASM สามารถดึงประสิทธิภาพกลับมาได้ดีกว่าแบบจำลองต้นฉบับ เมื่อทำการยุบตัวกรองครบทั้ง 12 เลเยอร์

เมื่อทำการยุบตัวกรองทั้ง 12 เลเยอร์ ในแต่ละเทคนิคและทำการวิเคราะห์ผลประสิทธิภาพความแม่นยำของแต่ละเทคนิคร่วมกัน แสดงในรูปที่ 6.1 โดยแกนตั้งแสดงประสิทธิภาพในการทำนายผลของแบบจำลองที่ถูกยุบตัวกรอง แกนนอนแสดงอัตราการบีบอัดแบบจำลอง ในช่วงของการอัตราการบีบอัดเกิน 70% ค่าเฉลี่ยพื้นที่ศูนย์ (APoZ) หรือ LGAP แบบวิเคราะห์ตัวกรองเดี่ยวประสิทธิภาพลดลงเป็นอย่างมาก ค่าเฉลี่ยพื้นที่ศูนย์มีความเป็นไปได้สูงมากที่จะไม่สามารถแยกความสามารถของตัวกรองได้ในกรณีที่พื้นที่ศูนย์ใกล้เคียงกัน ฉะนั้นหากมีตัวกรองที่สามารถตอบรับกับข้อมูลได้ดี แต่พื้นที่ศูนย์สูงกว่า ตัวกรองที่ตอบรับไม่ดีอาจทำให้การวิเคราะห์ตัวกรองเกิดข้อผิดพลาดขึ้นได้ ในส่วนของวิธี LGAP แบบวิเคราะห์ตัวกรองเดี่ยว การวิเคราะห์แบบตัวกรองเดี่ยวกับภาพรวมของเลเยอร์เกิดความแตกต่างกันสูงมากในแง่ของคะแนนการวิเคราะห์ที่ได้มากสำหรับแต่ละตัวกรอง ซึ่งไม่สามารถตอบได้ว่าในกรณีที่ตัวกรองให้คะแนนต่ำกว่าหรือสูงกว่าตัวกรองนั้นจะมีประสิทธิภาพที่ดี เทคนิคสุ่มให้ผลออกมาที่ดีมากในแบบจำลองของ VGG-16 กับชุดข้อมูล CIFAR-10 ช่วยบอกได้ชัดเจนว่า แม้จะเลือกแบบสุ่มแบบจำลองยังคงทนต่อการบีบอัดที่สูงได้ อย่างไรก็ดีการบีบอัดที่สูงมาก ประสิทธิภาพที่ได้น้อยกว่าการใช้เทคนิคแบบผลรวมค่าน้ำหนักและ LGAP แบบตัวกรองโดยรวม และมีความเป็นไปได้ที่อาจเกิดที่แย่กว่านี้ในการเลือก random seed ที่แตกต่างออกไป สำหรับผลรวมค่าน้ำหนัก (Weighted sum) วิเคราะห์จากปริมาณค่าน้ำหนักหากสูงถูกว่าดี ซึ่งก็ขาดความสัมพันธ์กับการวิเคราะห์ในเชิงข้อมูลไป ในการทดลองกับแบบจำลอง VGG-16 ถือว่าทำผลออกมาได้ดีมาก หากข้อมูลที่ใช้มีความซับซ้อนสูงขึ้น ค่าน้ำหนักอาจให้ประสิทธิภาพลดลงในการใช้งาน สำหรับ LGAP วิเคราะห์ตัวกรองรวม เป็นการวิเคราะห์ตัวกรองที่หายไปจากการทำงานในเลเยอร์ เมื่อตัวกรองที่ถูกวิเคราะห์หายไป จะถูกนำมาเทียบว่าการทำงานของเลเยอร์ต่างไปจากเดิมแค่ไหน ถ้าผลการกรองของภาพรวมทั้งเลเยอร์หายไปมากแสดงว่าตัวกรองนั้นส่งผลต่อการทำงานโดยภาพรวมสูงมากจำเป็นต้องเก็บไว้ ในการบีบอัดที่สูงการใช้ความสัมพันธ์ตัวกรองเชิงพื้นที่ถูกวิเคราะห์ร่วมกับการทำนายผลคำตอบ ทำให้การวิเคราะห์ทำได้ครอบคลุม ซึ่งถือเป็นจุดแข็งของเทคนิคที่นำเสนอ หลังจากการเปลี่ยนเฟรมเวิร์คพื้นฐานเป็นเฟรมเวิร์ค CASM ประสิทธิภาพในช่วงอัตราการบีบอัด 10%-50% ประสิทธิภาพสูงกว่าเล็กน้อย คาดว่าเป็นผลจากการ CASM สามารถที่จะดึงการทำงานของเลเยอร์หลังจากเลเยอร์ที่ถูกตัวกรองกลับมาได้เหมือนกับแบบจำลองต้นฉบับทำให้การทำงานออกมาได้ผลดี แต่เมื่อใช้การบีบอัดที่เกิน 50% ความสามารถของ CASM ไม่เพียงพอที่ทำการเรียกคืนงานทำงานให้เหมือนกับต้นฉบับ ทำให้ประสิทธิภาพลดลง



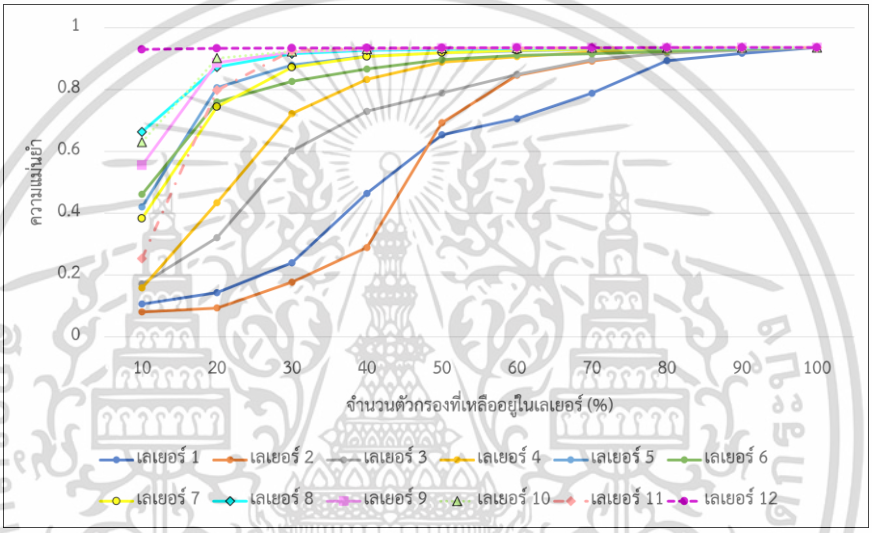
รูปที่ 6.1 ความแม่นยำของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ในการยุบตัวกรองทั้ง 12 เลเยอร์ในแต่ละเทคนิค

6.1.2 ผลประสิทธิภาพของแบบจำลอง VGG-16 ถูกยุบตัวกรองแบบแต่ละเลเยอร์แยกจากกัน

แบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 มีความแม่นยำในการทำนายผลในการแยกประเภทข้อมูลเป็น 0.9359 โดยในส่วนนี้จะแสดงผลการยุบตัวกรองในแต่ละเลเยอร์ที่ไม่ต่อเนื่องกันเพื่อสังเกตประสิทธิภาพของแบบจำลองหลังยุบตัวกรองแต่ละเลเยอร์ว่าลดลงไปเพียงใดในแต่ละเทคนิค แสดงในรูปที่ 6.2 - 6.6 โดยแกนตั้ง คือ ความสามารถในการทำนายผลของตัวกรองของแบบจำลอง VGG-16 ที่ผ่านการยุบตัวกรอง แกนนอน คือ จำนวนตัวกรองที่เหลืออยู่ในเลเยอร์ คิด

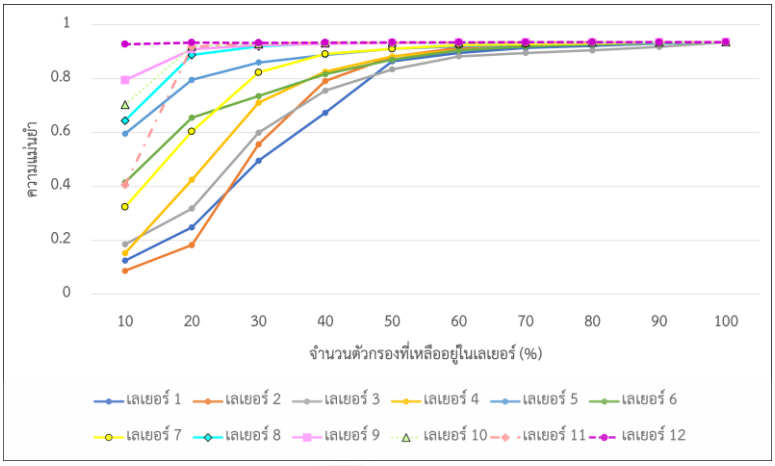
เป็น % (จำนวนตัวกรองที่เหลืออยู่ในเลเยอร์ (%) = 100 – อัตราการบีบอัด) เริ่มทำการยุบจาก เลเยอร์ 1 คือ conv2d_1 ไปจนถึง เลเยอร์ 12 คือ conv2d_12

การยุบตัวกรองในแต่ละเยอร์แบบวิธีสุ่ม แสดงในรูปที่ 6.2 เลเยอร์ที่มีความอ่อนไหว เมื่อจำนวนตัวกรองลดลง คือ เลเยอร์แรก ๆ ได้แก่ เลเยอร์ 1, เลเยอร์ 2, เลเยอร์ 3 สังเกตได้จากจำนวนตัวกรองที่เหลือในช่วง 10%-80% ค่าประสิทธิภาพลดลงเร็วกว่าเลเยอร์อื่น ๆ เลเยอร์ที่อยู่ลึกขึ้นส่วนใหญ่ทนต่อการบีบอัดได้สูงขึ้นตามลำดับ แต่เลเยอร์ที่ 11 เมื่อจำนวนตัวกรองเหลือ 10% ประสิทธิภาพจะลดลงอย่างมากจาก 20% ความแม่นยำจากประมาณ 0.8 ลงมาอยู่ที่ประมาณช่วง 0.2 – 0.3 เลเยอร์ที่ทนต่อการบีบมากที่สุด คือ เลเยอร์ 12



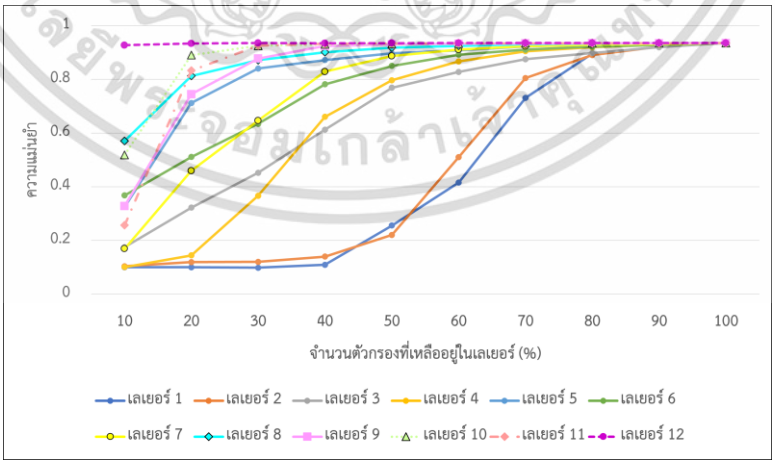
รูปที่ 6.2 ความแม่นยำของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ในการยุบตัวกรองแต่ละเลเยอร์แยกจากกันโดยเทคนิคแบบสุ่ม

การยุบตัวกรองในแต่ละเยอร์แบบผลรวมของค่าน้ำหนัก (Weighted sum) แสดงในรูปที่ 6.3 เลเยอร์ที่มีความอ่อนไหว เมื่อจำนวนตัวกรองลดลง คือ เลเยอร์ 1, เลเยอร์ 2, เลเยอร์ 3 สังเกตได้จากจำนวนตัวกรองที่เหลือในช่วง 10%-40% ค่าประสิทธิภาพลดลงเร็วกว่าเลเยอร์อื่น ๆ แต่เทคนิคผลรวมของค่าน้ำหนักลดลงช้ากว่าแบบสุ่ม เลเยอร์ที่อยู่ลึกขึ้นส่วนใหญ่ทนต่อการบีบอัดได้สูงขึ้นตามลำดับ เลเยอร์ที่ 11 เมื่อจำนวนตัวกรองเหลือ 10% ประสิทธิภาพจะลดลงอย่างมากจาก 20% ความแม่นยำจากประมาณช่วง 0.8 – 0.9 ลงมาอยู่ที่ประมาณ 0.4 เลเยอร์ที่ทนต่อการบีบมากที่สุด คือ เลเยอร์ 12 ซึ่งเห็นได้ชัดว่าเทคนิคผลรวมของค่าน้ำหนักในแต่ละเลเยอร์ที่ถูกยุบทนต่อการบีบอัดได้ดีกว่าแบบสุ่ม



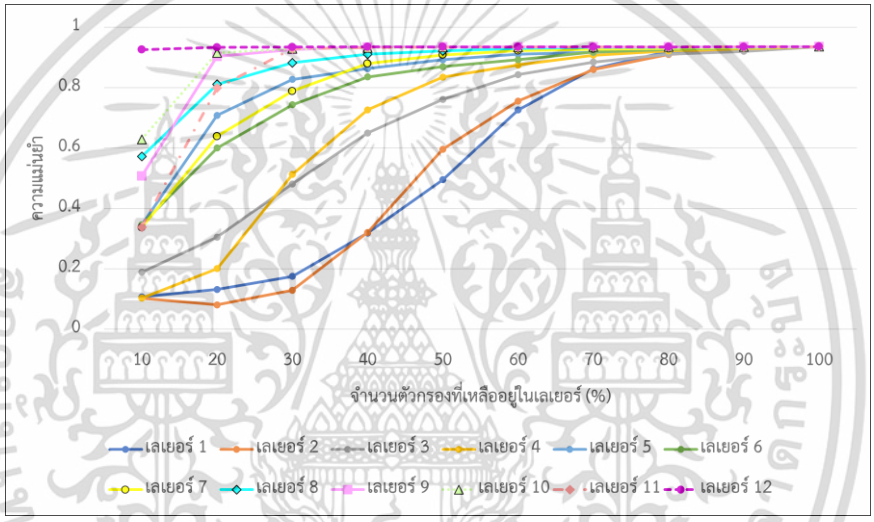
รูปที่ 6.3 ความแม่นยำของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ในการยุบตัวกรองแต่ละเลเยอร์แยกจากกันโดยเทคนิคผลรวมของค่าน้ำหนัก

การยุบตัวกรองในแต่ละเยอร์แบบค่าเฉลี่ยพื้นที่ศูนย์ (APoZ) แสดงในรูปที่ 6.4 เลเยอร์ที่มีความอ่อนไหว เมื่อจำนวนตัวกรองลดลง คือ เลเยอร์ 1, เลเยอร์ 2 สังเกตได้จากจำนวนตัวกรองที่เหลือในช่วง 10%-70% ค่าประสิทธิภาพลดลงเร็วกว่าเลเยอร์อื่น ๆ ซึ่งแย่กว่าวิธีการแบบสุ่ม เลเยอร์ที่อยู่ลึกขึ้นส่วนใหญ่ทนต่อการบีบอัดได้สูงขึ้นตามลำดับ หลายเลเยอร์ เมื่อจำนวนตัวกรองเหลือ 10% ประสิทธิภาพจะลดลงอย่างมากจาก 20% คือ เลเยอร์ 5 ความแม่นยำจากประมาณช่วง 0.7 – 0.8 ลงมาอยู่ที่ประมาณ 0.3, เลเยอร์ 9 ความแม่นยำจากประมาณช่วง 0.7 – 0.8 ลงมาอยู่ที่ประมาณ 0.3 และเลเยอร์ 11 ความแม่นยำจากประมาณช่วง 0.8 – 0.9 ลงมาอยู่ที่ประมาณ 0.2 เลเยอร์ที่ทนต่อการบีบมากที่สุด คือ เลเยอร์ 12 ซึ่งวิธีการนี้เลเยอร์ที่ถูกยุบทนทานน้อยว่าแบบสุ่มและผลรวมค่าน้ำหนัก (Weighted sum)



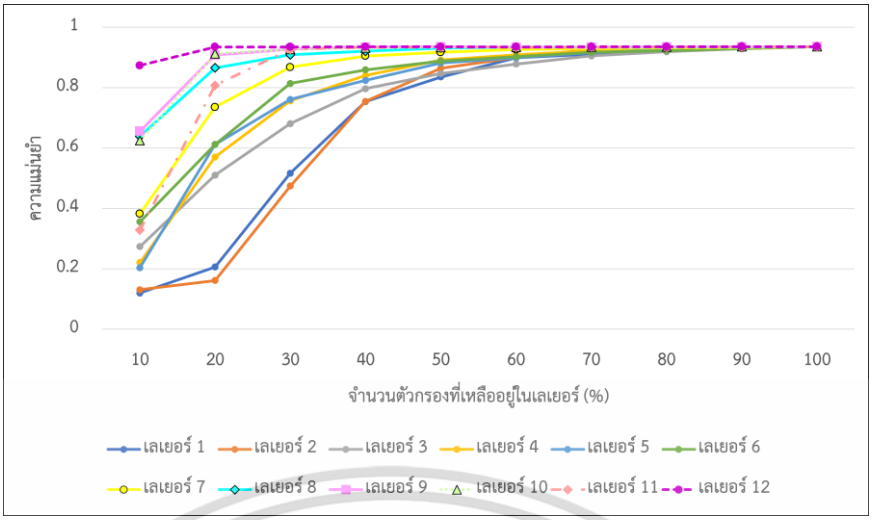
รูปที่ 6.4 ความแม่นยำของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ในการยุบตัวกรองแต่ละเลเยอร์แยกจากกันโดยเทคนิคค่าเฉลี่ยพื้นที่ศูนย์

การยุบตัวกรองในแต่ละเยอร์แบบค่าเฉลี่ย gradient แสดงในรูปที่ 6.5 เลเยอร์ที่มีความอ่อนไหว เมื่อจำนวนตัวกรองลดลง คือ เลเยอร์ 1, เลเยอร์ 2 สังเกตได้จากจำนวนตัวกรองที่เหลือในช่วง 10%-60% ค่าประสิทธิภาพลดลงเร็วกว่าเลเยอร์อื่น ๆ เลเยอร์ที่อยู่ลึกขึ้นส่วนใหญ่ทนต่อการบีบอัดได้สูงขึ้นตามลำดับ หลายเลเยอร์ เมื่อจำนวนตัวกรองเหลือ 10% ประสิทธิภาพจะลดลงอย่างมากจาก 20% คือ เลเยอร์ 5 ความแม่นยำจากประมาณช่วง 0.7 – 0.8 ลงมาอยู่ที่ประมาณ 0.3 - 0.4, เลเยอร์ 9 ความแม่นยำจากประมาณช่วง 0.8 – 0.9 ลงมาอยู่ที่ประมาณ 0.4 - 0.5 และเลเยอร์ 11 ความแม่นยำจากประมาณ 0.8 ลงมาอยู่ที่ประมาณ 0.3 – 0.4 เลเยอร์ที่ทนต่อการบีบมากที่สุด คือ เลเยอร์ 12 ซึ่งเทคนิคค่าเฉลี่ย gradient นี้เลเยอร์ที่ถูกยุบทนทานน้อยว่าแบบสุ่มและผลรวมค่าน้ำหนัก (Weighted sum) แต่ดีกว่าแบบค่าเฉลี่ยนพื้นที่ศูนย์ (APoZ)



รูปที่ 6.5 ความแม่นยำของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ในการยุบตัวกรองแต่ละเลเยอร์แยกจากกันโดยเทคนิคค่าเฉลี่ย gradient

การยุบตัวกรองในแต่ละเยอร์เทคนิค LGAP แบบตัวกรองโดยรวม ดีกว่าแบบตัวกรองเดี่ยว ซึ่งการดูผลกระทบของการยุบตัวกรองจึงเน้นไปที่แบบตัวกรองรวม แสดงในรูปที่ 6.6 เลเยอร์ที่มีความอ่อนไหว เมื่อจำนวนตัวกรองลดลง คือ เลเยอร์ 1, เลเยอร์ 2 สังเกตได้จากจำนวนตัวกรองที่เหลือในช่วง 10%-40% ค่าประสิทธิภาพลดลงเร็วกว่าเลเยอร์อื่น ๆ เลเยอร์ที่อยู่ลึกขึ้นส่วนใหญ่ทนต่อการบีบอัดได้สูงขึ้นตามลำดับ หลายเลเยอร์ เมื่อจำนวนตัวกรองเหลือ 10% ประสิทธิภาพจะลดลงอย่างมากจาก 20% คือ เลเยอร์ 5 ความแม่นยำจากประมาณช่วง 0.6 ลงมาอยู่ที่ประมาณ 0.2 และเลเยอร์ 11 ความแม่นยำจากประมาณ 0.8 ลงมาอยู่ที่ประมาณ 0.3 – 0.4 เลเยอร์ที่ทนต่อการบีบมากที่สุด คือ เลเยอร์ 12 เทคนิค LGAP แบบตัวกรองโดยรวม ช่วงจำนวนตัวกรองที่เหลือ 10%-20% จะมีเพียง 2 เลเยอร์เท่านั้นที่ตกลงมาต่ำที่สุด เมื่อเทียบกับเทคนิคอื่น ๆ ที่อาจมีเลเยอร์ 3 และเลเยอร์ 4 ที่ความแม่นยำลดต่ำลงมาก ซึ่งจุดนี้จึงบอกได้ว่า LGAP แบบตัวกรองโดยรวมทนทานกว่าเทคนิคอื่น ๆ เมื่อเจออัตราการบีบอัดที่สูง



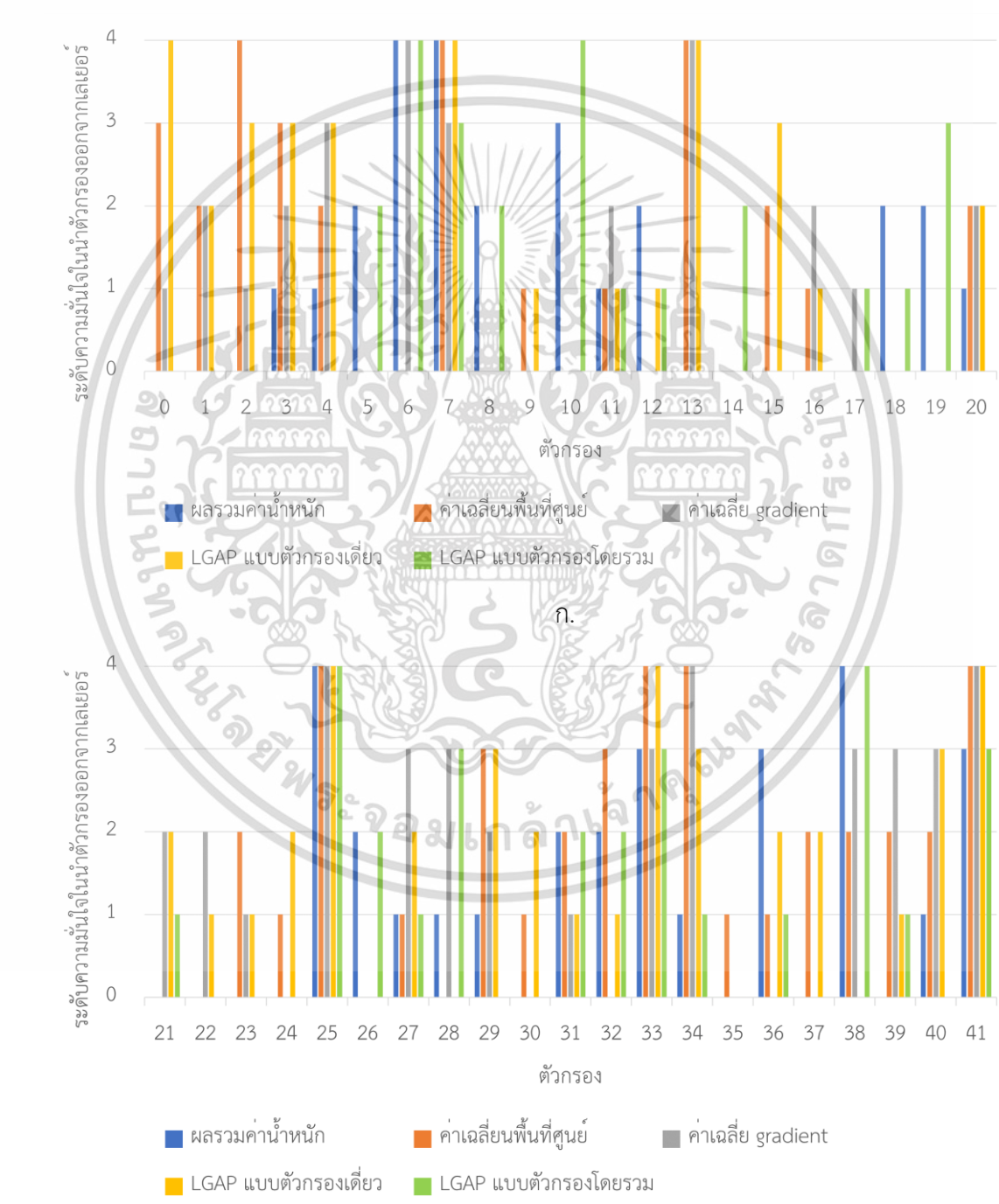
รูปที่ 6.6 ความแม่นยำของแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 ในการยุบตัวกรองแต่ละเลเยอร์แยกจากกันโดยเทคนิค LGAP แบบตัวกรองโดยรวม

จากผลของการยุบตัวกรองในแต่ละเยอร์ช่วยให้ทราบว่าเลเยอร์ช่วงต้น เลเยอร์ 1 - เลเยอร์ 4 มีความเปราะบางด้านประสิทธิภาพเป็นอย่างมากในการยุบตัวกรองออกจากเลเยอร์ และเลเยอร์ที่ทนต่อการยุบตัวกรองเป็นอย่างมาก คือ เลเยอร์ 12 เมื่อเทียบกันที่ 20% ดังนั้นการในยุบตัวกรองของเลเยอร์ในช่วงแรกจะมีความสำคัญต่อการรักษาประสิทธิภาพและการเรียกคืนประสิทธิภาพเป็นอย่างมาก

6.1.3 ผลความมั่นใจในการยุบตัวกรองของแต่ละเทคนิค

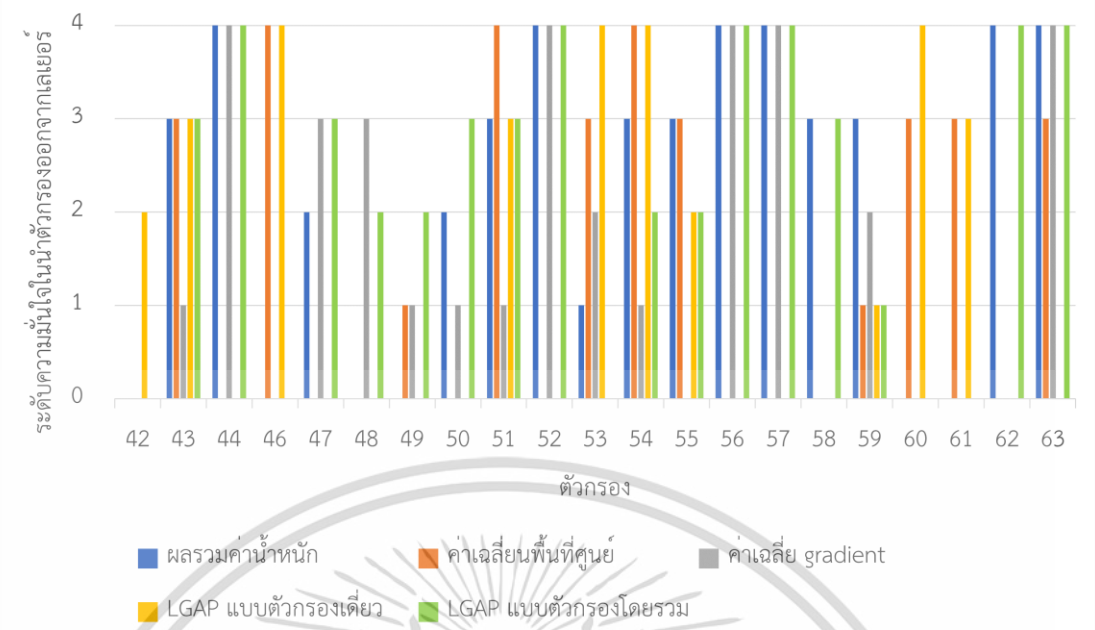
ในการยุบตัวกรองตัวกรองออกจากเลเยอร์ แต่ละเทคนิคจะใช้วิธีการวิเคราะห์ไม่เหมือนกัน ทำให้ตัวกรองที่ถูกเลือกยุบออกจากเลเยอร์ต่างกันออกไปเช่นกัน เพื่อวิเคราะห์การพิจารณาตัวกรองของแต่ละเทคนิคในรูปแบบของความมั่นใจในการนำตัวกรองออกจากเลเยอร์ แสดงในรูปที่ 6.7 การวิเคราะห์ด้านความมั่นใจในการนำตัวกรองออกจากเลเยอร์ เป็นตัวอย่างการวิเคราะห์ในเลเยอร์ 1 ซึ่งประกอบไปด้วย 64 ตัวกรอง โดยเริ่มจาก 0 – 63 ตัว ในตัวอย่างนี้เป็นการวิเคราะห์ในกรณีกำหนดให้จำนวนตัวกรองเหลือ 40% ในเลเยอร์ทั้งหมด แกนตั้งเป็นระดับความมั่นใจในการนำตัวกรองออกจากเลเยอร์ ค่า 0 หมายถึง ตัวกรองถูกเก็บในเลเยอร์ในการวิเคราะห์ของเทคนิคนั้น ๆ และ 4 หมายถึง ตัวกรองนั้นถูกยุบออกจากเลเยอร์แน่นอนตามเทคนิคที่ใช้วิเคราะห์ ระดับคะแนน 1-4 หมายถึง ตัวกรองถูกนำออกจากเลเยอร์ในเทคนิคนั้น ๆ โดยการแบ่งระดับพิจารณาจากจำนวนตัวกรองที่นำออกและหารด้วย 4 เพื่อแบ่งระดับความมั่นใจในการวิเคราะห์ว่าจะนำตัวกรองนั้นออก แกนนอน คือ หมายเลขตัวกรองแต่ละตัวที่แต่ละวิธีทำการวิเคราะห์ว่าต้องนำออกจากเลเยอร์ ในกรณีที่ตัวกรองบางตัวไม่อยู่ในรูปที่ 6.7 หมายความว่าตัวกรองเหล่านั้นถูกจัดเก็บในเลเยอร์ของทุกเทคนิค โดยเทคนิค LGAP แบบตัวกรองเดี่ยว จะเลือกตัวกรองที่ต่างจากเทคนิค LGAP แบบตัวกรองรวมค่อนข้างมาก และมีทิศทางไปในทางเดียวกับค่าเฉลี่ยพื้นที่ศูนย์ (APoZ) กับค่าเฉลี่ย gradient สำหรับ

เทคนิค LGAP แบบตัวกรองรวมมีการเลือกในทางทิศทางเดียวกับผลรวมค่าน้ำหนัก (Weighted Sum) สำหรับตัวกรองที่มีค่าระดับความมั่นใจสูงสุดในแต่ละเทคนิค จะถูกยุบออกเอาจากเลเยอร์ 1 มากที่สุด คือ ตัวกรองที่ 25 และต่อมา คือ 7, 33 และ 41 โดยเมื่อนำตัวกรองที่มีระดับความมั่นใจสูงนี้ ไปดูค่าคะแนน (Score) จะเห็นว่าตัวกรองเหล่านี้ได้คะแนนในแต่ละวิธีต่ำสุด แสดงให้เห็นว่าตัวกรองเหล่านี้มีการตอบรับกับข้อมูลที่ส่งเข้ามาในเลเยอร์น้อยมาก ซึ่งไม่มีความจำเป็นต้องเก็บไว้ในเลเยอร์ 1



ก.

ข.



ระดับ 0 หมายถึง ตัวกรองไม่ถูกนำออกจากเลเยอร์ ในการวิเคราะห์ของเทคนิคนั้น ๆ
ระดับ 1 – 4 หมายถึง ตัวกรองถูกวิเคราะห์ว่าต้องนำออกจากเลเยอร์ตามอัตราการบีบอัดของแบบจำลอง
ระดับ 4 หมายถึง ตัวกรองถูกวิเคราะห์ว่าต้องนำออกจากเลเยอร์มากที่สุดตามการวิเคราะห์ของเทคนิคนั้น ๆ
ตัวกรอง หมายถึง ตัวกรองที่ถูกวิเคราะห์ว่าต้องนำออกจากเลเยอร์ในแต่ละเทคนิค

ค.

รูปที่ 6.7 ระดับความมั่นใจในการยุบตัวกรองออกจากเลเยอร์ 1 ของแบบจำลอง VGG-16

6.1.4 ผลประสิทธิภาพของแบบจำลอง VGG-16 ถูกยุบตัวกรองของ LGAP ในแต่ละเฟรมเวิร์ค

การยุบตัวกรองโดยเทคนิค LGAP แบบตัวกรองรวมได้ถูกแสดงความสามารถในการวิเคราะห์ตัวกรองออกมาแล้วว่ามีประสิทธิภาพที่ดีและทนต่ออัตราการบีบที่สูงได้ ในส่วนนี้จะเป็นการพิจารณาถึงการเรียกคืนประสิทธิภาพของแบบจำลอง VGG-16 หลังจากการยุบตัวกรองด้วยเฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM ในตารางที่ 6.8 เป็นการเปรียบเทียบประสิทธิภาพทางด้านความผิดพลาดในการทำนายผลของแบบจำลอง หลังผ่านการยุบเลเยอร์ทั้ง 12 เลเยอร์ และการเรียกคืนประสิทธิภาพในแต่ละเลเยอร์ ซึ่งอัตราการบีบอัด 0% หมายถึง แบบจำลองไม่มีการยุบตัวกรองออกจากเลเยอร์ และอัตราการบีบอัด 90% หมายถึง แบบจำลองถูกนำตัวกรองออก 90% (จำนวนตัวกรองเหลือ 10%) ทุกเลเยอร์จะใช้อัตราการบีบอัดที่เท่ากัน เช่น ใช้อัตราส่วนการบีบอัด 90% จะทำการนำตัวกรองออกจากทุกเลเยอร์ 90% ของตัวกรองที่มีอยู่ในแต่ละเลเยอร์ ค่า FLOPs เป็นการวัด complexity ที่วัดจากเลเยอร์ที่เป็นคอนโวลูชั่นเลเยอร์ทั้ง 12 เลเยอร์ พารามิเตอร์ คือ จำนวนพารามิเตอร์ทั้งหมดที่มีอยู่ในแบบจำลอง VGG-16 ซึ่งจะแบ่งออกเป็น 2 ประเภท คือ พารามิเตอร์ที่สามารถสอนข้อมูลได้ (ส่วนใหญ่อยู่ในเลเยอร์คอนโวลูชั่น) และพารามิเตอร์ที่ไม่สามารถสอนข้อมูลได้ พื้นที่เก็บข้อมูล คือ ขนาดของแบบจำลองที่ใช้ในการเก็บข้อมูลวัดในหน่วย Megabyte เฟรมเวิร์ค

CASM จะใช้ข้อมูลจากชุดข้อมูลสำหรับสอนของ CIFAR-10 จำนวน 5000 รูป ใช้สำหรับการสอนข้อมูล และ 1000 รูป ใช้สำหรับการทดสอบการทำงาน ในกระบวนการ CASM เพื่อให้การทำงานกลับมาคล้ายกับแบบจำลองต้นฉบับ ในช่วงอัตราการบีบอัด 10%-50% LGAP กับเฟรมเวิร์ค CASM มีประสิทธิภาพในการทำนายผลผิดพลาดน้อยกว่าแบบ LGAP กับเฟรมเวิร์คพื้นฐาน ในขณะที่ค่า FLOPs, พารามิเตอร์ และพื้นที่เก็บข้อมูลมีขนาดเท่ากัน เมื่ออัตราการบีบสูงขึ้นเกิน 50% ความสามารถในการทำนายผลของ LGAP กับเฟรมเวิร์คพื้นฐานจะมีค่าผิดพลาดน้อยกว่า จะเห็นว่า CASM สามารถทำงานได้ดีกว่าเฟรมเวิร์คพื้นฐาน เมื่ออัตราการบีบอัดไม่เกิน 50% สาเหตุที่เป็นเช่นเพราะในการทำ CASM ต้องทำการสอนข้อมูลโดยใช้ข้อมูลขาออกจากแบบจำลองต้นฉบับการใช้อัตราบีบที่สูงเกิน 50% CASM จึงมีจำนวนตัวกรองไม่เพียงพอที่จะให้การทำงานของตัวเองกลับมาทำงานให้เหมือนแบบจำลองต้นฉบับ ฉะนั้นข้อมูลแผนผังคุณลักษณะที่ถูกส่งต่อภายในโครงข่ายประสาทเทียมแบบคอนโวลูชั่นอาจจะมีความแตกต่างไปจากแบบจำลองต้นฉบับมากขึ้น ในกรณีที่ต้องการใช้งานแบบจำลองที่ถูกบีบอัดให้มีขนาดลดลงไปประมาณ 2 เท่า (2X) ทั้ง FLOPs, พารามิเตอร์ และพื้นที่เก็บข้อมูล สามารถใช้อัตราการบีบอัดที่ 30% ได้ ซึ่งอัตราการบีบอัด 90% สามารถลด FLOPs ประมาณ 44 เท่า, พารามิเตอร์ประมาณ 84 เท่า และพื้นที่เก็บข้อมูลประมาณ 70 เท่า

ตารางที่ 6.8 ค่าความผิดพลาดในการทำนายของ LGAP ที่ใช้เฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM

เทคนิค, เฟรมเวิร์ค	**อัตราส่วนการบีบอัด**	**ค่าการทำนายผิดพลาด (%)**	**FLOPs**	**พารามิเตอร์**	**พื้นที่เก็บข้อมูล (MB)**
LGAP, พื้นฐาน	0%	6.41	6.3E+08	1.5E+07	57.4
LGAP, CASM		**6.41**			
LGAP, พื้นฐาน	10%	6.32	5.1E+08	1.2E+07	46.2
LGAP, CASM		**6.19**			
LGAP, พื้นฐาน	20%	6.64	4.1E+08	9.6E+06	36.6
LGAP, CASM		**6.47**			
LGAP, พื้นฐาน	30%	7.12	3.2E+08	7.4E+06	28.2
LGAP, CASM		**6.88**			
LGAP, พื้นฐาน	40%	7.28	2.4E+08	5.4E+06	20.9
LGAP, CASM		**7.25**			
LGAP, พื้นฐาน	50%	8.07	1.7E+08	3.8E+06	14.6
LGAP, CASM		**7.93**			

ตารางที่ 6.8 (ต่อ) ค่าความผิดพลาดในการทำนายของ LGAP ที่ใช้เฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM

เทคนิค, เฟรมเวิร์ค	อัตราส่วนการบีบอัด	ค่าการทำนายผิดพลาด (%)	FLOPs	พารามิเตอร์	พื้นที่เก็บข้อมูล (MB)
LGAP, พื้นฐาน	60%	8.53	1.2E+08	2.4E+06	9.4
LGAP, CASM		**8.97**			
LGAP, พื้นฐาน	70%	10.34	7.2E+07	1.4E+06	5.5
LGAP, CASM		**11.41**			
LGAP, พื้นฐาน	80%	13.88	3.7E+07	6.4E+05	2.6
LGAP, CASM		**15.82**			
LGAP, พื้นฐาน	90%	23.48	1.4E+07	1.8E+05	0.8
LGAP, CASM		**25.97**			

นอกจากนี้ผลข้างต้นนี้มีการแสดงผลในแง่ของการใช้จำนวนข้อมูลที่ใช้ในเรียกคืนประสิทธิภาพและเวลาเฉลี่ยที่ใช้ในการเรียกคืนประสิทธิภาพของแบบจำลอง VGG-16 ของทั้ง 2 เฟรมเวิร์ค แสดงในตารางที่ 6.9 โดยผลในส่วนนี้แบบจำลองถูกยุบโดยใช้อัตราการบีบอัดที่ 50% และแสดงประสิทธิภาพก่อนและหลังการเรียกคืนประสิทธิภาพ โดยจะเป็นการยุบตัวกรองในเลเยอร์แบบต่อเนื่องกัน ซึ่งเฟรมเวิร์ค CASM จะใช้การสอนข้อมูลเพียง 10 epoch และเฟรมเวิร์คพื้นฐานจะใช้ 1 epoch จำนวนข้อมูลที่ถูกใช้จะใช้ 200, 500 และ 1000 [10] จากชุดข้อมูลสำหรับสอน ใน CASM จะแบ่ง 80% สำหรับการสอน และ 20% สำหรับการทดสอบ ซึ่งในการทำงานการเรียกคืนประสิทธิภาพเวลาในการประมวลผลของแต่เลเยอร์จะถูกนำมาเฉลี่ยและแสดงในเลเยอร์ 12 โดยการวัดเวลาจะวัดจากการทำงานของการ์ดจอ RTX 3090 เห็นได้ว่าเฟรมเวิร์ค CASM สามารถเรียกคืนประสิทธิภาพของแบบจำลองกลับมาได้ดีกว่าเฟรมเวิร์คพื้นฐานที่ใช้ fine-tuning ซึ่งใช้ข้อมูลเพียง 200 ก็สามารถเรียกคืนประสิทธิภาพกลับมาได้สูงกว่าเฟรมเวิร์คพื้นฐานที่ใช้ 1000 ข้อมูล โดยประสิทธิภาพต่างกันอยู่ที่ 0.1498 (0.8050 CASM และ 0.6552 พื้นฐาน) นอกจากนี้ผลในด้านเวลาในการประมวลผลเฉลี่ยในทุกเลเยอร์ของ CASM สามารถประมวลผลได้เร็วกว่าในจำนวนข้อมูล 200, 500, และ 1000 เป็น 3.3 เท่า, 2.4 เท่า และ 1.7 เท่า ตามลำดับ

ตารางที่ 6.9 ค่าความแม่นยำในการเรียกคืนประสิทธิภาพของแบบจำลอง VGG-16 ในแต่ละจำนวนข้อมูลและแต่ละเฟรมเวิร์ค

จำนวนตัวอย่าง	สถานะ/เฟรมเวิร์ค	เลเยอร์					
		1	2	3	4	5	6
200	ก่อน/CASM	0.8357	0.8439	0.8209	0.8431	0.8124	0.7781
	หลัง/CASM	**0.9204**	**0.9220**	**0.9031**	**0.8793**	**0.8400**	**0.8076**
	ก่อน/fine-tuning	0.8357	0.4852	0.4413	0.4637	0.2885	0.3357
	หลัง/fine-tuning	0.833	0.5623	0.4915	0.341	0.3215	0.2432
500	ก่อน/CASM	0.8357	0.8418	0.8336	0.8487	0.8144	0.8016
	หลัง/CASM	**0.9246**	**0.9232**	**0.904**	**0.8859**	**0.862**	**0.8296**
	ก่อน/fine-tuning	0.8357	0.4496	0.4755	0.5213	0.4862	0.4081
	หลัง/fine-tuning	0.8315	0.5922	0.5618	0.5131	0.4383	0.4926
1000	ก่อน/CASM	0.8357	0.8498	0.8106	0.8397	0.8076	0.7774
	หลัง/CASM	**0.925**	**0.9243**	**0.9062**	**0.8902**	**0.8642**	**0.8364**
	ก่อน/fine-tuning	0.8357	0.6569	0.6102	0.6162	0.5167	0.5068
	หลัง/fine-tuning	0.8768	0.7498	0.7052	0.6075	0.5441	0.5602

ตารางที่ 6.9 (ต่อ) ค่าความแม่นยำในการเรียกคืนประสิทธิภาพของแบบจำลอง VGG-16 ในแต่ละจำนวนข้อมูลและแต่ละเฟรมเวิร์ค

จำนวนตัวอย่าง	สถานะ/เฟรมเวิร์ค	เลเยอร์					
		7	8	9	10	11	12
200	ก่อน/CASM	0.7887	0.7755	0.8015	0.8005	0.8046	0.805
	หลัง/CASM	**0.8021**	**0.8054**	**0.8024**	**0.8041**	**0.8048**	**0.805 (1.25s)**
	ก่อน/fine-tuning	0.2342	0.2522	0.3202	0.4046	0.2634	0.2251
	หลัง/fine-tuning	0.2686	0.3346	0.4049	0.3808	0.2935	0.1961 (4.13s)
500	ก่อน/CASM	0.8127	0.8007	0.8216	0.8238	0.8291	0.8291
	หลัง/CASM	**0.8259**	**0.827**	**0.8271**	**0.8293**	**0.8308**	**0.8291 (1.85s)**
	ก่อน/fine-tuning	0.4369	0.3972	0.43	0.5648	0.4453	0.4478
	หลัง/fine-tuning	0.427	0.483	0.5187	0.5865	0.5113	0.5717 (4.35)
1000	ก่อน/CASM	0.8192	0.8091	0.833	0.835	0.837	0.8365
	หลัง/CASM	**0.8344**	**0.8389**	**0.8341**	**0.8355**	**0.8373**	**0.8365 (2.59s)**
	ก่อน/fine-tuning	0.5021	0.5349	0.6071	0.6206	0.6029	0.6469
	หลัง/fine-tuning	0.5997	0.6471	0.6295	0.6651	0.6794	0.6552 (4.43s)

6.2 ผลลัพธ์ของการยุบโครงสร้าง ResNet-50 กับชุดข้อมูล ImageNet

ในส่วนนี้จะแสดงผลลัพธ์การยุบโครงสร้างของ ResNet-50 กับชุดข้อมูล ImageNet โดยผลลัพธ์ที่ถูกแสดงมีดังต่อไปนี้

1. ผลการปรับพารามิเตอร์ในการสอนข้อมูลเพื่อเรียกคืนประสิทธิภาพหลังทำการยุบตัวกรองด้วยเทคนิค LGAP ในเลเยอร์
2. ผลประสิทธิภาพของแบบจำลองที่ถูกยุบตัวกรองของ LGAP ที่ใช้เฟรมเวิร์คพื้นฐานกับเฟรมเวิร์ค CASM และเทคนิคการวิเคราะห์อื่น ๆ ที่ใช้เฟรมเวิร์คพื้นฐาน ในการยุบตัวกรองแบบจำลอง ResNet-50

3. ผลความมั่นใจในการยุบตัวกรองของ LGAP ในเฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM
4. ผลการวิเคราะห์เชิงภาพของแบบจำลองก่อนและหลังทำการยุบตัวกรอง

6.2.1 ผลการปรับพารามิเตอร์ในการสอนข้อมูลเพื่อเรียกคืนประสิทธิภาพหลังทำการยุบตัวกรองด้วยเทคนิค LGAP ในเลเยอร์

ในส่วนนี้จะแสดงผลของการหาพารามิเตอร์ในการเรียกคืนประสิทธิภาพของแบบจำลอง ResNet-50 หลังจากยุบตัวกรองในแต่ละเยอร์ครบทั้ง 32 เลเยอร์ (Residual block ใน ResNet-50 มี 16 block ซึ่งแต่ละ block สามารถยุบตัวกรองในเลเยอร์ได้ 2 เลเยอร์) กับชุดข้อมูล ImageNet ในการทดลองนี้จะใช้อัตราการบีบอัด 50% แสดงในตารางที่ 6.10 การหาพารามิเตอร์จะต้องหาพารามิเตอร์ในการยุบตัวกรองเลเยอร์ก่อนเลเยอร์สุดท้าย (1-31) และพารามิเตอร์ของการยุบตัวกรองเลเยอร์สุดท้าย พารามิเตอร์เหล่านี้จะถูกใช้ในกระบวนการเรียกคืนประสิทธิภาพ โดย ข้อมูล คือ จำนวนข้อมูลที่ใช้ในการสอนข้อมูลซ้ำ ซึ่งได้มาจากชุดข้อมูลสำหรับสอนใน ImageNet, epoch คือ จำนวนรอบที่ใช้ในการสอนข้อมูลซ้ำ, learning rate พารามิเตอร์ที่ใช้ในตั้งค่า optimization

การทดลองที่ 1-3 เป็นการทดลองใช้จำนวนข้อมูลน้อย เพื่อเรียกคืนประสิทธิภาพ ซึ่งผลปรากฏว่าความแม่นยำของแบบจำลองที่ถูกเรียกคืนประสิทธิภาพออกมาต่ำที่สุด การทดลองที่ 4-6 เราจึงขยายจำนวนข้อมูลเพิ่มมากขึ้นทำให้ได้ประสิทธิภาพสูงขึ้นตามลำดับ เป็นผลให้ในการทดลองที่ 7-8 จึงทำการขยายจำนวนข้อมูลเป็นการใช้ทุกรูปในชุดข้อมูลเพื่อใช้ในการสอนแบบจำลอง ดังนั้นเฟรมเวิร์คพื้นฐานในการเรียกคืนประสิทธิภาพจำเป็นต้องใช้ข้อมูลสำหรับการสอนข้อมูลทุกรูป เมื่อต้องการรักษาแบบจำลองให้ได้ประสิทธิภาพสูงสุด สำหรับพารามิเตอร์ epoch ในการยุบตัวกรองที่เลเยอร์ 1-19 ไม่จำเป็นต้องเพิ่ม epoch เพราะประสิทธิภาพจะลดลงกลับมาใกล้เคียงเดิม แม้ว่าจะเพิ่มจำนวน epoch แล้วก็ตาม ในเลเยอร์หลังช่วง 29-32 มีความสำคัญในการเรียกคืนประสิทธิภาพเป็นอย่างมากจึงต้องเพิ่ม epoch เพิ่มขึ้น สำหรับ ResNet-50 พารามิเตอร์ของ learning rate เหมาะที่จะใช้ในช่วง 1E-3 – 1E-5 ซึ่งในเลเยอร์ช่วงต้นเหมาะที่จะใช้ 1E-04 ในกรณีที่ไม่ต้องใช้ epoch มาก แต่เมื่อต้องการให้ประสิทธิภาพค่อยปรับขึ้นต้องเริ่มจาก 1E-3 – 1E-5 เมื่อปรับมาใช้เฟรมเวิร์ค CASM (แสดงในตารางที่ 6.11) ในส่วนของกระบวนการ ประมาณค่าน้ำหนัก (ขั้นตอนที่ 3 ของ CASM) ใช้จำนวนข้อมูลรูปภาพ 2 รูป / class จากชุดข้อมูลสำหรับสอนใน ImageNet แบ่งออกเป็น 2 ส่วน ส่วนที่ 1 ใช้ในการสอนข้อมูล 1600 รูป กับส่วนที่ 2 ใช้ในการทดสอบ 400 รูป หลังจากกระบวนประมาณค่าน้ำหนักต้องใช้การสอนข้อมูลซ้ำแบบบางส่วน คือ การสอนข้อมูลในเลเยอร์ก่อนหน้าทั้งหมดจนถึงเลเยอร์ที่มีการประมาณค่าน้ำหนักและเลเยอร์ส่วน fully-connected ทั้งหมด นอกนั้นส่วนอื่นในโครงข่ายจะไม่มีการเปลี่ยนเปลงของค่าน้ำหนัก ซึ่งเฟรมเวิร์คช่วยลดจำนวน epoch ในการใช้การสอนข้อมูลได้ ทำให้เวลาในการเรียกประสิทธิภาพของเฟรมเวิร์ค CASM ทำได้เร็วขึ้นและนอกจากนี้ยังให้ประสิทธิภาพยังสูงกว่าแบบจำลองเฟรมเวิร์คพื้นฐานอีกด้วย

ตารางที่ 6.10 พารามิเตอร์และค่าความแม่นยำในการเรียกคืนประสิทธิภาพของแบบจำลอง ResNet-50 หลังจากยุบตัวกรองด้วยเทคนิค LGAP และใช้เฟรมเวิร์คพื้นฐาน

case	พารามิเตอร์ก่อนเลเยอร์สุดท้าย				พารามิเตอร์เลเยอร์สุดท้าย				accuracy
	ข้อมูล (รูป/class)	Epoch	เลเยอร์	learning rate	ข้อมูล (รูป/class)	Epoch	เลเยอร์	learning rate	
1	200	1	1-31	1E-04	ทุกรูป	7	32	1E-3, 1E-4	69.59
2	200	1	1-22	1E-04	ทุกรูป	9	32	1E-3, 1E-4	70.16
	400	2	23-31	1E-04					
3	200	1	1-15	1E-04	ทุกรูป	9	32	1E-3, 1E-4	70.19
	600	2	16-31	1E-04					
4	600	1	1-28	1E-04	ทุกรูป	7	32	1E-3, 1E-4	70.07
	800	2	29-31	1E-04					
5	200	1	1-15	1E-04	ทุกรูป	9	32	1E-3, 1E-4	70.89
	600	1	16-19,20-22	1E-4, 1E-3					
	800	2	23-31	1E-03					
6	400	1	1-22	1E-04	ทุกรูป	9	32	1E-3, 1E-4	70.93
	800, ทุกรูป	2	23-31	1E-03					
7	ทุกรูป	1	1-31	1E-04	ทุกรูป	9	32	1E-3, 1E-4	71.06
8	ทุกรูป	1	1-19	1E-04	ทุกรูป	50	32	1E-3, 1E-4, 1E-5	71.34
	ทุกรูป	2	20-28	1E-03					
	ทุกรูป	3	28	1E-03					
	ทุกรูป	12	29-30	1E-03					
	ทุกรูป	20	31	1E-03					

ตารางที่ 6.11 พารามิเตอร์และค่าความแม่นยำในการเรียกคืนประสิทธิภาพของแบบจำลอง ResNet-50 หลังจากยุบตัวกรองด้วยเทคนิค LGAP และใช้ CASM เฟรมเวิร์ค

	พารามิเตอร์ก่อนเลเยอร์สุดท้าย				พารามิเตอร์เลเยอร์สุดท้าย				accuracy
case	ข้อมูล (รูป/class)	Epoch	เลเยอร์	learning rate	ข้อมูล (รูป/class)	Epoch	เลเยอร์	learning rate	
1	2 (CASM)	500	1-31	1E-3	2 (CASM)	500	32	1E-3	
	ทุกรูป (การสอนข้อมูลบางส่วน)	1	1-19, 20-31	1E-4, 1E-3	ทุกรูป (การสอนข้อมูลทั้งหมด)	40	32	1E-3, 1E-4, 1E-5	71.47

6.2.2 ผลประสิทธิภาพของแบบจำลองที่ถูกยุบตัวกรองของ LGAP ที่ใช้เฟรมเวิร์คพื้นฐานกับเฟรมเวิร์ค CASM และเทคนิคการวิเคราะห์อื่น ๆ ที่ใช้เฟรมเวิร์คพื้นฐาน ในการยุบตัวกรองแบบจำลอง ResNet-50

พารามิเตอร์ที่ได้จากผลก่อนหน้าถูกนำมาใช้ในการยุบตัวกรองและเรียกคืนประสิทธิภาพในแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet ในส่วนนี้จะแสดงผลการยุบตัวกรองของทุกเลเยอร์แบบต่อเนื่องที่ใช้อัตราการบีบ 50% เทียบกับวิธีอื่น ๆ ResNet-50 ที่ใช้ชุดข้อมูล ImageNet ข้อมูลแผนผังคุณลักษณะที่ได้ในแต่ละเลเยอร์จะมีขนาดใหญ่กว่าชุดข้อมูล CIFAR-10 มากเพราะรับข้อมูลขาเข้าเป็นขนาด 224x224 และประเภทคำตอบที่มี 1000 ประเภท ผลประสิทธิภาพของแต่ละเทคนิคในการวิเคราะห์ตัวกรองถูกแสดงในตารางที่ 6.12 ซึ่งจะเปรียบเทียบจากประสิทธิภาพในการทำนายผลประเภทข้อมูลที่ลดลง โดยจะพิจารณาค่าความแม่นยำที่ลดลงในการทำนายผลคำตอบอันดับ 1 คือ ประสิทธิภาพที่ลดลงในการทำนายผลข้อมูลที่เป็นคำตอบของประเภทข้อมูลนั้น และการลดลงของความแม่นยำในการทำนายผลคำตอบใน 5 อันดับแรก คือ ประสิทธิภาพที่ลดลงในการทำนายผลข้อมูลที่เป็นคำตอบของประเภทข้อมูลที่อยู่ใน 5 อันดับแรก FLOPs คือ complexity ของแบบจำลองที่วัดจากเลเยอร์คอนโวลูชั่นทั้งหมดในแบบจำลอง ResNet-50 ในการเปรียบเทียบจะเทียบแค่อัตราการบีบอัด 50% ซึ่งจากผลของ VGG-16 กับชุดข้อมูล CIFAR-10 แม้วิธีการแบบสุ่มจะมีประสิทธิภาพที่สูง แต่เมื่อต้องเจอแบบจำลองที่ข้อมูลมีขนาดใหญ่ขึ้นและประเภทข้อมูลมากขึ้น วิธีการสุ่มรักษาความแม่นยำในการทำนายจึงลดลงมากที่สุด สำหรับเทคนิคผลรวมของค่าน้ำหนัก (Weighted sum) ใช้การวิเคราะห์เพียงค่าน้ำหนักอย่างเดียว เมื่อต้องการรักษาให้ประสิทธิภาพลดลงให้น้อยที่สุด การวิเคราะห์เพียงค่าน้ำหนักอาจไม่เพียงพอในแบบจำลองที่ข้อมูลที่ขนาดใหญ่และมีประเภทข้อมูลมาก จะเห็นได้ว่าวิธีอื่น ๆ อาศัยการใช้ข้อมูลส่งเข้าไปในโครงข่ายประสาทเทียม เพื่อวิเคราะห์การทำงานของตัวกรอง การใช้ข้อมูลในการวิเคราะห์ตัวกรองของเทคนิคค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์ (APoZ) จะวิเคราะห์เพียงพิจารณาพื้นที่ศูนย์ที่เกิดขึ้นในตัวกรองหลังจากที่ได้ผลลัพธ์แผนผัง

คุณลักษณะมา ในกรณีที่พื้นที่ศูนย์ต่างกันชัดเจน การวิเคราะห์จะทำได้ดี แต่เมื่อเจอกรณีพื้นที่ศูนย์ที่ใกล้เคียงกันการวิเคราะห์ว่าตัวกรองที่พื้นที่ศูนย์น้อยกว่าเล็กน้อยอาจทำให้การวิเคราะห์ใช้งานไม่ได้หรือพื้นที่ศูนย์เยอะ แต่ตัวกรองดึงคุณลักษณะที่สำคัญที่ใช้ เทคนิคนี้จะทำการวิเคราะห์ไม่ได้ชัดเจนนัก สำหรับเทคนิค SSL และ SSR-L2 มีการทำที่คล้ายกันเพราะ SSR-L2 ใช้หลักการของ SSL ซึ่ง SSR-L2 จะมีความเร็วและประสิทธิภาพสูงกว่าในการทำงาน โดยการวิเคราะห์ตัวกรองของ SSR-L2 จะทำการวิเคราะห์ตัวกรองโดยการใช้ข้อมูลส่งผ่านไปในโครงข่าย เพื่อใช้ผลลัพธ์ของการทำนายผลกับผลลัพธ์ที่เป็นจริงในฟังก์ชันการสูญเสีย และมีการเพิ่ม regularizer เข้าไปในส่วนของฟังก์ชันสูญเสีย regularizer จะเป็นตัวควบคุมการทำ structured sparsity ในแบบจำลอง โดยการ ที่มีพารามิเตอร์ penalty ควบคุมความสัมพันธ์ (trade-off) ระหว่างประสิทธิภาพของแบบจำลองและ structured sparsity ฉะนั้นในการยุบโครงสร้างผ่าน SSR-L2 ต้องควบคุมการบีบอัดผ่านพารามิเตอร์ penalty ซึ่งอาจมีความยุ่งยาก เมื่อเราต้องการกำหนดอัตราการบีบอัดที่ชัดเจน เพราะแต่ละเลเยอร์จะมีอัตราการบีบที่ไม่เท่ากัน ในแง่ของการวิเคราะห์ตัวกรอง SSR-L2 วิเคราะห์ผลของการทำนาย เมื่อมีการทำการยุบตัวกรอง ถือเป็นจุดเด่นที่ช่วยให้รักษาประสิทธิภาพไว้ได้เป็นอย่างดี สำหรับเทคนิค ThiNet จะทำการวิเคราะห์โดยการสุ่มจุดบนแผนผังคุณลักษณะ แล้ววิเคราะห์ว่าเมื่อข้อมูลขาเข้าทำคอนโวลูชั่นกับเลเยอร์คอนโวลูชั่น ตัวกรองตัวใดบ้างที่สามารถทำให้ผลลัพธ์ ณ ตำแหน่งเดียวกัน ให้ผลค่าความผิดพลาด ออกมาต่ำที่สุด สำหรับตัวที่ทำให้เกิดความผิดพลาดสูงจะนำออกจากแบบจำลอง การวิเคราะห์แบบนี้เป็นการวิเคราะห์ความสัมพันธ์เชิงพื้นที่ ซึ่งถือว่าเป็นจุดที่ดี แต่การสุ่มจุดที่ไม่ครอบคลุม อาจจะทำให้การพิจารณาการทำงานของตัวกรองผิดพลาดได้ และการวิเคราะห์ของ ThiNet ยังขาดการวิเคราะห์ไปจนถึงการทำนายผล ในส่วนของเทคนิคค่าเฉลี่ยของ gradient เป็นการคำนวณ gradient จากการทำนายผลมาถึงจุดแผนผังคุณลักษณะ เพื่อคำนวณ gradient ผลลัพธ์ของตัวกรองว่ามีผลแค่ไหนในการทำนายผล ซึ่งถือว่าเป็นอีกหนึ่งวิธีที่มีประสิทธิภาพดี อย่างไรก็ดีในกรณีที่ค่าเฉลี่ย gradient ใกล้เคียงกันจะทำให้การแยกการทำงานตัวกรองที่แตกต่างกันออกจากกันยาก สำหรับวิธี LGAP แบบตัวกรองโดยรวม ถือเป็นวิธีที่รวมจุดเด่นของแต่ละวิธีเข้ามาด้วยกัน ทำให้ประสิทธิภาพในการรักษาความแม่นยำในการทำนายผลสูงกว่าวิธีอื่น เนื่องจาก LGAP ใช้ค่าเฉลี่ย gradient ของการทำนายผลกับแผนผังคุณลักษณะแต่ละตัวที่ได้มาให้น้ำหนักกับแผนผังคุณลักษณะของตัวเองและทำการประมวลผลต่อ เพื่อดูการทำงานของเลเยอร์จากแผนผังคุณลักษณะทั้งหมด ในการวิเคราะห์ตัวกรองจะนำตัวที่ถูกวิเคราะห์ออกจากเลเยอร์และดูผลกระทบของการทำงานของเลเยอร์ ทำให้การวิเคราะห์ของ LGAP มีความสมบูรณ์กว่าวิธีอื่น เพิ่มมาขึ้น เพราะการวิเคราะห์ต้องอิงจุดการทำนายผลที่เป็นผลลัพธ์ของประเภทข้อมูล การวิเคราะห์ตัวกรองร่วมกันแบบ gradient เชิงพื้นที่เกิดโดยการเปรียบเทียบการทำงานที่เปลี่ยนแปลงไป สำหรับเฟรมเวิร์ค CASM ถือเป็นตัวช่วยสำคัญในการเรียกคืนประสิทธิภาพให้กับเทคนิค LGAP เนื่องจากสามารถช่วยเพิ่มประสิทธิภาพในการทำนายผลได้ และลดจำนวน epoch ในการสอนลงเป็นอย่างมาก

ตารางที่ 6.12 ค่าความแม่นยำในการทำนายผลที่ลดลงของแบบจำลอง ResNet-50 ที่ถูกยุบตัวกรอง 50% ในแต่ละเทคนิค

เทคนิค	การลดลงของความแม่นยำในการทำนายผลคำตอบอันดับ 1 (%)	การลดลงของความแม่นยำในการทำนายผลคำตอบใน 5 อันดับแรก (%)	FLOPs	พารามิเตอร์	อัตราการบีบอัด
สุ่ม	4.65%	2.75%	3.4E+09	1.2E+07	50%
SSL [6]	4.58%	2.68%	4.2E+09	1.3E+07	≈ 50%
ThiNet [15]	4.12%	2.28%	3.4E+09	1.2E+07	50%
ค่าเฉลี่ยเปอร์เซ็นต์ของพื้นที่ศูนย์ [8]	4.25%	2.41%	3.4E+09	1.2E+07	50%
ผลรวมของค่าน้ำหนัก [7]	4.31%	2.42%	3.4E+09	1.2E+07	50%
SSR-L2 [16]	3.65%	2.11%	3.4E+09	1.2E+07	50%
ค่าเฉลี่ยของ gradient [9]	3.90%	1.94%	3.4E+09	1.2E+07	50%
LGAP กับเฟรมเวิร์คพื้นฐาน	3.56%	1.89%	3.4E+09	1.2E+07	50%
LGAP กับเฟรมเวิร์ค CASM	**3.43%**	**1.81%**	**3.4E+09**	**1.2E+07**	**50%**

ตารางที่ 6.12 แสดงผลลัพธ์ของเทคนิค LGAP แบบตัวกรองรวมใช้เฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM เปรียบเทียบกับเทคนิคอื่น ๆ แล้ว ยังมีการแสดงผลลัพธ์ของเทคนิค LGAP แบบตัวกรองรวม ที่ใช้อัตราการบีบอัดอื่น ๆ เพิ่มเติม เพื่อตรวจสอบประสิทธิภาพของทั้ง 2 เฟรมเวิร์คในแบบจำลอง ResNet-50 โดยได้แสดงการใช้อัตราการบีบอัดเป็น 30%, 50% และ 70% ในตารางที่ 6.13 ซึ่งจะเป็นการแสดงค่าความผิดพลาดในการทำนายคำตอบอันดับ 1 (top-1 error) และ อันดับ 5 (top-5 error) เมื่อไม่มีการบีบอัด ค่าความผิดพลาดในการทำนายผลคำตอบอันดับ 1 และอันดับ 5 จะเป็น 25.10% และ 7.90% ตามลำดับ ซึ่งในช่วงอัตราการบีบอัด 30%-50% LGAP แบบตัวกรองรวมที่ใช้เฟรมเวิร์ค CASM จะมีค่าความผิดพลาดในการทำนายผลคำตอบอันดับ 1 และอันดับ 5 น้อยกว่าเฟรมเวิร์คพื้นฐาน หากแต่ในกรณีที่อัตราการบีบ 70% เฟรมเวิร์คพื้นฐานมีค่าความผิดพลาดในการทำนายผลคำตอบอันดับ 1 และอันดับ 5 สูงกว่า โดยในการยุบตัวกรองด้วยอัตราการบีบอัด 50% สามารถลด FLOPs ได้ 2.3 เท่า, ลดจำนวนพารามิเตอร์ ได้ 2.1 เท่า และลดขนาดพื้นที่เก็บข้อมูลได้ 2.1 เท่า ตามลำดับ และที่อัตราการบีบอัด 70% สามารถ FLOPs ได้ 3.6 เท่า, ลดจำนวนพารามิเตอร์ได้ 3.0 เท่า และ ลดขนาดพื้นที่เก็บข้อมูลได้ 2.9 เท่า ตามลำดับ

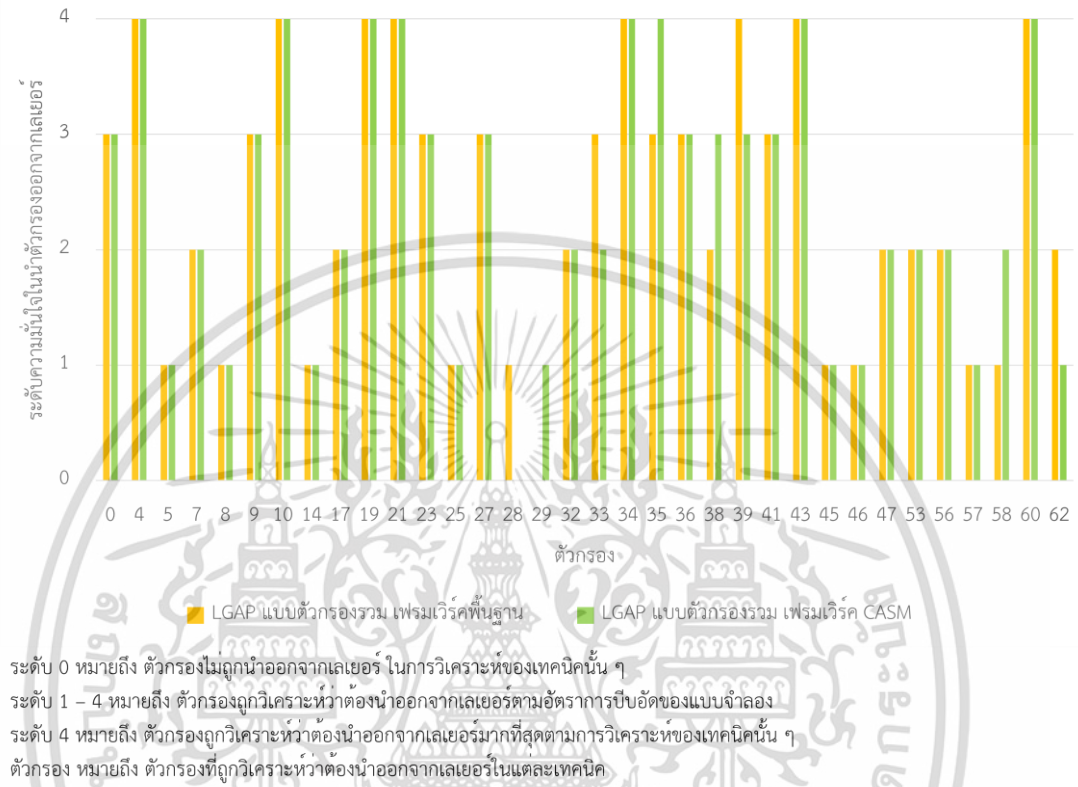
ตารางที่ 6.13 ค่าความผิดพลาดในการทำนายผลของแบบจำลอง ResNet-50 ที่ถูกยุบตัวกรองในเทคนิค LGAP

เฟรมเวิร์ค	อัตราการบีบอัด	ค่าความผิดพลาดในการทำนายคำตอบอันดับ 1 (%)	ค่าความผิดพลาดในการทำนายคำตอบใน 5 อันดับแรก (%)	FLOPs	พารามิเตอร์	พื้นที่เก็บข้อมูล (MB)
-	0%	25.10	7.90	7.7E+09	2.6E+07	98.2
พื้นฐาน	30%	27.55	8.94	4.8E+09	1.7E+07	65.2
CASM		**26.80**	**8.67**			
พื้นฐาน	50%	28.66	9.79	3.4E+09	1.2E+07	47.8
CASM		**28.53**	**9.71**			
พื้นฐาน	70%	31.44	11.26	2.2E+09	8.7E+06	33.6
CASM		**31.47**	**11.45**			

6.2.3 ผลความมั่นใจในการยุบตัวกรองของ LGAP ในเฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM

ในการยุบตัวกรองตัวกรองออกจากเลเยอร์ ทั้ง LGAP แบบตัวกรองโดยรวม 2 เฟรมเวิร์ค จะใช้เทคนิคในการเรียกคืนประสิทธิภาพที่แตกต่างกัน ทำให้อาจมีการเลือกตัวกรองที่แตกต่างกันได้ ในส่วนนี้จะแสดงผลความมั่นใจการยุบตัวกรองของทั้ง 2 เฟรมเวิร์ค โดยเลเยอร์ที่จะแสดงมี 2 เลเยอร์ คือ เลเยอร์ 2 (64 ตัวกรอง) และ เลเยอร์ 32 (512 ตัวกรอง) ซึ่งการการวิเคราะห์จะพิจารณาจากอัตราการบีบอัด 50% การแสดงผลเริ่มจากเลเยอร์ 2 เนื่องจากในเลเยอร์ 1 การวิเคราะห์ตัวกรองด้วย LGAP ทั้ง 2 เฟรมเวิร์คให้ผลลัพธ์การเลือกยุบตัวกรองเหมือนกัน อาจมีแตกต่างกันเล็กน้อยในกรณีที่ใช้จำนวนข้อมูลในการทำการวิเคราะห์ไม่เหมือนกัน ในรูปที่ 6.8 แสดงระดับความมั่นใจในการยุบตัวกรองออกจากเลเยอร์ 2 ของแบบจำลอง ResNet-50 แกนตั้ง คือ ระดับความมั่นใจในการยุบตัวกรองออกจากเลเยอร์ แกนนอน คือ ตำแหน่งของตัวกรอง โดยค่าความมั่นใจระดับ 0 หมายถึง พิจารณาว่าตัวกรองไม่ควรนำออกจากเลเยอร์ และที่ระดับ 4 หมายถึง พิจารณาว่าตัวกรองต้องถูกนำออกจากเลเยอร์ ซึ่งในรูปที่ 6.8 จะแสดงถึงตัวกรองทุกตัวที่เทคนิค LGAP แต่ละเฟรมเวิร์คนำออกในกรณีที่มีระดับความมั่นใจมากกว่า 0 แม้ว่าระดับความมั่นใจจะอยู่ที่ระดับ 1 จะต้องถูกนำออกด้วยอัตราการบีบอัดที่ 50% กรณีที่ตัวกรองไม่ปรากฏระดับความมั่นใจ (ระดับความมั่นใจ เท่ากับ 0) แสดงว่าตัวกรองตัวนั้นไม่ถูกนำออกจากเลเยอร์ในเฟรมเวิร์คนั้น ๆ จำนวนตัวกรองตัวกรองที่ถูกเก็บคือ (จำนวนตัวกรองทั้งหมด x จำนวนเปอร์เซ็นต์ตัวกรองที่เหลืออยู่) - 1 ในส่วนของเลเยอร์ 2 ทั้ง 2

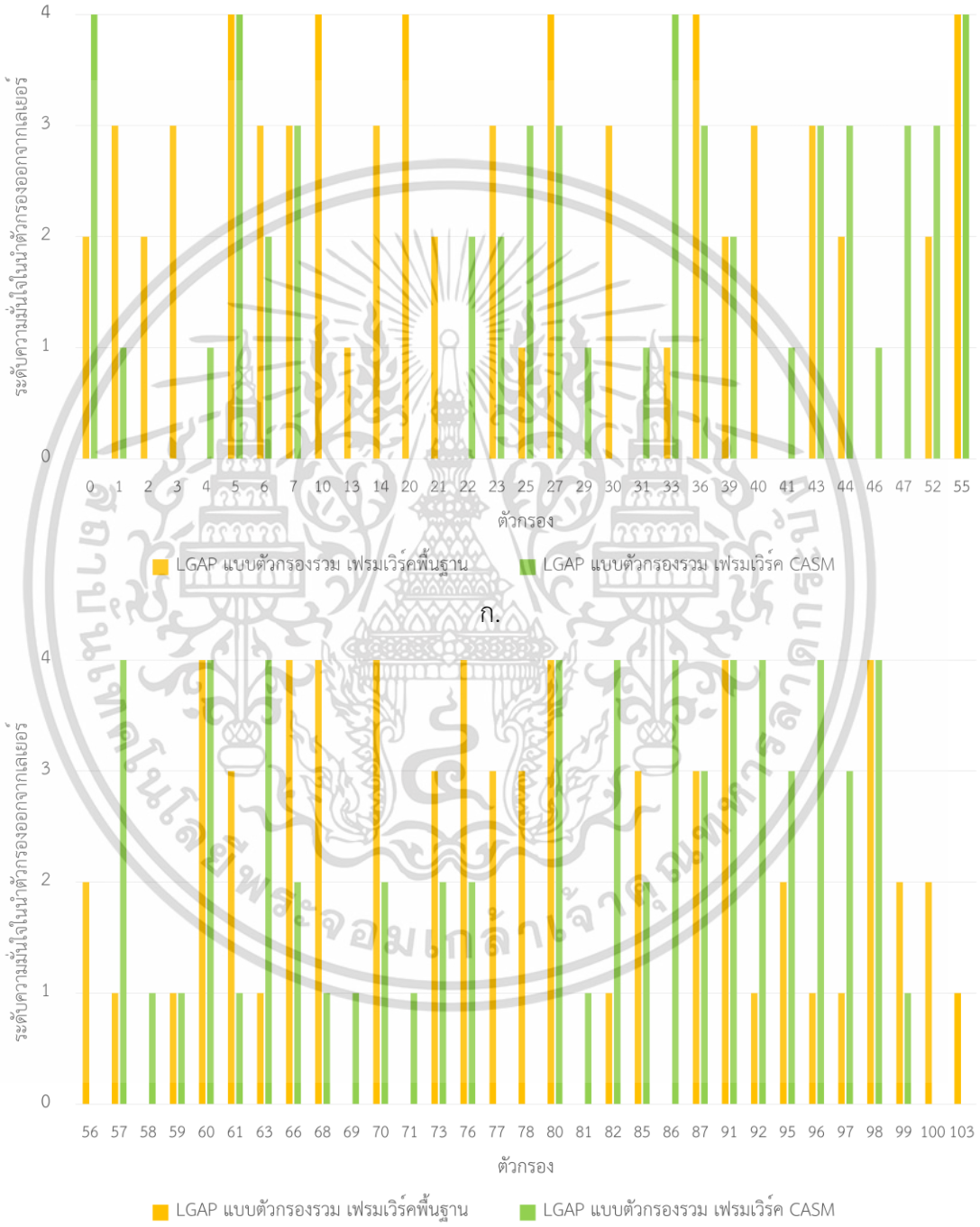
เฟรมเวิร์คเลือกตัวกรองที่นำออกเหมือนกัน แต่ต่างกันตรงที่ระดับความมั่นใจในบางตัวกรอง และมีตัวกรองที่นำออกต่างกัน คือ ตัวกรอง 28 และ ตัวกรอง 29 โดยที่อัตราการบีบอัด 50% เลเยอร์ 2 จะยุบตัวกรองออก 33 ตัว



รูปที่ 6.8 ระดับความมั่นใจในการยุบตัวกรองออกจากเลเยอร์ 2 ของแบบจำลอง ResNet-50

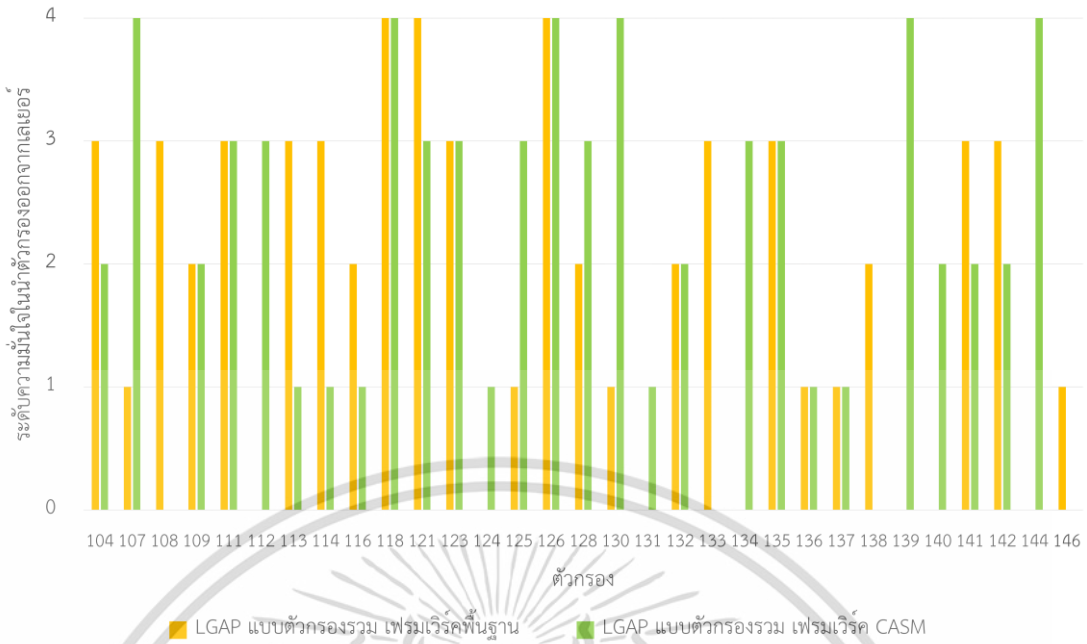
ระดับความมั่นใจในการยุบตัวกรองออกจากเลเยอร์ 32 ของแบบจำลอง ResNet-50 แสดงในรูป 6.9 จะแสดงถึงตัวกรองทุกตัวที่เทคนิค LGAP แบบตัวกรองรวมกับเฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM มีความมั่นใจในการยุบออกในระดับ 0 – 4 โดยในกรณีที่มีระดับความมั่นใจเป็น 0 (ไม่มีแท่งกราฟ) จะหมายถึงกรณีที่เทคนิคนั้นๆ ไม่ต้องการยุบตัวกรองนั้นออกจากแบบจำลอง หากค่าระดับความมั่นใจมากกว่า 0 ตั้งแต่ระดับ 1 ขึ้นไป จะถือว่าสามารถยุบออกจากแบบจำลองได้ และหากตัวกรองใดไม่ปรากฏในกราฟ หมายถึงทุกเทคนิคไม่ต้องการยุบตัวกรองนั้นด้วยอัตราการบีบอัดที่ 50% สำหรับแบบจำลอง ResNet50 ในเลเยอร์ 32 ประกอบไปด้วย 512 ตัวกรอง เมื่อใช้อัตราการบีบอัด 50% ตัวกรองที่ถูกยุบในเลเยอร์ 32 จะมีจำนวน 257 ตัว ในเลเยอร์นี้ทั้ง 2 เฟรมเวิร์คเลือกตัวกรองต่างกันอยู่ 70 ตัว หมายความว่าตัวกรอง 70 ตัวที่อยู่ในกลุ่มของ 257 ตัว ที่ต้องถูกยุบ หรือประมาณ 27% ((70/257)x100) ที่เฟรมเวิร์คพื้นฐานให้ความเชื่อมั่นว่าต้องนำออก (ระดับ 1-4) จะไม่อยู่ในฝั่งของเฟรมเวิร์ค CASM จำนวนตัวกรองที่อยู่ในระดับความเชื่อมั่นเดียวกันของทั้ง 2 เฟรมเวิร์คมีทั้งหมด 67 ตัวกรอง จำนวนตัวกรองที่ระดับความเชื่อมั่นต่างกัน 1 ระดับ มีทั้งหมด 111 ตัวกรอง

จำนวนตัวกรองที่ระดับความเชื่อมั่นต่างกัน 2 ระดับ มีทั้งหมด 70 ตัวกรอง ดังนั้นแล้วทั้ง 2 เฟรมเวิร์คมีความแตกต่างกันในการเลือกตัวกรองและการให้ระดับความเชื่อมั่นในการยุบตัวกรองที่แตกต่างกันสำหรับการยุบออกจากแบบจำลองในเลเยอร์ 32 ทำให้สามารถสรุปได้ว่าทั้ง 2 เฟรมเวิร์คจะเกิดการเลือกตัวกรองที่แตกต่างกัน เมื่อทำการยุบตัวกรองออกจากเลเยอร์แบบต่อเนื่องกัน

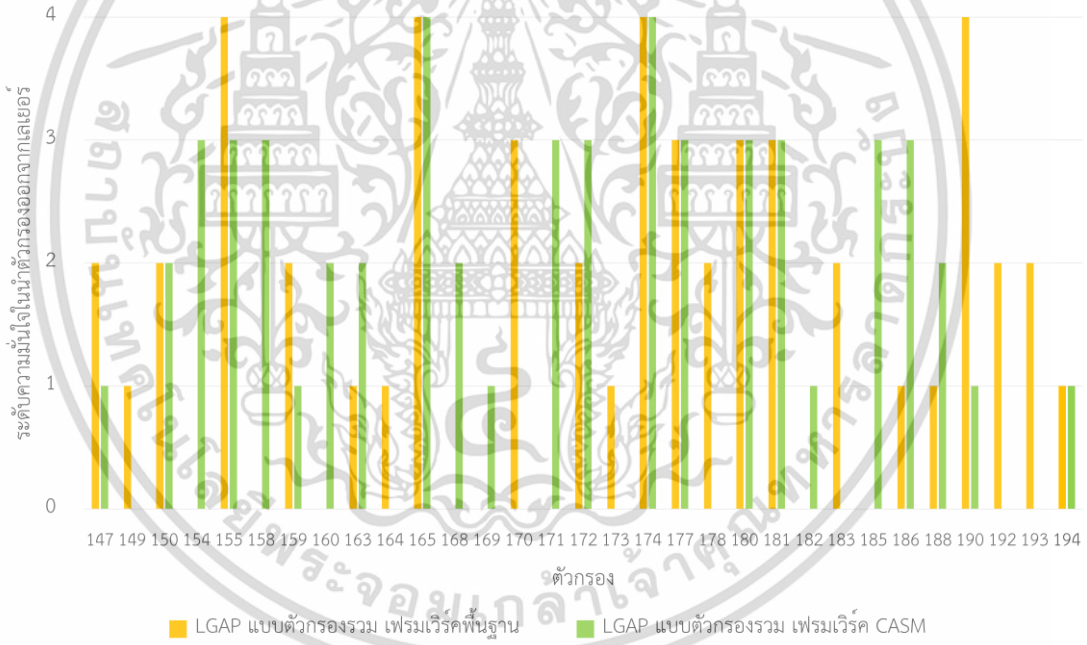


ก.

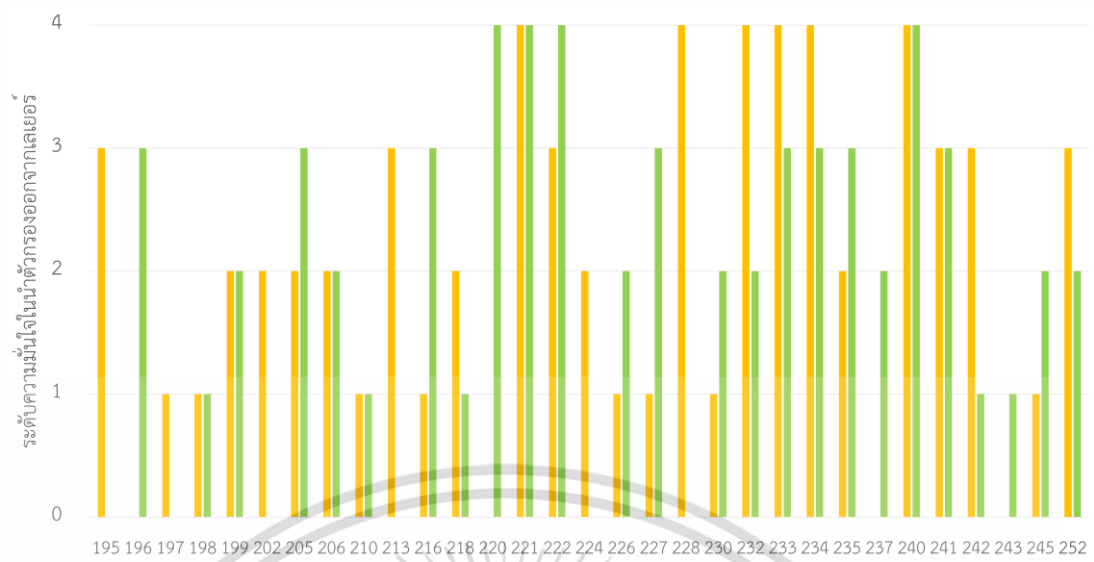
ข.



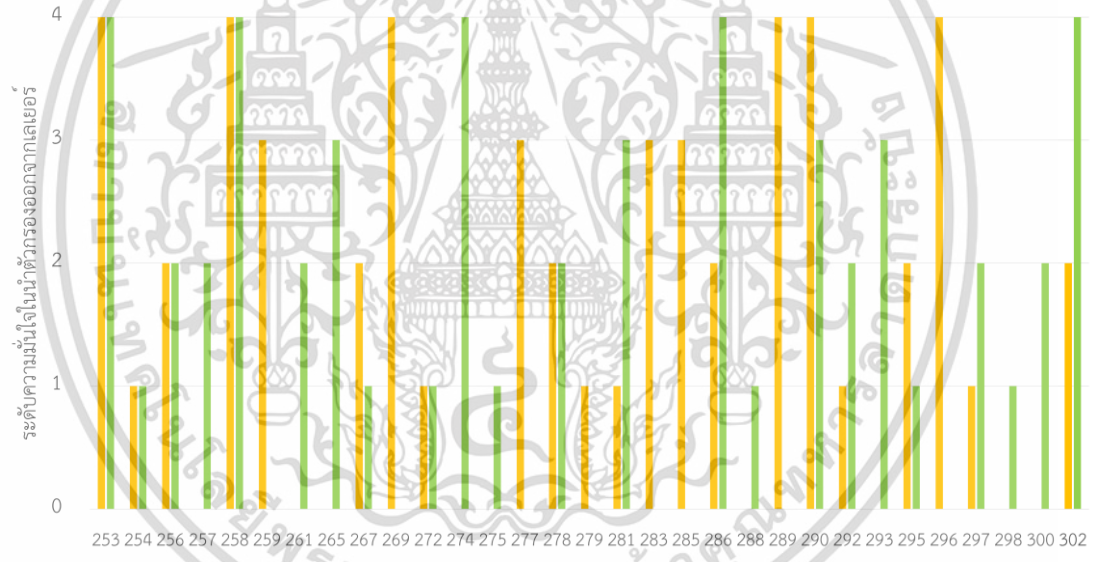
ค.



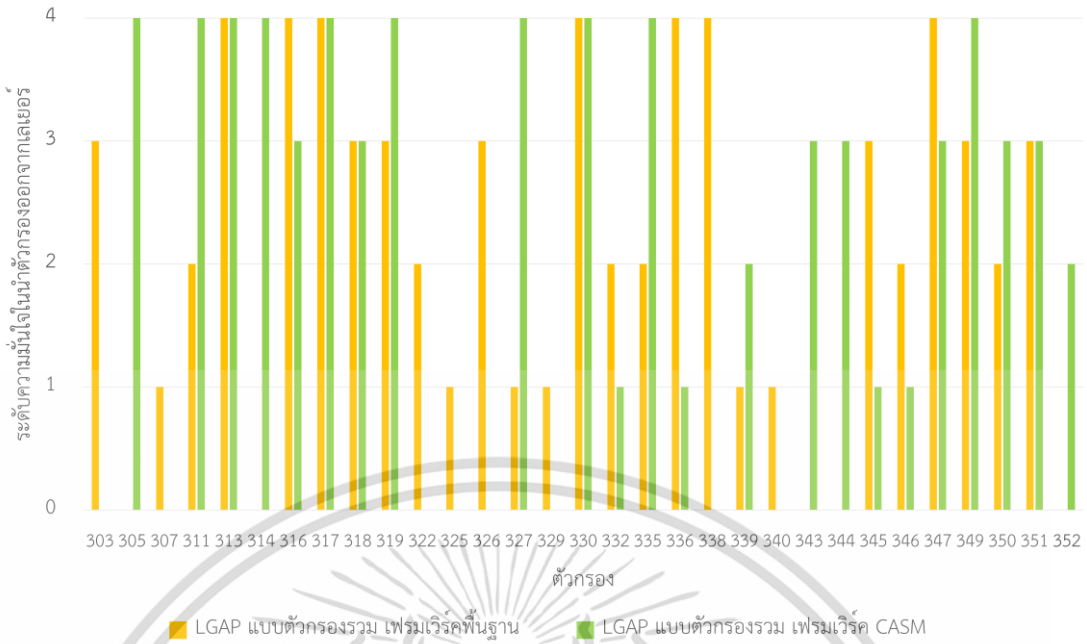
ง.



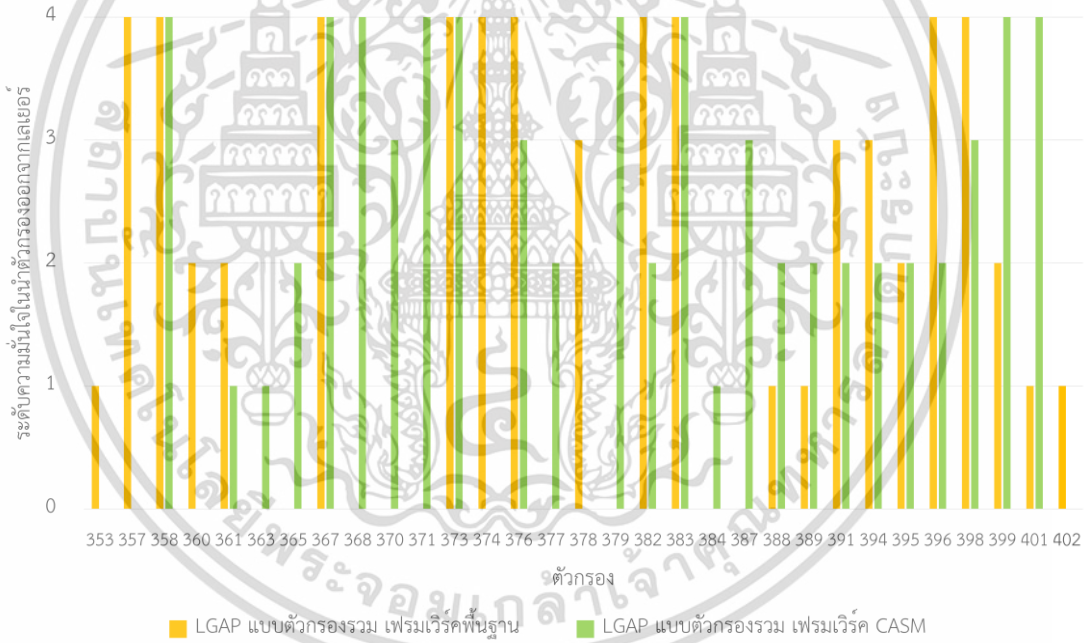
จ.



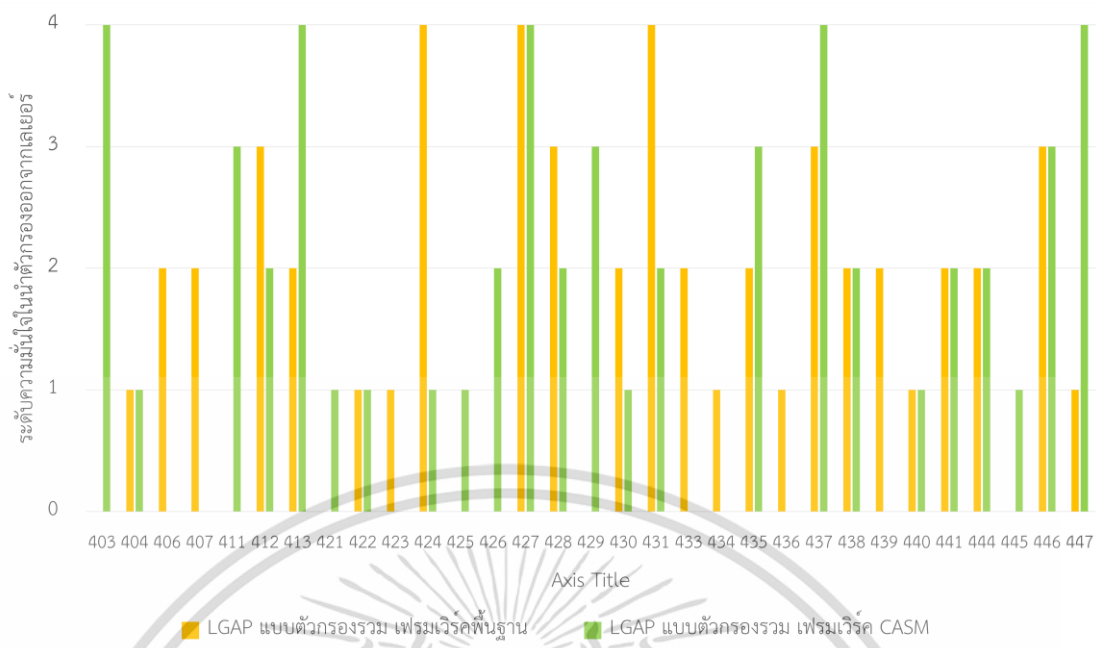
ฉ.



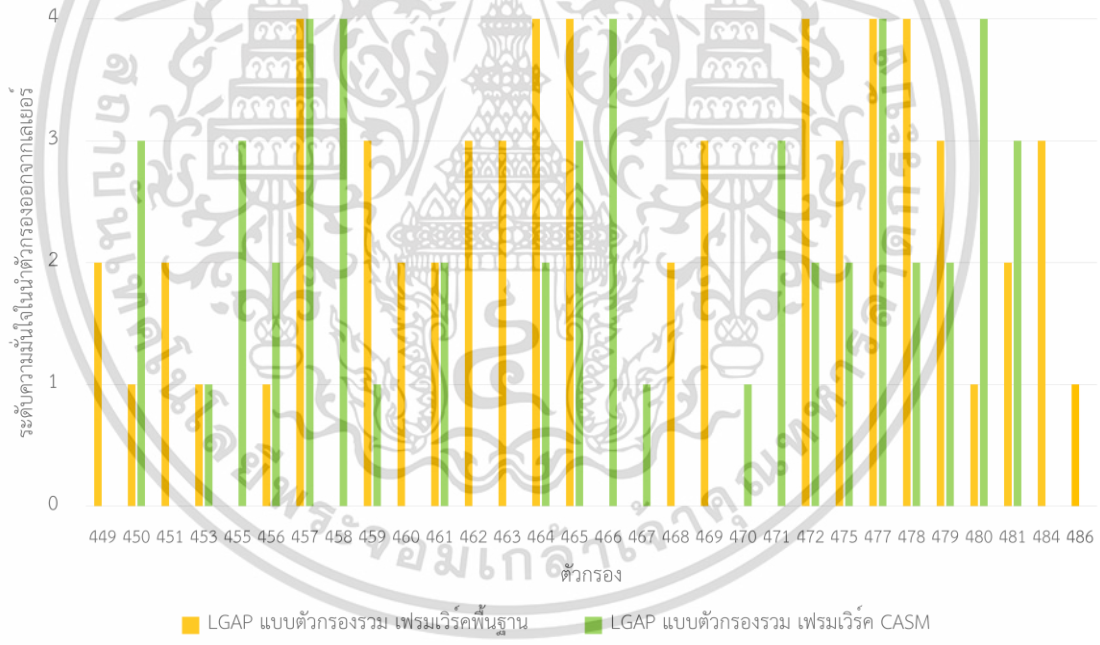
ช.



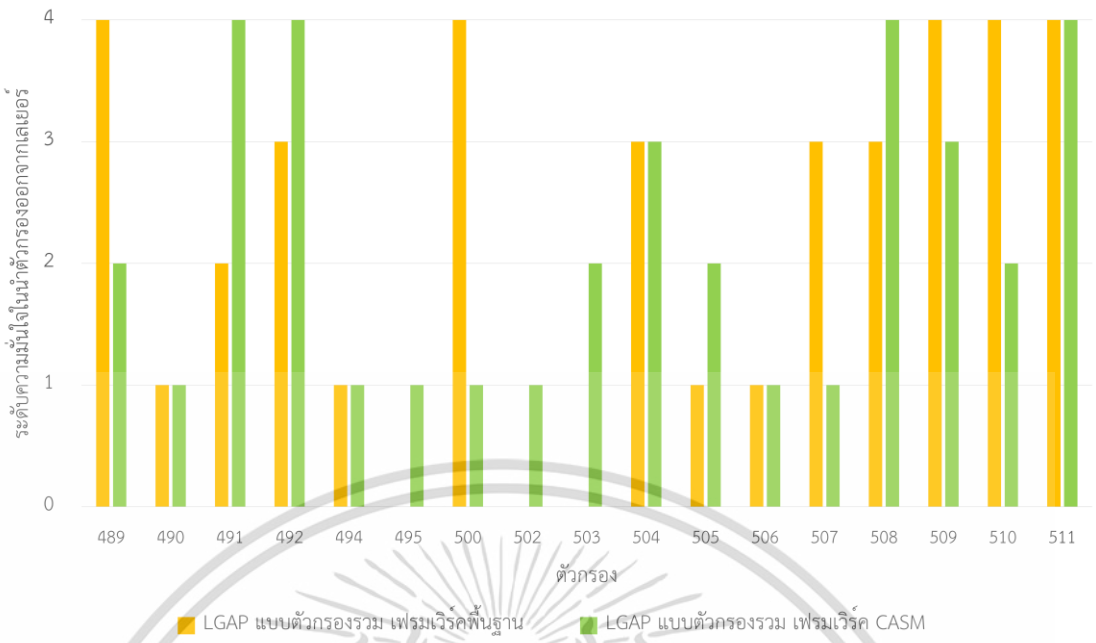
ซ.



ด.



ต.



ระดับ 0 หมายถึง ตัวกรองไม่ถูกนำออกจากเลเยอร์ ในการวิเคราะห์ของเทคนิคนั้น ๆ
ระดับ 1 – 4 หมายถึง ตัวกรองถูกวิเคราะห์ว่าต้องนำออกจากเลเยอร์ตามอัตราการบีบอัดของแบบจำลอง
ระดับ 4 หมายถึง ตัวกรองถูกวิเคราะห์ว่าต้องนำออกจากเลเยอร์มากที่สุดตามการวิเคราะห์ของเทคนิคนั้น ๆ
ตัวกรอง หมายถึง ตัวกรองที่ถูกวิเคราะห์ว่าต้องนำออกจากเลเยอร์ในแต่ละเทคนิค

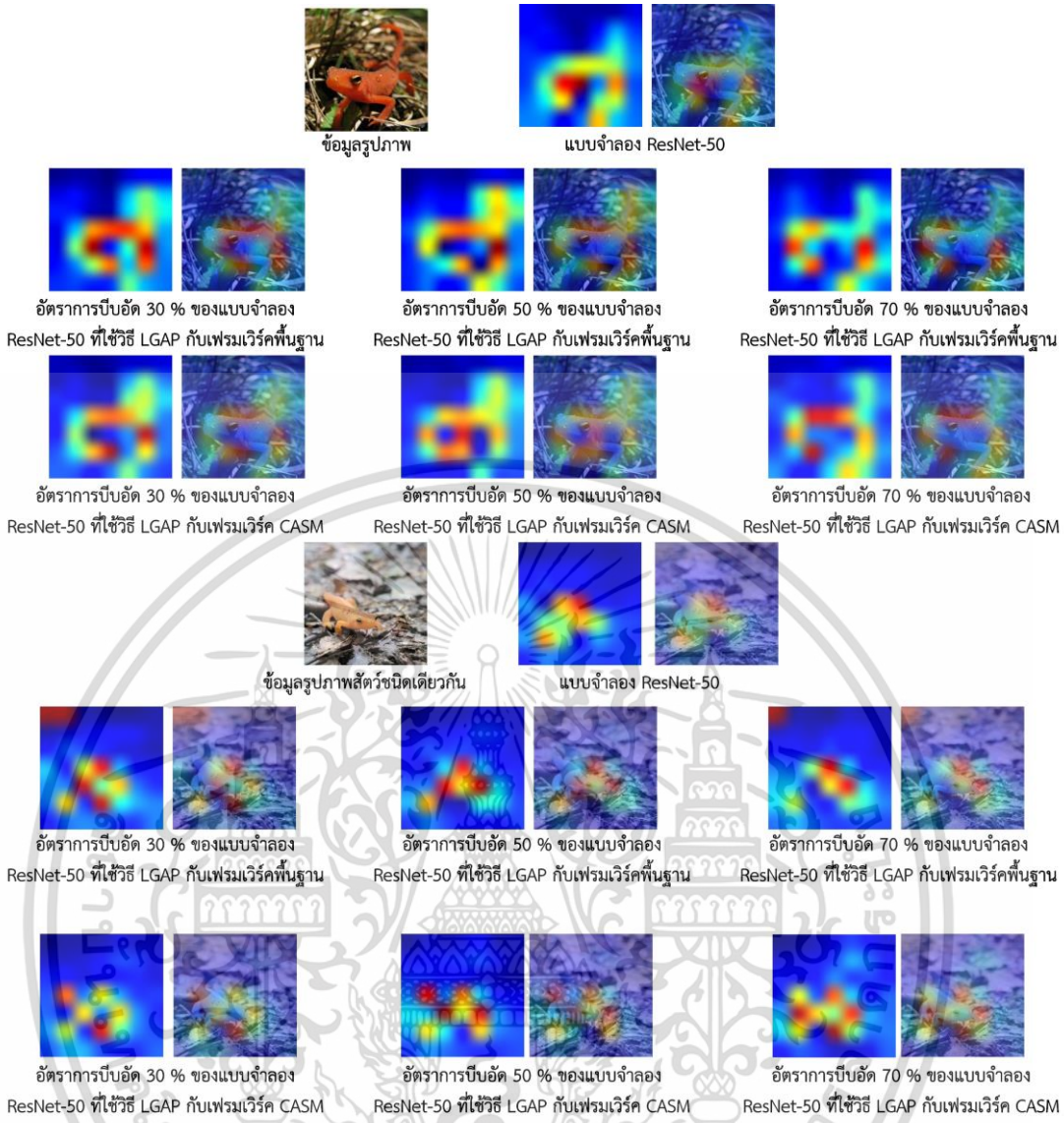
ถ.

รูปที่ 6.9 ระดับความมั่นใจในการยุบตัวกรองออกจากเลเยอร์ 32 ของแบบจำลอง ResNet-50

6.2.4 ผลการวิเคราะห์เชิงภาพของแบบจำลองก่อนและหลังทำการยุบตัวกรอง

ในการแสดงผลด้านการวิเคราะห์เชิงภาพของแบบจำลอง ResNet-50 จะแสดงถึงแผนที่ความร้อนที่เลเยอร์แผนผังคุณลักษณะสุดท้ายของแบบจำลอง โดยการใช้หลักการของ gradient เชิงพื้นที่ในการแสดงผล แสดงในรูปที่ 6.10 โดยการแสดงผลจะแสดงในแบบจำลองของ ResNet-50 ที่เป็นต้นฉบับและแบบจำลอง ResNet-50 ที่ผ่านการยุบตัวกรองด้วยเทคนิค LGAP แบบตัวกรองรวมในเฟรมเวิร์คพื้นฐานและเฟรมเวิร์ค CASM ที่ทำการบีบอัดโดยใช้อัตราการบีบอัดที่ 30%, 50% และ 70% เพื่อดูว่าในการทำนายผลแบบจำลอง ResNet-50 ในแต่ละแบบมีจุดสนใจส่วนใดในภาพ เพื่อใช้ในการทำนายผลประเภทคำตอบ ข้อมูลรูปภาพที่นำมาใช้เป็นรูปภาพจากชุดข้อมูล ImageNet [2] ประเภทข้อมูล eft (n01631663) และได้มีแสดงผลเพิ่มเติม จากการใช้ข้อมูลรูปภาพจากประเภทข้อมูลเดียวกัน แต่เป็นข้อมูลที่ได้จากแหล่งข้อมูล flickr [17] เมื่อนำข้อมูลรูปส่งเข้าไปในแบบจำลอง ResNet-50 ต้นฉบับ จะได้แผนที่ความร้อนแสดงการทำงานของแบบจำลองตามในรูปที่ 6.10 (แบบจำลอง ResNet-50) ในส่วน LGAP กับเฟรมเวิร์คพื้นฐาน เมื่อใช้อัตราบีบอัด 30% แผนที่ความร้อนเปลี่ยนจากแบบจำลองต้นฉบับ เมื่ออัตราการบีบเพิ่มสูงขึ้นเป็น 50% และ 70% แผนที่ความร้อนเปลี่ยนแปลงเพิ่มมากขึ้นและมีจุดที่กระจายออกจากวัตถุเพิ่มมากขึ้น สำหรับ LGAP กับเฟรมเวิร์ค CASM แผนที่ความร้อนในอัตราการบีบ 30% มีจุดสนใจที่บริเวณวัตถุเปลี่ยนแปลงไป เมื่อใช้อัตราการ

บีบที่สูงขึ้นเป็น 50% และ 70% จุดสนใจบริเวณวัตถุเพิ่มมากขึ้น ซึ่งมีความเป็นได้สูงที่การทำนายของแบบจำลองในการทำนายผลประเภทข้อมูลนี้จะทำงานได้ดีขึ้นในเฟรมเวิร์ค CASM ข้อมูลเชิงตัวเลขที่ช่วยยืนยันได้ คือ FI-score ที่วัดความแม่นยำในการทำนายผลของข้อมูลประเภทนี้ ในแบบจำลอง ResNet-50 ต้นฉบับ F1-score ของประเภทข้อมูล n01631663 มีค่าเป็น 0.771 สำหรับแบบจำลอง ResNet-50 ที่ใช้ LGAP ร่วมกับเฟรมเวิร์คพื้นฐาน โดยอัตราการบีบอัด 30%, 50% และ 70% มีค่าเป็น 0.766, 0.804, และ 0.787 ตามลำดับ สำหรับแบบจำลอง ResNet-50 ที่ใช้ LGAP ร่วมกับเฟรมเวิร์ค CASM โดยอัตราการบีบอัด 30%, 50% และ 70% มีค่า 0.776, 0.796, และ 0.817 ตามลำดับ ซึ่งผลในเชิงตัวเลขด้านประสิทธิภาพการทำนายผลประเภทข้อมูลโดยรวมทุกเฟรมเวิร์คประสิทธิภาพลดลง แต่สำหรับประเภทข้อมูล n01631663 เฟรมเวิร์คพื้นฐานจะสูงขึ้น เมื่อใช้อัตราการบีบอัด 50% ส่วนเฟรมเวิร์ค CASM จะเพิ่มขึ้น เมื่ออัตราการบีบอัดเพิ่มขึ้น แสดงให้เห็นว่า CASM ใช้เรียกคืนประสิทธิภาพได้ดีขึ้นในประเภทของข้อมูล อย่าง n01631663 และผลการวิเคราะห์เชิงภาพแสดงให้เห็นว่าแบบจำลอง ResNet-50 ที่ผ่านการยุบตัวกรองมาสามารถใช้งานได้กับข้อมูลมาจากแหล่งอื่นที่ไม่ได้อยู่ในชุดข้อมูล ImageNet แต่เป็นประเภทข้อมูลเดียวกัน สามารถใช้งานได้ดีเช่นเดียวกัน



รูปที่ 6.10 ผลการวิเคราะห์เชิงภาพด้วยเทคนิค gradient เชิงพื้นที่ โดยข้อมูลรูปภาพขาเข้าเป็นรูปจาก ImageNet [2] และ flickr [17] เป็นข้อมูลประเภท n01631663

บทที่ 7
สรุปผลและแนวทางในการพัฒนา

วิทยานิพนธ์ฉบับนี้ได้นำเสนอเทคนิคการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่หรือ LGAP และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น หรือ CASM เพื่อใช้ในการวิเคราะห์ตัวกรองสำหรับยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่นและการเรียกคืนประสิทธิภาพของแบบจำลองหลังจากยุบตัวกรองในโครงข่ายประสาทเทียม ซึ่งเทคนิคนี้ได้ทำการทดลองยุบตัวกรองในแบบจำลอง VGG-16 กับชุดข้อมูล CIFAR-10 และแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet ซึ่งประสิทธิภาพความแม่นยำในการทำนายผลของแบบจำลองยังสามารถรักษาไว้ได้ดีเมื่อเทียบเทคนิคการวิเคราะห์ตัวกรองแบบอื่น ๆ และเฟรมเวิร์ค CASM รักษาเรียกคืนประสิทธิภาพของแบบจำลองที่ถูกยุบกรองกลับมาได้ดีกว่าแบบเฟรมเวิร์คพื้นฐานในการยุบกรองที่น้อยกว่าหรือเท่ากับ 50% นอกจากนี้เฟรมเวิร์ค CASM สามารถลดจำนวน epoch ในช่วงการทำการเรียกคืนประสิทธิภาพแบบจำลองหลังการยุบตัวกรองโดยการสอนข้อมูลซ้ำแบบสอนข้อมูลบางส่วน (ทำการสอนข้อมูลตั้งแต่เลเยอร์แรกจนถึงเลเยอร์ที่ทำกระบวนการ CASM และเลเยอร์ fully connected) และแบบสอนข้อมูลทุกส่วนได้ (สอนข้อมูลทุกเลเยอร์ในโครงข่าย) ส่งผลให้ลดเวลาในกระบวนการเรียกคืนประสิทธิภาพ

7.1 สรุปผลการดำเนินงานวิจัย

ในงานวิจัยนี้ ได้นำเสนอวิธีการยุบโครงข่ายประสาทเทียมแบบคอนโวลูชั่นโดยการยุบตัวกรอง ด้วยเทคนิคการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่หรือ LGAP ในการวิเคราะห์ความสำคัญของตัวกรองจะทำการวิเคราะห์ความสัมพันธ์ของตัวกรองกับความสัมพันธ์ในการทำนายผลร่วมกัน ทำให้สามารถวิเคราะห์ในรูปแบบของความสัมพันธ์เชิงพื้นที่ได้ เมื่อนำไปวิเคราะห์ตัวกรองเพื่อจัดอันดับความสำคัญของตัวกรองก่อนทำการยุบออกจากแบบจำลอง ช่วยให้เกิดประสิทธิภาพในการยุบโครงข่ายประสาทเทียบได้ดีที่ขึ้น เมื่อเทียบกับเทคนิคอื่น ๆ ที่ถูกนำมาเปรียบเทียบ โดยแบบจำลองที่นำมาใช้ในการยุบตัวกรองได้แต่ VGG-16 กับชุดข้อมูล CIFAR-10 และแบบจำลอง ResNet-50 กับชุดข้อมูล ImageNet โดยในการยุบแบบจำลอง VGG-16 เมื่อถูกยุบตัวกรองโดยใช้อัตราการบีบอัดที่ 90% ยังสามารถรักษาประสิทธิภาพได้สูงที่สุด สำหรับในแบบจำลอง ResNet-50 เมื่อเปรียบเทียบการยุบตัวกรองในอัตราการบีบอัด 50% สามารถรักษาประสิทธิภาพตัวกรองได้สูงสุด นอกจากนี้เฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น หรือ CASM เป็นเฟรมเวิร์คที่ช่วยเพิ่มประสิทธิภาพในการเรียกคืนประสิทธิภาพของแบบจำลองหลังยุบตัวกรองได้ดียิ่งขึ้น เมื่อแบบจำลองใช้อัตราการบีบอัดน้อยกว่าหรือเท่ากับ 50% ประสิทธิภาพสามารถกลับมาได้ดีกว่าแบบเฟรมเวิร์คพื้นฐาน ซึ่งเฟรมเวิร์ค CASM สามารถช่วยลดระยะเวลาในการเรียกคืนประสิทธิภาพได้

จากการลด epoch ในการสอนข้อมูลซ้ำ ซึ่งเทคนิคการจับคู่แผนผังคุณลักษณะกับ gradient เชิงพื้นที่ หรือ LGAP และเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น หรือ CASM เป็นเฟรมเวิร์ค ได้ถูกพิสูจน์แล้วว่ามีความสามารถเพียงพอในการใช้งานด้านการยุบโครงสร้างในโครงข่ายประสาทเทียมแบบคอนโวลูชั่น

7.2 แนวทางในการพัฒนา

ในงานวิจัยพบว่าเฟรมเวิร์คแบบจำลองขนาดเล็กสำหรับการประมาณคอนโวลูชั่น หรือ CASM ยังเรียกคืนประสิทธิภาพได้ไม่ดีกว่าเฟรมเวิร์คพื้นฐาน เมื่อใช้อัตราการบีบอัดเกินกว่า 50% ในการยุบตัวกรอง ซึ่งน่าจะเป็นผลจากการนำตัวกรองออกมากเกินไป ทำให้ไม่สามารถจำลองการทำงานของตัวกรองกลับมาได้เหมือนกับแบบจำลองต้นฉบับ ซึ่งในการพัฒนางานวิจัยต่อไป เฟรมเวิร์ค CASM น่าจะสามารถเพิ่มประสิทธิภาพได้โดยการอิงการใช้แบบจำลองต้นฉบับใหม่ได้ เมื่ออัตราการบีบอัดแบบจำลองสูงขึ้น กล่าวคือ เมื่อทำการบีบอัดแบบจำลองเกิน 50% อาจจะใช้แบบจำลองที่บีบอัดที่ 40% หรือน้อยกว่านั้นมาใช้เป็นต้นฉบับในระหว่างกระบวนการการเรียกคืนประสิทธิภาพโดยวิธี CASM เพราะมีความเป็นไปได้ที่จะสามารถเรียกคืนประสิทธิภาพกลับไปได้สูงกว่าการใช้แบบจำลองต้นฉบับที่ไม่มีการบีบอัด เพราะจำนวนตัวกรองที่เหลืออยู่อาจจะเพียงพอในการเรียกประสิทธิภาพของแบบจำลองที่ผ่านการบีบอัดมาแล้วได้

นอกจากนี้หากต้องการนำงานวิจัยไปพัฒนาเพื่อใช้งานจริง สามารถทำได้กับสถาปัตยกรรมที่เป็นเลเยอร์คอนโวลูชั่น สำหรับชุดคำสั่งของ Keras แต่สำหรับการใช้ในสถาปัตยกรรมใหม่ที่แตกต่างออกไปจากนี้จำเป็นต้องทดสอบเพื่อวิเคราะห์พารามิเตอร์กับผลของประสิทธิภาพของแบบจำลองเพิ่มเติมก่อน

เอกสารอ้างอิง

[1] A. Krizhevsky. “Learning multiple layers of features from tiny images” **Learning Multiple Layers of Features from Tiny Images,** 2009.

[2] O. Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge” **International Journal of Computer Vision,** vol. 115, no. 3, 2015. pp. 211–252.

[3] K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition” **Very Deep Convolutional Networks for Large-scale Image Recognition,** 2014.

[4] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition” **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,** 2016. pp. 770–778.

[5] V. Lebedev and V. Lempitsky. “Fast ConvNets Using Group-Wise Brain Damage” **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,** 2016. pp. 2554–2564.

[6] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. “Learning structured sparsity in deep neural networks” **Advances in Neural Information Processing Systems,** 2016. pp. 2082–2090.

[7] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. “Pruning filters for efficient convnets” **arXiv,** 2016.

[8] H. Hu, R. Peng, Y. Tai, and C. Tang. “Network trimming: A data-driven neuron pruning approach towards efficient deep architectures” **CoRR,** 2016.

[9] C. Liu and H. Wu. “Channel pruning based on mean gradient for accelerating Convolutional Neural Networks” **Signal Processing,** vol. 156, 2019. pp. 84–91.

[10] A. Alqahtani, X. Xie, M. W. Jones, and E. Essa. “Pruning CNN filters via quantifying the importance of deep visual representations” **Computer Vision and Image Understanding,** vol. 208–209, 2021.

[11] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization” **Proceedings of the IEEE International Conference on Computer Vision,** 2017. pp. 618–626.

[12] S. Liu and W. Deng. “Very deep convolutional neural network based image classification using small training sample size” **Proceedings - 3rd IAPR Asian Conference on Pattern Recognition,** ACPR 2015, 2016. pp. 730–734.

[13] S. Ioffe and C. Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift” **32nd International Conference on Machine Learning,** ICML 2015, vol. 1, 2015. pp. 448–456.

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. “Dropout: A simple way to prevent neural networks from overfitting” **Journal of Machine Learning Research,** vol. 15, 2014. pp. 1929–1958.

[15] J.-H. Luo, J. Wu, and W. Lin. “ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression” **2017 IEEE International Conference on Computer Vision (ICCV),** 2017. pp. 5068–5076.

[16] S. Lin, R. Ji, Y. Li, C. Deng, and X. Li. “Toward Compact ConvNets via Structure-Sparsity Regularized Filter Pruning” **IEEE Transactions on Neural Networks and Learning Systems,** vol. 31, no. 2, 2020. pp. 574–588.

[17] L. Enders. **“Red Eft in Leaf Litter.”** [Online]. Available: https://www.flickr.com/photos/usfwsnortheast/51277223193/. 2021.

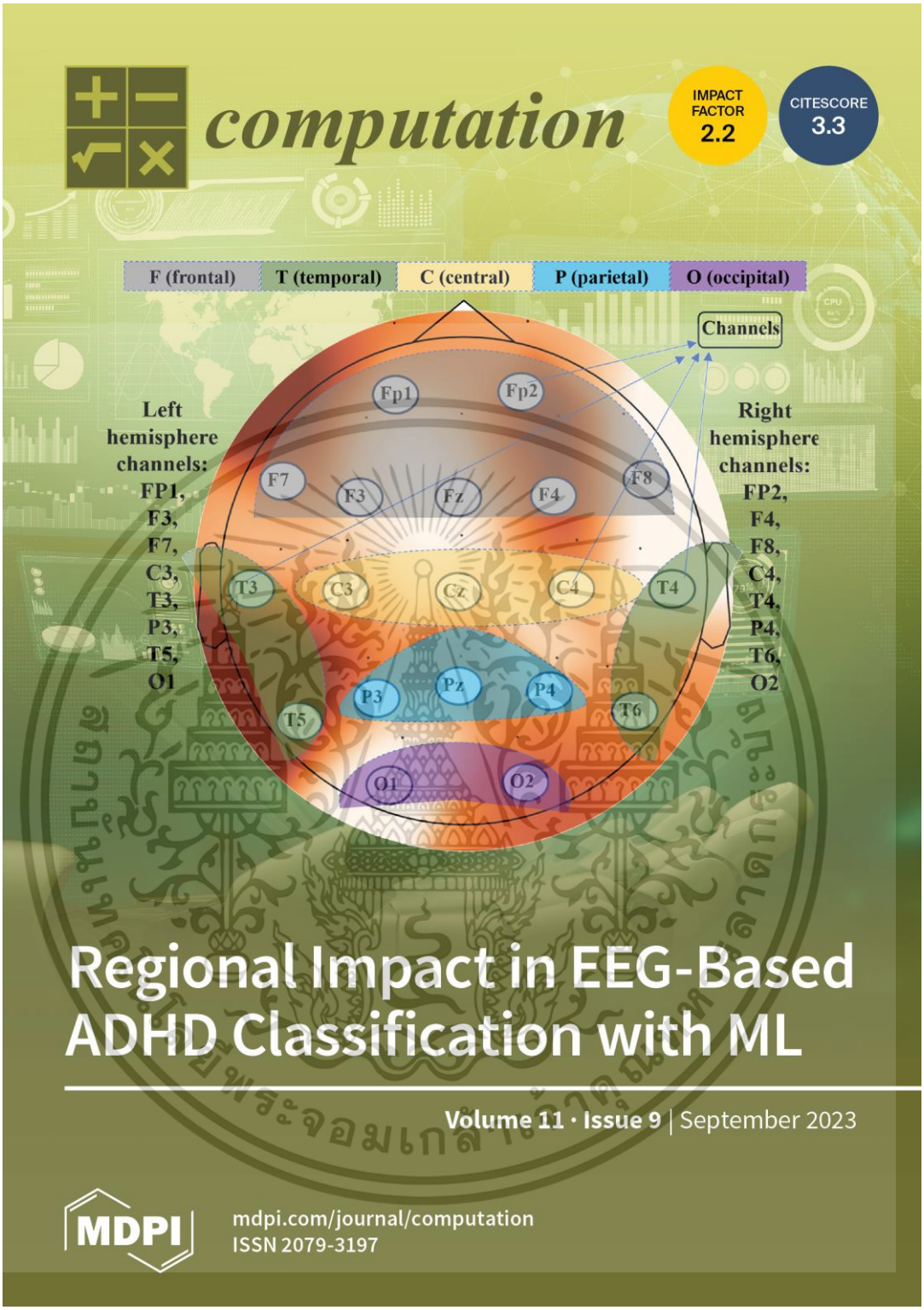
[18] M. Intaraprasit and O. Chitsobhuk. “Filter Pruning Based on Local Gradient Activation Mapping in Convolutional Neural Networks” **International Journal of Innovative Computing, Information and Control,** vol. 19, no. 6, 2023.

[19] M. Intaraprasit and O. Chitsobhuk. “Filter Pruning with Convolutional Approximation Small Model Framework” **Computation,** vol. 11, no. 9, 2023.

ภาคผนวก

ภาคผนวก ก. ผลงานที่ได้รับการตีพิมพ์

ภาคผนวก ก.
ผลงานที่ได้รับการตีพิมพ์



computation

IMPACT FACTOR 2.2

CITESCORE 3.3

Article

Filter Pruning with Convolutional Approximation Small Model Framework

Monthon Intraraprasit and Orachat Chitsobhuk

MDPI

https://doi.org/10.3390/computation11090176

computation

MDPI

Article

Filter Pruning with Convolutional Approximation Small Model Framework

Monthon Intraraprasit * and Orachat Chitsobhuk *

School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Chalongkrung Road, Ladkrabang, Bangkok 10520, Thailand
* Correspondence: monthon.int@gmail.com (M.I.); orachat.ch@kmitl.ac.th (O.C.)

Abstract: Convolutional neural networks (CNNs) are extensively utilized in computer vision; however, they pose challenges in terms of computational time and storage requirements. To address this issue, one well-known approach is filter pruning. However, fine-tuning pruned models necessitates substantial computing power and a large retraining dataset. To restore model performance after pruning each layer, we propose the Convolutional Approximation Small Model (CASM) framework. CASM involves training a compact model with the remaining kernels and optimizing their weights to restore feature maps that resemble the original kernels. This method requires less complexity and fewer training samples compared to basic fine-tuning. We evaluate the performance of CASM on the CIFAR-10 and ImageNet datasets using VGG-16 and ResNet-50 models. The experimental results demonstrate that CASM surpasses the basic fine-tuning framework in terms of time acceleration (3.3× faster), requiring a smaller dataset for performance recovery after pruning, and achieving enhanced accuracy.

Keywords: filter pruning framework; convolutional neural networks; deep learning; model compression

Citation: Intraraprasit, M.; Chitsobhuk, O. Filter Pruning with Convolutional Approximation Small Model Framework. *Computation* **2023**, *11*, 176. https://doi.org/10.3390/computation11090176

Academic Editor: Junlin Hu

Received: 21 June 2023
Revised: 4 August 2023
Accepted: 16 August 2023
Published: 5 September 2023

1. Introduction

The use of convolutional neural networks (CNNs) for the analysis of human issues is widespread throughout both the commercial and academic domains. They are particularly helpful in computer vision for problems such as image classification [1–3], object recognition [4–6], object detection [7,8], semantic segmentation [9–11], etc. CNNs have demonstrated remarkable effectiveness in recent years. However, they have complex, multi-layered structures with numerous interconnections. The use of filters at each level of a CNN's structure makes it possible to record recurring patterns in the data. For example, ResNet-50 [12] includes 25.6 million parameters with required 98.2 Megabytes (MB) storage space. This massive structure impacts the total amount of time spent on learning and prediction. Moreover, it is difficult to execute CNN models on hardware with limited resources, such as a microcontroller or a smartphone, which may have a limited amount of battery life, computational units, or storage space [13]. Compressing models is an essential technique to operate CNN models efficiently with constrained resources. Model compression approaches can be categorized into the following three categories [14]: (1) Precision reduction is a method aiming to reduce the number of bits needed to represent CNN weights without sacrificing the model's performance [14]. However, the model effectiveness is sensitive to the criteria for minimizing the number of bits, despite the fact that this approach can minimize storage and power consumption. (2) Network Pruning is a compression approach used to assess weights with minimal influence on model performance. Retraining is completed once the specified weights are set to zero or removed from the CNN. The weight removal approach is challenging, and the outcomes are heavily reliant on weight analysis. Inadequate weight analysis can lead to decreased accuracy and unproductive compression issues [14,15]. (3) Compact network is a method, where the number of weights in a CNN are reduced by modifying the kernel design, resulting in a

more compact network architecture. The underlying structure has been replaced by a series of convolution layers with smaller kernel sizes.

Filter-level pruning stands out as a prominent network pruning technique, being acknowledged as one of the most favorable approaches for compressing Convolutional Neural Networks (CNNs). The utilization of this strategy in network training implies its potential in removing a number of unnecessary weights, all while minimizing the impact on the model's overall performance. Generally, the framework for filter-level pruning encompasses three key processes: filter analysis, filter pruning, and model fine-tuning. The efficacy of filter-level pruning predominantly depends on the criteria employed for filter analysis. Moreover, we observe that the process of fine-tuning can exert an influence on the performance of filter-level pruning during the recovery phase, owing to the varying sensitivity of individual layers. It is conceivable that relying solely on fine-tuning might prove inadequate in fully restoring model performance after each layer's pruning. Prolonging the fine-tuning phase by increasing the number of epochs, especially when reliant on an extensive dataset, would demand a significant time investment to restore model performance, with no assurance of achieving optimality due to the disparate sensitivities exhibited by different layers.

Therefore, we propose a new framework that combines the filter pruning strategy with the Kernel Recovery (KR) technique. The approach for filter pruning is based on our Localized Gradient Activation heatmaP (LGAP), which considers the spatial correlation between the investigated filter and the target prediction [16]. The weak filters with the high filter score are considered as insignificant filters and removed from the layer. The KR is integrated in order to minimize the damage of the pruned model. The proposed KR is based on a kernel weight reconstruction, wherein the weights of the pruned layer are optimized to maintain the feature map outputs at the subsequent layer as closely aligned as possible with those of the original unpruned model. This KR process requires less computational time compared to the fine-tuning technique, resulting in faster implementation of the model compression with comparable performance.

2. Related Work

Compression methods can be utilized to tackle the issue of an overwhelming number of parameters in deep neural networks. The ideal version of the brain damage approach was first introduced in the field of network pruning to eliminate unnecessary model coefficients in CNN acceleration [17]. A second-order Taylor expansion was used to determine the regularization parameters. Nevertheless, as compared to traditional fine-tuning, this approach involved the computation of the Hessian matrix, which required extra memory and increased processing costs. Lebedev and Lempitsky [18] proposed novel research that integrates group-wise pruning into the learning process, which empowered the regularization of group sparsity. Since the majority of their values were nearly zero, it might be possible to remove some weight groups. Structured Sparsity Learning (SSL) was implemented in [19], where regularization of CNN models is feasible for a wide range of techniques, including the application of filters, filter shapes, channels, and layer depth. SSL offered efficient pruned architecture. However, the fundamental architecture was destroyed resulting in unsupported libraries. Other filter level pruning approaches that were designed to preserve the network structure include the random technique, the weight sum technique [20], the average percentage of zeros (APoZ) [21], the mean gradient [22], and TriNet [23]. They are established on the same premise.

The first process is filter selection; each technique analyzes filters based on criteria for identifying strong and weak filters. (1) Pruning by random, in which filters were eliminated at random. (2) The weight sum approach [20] measured filter significance using the sum of absolute weights. The larger the weight sum, the greater the significance of the filter. (3) The Average Percentage of Zeros (APoZ) [21] measured the percentage of zero values in the output activation with a high APoZ indicating that the filter was unnecessary and should be eliminated. (4) Mean gradient [22] utilized the concept of a mean gradient to

examine the layer feature maps. (5) TriNet [23] implemented a greedy channel selection strategy, where the input channels with the least increase in reconstruction error were pruned from the target layer. (6) Structured sparsity regularization (SSR) [24] was an approach where the relation between the global output loss and the local filter removal was introduced into the objective function along with the structured sparsity constraint. Alternative Updating with Lagrange Multipliers (AULM) was applied with an adaptive filter pruning task. AULM utilizes structure sparse regularizers, with two different types to handle convergence challenge. It was claimed that the method could achieve a fast convergence rate.

The second process is filter pruning; The ineffective filters are eliminated from the CNN model. The third procedure is model fine-tuning, which is employed to restore performance after the pruning process. Unfortunately, fine-tuning takes time and does not ensure optimal performance. Certain layers with high sensitivity [20] may require more than one fine-tuning period to be adequately recovered. This indicates that each layer has a different amount of sensitivity, and as a result, model pruning with the same pruning ratio can lead to either significant or minimal influence on the model's performance.

One of the key differences among various filter pruning techniques lies in the filter selection strategies. The majority of existing techniques mainly rely on statistics or activation-based criteria (such as weighted sum, Average Percentage of Zeros (APoZ), and Structured sparsity regularization (SSR)) to identify important filters for model pruning. However, these approaches lack a general understanding of the entire model network since they perform filter analysis on the individual kernel basis, rather than utilizing the layer-wise kernel relationship that should encourage data forward to the prediction layer. Consequently, they may suffer from incomplete relational layer data analysis, limiting their ability to establish a strong connection between individual neurons and the final prediction. For instance, TriNet [23] tries to determine the least impactful filter channels on the output feature map by employing reconstruction loss through least-square estimation. While statistical analyses have their place, they lack spatial data relevant to target prediction, which can be crucial for understanding model predictions in depth.

In contrast, our previous work introduced the Localized Gradient Activation heatmap (LGAP) [16], which takes a data feed-forward approach to explore the spatial relationships among filters in each layer. LGAP, as opposed to single filter analysis, concentrates on distinct weak filters within localized regions, allowing for the discovery of critical locations in target prediction. This method has proven to outperform conventional statistical pruning strategies and effectively preserves the performance of the pruned model.

However, one challenge that most pruning techniques face is the time-consuming fine-tuning process required for recovering the model's performance. To address this issue, an alternative framework was proposed [25] that aimed to estimate new kernels using partial convolution to restore the pruned model's performance. Unfortunately, this new framework lacks clarity in its multi-channel estimates, leading to the repetitive execution of single channel and single filter approximations. As a result, the feasibility of this approach may be uncertain, and it could potentially impact the overall effectiveness of the fine-tuning process.

In conclusion, LGAP stands out as a promising approach by effectively exploring spatial relationships among filters and achieving superior performance compared to statistical pruning techniques. However, further improvements are needed in the proposed framework for fine-tuning to ensure its effectiveness and feasibility in restoring the model's performance after pruning. Obtaining the right balance between spatial understanding and performance recovery remains a challenge for successful filter pruning and model compression.

Therefore, in our research, we introduce a novel framework that combines filter pruning with Kernel Recovery (KR) to enhance model compression and performance preservation. The core of this framework is our Localized Gradient Activation heatmap (LGAP) algorithm [16], which takes into account the spatial relationships between filters

and the objective prediction during the analysis, with an integration of KR, where kernel weight reconstruction is performed to ensure that the feature map results at subsequent layers closely resembling those of the original unpruned model. In our KR technique, we utilize regression with cross correlation (cosine similarity) for global loss to train new kernel weights, allowing the process to be applied to multi-channel and multi-filter convolutional layers. The advantage of KR lies in its computational efficiency compared to the time-consuming fine-tuning method, allowing for a more rapid deployment of model compression while maintaining performance. It can be seen that LGAP leverages a data feed-forward technique to explore the spatial relationships among filters within each layer, going beyond traditional single filter analysis. By identifying weak filters within localized regions through a localized prediction map, we can isolate crucial areas for the target prediction. This approach contrasts with other filter pruning techniques that rely heavily on statistical criteria, such as weighted sum or output feature maps such as APoZ and SSR. Such statistical methods often overlook important spatial data, leading to incomplete relational layer analysis and a limited understanding of the model's predictions.

3. Methods

In this section, we present our new framework for filter pruning in two sections: an overview of the framework in Section 3.1, the convolutional approximation of a small model (CASM) in Section 3.2.

3.1. Overview of Our Framework

The fundamental framework of filter pruning consists of four main processes: filter selection, filter pruning, CASM, and performance recovery. The overview of our framework is illustrated in Figure 1.

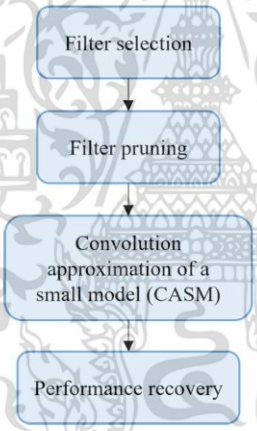


Figure 1. Overview of our framework.

In filter selection, filters are assessed and scored based on their significance to layer performance. We adopt our Localized Gradient Activation heatmap (LGAP) [16], which provides the spatial relationships between the feature map activation of the explored filter and the target prediction. The filter score is then estimated in term of the layer-wise loss persistence, which is the gradient difference between a full layer and a layer that does not have an inspected filter.

The first step involves the calculation of the gradient. The estimation of this gradient entails examining the predicted outcome of the top-ranked class in relation to the

output feature map of the activation layer. These output feature maps are generated by inputting the training dataset into the network and extracting them before the softmax layer. The representation of this relationship can be observed in Equation (1) [26].

$$G^{l,k} = \frac{\partial y^c}{\partial A^{l,k}} \tag{1}$$

$G^{l,k}$ is the gradient of kth filter in the layer l, y^c is the prediction result of top-class c. $A^{l,k}$ is the feature map from the activation layer of the kth filter in layer l.

The alpha weight can be calculated after estimating the gradient by performing a global-average-pooling of the gradient, as described in Equation (2) [26].

$$\alpha_{l,k}^c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} G_{i,j}^{l,k} \tag{2}$$

$\alpha_{l,k}^c$ is the alpha weight of the kth filter in layer l, $H \times W$ is the Height and Width of the output gradient class c.

Finally, the layer-wise loss persistence can be calculated, as presented in Equation (3) [16].

$$H_{l,f} = norm(|(\sum_k \alpha_{l,k}^c A^{l,k}) - \alpha_{l,f}^c A^{l,f}|) \tag{3}$$

where H is the loss persistence heatmap, f is the filter that is analyzed in layer l, kth is the filter within layer l, c is the top predicted class and α is the significant weight that is obtained by neuron in the form of global-average-pooling of the gradient.

The persistence score quantifies the impact of a filter's loss and aids in distinguishing between significant and insignificant filters. The higher the persistence score, the less the consequence of losing the examined filter from the layer. Weak filters with minimal loss impact are considered as unnecessary kernels and their weight parameters are removed from the layer. Depending on the layer's sensitivity, the model performance may decrease after pruning weak filters. To preserve model performance, most researchers engaged in a process of fine-tuning. A range of fine-tuning strategies, including layer-by-layer and all-layer fine-tuning, were utilized. Nevertheless, not all parameters used for fine-tuning and optimization were disclosed. It would be quite challenging to estimate how researchers configured their algorithms for iterative pruning with fine-tuned parameters. In order to effectively restore the performance of the pruned model relative to the original unpruned model, the process of fine-tuning takes not only an excessive amount of computing effort but also a substantial amount of retraining data.

To alleviate the complexity of fine-tuning aimed at preserving model performance, we introduced a Kernel Recovery (KR) strategy. The purpose is to restore the kernel coefficients of the pruned model, aiming to recover the subsequent layer's output activation as closely as possible to that of the original unpruned kernel. Our KR strategy, named Convolutional Approximation of a Small Model (CASM), was utilized to approximate the kernel weights of the pruned model from a partial convolution kernel.

Following KR, the model structure was nearly reinstated, allowing further refinement with just a single round of either partial or full fine-tuning for overall adjustment. With KR, we are able to reduce the complexity of restoring model performance, as the approximation of a small model requires significantly less computational time and involves fewer retraining data.

3.2. Convolutional Approximation of a Small Model (CASM)

CASM is a method for restoring kernel weights following the pruning process, as illustrated in Figure 2. Filters are assessed in process 1 (Filter Selection), and the results of weak filters are marked in purple highlighted blocks, where the dimension of the filter kernel is given as H, W: the height and width of the feature map, C: the number of filter

channels, and F: the number of filters. In process 2 (Filter Pruning), highlighted weak filters in process 1 are removed from the CNN model while process 3 (CASM) performs convolution approximation of the kernels at layer $l+1$, which are affected by the reduction of output activations at layer l. As a result of the removal of the filter, only fragments of output activations remain, and partial convolutions are conducted at layer $l+1$. Due to the loss of activation output from layer l as the input to layer $l+1$, the remaining channel weights are no longer suitable to maintain layer performance. To restore the layer's performance, the surviving channel weights must be relearned. We create a small model of a kernel based on the remaining structure of layer $l+1$, and we copy the remaining weights onto this small model. This small model is then trained using fundamental optimization in the spatial domain, with an objective function defined in Equation (4) [25].

$$J = \frac{1}{2}||h * x - y||^2 \tag{4}$$

where J represents the objective function, h is the filter kernel of a small model, $*$ is convolution, x is the feature map of layer l from process 2 (Filter Pruning), and y is the output feature map of layer $l+1$ from process 1 (from original unpruned layer).

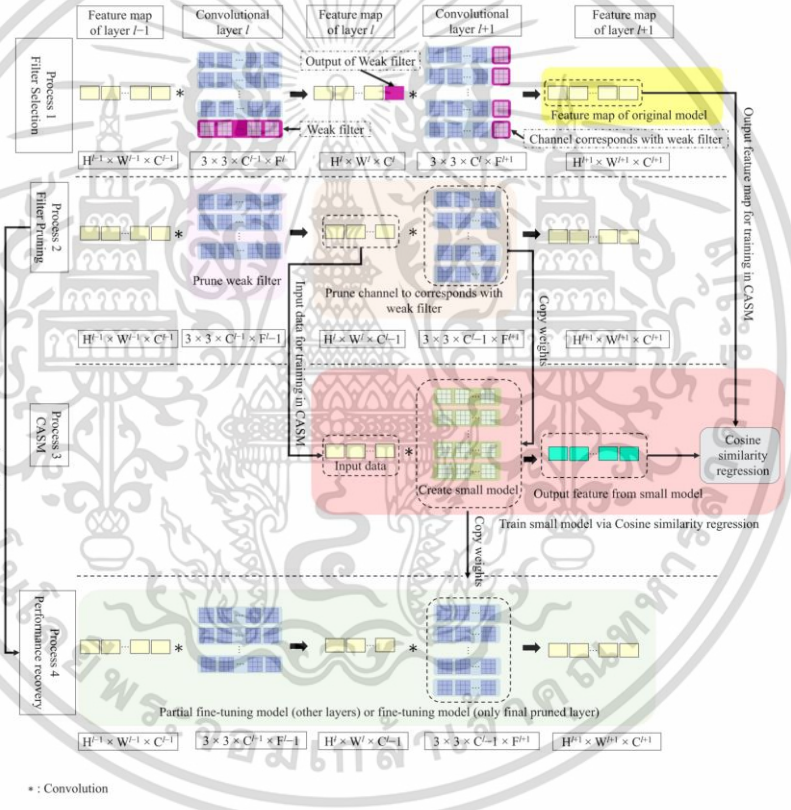


Figure 2. Procedure of our CASM framework.

During the training process, kernel weight optimization is performed, and a cross-correlation (cosine similarity) metric (Equation (5)) is used as a performance assessment between the feature maps resulting from a small model and the original unpruned model. The optimization parameters are listed in Table 1.

Table 1. A summarized table of optimization parameters for training CASM.

Parameter	Value
Optimizer	SGD
Learning rate	0.001
Momentum	0.9
Weight decay	0.0005
Epoch for training	500

The optimum weights of the small model are then reassigned to the pruned model at convolutional layer $l+1$ in process 4 (partial fine-tuning). In process 4, partial fine-tuning is executed from the initial layer up to convolutional layer $l+1$ (as depicted by the green highlighted block in Figure 2), encompassing all segments of the fully connected layer. Subsequent to the elimination of the final pruning layer, the model's performance is reestablished through full fine-tuning. During this phase, all layers within the pruned model are unfrozen.

$$S = \frac{A \cdot B}{||A|| ||B||} \tag{5}$$

where S represents cross-correlation (cosine similarity), A is a feature map from small model prediction and B is feature map $l+1$ from the original model.

4. Experiments and Discussion

This session evaluates the performance of the CASM framework using a CNN model that has been pre-trained via Keras library [27]. In Section 4.1, we explain the details of the pre-trained model and data to be implemented for the CASM framework evaluation experiment. In Section 4.2, we compare the results of CASM and its pruning model to those of other techniques utilizing basic or other frameworks.

4.1. A Detail of Experimental Implementation

The performance of the CASM framework was evaluated using two datasets and two pre-trained CNN models.

The utilized experimental datasets are the following:

1. CIFAR-10 [28] is a small dataset, encompassing 32 × 32 pixel color images. This dataset contains 60,000 images with 10 different classes, with 50,000 allocated for training and 10,000 for testing.
2. ImageNet (ILSVRC 2012) [29] is a large-scale dataset, consisting of 1.2 million images for the training set, with 1000 different classes, and 50,000 images for the validation set. The images are resized to 256 × 256 pixels, cropped to 224 × 224 pixels based on their centers, and horizontally flipped during the data augmentation process.

The utilized pre-trained CNN models are the following:

1. VGG-16 [30] is a state-of-the-art CNN model designed for image classification. The basic structure consists of 13 convolutional layers and was designed to be used with large-scale image datasets. In each layer, the size of the kernel is 3 × 3. The performance of VGG-16 for image classification is outstanding. It was trained on datasets from ImageNet to CIFAR-10 [31] by reducing the number of object classes in the fully connected layer from 1000 to 10. In addition, a batch normalization layer [32] and a dropout layer [33] were added to each convolutional layer. All convolutional layers of VGG-16 on CIFAR-10 have parameters of 1.5×10^7 and FLOPs (Floating Point Operation) of

6.3×10^8. In this article, we equated 2× Multiply-Accumulate Computations (MACs) with FLOPs.

2. ResNet-50 [12] is a state-of-the-art CNN model that is produced with a baseline of VGG-16 and 16 unique structures called residue blocks. This model can only have the first two convolutional layers of each block pruned, as the final convolutional layer cannot be pruned due to its unique structure. If it is pruned, the output feature map of the final convolutional layer of the block will not be compatible with the output feature map from the shortcut connection due to their difference in dimension. ResNet-50 has 2.3×10^7 parameters and 7.7×10^9 FLOPs.

For VGG-16, pruning begins at the second convolutional layer. For partial-fine tuning, the optimizer is set to SGD, one epoch, 0.001 learning rate, 0.9 momentum, and 0.0005 weight decay. For the final layer, a 10^{-3} to 10^{-5} learning rate and forty epochs are implemented. The batch size for this operation is set to 32. We prune model VGG-16 at a ratio of 50%, pruning with a comparison of 32 and 128 batch sizes. The accuracy results are 92.07% and 92.1%, respectively, and are not statistically distinguishable. In the CASM process, the CIFAR-10 dataset is utilized with 5000 images for training and 1000 images for testing. For the filter selection procedure, 10 images per class were used. Since VGG-16 on CIFAR-10 requires less processing time to prune the model, we can experiment in detail using a pruning ratio of 10–90%. The pruning ratio is the same for all convolutional layers.

For ResNet-50, pruning begins at the first convolutional layer of the first residual block. For partial-fine tuning, the optimizer is set to SGD, one epoch, 0.0001 learning rate, 0.9 momentum, and 0.0005 weight decay. For the CASM process, the ImageNet dataset is utilized with 1600 for training and 400 for testing. The remaining parameters are the same as those used for VGG-16. Pruning ratios of 30%, 50%, and 70% are conducted for ResNet-50 due to the large processing time of pruning on ImageNet.

To evaluate the performance of CASM and the basic fine-tuning framework, we conducted experiments using the VGG-16 model on the CIFAR-10 dataset, with a pruning ratio of 50%. We randomly selected three sets of 200, 500, and 1000 images from the dataset for training the model. The fundamental training parameters, including the SGD optimizer, learning rate of 0.001, momentum of 0.9, and weight decay of 0.0005, were kept consistent for both frameworks.

Since the development time for the CASM framework is significantly shorter compared to basic fine-tuning, we carried out experiments to assess layer recovery performance and training time for each framework.

4.2. Pruning Model Results

In this section, we demonstrate the results of CASM and the CASM framework's pruning of the VGG-16 and ResNet-50 models.

4.2.1. Kernel Recovery Results of CASM

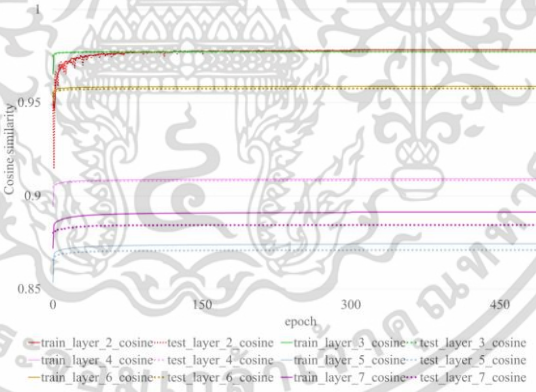
Cosine similarity is used to evaluate kernel recovery performance. The measurement reflects how close the results of the restored kernel are to those of the original kernel prior to pruning. Performance of the CASM on VGG-16 with CIFAR-10 is illustrated in Figure 3. Each color represents the outcome of each layer. The dotted line represents the training outcome, while the solid line represents the testing outcome of the CASM procedure. See also Figure 3a, which illustrates the outcome of layers 2–7 and Figure 3b, which illustrates the outcome of layers 8–12. Even though the CASM training results are nearly constant at epoch 300, as shown in Table 2, its performance has been satisfied since epoch 10. The final layer recovery is not performed due to absence of output activation at the subsequent layer of the original unpruned model used to train the small model. To achieve satisfactory performance, the CASM framework requires at least ten epochs, whereas the basic fine-tuning framework requires only one. However, as shown in Table 2, even with ten epochs of CASM training, the required time is significantly less than that of simple fine-tuning. With limited sample sizes (200, 500, 1000), CASM is able to regain performance that is

superior to that of fine-tuning. Using the RTX 3090 (Nvidia Corporation, Santa Clara, CA, USA), we determined the average process time of all layers (see Table 2). CASM demonstrates superior processing speed compared to fine-tuning for different sample sizes. Specifically, for 200, 500, and 1000 samples, CASM processes 3.3 times, 2.4 times, and 1.7 times faster, respectively. The basic fine-tuning framework, on the other hand, requires a larger number of samples to achieve performance recovery. Even with 1000 samples, the performance of the basic framework did not match that of CASM, which utilized only 200 samples. CASM achieved superior performance, despite using less than five times the number of samples, and a performance gain greater than the basic fine-tuning framework, with a margin of 0.1498 (0.8050–0.6552). The superiority of CASM over basic fine-tuning becomes evident when considering the comparative aspects of sample size and processing time demanded by each approach.

Table 2. A performance comparison of layer-by-layer pruning using CASM and basic frameworks based on the pruned VGG-16 model with CIFAR-10 for 200, 500, and 1000 training samples.

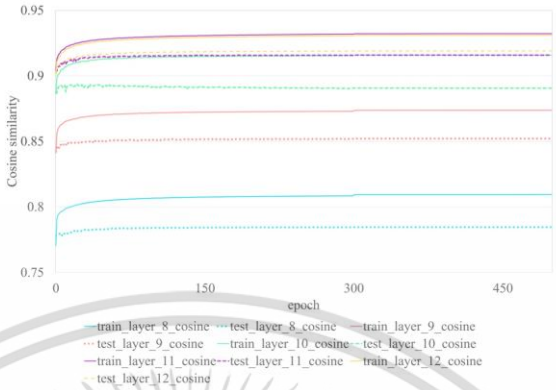
Sample	State/Process	Layer 3	Layer 5	Layer 7	Layer 9	Layer 11	Layer 13
200	Before/CASM	0.8439	0.8431	0.7781	0.7755	0.8005	0.8050
	After/CASM	0.9220	0.8793	0.8076	0.8054	0.8041	0.8050 (1.25 s)
	Before/fine-tuning	0.4852	0.4637	0.3357	0.2522	0.4046	0.2251
	After/fine-tuning	0.5623	0.3410	0.2432	0.3346	0.3808	0.1961 (4.13 s)
500	Before/CASM	0.8418	0.8487	0.8016	0.8007	0.8238	0.8291
	After/CASM	0.9232	0.8859	0.8296	0.8270	0.8293	0.8291 (1.85 s)
	Before/fine-tuning	0.4496	0.5213	0.4081	0.3972	0.5648	0.4478
	After/fine-tuning	0.5922	0.5131	0.4926	0.4830	0.5865	0.5717 (4.35)
1000	Before/CASM	0.8498	0.8397	0.7774	0.8091	0.8350	0.8365
	After/CASM	0.9243	0.8902	0.8364	0.8389	0.8355	0.8365 (2.59 s)
	Before/fine-tuning	0.6569	0.6162	0.5068	0.5349	0.6206	0.6469
	After/fine-tuning	0.7498	0.6075	0.5602	0.6471	0.6651	0.6552 (4.43 s)

s represents seconds.



(**a**)

Figure 3. *Cont.*



(**b**)

Figure 3. Performance (Cosine Similarity) of the CASM on VGG-16 with CIFAR-10 at 50% pruning ratio and the original model, where a dotted line represents the training result and a solid line represents the testing result, (**a**) the outcome of layers 2–7, (**b**) the outcome of layers 8–12.

4.2.2. Pruning Performance of VGG-16 on CIFAR-10

Table 3 illustrates the results of LGAP's pruning with a combination of either the basic fine-tuning framework or the CASM framework. The results indicate that LGAP with CASM produces fewer errors than LGAP with the basic framework for pruning ratios less than or equal to 50%. With the range of 10% to 50% pruning ratio, the different errors between the basic and CASM frameworks are 0.13%, 0.17%, 0.24%, 0.03%, and 0.14%, respectively. At a 20% pruning ratio, the result of CASM pruning is roughly equivalent to the original model. This indicates that the CASM framework could restore excellent performance with a pruning ratio of 50% or less. If the pruning ratio is greater than 50%, the recovery performance of the CASM framework tends to degrade. Therefore, we would suggest using the CASM framework for a pruning ratio of 50% or less. Pruning VGG-16 could reduce FLOPs, number of parameters, and storage by 3.6×, 3.9×, and 3.9×, respectively, at a pruning ratio of 50%. Figure 4 illustrates a comparison of the performance of our LGAP with either the CASM or the basic framework and other pruning techniques: Random, Weighted sum, APoZ, and Mean gradient. It can be seen that at a pruning ratio less than or equal to 50%, the performances of all techniques are comparable.

Table 3. Pruning performance comparison of LGAP with basic framework and CASM framework based on VGG-16.

Technique, Framework	Pruning Ratio	Error (%)	FLOPs	FLOPs Reductions (×)	Parameters	Parameter Reductions (×)	Storage Size (MB)	Storage Reductions (×)
-	0%	6.41	6.3×10^8	0	1.5×10^7	0	57.4	0.0
LGAP, Basic	10%	6.32	5.1×10^8	1.2	1.2×10^7	1.2	46.2	1.2
LGAP, CASM		6.19						
LGAP, Basic	20%	6.64	4.1×10^8	1.5	9.6×10^6	1.6	36.6	1.6
LGAP, CASM		6.47						

Table 3. *Cont.*

Technique, Framework	Pruning Ratio	Error (%)	FLOPs	FLOPs Reductions (×)	Parameters	Parameter Reductions (×)	Storage Size (MB)	Storage Reductions (×)
LGAP, Basic	30%	7.12	3.2×10^8	2.0	7.4×10^6	2.0	28.2	2.0
LGAP, CASM		6.88						
LGAP, Basic	40%	7.28	2.4×10^8	2.6	5.4×10^6	2.8	20.9	2.8
LGAP, CASM		7.25						
LGAP, Basic	50%	8.07	1.7×10^8	3.6	3.8×10^6	3.9	14.6	3.9
LGAP, CASM		7.93						
LGAP, Basic	60%	8.53	1.2×10^8	5.4	2.4×10^6	6.2	9.4	6.1
LGAP, CASM		8.97						
LGAP, Basic	70%	10.34	7.2×10^7	8.7	1.4×10^6	10.8	5.5	10.5
LGAP, CASM		11.41						
LGAP, Basic	80%	13.88	3.7×10^7	16.9	6.4×10^5	23.5	2.6	22.2
LGAP, CASM		15.82						
LGAP, Basic	90%	23.48	1.4×10^7	44.1	1.8×10^5	84.4	0.8	69.5
LGAP, CASM		25.97						

× is times. MB is Megabyte.

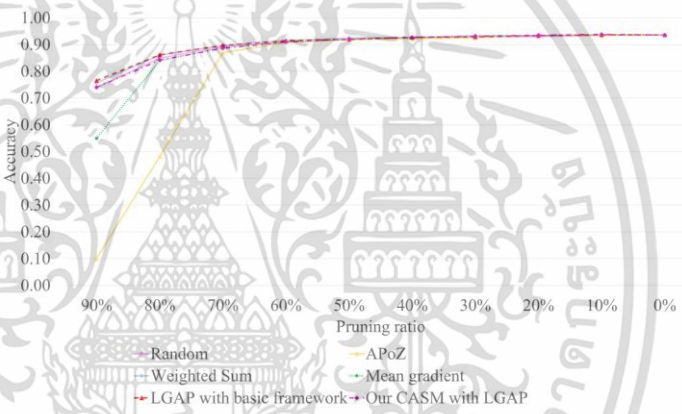


Figure 4. Performance comparison of a pruned VGG-16 with CIFAR-10 using CASM framework and other techniques using basic fine-tuning framework.

4.2.3. Pruning Performance of ResNet-50 on ImageNet

Table 4 illustrates the performance of pruning Resnet-50 using the LGAP technique with a combination of either the CASM or the basic fine-tuning framework. The experimental results show that LGAP with the CASM framework can provide superior recovery performance by 0.75% (30% pruning ratio) and 0.13% (50% pruning ratio) fewer errors for the top-1 and 0.13% (30% pruning ratio) and 0.08% (50% pruning ratio) fewer errors for the top-5. However, when the pruning ratio exceeds 70%, the basic framework is superior to the CASM framework. We believe that after a substantial amount of pruning, there is insufficient data for recovery since only a small number of filters remained. Thus, the basic fine-tuning framework with a massive number of retraining samples and several retraining cycles continues to be essential.

Table 4. Pruning performance comparison of LGAP with basic framework and CASM framework based on ResNet-50.

Framework	Pruning Ratio	Top-1 Error (%)	Top-5 Error (%)	FLOPs	Parameters	Storage Size (MB)
-	0%	25.10	7.90	7.7×10^9	2.6×10^7	98.2
Basic	30%	27.55	8.94	4.8×10^9	1.7×10^7	65.2
CASM		26.80	8.67			
Basic	50%	28.66	9.79	3.4×10^9	1.2×10^7	47.8
CASM		28.53	9.71			
Basic	70%	31.44	11.26	2.2×10^9	8.7×10^6	33.6
CASM		31.47	11.45			

Top-1 represents the highest probability (top-ranked) prediction that corresponds with the ground truth. Top-5 represents the five highest probability predictions that correspond with the ground truth. MB represents Megabytes.

In Table 5, we compare the performance of LGAP with CASM to those of other techniques. At a 50% pruning ratio, the Top-1 accuracy of LGAP with CASM is superior to Random, SSL, ThiNet, APoZ, weighted sum, SSR-L2, Gradient mean, and LGAP with a basic framework by 1.22%, 1.15%, 0.69%,0.82%, 0.88%, 0.22%, 0.47%, and 0.13%, respectively. For top-5 accuracy, LGAP with CASM outperforms Random, SSL, ThiNet, APoZ, weighted sum, SSR-L2, Gradient mean, and LGAP with the basic framework by 0.94%, 0.87%, 0.47%, 0.60%, 0.61%, 0.30%, 0.13%, and 0.08%, respectively. ResNet-50 with 50% pruning can help reduce FLOPs, number of parameters, and storage by 2.3×, 2.1×, and 2.1×, respectively. The random technique for filter selection in model pruning is considered unstable, given its potential to yield either the best or the worst outcomes. It possesses the capacity to prune both robust and weak filters from the model. Although SSL can perform model pruning, it frequently alters the original model structure. The weighted sum approach relies on statistical weights to evaluate filters, constraining its ability to comprehend the spatial correlations and behavior of filters within a layer. Data-driven methodologies like ThiNet, APoZ, SSR-L2, Gradient Mean, and LGAP have demonstrated superior performance when compared to using only weight analysis. APoZ assesses filters based on the average percentage of zero activations within the activation layer. However, it fails to distinguish between two filters sharing the same zero activation area, potentially resulting in suboptimal pruning choices. Techniques such as ThiNet, SSR-L2, and Gradient Mean analyze filters using output feature maps and gradients, leading to a deeper grasp of filter significance in contrast to APoZ. Nevertheless, these methods still lack spatial relationship corresponding to target prediction.

In contrast, LGAP can analyze the spatial relationships among filters in each layer in a data-driven manner. Unlike methods that focus on a single weak filter, LGAP emphasizes distinct weak filters, allowing for the identification of crucial areas for target prediction. Consequently, LGAP demonstrates superior performance compared to alternative techniques [16].

Figure 5 illustrates the visualization of the feature maps resulting from the pruned ResNet-50 using LGAP with either the CASM framework or the basic fine-tuning framework and the original model at different pruning ratios (30%, 50%, and 70%). When gradient localization was applied to LGAP and the basic fine-tuning framework with a 30% pruning ratio, our past research showed that the gradient localization of the model derived from both frameworks was comparable to that of the original model. However, during these experiments, we have observed that some sample images resulted in more gradient localization deviations with improved performance, such as the input image of class n01631663 in Figure 5 (upper). Additionally, we discovered that even when using data from the same class but sourced from outside ImageNet, the results still followed the same trend, as seen in Figure 5 (lower). These results indicate that such images can

be effectively utilized for our purposes. The F1-score of the original model is 0.771 while the F1-scores of 30%, 50%, and 70% pruning ratios for LGAP with a CASM framework are 0.776, 0.796, and 0.817, respectively and for LGAP with a basic framework, they are 0.766, 0.804, and 0.787, respectively. When the compression ratio increased, the gradient localization varied more from the original model, as seen in Figure 5. Even at a pruning ratio of 70%, the F1-score of LGAP with CASM is proven to be superior to the original model and other pruned models. Even though pruning performance degrades at a high pruning ratio, some classes may outperform models with a lower pruning ratio or the original model. In addition, we analyzed the results from pruning the ResNet-50 model obtained from CASM and the basic framework in Table 6. We fed two images per class (2000 images) from the validation set into the network of the original model, pruned the model by CASM and pruned the model by basic framework to obtain the output feature map from the final activation layer of each model. Cosine similarity is used to compare the similarity of the output feature map from the original model and pruned models (pruning by CASM and basic framework). We found that the cosine similarity between the original model and the pruned model by CASM at 30% and 50% pruning ratios was higher than the cosine similarity of the original model and the pruned model by the basic framework. This shows that pruning the model by the basic framework has a greater impact on changes from the original in the output feature map than CASM. When using the basic framework, all the layers are fine-tuned, which causes significant shifts in the model weights. Moreover, the time complexity of the basic framework is relatively high due to the fact that some of the layers are sensitive to being retrained and require additional fine-tuning (more than one epoch). In contrast, CASM attempts to restore performance by training the layer adjacent to the pruned layer and fine-tuning just certain parts of the model.

Table 5. Performance comparison of pruned ResNet-50 using LGAP with either CASM or basic framework with other pruning techniques at 50% pruning ratio.

Technique	Top-1 Accuracy Decrease (%)	Top-5 Accuracy Decrease (%)	FLOPs	Parameters	Pruned
Random	4.65%	2.75%	3.4×10^9	1.2×10^7	50%
SSL [19]	4.58%	2.68%	4.2×10^9	1.3×10^7	≈50%
ThiNet [23]	4.12%	2.28%	3.4×10^9	1.2×10^7	50%
APoZ [21]	4.25%	2.41%	3.4×10^9	1.2×10^7	50%
Weighted sum [20]	4.31%	2.42%	3.4×10^9	1.2×10^7	50%
SSR-L2 [24]	3.65%	2.11%	3.4×10^9	1.2×10^7	50%
Gradient Mean [22]	3.90%	1.94%	3.4×10^9	1.2×10^7	50%
LGAP with basic framework [16]	3.56%	1.89%	3.4×10^9	1.2×10^7	50%
LGAP with CASM framework	3.43%	1.81%	3.4×10^9	1.2×10^7	50%

Top-1 represents the highest probability (top-ranked) prediction that corresponds with the ground truth. Top-5 represents the five highest probability predictions that correspond with the ground truth.

Table 6. Cosine similarity of the output feature map at the final activation layer after pruning all layers was compared between the original model and CASM and basic framework on ResNet-50.

Pruning Ratio (%)	CASM Framework	BASIC Framework
30	0.6970	0.6752
50	0.6105	0.5996
70	0.5076	0.5087

The introduction of the CASM has led to significant improvements in both performance and efficiency compared to the original model. By minimizing the deviation of the

output feature map, CASM achieves greater similarity and improved accuracy. The CASM framework offers not only improved performance but also a significant reduction in time complexity, achieving a speed increase of 3.3 times compared to the basic framework. Moreover, CASM exhibits a capability to achieve higher model accuracy with a smaller dataset as compared to the basic framework (Table 2). With the integration of CASM into the pruning technique, our LGAP filter pruning becomes more effective, while ensuring that computational resources such as FLOPs and storage space remain consistent with the basic framework.

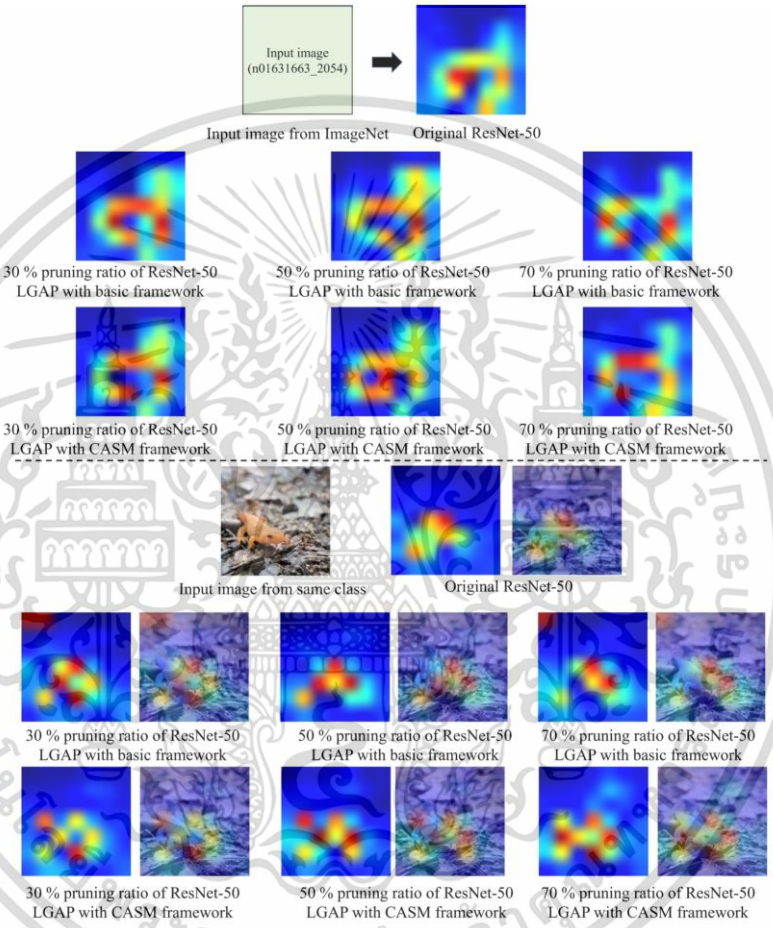


Figure 5. Visualization of feature map results from our pruned models with 30%, 50%, and 70% pruning ratio. The input image (n01631663_2054) is an image from the ImageNet [29] and the input image from the same class is from [34].

However, it is essential to note that CASM has a limitation when the pruned model exceeds a 50% threshold. In such cases, the recovery performance tends to deteriorate due to an insufficient number of channels in the remaining weights of the convolutional layer. This makes it difficult to accurately reconstruct the output feature map, as illustrated in Table 6. Despite this limitation, CASM's overall contributions to performance enhancement, efficiency, and resource preservation make it a valuable addition to the existing framework.

5. Conclusions

In this research, we propose a new filter pruning framework for CNN models. Our new framework, known as the CASM framework, is a kernel recovery process, which creates a small model of remaining kernels. The kernel weights of the small model are fine-tuned to restore the feature map of the remaining kernel as closely as possible to that of the original unpruned kernel. With the same amount of training samples, CASM yields better results than the basic fine-tuning framework, while demanding less computational effort. The results clearly indicate that the CASM framework can restore the performance of a pruned model more effectively than the basic framework, especially at a pruning ratio of 50% or lower. This holds true for both VGG-16 with CIFAR-10 and ResNet-50 with ImageNet. Hence, CASM assists in recovering the remaining kernels with reduced complexity, fewer training samples, and less storage usage compared to the basic fine-tuning framework. The experimental results demonstrate CASM's superiority over the basic fine-tuning framework, showing faster time acceleration (3.3×), requiring a smaller dataset volume for performance recovery post-pruning, and enhancing accuracy. For future research, we plan to explore a new framework for minimizing the time required to restore a pruned model's performance by reducing both partial and full fine-tuning processes.

Author Contributions: Conceptualization, M.I. and O.C.; methodology, M.I.; software, M.I.; validation, M.I. and O.C.; formal analysis, M.I. and O.C.; investigation, O.C.; resources, M.I. and O.C.; data curation, M.I. and O.C.; writing—original draft preparation, M.I.; writing—review and editing, O.C.; visualization, M.I. and O.C.; supervision, O.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data proposed in this research can be accessed in [28,29].

Acknowledgments: This research was advocated by King Mongkut's Institute of Technology Ladkrabang, Thailand.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

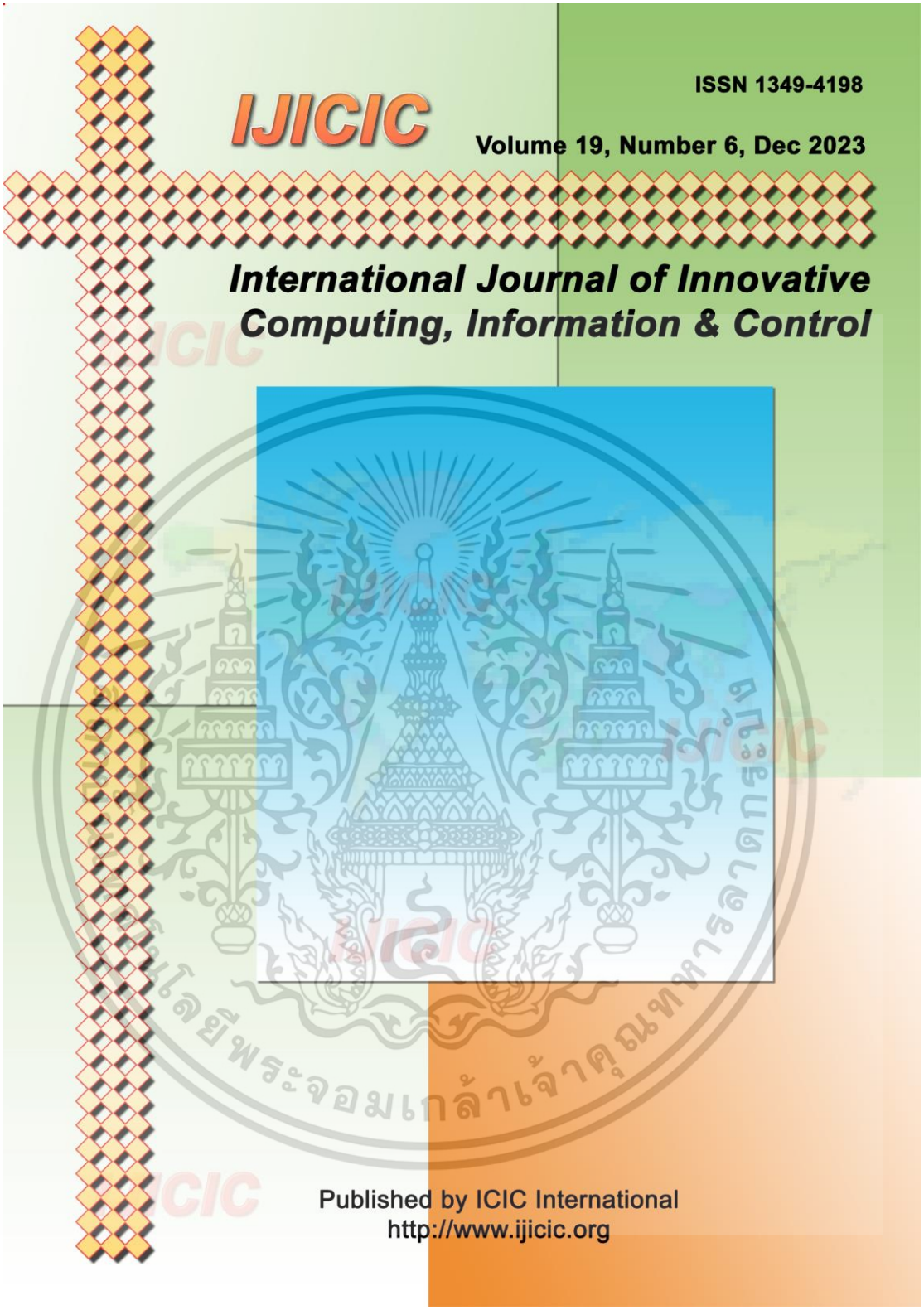
CNNs	Convolutional neural networks
CASM	Convolutional Approximation Small Model
KR	Kernel Recovery
LGAP	Localized Gradient Activation heatmaP
SSL	Structured Sparsity Learning
APoZ	Average Percentage of Zeros
SSR	Structured sparsity regularization

References

1. Mohammed, H.R.; Hussain, Z.M. Hybrid Mamdani Fuzzy Rules and Convolutional Neural Networks for Analysis and Identification of Animal Images. *Computation* **2021**, *9*, 35. [CrossRef]
2. Varma, G.T.; Krishna, A.S. Transfer Learning-based Optimal Feature Selection with DLCNN for Shrimp Recognition and Classification. *Int. J. Intell. Eng. Syst.* **2022**, *15*, 91–102.

3. Lee, K.; Kim, S.; Lee, E.C. Fast and Accurate Facial Expression Image Classification and Regression Method Based on Knowledge Distillation. *Appl. Sci.* **2023**, *13*, 6409. [CrossRef]
4. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807.
5. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated Residual Transformations for Deep Neural Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5987–5995.
6. Aung, H.M.L.; Pluempitiwiriyawej, C.; Hamamoto, K.; Wangsiripitak, S. Multimodal Biometrics Recognition Using a Deep Convolutional Neural Network with Transfer Learning in Surveillance Videos. *Computation* **2022**, *10*, 127. [CrossRef]
7. Yuan, J.; Xiong, H.C.; Xiao, Y.; Guan, W.; Wang, M.; Hong, R.; Li, Z.Y. Gated CNN: Integrating multi-scale feature layers for object detection. *Pattern Recognit.* **2020**, *105*, 107131. [CrossRef]
8. Maltezos, E.; Douklias, A.; Dadoukis, A.; Misichroni, F.; Karagiannidis, L.; Antonopoulos, M.; Voulgary, K.; Ouzounoglou, E.; Amditis, A. The INUS Platform: A Modular Solution for Object Detection and Tracking from UAVs and Terrestrial Surveillance Assets. *Computation* **2021**, *9*, 12. [CrossRef]
9. Guvenoglu, E. Determination of the Live Weight of Farm Animals with Deep Learning and Semantic Segmentation Techniques. *Appl. Sci.* **2023**, *13*, 6944. [CrossRef]
10. Vadukkal, U.K.V.; Palumbo, M.; Attolico, G. Semantic Segmentation of Packaged and Unpackaged Fresh-Cut Apples Using Deep Learning. *Appl. Sci.* **2023**, *13*, 6969. [CrossRef]
11. De Silva, K.D.M.; Lee, H.J. Distorted Aerial Images Semantic Segmentation Method for Software-Based Analog Image Receivers Using Deep Combined Learning. *Appl. Sci.* **2023**, *13*, 6816. [CrossRef]
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
13. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2818–2826.
14. Pilipović, R.; Bulić, P.; Risojević, V. Compression of convolutional neural networks: A short survey. In Proceedings of the 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), Sarajevo, Bosnia and Herzegovina, 21–23 March 2018; pp. 1–6.
15. Ghimire, D.; Kil, D.; Kim, S.H. A Survey on Efficient Convolutional Neural Networks and Hardware Acceleration. *Electronics* **2022**, *11*, 945. [CrossRef]
16. Intraraprasit, M.; Chitsobhuk, O. Filter Pruning Based on Local Gradient Activation Mapping in Convolutional Neural Networks. *Int. J. Innov. Comput. Inf. Control* **2023**, *19*, *in press*.
17. Le Cun, Y.; Denker, J.S.; Solla, S.A. Optimal Brain Damage. In Proceedings of the 2nd International Conference on Neural Information Processing Systems (NIPS'89), Denver, CO, USA, 27–30 November 1989; MIT Press: Cambridge, MA, USA, 1989; pp. 598–605.
18. Lebedev, V.; Lempitsky, V. Fast ConvNets Using Group-Wise Brain Damage. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2554–2564.
19. Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; Li, H. Learning Structured Sparsity in Deep Neural Networks. In Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16), Barcelona, Spain, 5–10 December 2016; Curran Associates Inc.: Red Hook, NY, USA, 2016; pp. 2082–2090.
20. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, H.P. Pruning Filters for Efficient ConvNets. *arXiv* **2016**, arXiv:1608.08710.
21. Hu, H.; Peng, R.; Tai, Y.; Tang, C. Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures. *arXiv* **2016**, arXiv:1607.03250.
22. Liu, C.; Wu, H. Channel pruning based on mean gradient for accelerating Convolutional Neural Networks. *Signal Process.* **2019**, *156*, 84–91.
23. Luo, J.H.; Wu, J.; Lin, W. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 5068–5076.
24. Lin, S.; Ji, R.; Li, Y.; Deng, C.; Li, X. Toward Compact ConvNets via Structure-Sparsity Regularized Filter Pruning. *IEEE Trans. Neural. Netw. Learn. Syst.* **2020**, *31*, 574–588. [CrossRef] [PubMed]
25. Alqahtani, A.; Xie, X.; Jones, M.W.; Essa, E. Pruning CNN filters via quantifying the importance of deep visual representations. *Comput. Vis. Image Underst.* **2021**, *208–209*, 103220. [CrossRef]
26. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 618–626. [CrossRef]
27. Chollet, F.; et al. Keras. Available online: https://keras.io (accessed on 31 May 2023).
28. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; Technical Report 0; University of Toronto: Toronto, ON, USA, 2009.
29. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]

30. Simonyan, K.; Zisserman, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition*; Computational and Biological Learning Society: Leesburg, VA, USA, 2015; pp. 1–14.
31. Liu, S.; Deng, W. Very deep convolutional neural network based image classification using small training sample size. In Proceedings of the 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), Kuala Lumpur, Malaysia, 3–6 November 2015; pp. 730–734.
32. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of Machine Learning Research, Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015*; Bach, F., Blei, D., Eds.; PMLR: Lille, France, 2015; Volume 37, pp. 448–456.
33. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
34. Enders, L. Red Eft in Leaf Litter. 2021. Available online: https://www.flickr.com/photos/usfwsnortheast/51277223193/ (accessed on 3 August 2023).



International Journal of Innovative Computing, Information and Control
Volume **19**, Number **6**, December **2023**

ICIC International ©2023 ISSN 1349-4198
pp. **1697–1715**

FILTER PRUNING BASED ON LOCAL GRADIENT ACTIVATION MAPPING IN CONVOLUTIONAL NEURAL NETWORKS

Monthon Intraraprasit and Orachat Chitsobhuk*

School of Engineering
King Mongkut's Institute of Technology Ladkrabang
Chalongkrung Road, Ladkrabang, Bangkok 10520, Thailand
61601175@kmitl.ac.th; *Corresponding author: orachat.ch@kmitl.ac.th

Received January 2023; revised April 2023

Abstract. *Convolutional Neural Network (CNN) is a well-known Deep learning model utilized extensively in the field of computer vision. The structure of convolutional neural networks is quite complicated and necessitates a substantial amount of computational time and storage resources. As a result, it is difficult to adopt a CNN model on a resource-constraint device. Model pruning can help to reduce computation time and storage requirements. In this research, we propose a filter pruning technique based on Localized Gradient Activation heatmaP (LGAP) for the purpose of pruning CNNs. Analyzing a filter based on statistical criterion of single neuron can lead to a loss in spatial relations within the filter activation itself, the relationship to target prediction, as well as the relationship among filters in that specific layer. To minimize the limitations, we evaluate the significance of a filter through the spatial information of local gradient activation related to the target prediction in terms of the layer-wise loss of the investigated filter. The effect of loss of an investigated filter demonstrates the significance or insignificance of the filter. Our pruning criteria ensure that these significant filters are preserved, while maintaining the model accuracy. The performance of our pruning method was validated using VGG-16 and ResNet-50. With pruning ratio of 50%, VGG-16 tends to decrease 1.66% of its accuracy, 3.6× of FLOP and 3.9× of storage reduction. For ResNet-50, with 50% pruning ratio, the results show that Top-1 and Top-5 of our pruning techniques outperform all the baseline techniques with a reduction of top-1 accuracy by 3.56%, top-5 accuracy by 1.89%, Floating Point Operation by 2.3×, and storage by 2.05×.*

Keywords: Filter pruning, Convolutional neural networks, Deep learning, Model compression

1. **Introduction.** Convolutional Neural Networks (CNNs) are being used to solve human problems in both industry and academia. In computer vision, they are particularly helpful for tasks like image classification [1, 2], object detection [3, 4, 5, 6, 7, 8], object recognition [9, 10, 11, 12], and semantic segmentation [13, 14]. The performance of CNNs has been demonstrated to be exceptional in recent years. Their structures are complex and made up of numerous interconnected layers. This structure enables in capturing data patterns through filters in each layer. Normally, CNNs have filters for detecting edges and colors in their first layer. Deeper layers can recognize more complex patterns (grids or stripes) by using data from earlier layers. This results in the layer's filter having a large number of parameters and a large amount of storage space. For example, ResNet-50 [15] contains 25.6 million parameters and requires 98.2 megabytes (MB) of storage space. It influences the total amount of time necessary for the learning and prediction procedures.

DOI: 10.24507/ijicic.19.06.1697

The cost of the Graphic Processing Unit (GPU) to run the model is considerable. Moreover, it is difficult to execute CNNs model on hardware with limited resources [16], such as a microcontroller or smart phone. Limited resources refer to limited battery, computational unit, storage, etc. Model compression is a key strategy for running CNNs models effectively in constrained environments and resources.

Model compression is a technique for reducing the size of CNN's model, which can be classified into 3 categories [17, 18]. 1) Precision reduction is a technique for maintaining CNN performance while reducing the number of bits necessary to represent the CNN weights [17]. Although this technique can minimize the amount of storage and power requirement [19], it greatly depends on the criteria for selecting the number of bits, which could affect the model's accuracy. 2) Network pruning, a compression technique, is used to evaluate weights with less impact on the model's accuracy. The chosen weights are then set to zero or eliminated from CNN, and retraining is performed after that. The weight removal procedure is challenging, and the outcomes heavily depend on weight analysis. Inefficient weight analysis might result in decreased accuracy and ineffective compression issues [17]. 3) Compact network architecture is a technique for reducing the number of weights in CNN by altering the kernel design. Presently, a new structure in a sequence of convolutional layers with reduced kernel sizes has replaced the underlying structure. For instance, one of the most well-known architectures of the compact CNN is SqueezeNet [20]. The three strategies are implemented as follows: 1) The kernel size of the filter has been reduced from 3×3 to 1×1; 2) There is a reduction in the number of input channels that are transmitted to the 3×3 filters; 3) Subsequent layers reduce the feature map to a more manageable size. The compact nature of SqueezNet results in a significant reduction in the number of parameters, which in turn minimizes the amount of processing time required for CNN training and operation. A significant amount of information has been lost due to the squeeze convolutional layer's reduction of the channel number. Despite the fact that the model makes an effort during the expand process to recover the information loss, it is impossible to restore such a large amount of information that has been pruned, particularly when non-linear layers are being replaced.

Network pruning is the most intriguing technique among all CNN compression techniques. According to network training observation, this strategy can remove a variety of irrelevant weights with less impact on the pruned model's accuracy. The authors of [21] made use of the second derivative principle to enable tradeoffs between training errors and network complexity. Even though it is useful for real-world applications, the structure of a deep model necessitates a significant amount of storage space and processing capacity.

It was recently proposed to use the basic pruning technique to reduce weights when all connections were below a certain threshold. The model was then adjusted to restore accuracy [22]. After numerous iterations, the model became extremely sparse and non-structured, making it unsupportable by standard libraries. As a result, the model needed specialized hardware and software in order to collaborate effectively due to the unpredictable nature of network connections. The issue of non-structured random connectivity had an impact on memory accessibility and cache utilization [23]. Random connectivity caused poor cache locality, erratic memory access, and potential effects on practical acceleration. This restriction can occasionally cause a slowdown in acceleration.

Filter level pruning is one way to overcome non-structured limitations [24]. After redundant filters were removed, the pruned model was not sparse and did not cause a non-structure issue. This made it possible for the model to be more effective than non-structured pruning in the following ways. 1) This results in valid network connectivity, where implementation can be assisted by deep learning libraries, since the model structure is not harmed after pruning. As a result, the model can be further compressed using

other techniques, such as quantization of the parameters [25]. 2) As a result of model parameters being pruned, storage can be significantly reduced.

From previous research, it can be seen that the pruning criteria used for filter selection can have a great effect on the performance of the pruned models. The statistical based criteria are mostly adopted since the estimation is simple and can be used to represent the overall significance level of a neuron. Global representation might lack ability to deal with localizing the objects or Region-of-Interest (ROI), whereas single neuron analysis may lose information regarding the relationships within and between layers respected to the final prediction.

We, therefore, proposed a filter pruning method based on the Localized Gradient Activation heatmaP (LGAP), which considers the spatial correlation between the investigated filter and the target prediction. The LGAP was adopted as our criteria to prune filters in the CNN layers by analyzing the significance of a filter through the similarity of each CNN layer's behavior before and after pruning to estimate the filter's impact. The layer-wise loss of the investigated filter was determined in order to analyze the gradient difference between a complete layer and a layer without an investigated filter. The effect of loss of an investigated filter demonstrates the significance or insignificance of the filter. Our pruning criteria ensure that these critical filters are preserved, while maintaining the model accuracy. The model is then fine-tuned to recover prediction performance after pruning. The effectiveness of the proposed pruning method will be compared to other pruning techniques. From the experimental results, filter pruning based on our LGAP strategy can identify appropriate filters to be pruned, thus preserving the superior performance over other pruning techniques. The pruned model contributes to a reduction in the total amount of computational time, the number of parameters, and the amount of storage space required, depending on the selected pruning ratio.

2. **Related Work.** Compression techniques can be used to resolve the problem of an excessive number of parameters presented by deep learning models. To accelerate CNN [21], the model coefficients were initially removed using the optimal brain damage technique. This was one of the early techniques for pruning a network. The authors attempted to select parameters for regularization of the pruning process using a second-order Taylor expansion. In contrast to conventional fine-tuning, this method demanded the computation of the Hessian matrix, which added extra memory and computational overhead. Research on a new technique that integrates group-wise pruning into the learning phase was proposed by Lebedev and Lempitsky [26]. This allowed for group sparsity regularization. Since most of their values were close to zero, some weight groups might be eliminated. The approach in [23] adopted the Structured Sparsity Learning (SSL) technique. CNN structures can be regularized in a variety of ways, including the use of filters, filter shapes, channels, and layer depth. Even though SSL can effectively prune the network, the loss of the basic network structure may result in unsupported libraries. Other filter level pruning methods were proposed such as random pruning, weight sum technique [27], Average Percentage of Zeros (APoZ) [28], mean gradient [29], and Structured Sparsity Regularization (SSR) [30]. 1) Random pruning, in which filters are removed at random. 2) The technique of weight sum in [27] eliminated filters based on the sum of their absolute weights. As the weight sum increases, the filter becomes more significant. 3) Average Percentage of Zeros (APoZ) [28] technique determined the significance score of the filter, which was computed from the percentage of zero values in the activation output. A large percentage of zero was an indication that the filter was unnecessary and should be removed. 4) Mean gradient [29] is a pruning technique that analyzed layer feature maps based on its mean gradient. 5) Structured Sparsity Regularization (SSR) [30] was a technique in which the objective

function incorporated the structured sparsity constraint as well as the correlation between the global output loss and the local filter removal. Alternative Updating with Lagrange Multipliers (AULM) was employed for adaptive filter pruning based on two different types of structure sparse regularizers to deal with the convergence challenge. The approach was reportedly capable of achieving a fast convergence rate. 6) ThiNet [24] adopted a greedy strategy for channel selection, to prune the target layer by greedily selecting the input channel that has the least increase in reconstruction error.

The majority of currently available methods tend to concentrate on applying statistics to the filter weights (weight sum [27]) or feature map activations (APoZ [28], and SSR [30]), as a statistical criterion to distinguish the importance of filter neurons. However, these methods did not carry out data-driven (data feed forward) operations across the network of the model, which led to a lack of relational data analysis inside a layer as well as consequences on the data's propagation to the final prediction. Even though the statistical criterion can be used to represent overall significant level of a neuron, aiming to understand the predictions of a model by analyzing the individual units and seeking an explanation for specific activation, it does not provide confident relationship of the neuron respect to the final prediction. ThiNet [24] evaluates channel importance based on reconstruction loss from the least-square estimation to identify filter channels with the smallest impact to the output feature map. Nevertheless, previous mentioned techniques rely on the analysis of statistical evaluation of the filters, which lack the spatial information related to the target prediction. The localized prediction map is helpful in distinguishing the critical regions of the target prediction. Moreover, there have been fewer studies on the effects of pruned filter on the layer-wise activation.

Eliminating filter based on statistical criterion of single neuron analysis can lead to a loss in spatial relations within the filter activation itself, as well as the relationship to target prediction and the relationship among filters in that specific layer. To minimize the limitations, our contribution aims to

1) Analyze the localized gradient activation to estimate the spatial relationship between an individual filter and the output prediction;

2) Estimate filter pruning criterion based on the Localized Gradient Activation heatmaP (LGAP), which considers the spatial relationships between the feature map activation of the investigated filter and the target prediction;

3) Evaluate weak filter pruning strategy based on layer-wise loss of the investigated filter.

LGAP offers the advantage of using a data-driven method in conjunction with a study of the spatial relationship among filters in each layer based on layer-wise behavior rather than a single neuron analysis to identify weak filters while preserving the model's performance. As a result, the efficacy of our pruned model outperforms the previously discussed strategies [24, 28, 30].

3. **Methods.**

In this section, we detail our filter pruning technique into 4 sections: an overview of our filter level pruning method (Section 3.1), the LGAP estimation (Section 3.2), Significant filter scoring (Section 3.3), and weak filter pruning (Section 3.4).

3.1. **Overview of our filter level pruning.**

Our filter pruning process is performed on a pre-trained CNN model. To visualize the behavior of a filter neuron, the idea of gradient-based localization [31] is adopted. The localization can be achieved by passing a single forward of a set of training data to the pre-trained model. The gradient information is then estimated according to partial backpropagating gradient respected to feature map activations from the last convolutional layer of CNN to the beginning layers since the last

convolutional layer is expected to be the best approximation between high-level semantics from the fully-connected layers and detailed spatial relations from the CNN layers. The benefit of gradient-based localization is its ability to generate visual explanations from CNN neuron behaviors without architectural modifications. The global average pooling of local-gradient information of each filter is calculated as a filter significant weight.

Then, a layer-wise heatmap of weighted channel activations is generated as a layer summary. As a result, the activation from filter that corresponds best to the top class output will be emphasized, while the activation from the filter that corresponds less to the desired class will be diminished. The layer-wise heatmap can provide an excellent visualization on the layer's behavior. However, it may not be efficient illustration of the filter's behavior. Analyzing individual filter in each layer can be performed in two ways, through analysis of a significant weighted activation heatmap corresponding to a single filter neuron, or through analysis of heatmap loss from eliminating an investigated filter activation. We discover that analyzing significant weighted activation heatmap on a single layer basis may not be an effective way to determine the significance of a filter due to large variations in activations among filters in specific layers. This can have an impact on the development of filter scoring criterion for determining significant filters to be preserved and pruning strategy.

However, analyzing the similarity of a complete layer versus a layer without an investigated filter based on a local gradient emphasized heatmap, known as loss persistence heatmap, can provide an effect of loss of an investigated filter and prove to be better distinguish significant from insignificant filters. Therefore, we would use the loss persistence heatmap as our filter scoring criterion to prune the less effective filter to the desired output class. The overview of our filter level pruning technique is illustrated in Figure 1.

FIGURE 1. Overview of our pruning technique

3.2. **Localized Gradient Activation heatmaP (LGAP).** To estimate the LGAP, we first compute the gradient estimation of the top class prediction with respect to an output feature map activation obtained from training dataset prediction before softmax as shown in Equation (1) [31].

$$G^{l,k} = \frac{\partial y^c}{\partial A^{l,k}} \tag{1}$$

where $G^{l,k}$ is the gradient estimation of the kth filter in the layer l, y^c is the output of the top predicted class c, and $A^{l,k}$ is the output feature map activation of the kth filter neuron in the layer l.

The significant weight (α) of a neuron in terms of a global-average-pooling of the gradient $G^{l,k}$ over both its height (H) and width (W) corresponding to the output class c can be calculated in Equation (2) [31]. The results of significant weighted feature map help to emphasize the feature map of the significant filter, which can be visualized in Figure 2.

$$\alpha_{l,k}^c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} G_{i,j}^{l,k} \tag{2}$$

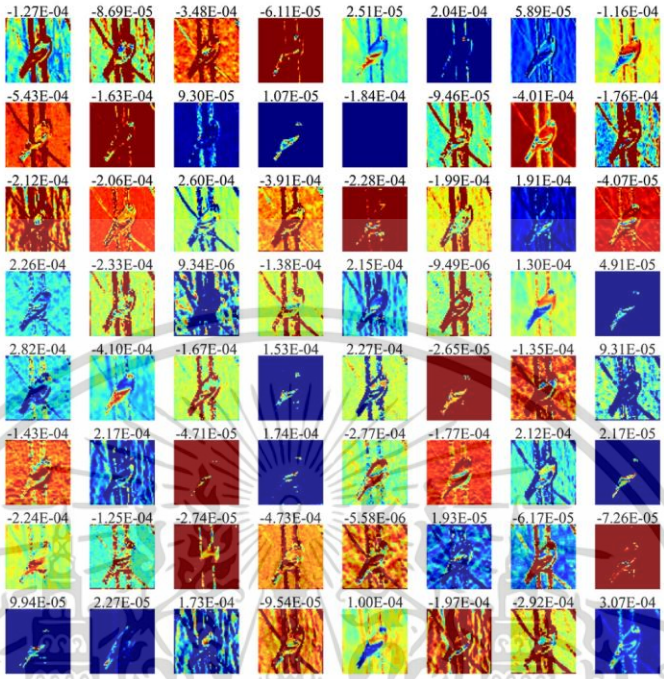


FIGURE 2. Significant weighted feature map

A layer summary can be expressed in terms of the layer-wise heatmap of weighted channel activations called Localized Gradient Activation heatmaP (LGAP), where the activations from filters best corresponding to the top-class output will be emphasized. We found that the layer-wise heatmap (H_c) can provide an excellent visualization on the layer's behavior and can be calculated in terms of a normalized sum of the significant weighted feature map activations of all neurons in the particular lth layer and we proposed those relations in Equation (3).

$$H_c = norm\left(\left|\sum_k \alpha_{l,k}^c A^{l,k}\right|\right) \tag{3}$$

where $|.|$ is absolute value and *norm* is based on the max-min normalization.

Localized gradient information estimates are typically based on regulated activation, with the assumption that negative activation may correspond to an undesired class [31]. Neglecting negative responses, on the other hand, may be beneficial for visualization, but may not be effective for filter significant analysis. To learn all filter's behaviors, both positive and negative activations are taken into account using a normalized activation heatmap visualization.

Analyzing individual filter in each layer can be performed through analysis of a significant weighted activation heatmap corresponding to a single filter neuron or analysis of the

heatmap of loss from eliminating an investigated filter activation. We discover that analyzing significant weighted activation heatmap on a single layer basis may not be an effective way to determine the significance of a filter due to large variations in activations among filters in specific layers. This can have an impact on the development of filter scoring criterion for determining significant filters to be preserved and pruning strategy. However, analyzing the similarity of a complete layer versus a layer without an investigated filter based on a local gradient emphasized heatmap, known as loss persistence heatmap, can provide an effect of loss of an investigated filter and prove to be better distinguished significantly from insignificant filters. Therefore, we would use the loss persistence heatmap ($H_{l,f}$) to calculate our filter scoring criterion to prune the less effective filter respected to the desired output class. The loss persistence heatmap can be calculated in Equation (4).

$$H_{l,f} = norm\left(\left|\left(\sum_k \alpha_{l,k}^c A^{l,k}\right) - \alpha_{l,f}^c A^{l,f}\right|\right) \tag{4}$$

where $A^{l,f}$ is the activation of fth filter kernel at layer l, which will be removed.

3.3. **Filter scoring.** This section explains our filter scoring criteria. Each filter is scored according to a similarity between the heatmap of a complete layer (H_c) and the loss persistence heatmap ($H_{l,f}$) (see in Figure 3). The similarity can be viewed as a persistence score when the particular filter f is removed from the layer. The higher the persistence score, the less the filter's removal effects on the desired output. We calculate the similarity based on cosine similarity method as shown in Equation (5). This comparison shows how the layer l is changed after the filter f is removed.

$$S_{c,f} = \frac{H_c, H_{l,f}}{||H_c||, ||H_{l,f}||} \tag{5}$$

where $S_{c,f}$ is the similarity score of the fth filter in terms of cosine similarity between H_c and $H_{l,f}$, and $||.||$ is L2 norm.

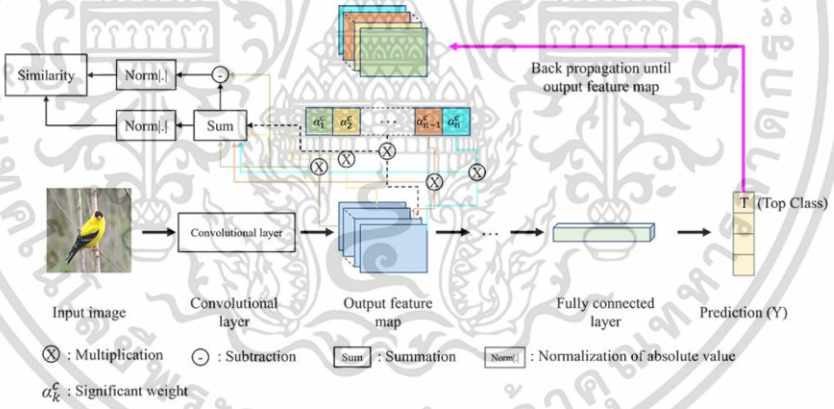


FIGURE 3. Overview of our filter scoring procedure for the lth layer

After obtaining cosine similarity of all filters, we rank the similarity results. The higher the similarity, the less effect the removed filters have on layer operation and are considered as weak filters that must be pruned from the layer.

3.4. **Weak filter pruning.** The procedure of weak filter pruning of layer l in CNN model is illustrated in Figure 4. In the first step, the output feature maps of layer l ($A^l \in \mathbb{R}^{H \times W \times K}$, where H is the height, W is the width and K is the number of channels) corresponding to the convolution filters at the lth layer ($conv^l \in \mathbb{R}^{S \times S \times K \times D}$, where $S \times S$ (3×3) is kernel size, K is the number of filter channels and D is the number of filters or

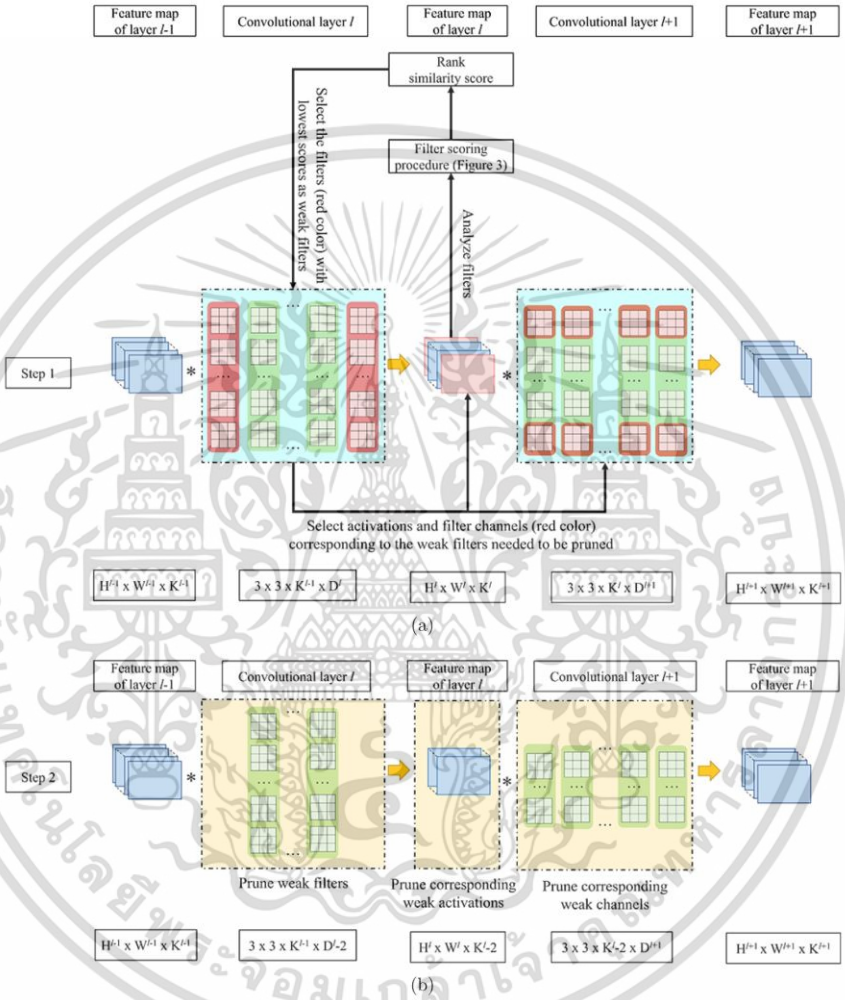


FIGURE 4. (color online) Weak filter pruning: (a) Step 1: Weak filter analysis; (b) Step 2: Prune weak filters and their corresponding weak activations and channels

number of channels of next $(l+1)$th layer) are extracted and the LGAP and filter score corresponding to the top-class output is estimated to obtain the loss persistence score of each filter. The n filters with the highest filter score are considered as weak filters since losses of those filters have less effect to the lth layer. The average filter scores of all images in the dataset for all filters in the specified layer are ranked and the filters corresponding to the highest scores are pruned in the second step. Pruning filters of layer l will affect their output feature maps of the next $(l+1)$th layer, which must also be removed as illustrated in Figure 4. After filter pruning, CNN model is fine-tuned to recover the model's prediction performance.

4. **Experiments.** In this session, we evaluate our filter level pruning method on pre-trained CNN models in Keras Applications [32]. We explain the experimental implementation in detail in Section 4.1, while, in Section 4.2, presenting a comparison of the pruning results obtained from our pruning technique and several baseline techniques.

4.1. **Implementation details.** In this section, we explain the dataset and the pre-trained models used to analyze and validate our pruning technique. To study the pruning impacts and performance losses, the models are pruned with different compression ratios.

4.1.1. *Dataset and models.* Our filter level pruning technique is evaluated on 2 different datasets.

• CIFAR-10 [33]: it is a color image dataset consisting of 60,000 images with 32×32 pixels in size. There are 10 different categories labeled from 1 to 10. The numbers of training and testing images are 50,000 and 10,000 images, respectively.

• ImageNet (ILSVRC 2012) [34]: it is a large image dataset with 1,000 different categories. There are 1.2 million images for training set and 50,000 images for validation set. All images are resized to 256×256 pixels and then cropped to 224×224 pixels based on their center.

For ImageNet, the images are resized to 256×256, and then cropped at its center area to obtain 224×224 pixels in size. An augmentation technique was used for generating transformation effects such as horizontal flip. The CIFAR-10 test set and ImageNet validation set are used to verify classification performance of the pruned model.

Two state-of-the-art classification models, VGG-16 and ResNet-50, are adopted.

• VGG-16 [35]: it is a CNN model developed from large scale of images for image classification. The base structure of model was trained with ImageNet dataset, and then modified to fit CIFAR-10 dataset [36] to illustrate the effective model performance. VGG-16 on CIFAR-10 consists of 13 convolutional layers with 3×3 kernel size. Filter pruning starts from the second convolutional layer since the first convolutional layer is necessary for creating the basis output feature map as mentioned in previous related works. The fully connected layers are modified to reduce the number of categories from 1000 ImageNet's categories down to 10 CIFAR-10's categories. The smaller size of CIFAR-10's images can greatly reduce the number of parameters from baseline model. Moreover, the model is integrated with a batch normalization layer [37] and a dropout layer [38] after each convolutional layer. The CNN structure of VGG-16 for CIFAR-10 is shown in Table 1. Floating Point Operation (FLOPs) in this paper referred to 2× of Multiply-Accumulate Computations (MACs). VGG-16 model has various convolutional layers with the total of FLOPs and parameter as 6.3E+08 and 1.5E+07, respectively.

• ResNet-50 [15]: it is a famous CNN model. This model was developed using the same large-scale dataset (ImageNet) as the baseline VGG-16, but with improved model performance. Since ResNet-50 has special structure of the residual blocks (as seen in Figure 5), the model can be pruned for the first two convolutional layers and the channels

TABLE 1. CNN structure of VGG-16 on CIFAR-10

Layer name	Filter ($S \times S \times K \times D$)	FLOPs	Parameter
conv2d	$3 \times 3 \times 3 \times 64$	3.5E+06	1.8E+03
conv2d_1	$3 \times 3 \times 64 \times 64$	7.5E+07	3.7E+04
conv2d_2	$3 \times 3 \times 64 \times 128$	3.8E+07	7.4E+04
conv2d_3	$3 \times 3 \times 128 \times 128$	7.5E+07	1.5E+05
conv2d_4	$3 \times 3 \times 128 \times 256$	3.8E+07	3.0E+05
conv2d_5	$3 \times 3 \times 256 \times 256$	7.5E+07	5.9E+05
conv2d_6	$3 \times 3 \times 256 \times 256$	7.5E+07	5.9E+05
conv2d_7	$3 \times 3 \times 256 \times 512$	3.8E+07	1.2E+06
conv2d_8	$3 \times 3 \times 512 \times 512$	7.5E+07	2.4E+06
conv2d_9	$3 \times 3 \times 512 \times 512$	7.5E+07	2.4E+06
conv2d_10	$3 \times 3 \times 512 \times 512$	1.9E+07	2.4E+06
conv2d_11	$3 \times 3 \times 512 \times 512$	1.9E+07	2.4E+06
conv2d_12	$3 \times 3 \times 512 \times 512$	1.9E+07	2.4E+06

conv2d: 2-dimensional operation of convolutional layer
$S \times S \times K \times D$: $S \times S$ is kernel size, K is the number of channels and D is number of filters
FLOPs: Floating Point Operation

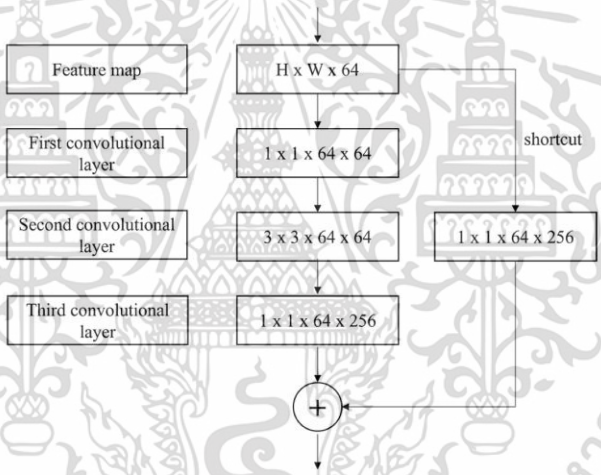


FIGURE 5. An example of the residual block of ResNet-50

of the next layers corresponding to the pruned layers must be removed. The output feature map of the third layer does not change dimension. The ResNet-50 trained on ImageNet is used to validate our filter pruning technique. Since ResNet-50 model consists of various convolutional layers and its structure is more complicated than VGG-16 as shown in Table 2, we will illustrate FLOPs and parameters of each layer in terms of a summarized base structure with the total of 7.7E+09 FLOPs and 2.3E+07 parameters, respectively.

TABLE 2. CNN structure of ResNet-50 on ImageNet

Layer name	Filter ($S \times S/D$)	Number of series	Output size
conv2d_1	$7 \times 7/64$	1	112×112
conv2d_2_x	$1 \times 1/64$	3	56×56
	$3 \times 3/64$		
	$1 \times 1/256$		
conv2d_3_x	$1 \times 1/128$	4	28×28
	$3 \times 3/128$		
	$1 \times 1/512$		
conv2d_4_x	$1 \times 1/256$	6	14×14
	$3 \times 3/256$		
	$1 \times 1/1024$		
conv2d_5_x	$1 \times 1/512$	3	7×7
	$3 \times 3/512$		
	$1 \times 1/2048$		

conv2d: 2-dimensional operation of convolutional layer
$S \times S/D$: $S \times S$ is kernel size and D is number of filters

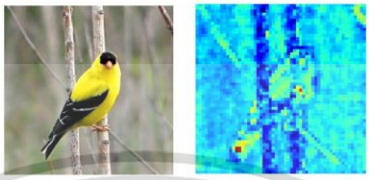
Prior to pruning the pre-trained model, we must explore parameters (learning rate, momentum, weight decay, and batch size) in order to fine-tune the original pre-trained models and produce results equivalent to the original pre-trained model. These parameters will be utilized to regain the pruning model's performance. We discovered that the VGG-16 model based on CIFAR-10 would be well fine-tuned with the following parameters: SGD optimizer, learning rate of 0.001, momentum of 0.9, weight decay of 0.0005, and batch size of 128; however, the parameters of ResNet-50 based on ImageNet would require the same SGD optimizer with a difference in batch size of 32. After pruning all layers based on a layer-by-layer basis, both models are fine-tuned to restore model performance. The number of fine-tuned epochs for VGG-16 on CIFAR-10 and ResNet-50 on ImageNet has been set to 40 and 50, respectively, with scheduled learning rate ranging from 10^{-3} to 10^{-5}.

4.1.2. *Pruning ratio.* It is a challenge to discover suitable percentage of pruning for each layer since each layer has different pruning sensibilities. For VGG-16 on CIFAR-10, our experiments are implemented on pruning ratio of 10%-90% for each layer. However, for ResNet-50 on ImageNet containing a large model size, we perform the experiments on 30%, 50% and 70% since it takes a long time to prune and fine-tune the model. Pruning can help to decrease complexity and size of the model. Large compression ratio is always preferred to reduce the complexity while maintaining the model performance.

4.1.3. *Filter scoring results.* A set of images from training dataset is randomly selected as inputs to obtain the output feature maps, which will be used to analyze filter and estimate filter pruning scores. Figure 6 shows an example of the input image (ImageNet) that is fed into the ResNet-50 to analyze first convolutional layer and obtain the corresponding layer-wise heatmap based on LGAP. Loss persistence heatmaps based on LGAP (from Equation (4)) are estimated from all filters in each layer (for example, 64 filters in the first layer ResNet-50 in Figure 7). Then, a filter pruning score (from Equation (5)) is calculated and the average score $Score_f$ (from Equation (6)) is investigated to analyze the significance of a filter.

$$Score_f = \frac{1}{N}\sum_{n=1}^{N} S_{c,f} \tag{6}$$

where N is number of input images. The filter scores are ranked in ascending order. The lower the score, the more significance the filter.



(a) (b)

FIGURE 6. (a) The input image and (b) the layer-wise heatmap based on LGAP (from Equation (3))

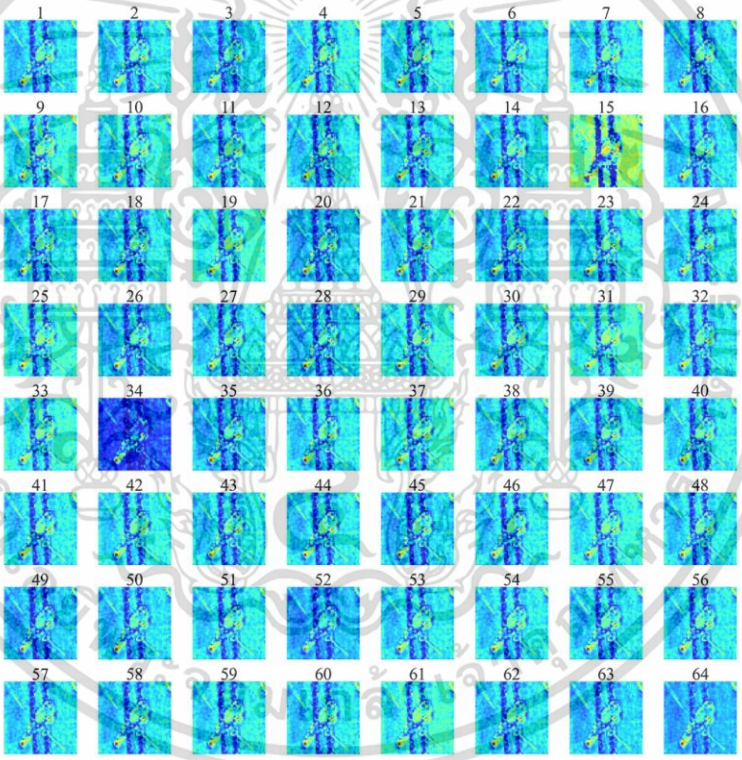


FIGURE 7. Loss persistence heatmap (Equation (4))

In our experiments, sets of 10, 100, and 200 images per category are adopted from CIFAR-10 and used to analyze our pruning technique on VGG-16. An example of filter pruning on the first convolutional layer of VGG-16 with pruning ratio 50% can achieve the accuracy of 93.24%, 93.33%, and 93.28%, respectively. It can be seen that increasing number of input images may not cause significant effects on the model accuracy. Hence, our experiment on VGG-16 will adopt a set of 10 images per class for filter analysis and scoring for both VGG-16 on CIFAR-10 and ResNet-50 on ImageNet.

4.2. **Pruning results.** In this section, we illustrate result from our pruning technique via VGG-16 and ResNet-50 compared to other related pruning techniques.

4.2.1. *Results from pruning VGG-16 on CIFAR-10.* First, we illustrate the results of our filter pruning for each convolutional layer with different pruning ratios in Figure 8. The capability of the model to endure a larger pruning ratio can be determined by the severity of the decline in accuracy caused by the pruned layer (sensibility). As the pruning ratio increases, the accuracy of the lower layers (conv2d_1 to conv2d_6) tends to be affected more than the accuracy of the deeper levels (conv2d_7 to conv2d_12).

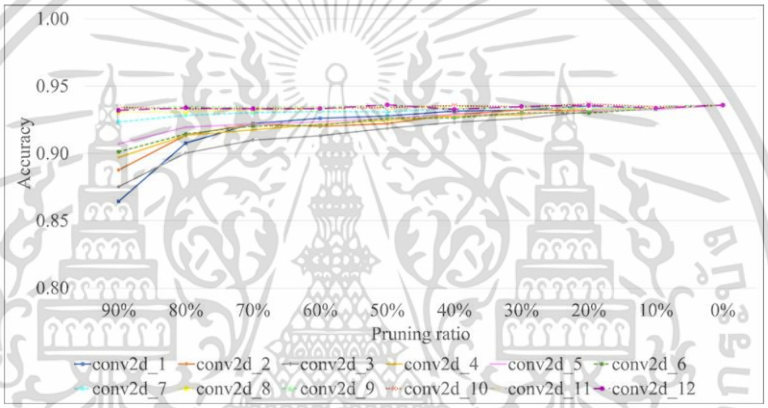


FIGURE 8. VGG-16 model accuracy after filter pruning each convolutional layer with different pruning ratios

We perform layer-wise pruning of all convolutional layers (conv2d_1 to conv2d_12) with the pruning ratio in range of 10%-90% with the same pruning ratio for all layers (see in Table 3). In Table 3, we calculate FLOPs from conv2d_1 layer to conv2d_12 layer. For test set of CIFAR-10 (10,000 images), the pruned model at 30% pruning ratio provides a reduction in FLOPs, parameters, and storage by 2×. Prediction time can be reduced about 1.7× while accuracy is decreased about 0.71%. The most reduction in FLOPs, parameters, storage and prediction time are 44.1×, 84.4×, 69.5× and 2.2×, respectively, with the accuracy reduction about 17.07% at 90% pruning ratio.

4.2.2. *A performance comparison for VGG-16 on CIFAR-10 with different filter pruning techniques.* In order to evaluate our filter pruning method, a performance comparison is conducted with several baseline techniques on N images, described as follows.

1) Random: filters are randomly pruned from layers. The result with the random seed is selected.

TABLE 3. The model performance after filter pruning in all layers on VGG-16

Pruning ratio	Error (%)	FLOPs	Parameters	Storage size (MB)	Prediction times (Second)
0%	6.41	6.3E+08	1.5E+07	57.4	2.25
10%	6.32	5.1E+08	1.2E+07	46.2	2.21
20%	6.64	4.1E+08	9.6E+06	36.6	2.02
30%	7.12	3.2E+08	7.4E+06	28.2	1.70
40%	7.28	2.4E+08	5.4E+06	20.9	1.31
50%	8.07	1.7E+08	3.8E+06	14.6	1.10
60%	8.53	1.2E+08	2.4E+06	9.4	1.10
70%	10.34	7.2E+07	1.4E+06	5.5	1.07
80%	13.88	3.7E+07	6.4E+05	2.6	1.08
90%	23.48	1.4E+07	1.8E+05	0.8	1.04

MB: Megabyte

2) Weighted sum [27]: a sum of absolute filter kernel weight values is calculated as a filter score. Weak filters with low weighted sum scores are pruned.

$$Score_i^{\text{weighted sum}} = \sum |W(:,:,:,i)|$$

where W is the weights of each filter i.

3) APoZ [28]: a percentage of zero activation after ReLU function from each filter is calculated as a filter score. The higher APoZ score, the weaker the filter.

$$Score_i^{\text{APoZ}} = \frac{1}{N \times H \times W} \sum_{n=1}^{N} (f(O_k(:,:,i) = 0))$$

where $H \times W$ is the dimension of output feature map $O_k \in \mathbb{R}^{H \times W \times D}$, $f(.) = 1$ if (.) is true, and $f(.) = 0$ if (.) is false.

4) Mean gradient [29]: a mean of gradient from each filter is calculated as filter score. Weak filters with low mean of gradient are pruned.

$$Score_i^{\text{Mean gradient}} = \frac{1}{N} \sum_{n=1}^{N} |mean(G(:,:,i))|$$

where G is the gradient calculated from each filter i.

5) Weighted Activation (WA): a normalized heatmap of absolute significant weighted feature map is calculated as a filter score below. Weak filters with low scores are pruned.

$$H_i^{WA} = norm\left(\left|\alpha_{l,i}^c A^{l,i}\right|\right)$$

$$Score_i^{WA} = \frac{1}{N} \sum_{n=1}^{N} \frac{H_c, H_i^{WA}}{||H_c|| ||H_i^{WA}||}$$

where H_i^{WA} is the normalized heatmap of absolute significant weighted feature map.

A performance comparison of our proposed filter pruning criteria and previous mentioned techniques is shown in Figure 9. The pruning ratios used in the experiments range from 10% to 90%. It can be shown that the accuracy of all strategies is comparable within the pruning ratio range of 10%-60%. However, as pruning ratio exceeds 60%, we start to see that the accuracy of APoZ, mean gradient, and Weight Activation begins to decline. Especially when the pruning ratio exceeds 90%, our filter pruning score based on loss

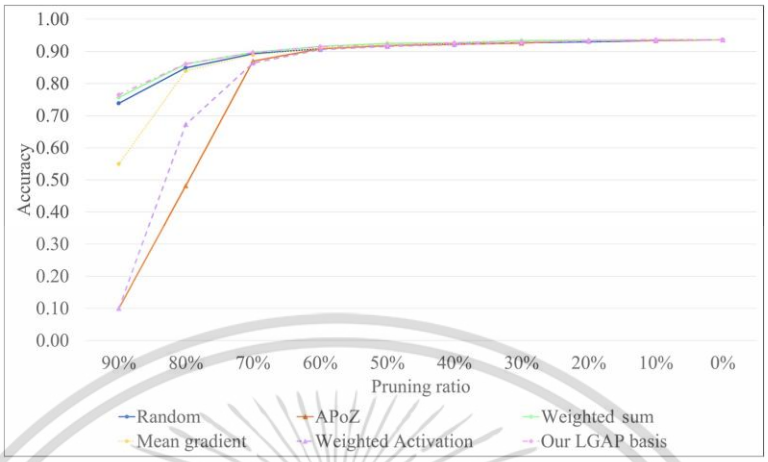


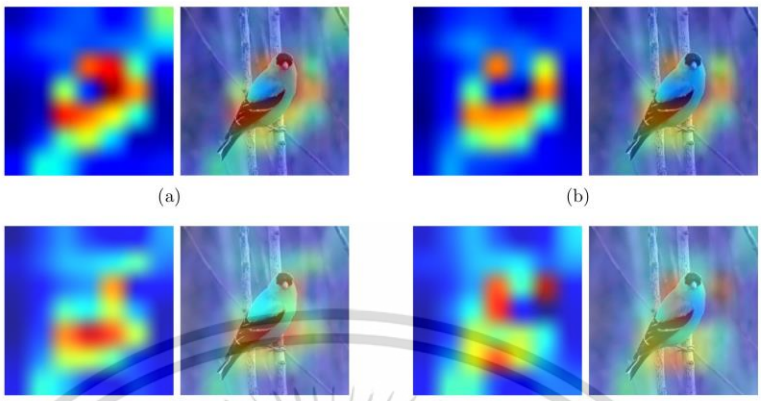
FIGURE 9. A performance comparison of our filter pruning criteria and other baseline filter selection criteria

persistence LGAP can maintain higher accuracy than the baseline techniques of random, weighted sum, APoZ, mean gradient and Weighted Activation by 2.66%, 0.83%, 66.52%, 21.57% and 66.52%, respectively. For the results of filter pruning techniques of VGG-16 on CIFAR-10, we observe that all the models can maintain their accuracies with pruning ratio less than 60%. This means that if we would like to compress the model less than 60%, any of the experimental techniques can provide comparable results. However, when the pruning ratio exceeds 60%, the decline in accuracy starts to straight down for most of the techniques while our technique can still maintain the model performance.

4.2.3. *A performance comparison for VGG-16 on CIFAR-10.* VGG-16, trained on a large-scale ImageNet dataset, provides a stronger model and tends to be well tolerant with filter pruning techniques. We observe that, at 60% pruning ratio, our filter pruning technique obtains a decrease in accuracy by 2.12%, reduction 5.4× in FLOPs and 6.2× in the number of parameters while pruning ratio at 90%, our filter pruning technique outperforms the other techniques: random, weighted sum, APoZ, mean gradient, and Weighted Activation by 2.66%, 0.83%, 66.52%, 21.57% and 66.52%, respectively.

4.2.4. *Model performance of pruned ResNet-50 on ImageNet.* The original input image is shown in Figure 6(a) while Figure 10 illustrates the LGAP estimation of the top class of prediction with respect to an output feature map activation obtained from the pruned ResNet-50 model with different filter pruning ratios compared to the original one. As the pruning ratio increases, local gradient heatmap is gradually changed, and the output activation is lessened and shifted from the gradient result of the original model. Even though it does not affect the classification result in this case, it causes errors in some of the images and the % error starts to increase with filter pruning ratio above 50%.

We illustrate the results of pruning ResNet-50 using our loss persistent LGAP pruning strategy in Table 4, which illustrates the percentage of errors from Top-1 and Top-5 results, the FLOPs, required size of parameters, storage, and prediction time from all convolutional layers in ResNet-50. Prediction times are measured from a set of ImageNet



(a) (b) (c) (d)

FIGURE 10. Localized Gradient Activation heatmaP (LGAP) of the original and pruned ResNet-50 with different pruning ratios: (a) Original ResNet-50, (b) 30% pruning ratio of ResNet-50, (c) 50% pruning ratio of ResNet-50 and (d) 70% pruning ratio of ResNet-50

TABLE 4. Filter pruning results on ResNet-50

Pruning ratio	Top-1 error (%)	Top-5 error (%)	FLOPs	Parameters	Storage size (MB)	Prediction time (Second)
0%	25.10	7.90	7.7E+09	2.6E+07	98.2	117.35
30%	27.55	8.94	4.8E+09	1.7E+07	65.2	108.30
50%	28.66	9.79	3.4E+09	1.2E+07	47.8	99.33
70%	31.44	11.26	2.2E+09	8.7E+06	33.6	90.21

MB: Megabyte

(50,000 images). At 30% pruning ratio, the model accuracy of the top-1 decreases by 2.45% and top-5 decreases by 1.04%, while FLOP, the number of parameters, the storage size, and the prediction time decrease about 1.6×, 1.5×, 1.5×, and 1.08×, respectively. At 50% pruning ratio, the model accuracy of top-1 and top-5 decreases by 3.56% and 1.89%, respectively. The FLOPs, the number of parameters, the storage size, and the prediction time decrease about 2.3×, 2.1×, 2.1×, and 1.18×, respectively. At 70% pruning ratio, the model accuracy of the top-1 decreases by 6.34% and top-5 decreases by 3.36%. The FLOPs decrease about 3.6×, parameters decrease about 3×, storage decreases about 2.9× and prediction times decrease about 1.3×. It can be seen that with large pruning ratio, FLOPs and parameter of ResNet-50 are greatly reduced; however, prediction times slightly decrease when compared with VGG-16. This may cause by the special structure (Residual block) inside of ResNet-50, which still requires large computation time. Even though the prediction time can be slightly reduced, the hardware resources can be significantly optimized and suitable for devices with limit resources (memory, and processing unit).

A performance comparison of filter pruning strategies for ResNet-50 on ImageNet is listed in Table 5. With 50% pruning ratio, the results show that Top-1 and Top-5 of our techniques outperform all the baseline techniques. Our technique can achieve lower

Top-1 error compared to the baseline techniques: random, SSL, ThiNet, APoZ, weighted sum, mean gradient and SSR-L2 by 1.09%, 1.02%, 0.56%, 0.69%, 0.75%, 0.34% and 0.09%, respectively. Top-5 error of our technique is lower than baseline techniques: random, SSL, ThiNet, APoZ, weighted sum, mean gradient and SSR-L2 by 0.86%, 0.79%, 0.39%, 0.52%, 0.53%, 0.05% and 0.22%, respectively. For image classification experiments in the real world, we examined the performance of the trimmed ResNet-50 on images not included in the ImageNet dataset. We found that pruned models can predict images belonging to their class with comparable accuracy and in less time than the original model.

TABLE 5. A performance comparison of filter pruning on ResNet-50 with pruning ratio 50%

Technique	Top-1 accuracy drop (%)	Top-5 accuracy drop (%)	FLOPs	Parameters	Pruned
Random	4.65	2.75	3.4E+09	1.2E+07	50%
SSL [23]	4.58	2.68	4.2E+09	1.3E+07	$\approx 50\%$
ThiNet [24]	4.12	2.28	3.4E+09	1.2E+07	50%
APoZ [28]	4.25	2.41	3.4E+09	1.2E+07	50%
Weighted sum [27]	4.31	2.42	3.4E+09	1.2E+07	50%
Mean gradient [29]	3.90	1.94	3.4E+09	1.2E+07	50%
SSR-L2 [30]	3.65	2.11	3.4E+09	1.2E+07	50%
Our technique	**3.56**	**1.89**	**3.4E+09**	**1.2E+07**	**50%**

5. **Conclusion.** In this article, we propose a new technique for filter level pruning to prune two CNN models (VGG-16 on CIFAR-10 and ResNet-50 on ImageNet) based on the Localized Gradient Activation heatmaP (LGAP). The LGAP is a layer-wise heatmap of weighted channel activations, where the activation from filters best corresponding to the top-class output will be emphasized. It provides the spatial relationship between the investigated filter and the target prediction. The filter score is then estimated based on the similarity between the heatmap of a complete layer and the loss persistence heatmap. The similarity can be viewed as a persistence score when a filter is removed from the layer. The high filter score shows that removed filters have less effect with layer operation and are considered as weak filters that must be pruned from the layer. Even though the other related techniques adopted data-driven approach, they lacked spatial relationship among filters in each layer and relationship between filters and output prediction due to their concentration on a single-neural analysis. From the experimental results, it can be seen that the performance of the pruned model using our filter pruning strategy outperforms the other filter pruning techniques especially with high compression ratio. However, since the fine-tuning process of a large pre-trained model required a significant amount of time to recover performance after pruning, our future research will concentrate on reducing the time required for fine-tuning performance recovery.

Acknowledgment. This work was supported by King Mongkut's Institute of Technology Ladkrabang, Thailand.

REFERENCES

[1] A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Commun. ACM*, vol.60, no.6, pp.84-90, 2017.

[2] A. F. Siregar and T. Mauritsius, Ulos fabric classification using android-based convolutional neural network, *International Journal of Innovative Computing, Information and Control*, vol.17, no.3, pp.753-766, 2021.
[3] R. Girshick, Fast R-CNN, *Proc. of 2015 IEEE International Conference on Computer Vision (IC CV)*, pp.1440-1448, 2015.
[4] J. Yuan et al., Gated CNN: Integrating multi-scale feature layers for object detection, *Pattern Recognition*, vol.105, 2020.
[5] E. Essa, D. Aldesouky, S. E. Hussein and M. Z. Rashad, Neuro-fuzzy patch-wise R-CNN for multiple sclerosis segmentation, *Medical & Biological Engineering & Computing*, vol.58, no.9, pp.2161-2175, 2020.
[6] R. Girshick, J. Donahue, T. Darrell and J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, *Proc. of 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.580-587, 2014.
[7] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, *Proc. of the 28th International Conference on Neural Information Processing Systems*, vol.1, pp.91-99, 2015.
[8] W. Yang, G.-L. Zhou, Z.-W. Gu, X.-D. Jiang and Z.-M. Lu, Safety helmet wearing detection based on an improved YOLOv3 scheme, *International Journal of Innovative Computing, Information and Control*, vol.18, no.3, pp.973-988, 2022.
[9] C. Szegedy et al., Going deeper with convolutions, *Proc. of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.1-9, 2015.
[10] F. Chollet, Xception: Deep learning with depthwise separable convolutions, *Proc. of 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.1800-1807, 2017.
[11] S. Xie, R. Girshick, P. Dollár, Z. Tu and K. He, Aggregated residual transformations for deep neural networks, *Proc. of 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.5987-5995, 2017.
[12] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, Densely connected convolutional networks, *Proc. of 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2261-2269, 2017.
[13] H. Noh, S. Hong and B. Han, Learning deconvolution network for semantic segmentation, *Proc. of 2015 IEEE International Conference on Computer Vision (ICCV)*, pp.1520-1528, 2015.
[14] Y. Wang, J. Liu, Y. Li, J. Fu, M. Xu and H. Lu, Hierarchically supervised deconvolutional network for semantic video segmentation, *Pattern Recognition*, vol.64, pp.437-445, 2017.
[15] K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition, *Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.770-778, 2016.
[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, Rethinking the inception architecture for computer vision, *Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2818-2826, 2016.
[17] R. Pilipović, P. Bulić and V. Risojević, Compression of convolutional neural networks: A short survey, *Proc. of the 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*, pp.1-6, 2018.
[18] D. Ghimire, D. Kil and S.-H. Kim, A survey on efficient convolutional neural networks and hardware acceleration, *Electronics (Switzerland)*, vol.11, no.6, 2022.
[19] U. Lotrič and P. Bulić, Applicability of approximate multipliers in hardware neural networks, *Neurocomputing*, vol.96, pp.57-65, 2012.
[20] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally and K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50× fewer parameters and <1MB model size, *CoRR*, vol.abs/1602.07360, 2016.
[21] Y. LeCun, J. S. Denker and S. A. Solla, Optimal brain damage, *Advances in Neural Information Processing Systems 2*, Denver, CO, USA, pp.598-605, 1989.
[22] S. Han, J. Pool, J. Tran and W. Dally, Learning both weights and connections for efficient neural network, *NIPS'15: Proc. of the 28th International Conference on Neural Information Processing Systems*, pp.1135-1143, 2015.
[23] W. Wen, C. Wu, Y. Wang, Y. Chen and H. Li, Learning structured sparsity in deep neural networks, *NIPS'16: Proc. of the 30th International Conference on Neural Information Processing Systems*, pp.2074-2082, 2016.

[24] J.-H. Luo, J. Wu and W. Lin, ThiNet: A filter level pruning method for deep neural network compression, *Proc. of 2017 IEEE International Conference on Computer Vision (ICCV)*, pp.5068-5076, 2017.
[25] J. Wu, C. Leng, Y. Wang, Q. Hu and J. Cheng, Quantized convolutional neural networks for mobile devices, *Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.4820-4828, 2016.
[26] V. Lebedev and V. Lempitsky, Fast ConvNets using group-wise brain damage, *Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2554-2564, 2016.
[27] H. Li, A. Kadav, I. Durdanovic, H. Samet and H. P. Graf, Pruning filters for efficient ConvNets, *CoRR*, vol.abs/1608.08710, 2017.
[28] H. Hu, R. Peng, Y.-W. Tai and C.-K. Tang, Network trimming: A data-driven neuron pruning approach towards efficient deep architectures, *CoRR*, vol.abs/1607.03250, 2016.
[29] C. Liu and H. Wu, Channel pruning based on mean gradient for accelerating convolutional neural networks, *Signal Processing*, vol.156, pp.84-91, 2019.
[30] S. Lin, R. Ji, Y. Li, C. Deng and X. Li, Toward compact ConvNets via structure-sparsity regularized filter pruning, *IEEE Transactions on Neural Networks and Learning Systems*, vol.31, no.2, pp.574-588, 2020.
[31] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, Grad-CAM: Visual explanations from deep networks via gradient-based localization, *Proc. of 2017 IEEE International Conference on Computer Vision (ICCV)*, pp.618-626, 2017.
[32] F. Chollet et al., *Keras*, https://github.com/fchollet/keras, 2015.
[33] A. Krizhevsky and G. Hinton, *Learning Multiple Layers of Features from Tiny Images*, Technical Report, University of Toronto, 2009.
[34] O. Russakovsky et al., ImageNet large scale visual recognition challenge, *International Journal of Computer Vision*, vol.115, no.3, pp.211-252, 2015.
[35] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, *Proc. of the 3rd International Conference on Learning Representations (ICLR 2015)*, pp.1-14, 2015.
[36] S. Liu and W. Deng, Very deep convolutional neural network based image classification using small training sample size, *Proc. of the 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pp.730-734, 2015.
[37] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, *Proc. of the 32nd International Conference on Machine Learning*, pp.448-456, 2015.
[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, *J. Mach. Learn. Res.*, vol.15, no.1, pp.1929-1958, 2014.

Author Biography

Monthon Intraraprasit received the M.E. degree in Computer Engineering from King Mongkut's Institute of Technology Ladkrabang, Thailand, in 2017. He is currently pursuing the D.Eng. degree with the King Mongkut's Institute of Technology Ladkrabang, Thailand. His research interests include machine learning, deep learning and computer vision.

Orachat Chitsobhuk received the Ph.D. degree in Electrical Engineering from University of Texas, Arlington, US, in 2001. She is currently an associate professor at King Mongkut's Institute of Technology Ladkrabang, Thailand. Her research interests include image and scene analysis, machine learning and pattern recognition.

ประวัติผู้เขียน

ชื่อ-นามสกุล	นายมณฑล อินทรประสิทธิ์
ที่อยู่	34 ซ.สมบูรณ์พัฒนา 3 ถ.ประชาสงเคราะห์ แขวงดินแดง เขตดินแดง จังหวัดกรุงเทพมหานคร 10400
ประวัติการศึกษา	2559 จบการศึกษาวิศวกรรมศาสตรบัณฑิต คณะวิศวกรรมศาสตร์ สาขาคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
	2560 จบการศึกษาวิศวกรรมศาสตรมหาบัณฑิต คณะวิศวกรรมศาสตร์ สาขาคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง