การเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก

กศตพรรษ นาคแนม

วิทยานิพนธ์เสนอบัณฑิตวิทยาลัย มหาวิทยาลัยนเรศวร
เพื่อเป็นส่วนหนึ่งของการศึกษา หลักสูตรวิทยาศาสตรมหาบัณฑิต
สาขาวิชาฟิสิกส์การแพทย์
ปีการศึกษา 2567

การเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก

กศตพรรษ นาคแนม

วิทยานิพนธ์เสนอบัณฑิตวิทยาลัย มหาวิทยาลัยนเรศวร
เพื่อเป็นส่วนหนึ่งของการศึกษา หลักสูตรวิทยาศาสตรมหาบัณฑิต
สาขาวิชาฟิสิกส์การแพทย์
ปีการศึกษา 2567

วิทยานิพนธ์ เรื่อง "การเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก"
ของ กศตพรรษ นาคแนม
ได้รับการพิจารณาให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาฟิสิกส์การแพทย์

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการสอบวิทยานิพนธ์
(รองศาสตราจารย์ ดร.กิติวัฒน์ คำวัน)

..... ประธานที่ปรึกษาวิทยานิพนธ์
(ผู้ช่วยศาสตราจารย์ ดร.ฐิติพงศ์ แก้วเหล็ก)

..... กรรมการผู้ทรงคุณวุฒิภายใน
(ผู้ช่วยศาสตราจารย์ ดร.นันทวัฒน์ อู่ดี)

อนุมัติ

.....
(รองศาสตราจารย์ ดร.กรองกาญจน์ ชูทิพย์)
คณบดีบัณฑิตวิทยาลัย

ชื่อเรื่อง การเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก

ผู้วิจัย กศตพรรษ นาคแนม

ประธานที่ปรึกษา ผู้ช่วยศาสตราจารย์ ดร.ฐิติพงศ์ แก้วเหล็ก

ประเภทสารนิพนธ์ วิทยานิพนธ์ วท.ม. ฟิสิกส์การแพทย์, มหาวิทยาลัยนเรศวร, 2567

คำสำคัญ การเรียนรู้เชิงลึก, การแบ่งส่วนข้อมูลภาพ, การเพิ่มคุณภาพของภาพ, มะเร็งไกลโอมา, ภาพเอ็มอาร์ไอสมอง

บทคัดย่อ

ไกลโอมาเป็นเนื้องอกที่ต้องได้รับการรักษาและกำหนดขอบเขตของเนื้องอกอย่างแม่นยำ ในปัจจุบันการกำหนดขอบเขตของเนื้องอกเป็นเรื่องที่ใช้เวลานานและขึ้นอยู่กับประสบการณ์ของนักรังสีแพทย์ ดังนั้นปัญญาประดิษฐ์จึงมีบทบาทสำคัญในการลดปัญหาดังกล่าว ดังนั้นการศึกษานี้จึงมีจุดมุ่งหมายเพื่อพัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง รวมถึงเปรียบเทียบผลของกระบวนการปรับปรุงภาพเพื่อเพิ่มประสิทธิภาพของโมเดล การดำเนินการมีการใช้เทคนิคการปรับปรุงภาพ Contrast-Limited Adaptive Histogram Equalization (CLAHE), Gamma Correction (GC), Non-Local Mean filter (NLMF), และ Median and Wiener filter (MWF) ใช้สถาปัตยกรรม U-net, UNet++, และ ThirdConv รวมไปถึงใช้ชุดข้อมูลภาพ BraTS2023 และประเมินคุณภาพของภาพจากค่า Structural Similarity Index (SSIM) และเปรียบเทียบความเหมือนของขอบเขตเนื้องอกจากค่า Dice Similarity Coefficient (DSC) ผลการประเมินพบว่าโมเดลที่ถูกฝึกฝนโดยใช้สถาปัตยกรรม ThirdConv และใช้ชุดข้อมูลปกติมีค่า DSC เท่ากับ 0.811 ขณะที่ชุดข้อมูลที่เพิ่มคุณภาพของภาพด้วยเทคนิค CLAHE, GC, และ NLMF มีค่า DSC เท่ากับ 0.803 และเทคนิค MWF มีค่า DSC เท่ากับ 0.807 สำหรับผลการประเมินคุณภาพของภาพ เทคนิค CLAHE, GC, NLNF, และ MWF มีค่า SSIM เท่ากับ 0.916, 0.968, 1.000, และ 0.936 ตามลำดับ สรุปได้ว่าการใช้ชุดข้อมูลที่เพิ่มคุณภาพของภาพด้วยเทคนิคต่าง ๆ สามารถเพิ่มหรือลดประสิทธิภาพให้กับโมเดลได้ อย่างไรก็ตาม โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลที่เพิ่มคุณภาพของภาพเพียงอย่างเดียวไม่เพียงพอที่จะเพิ่มประสิทธิภาพของโมเดลได้อย่างมีนัยสำคัญ

Title COMPARISON OF PRE-PROCESSING METHODS FOR IMPROVING PERFORMANCE OF IMAGE SEGMENTATION OF GLIOMA ON BRAIN MRI IMAGES USING DEEP LEARNING

Author Kasatapad Naknaem

Advisor Assistant Professor Titipong Kaewlek, Ph.D.

Academic Paper M.S. Thesis in Medical Physics, Naresuan University, 2024

Keywords Deep learning, Image segmentation, Image enhancement, Glioma, Brain MRI image

ABSTRACT

Glioma, a type of brain tumor, necessitates precise and accurate treatment, so tumor boundary delineation is crucial. Currently, determining the tumor's extent is time-consuming and highly dependent on the oncologist's expertise. Artificial intelligence (AI) can address these challenges. This study aims to develop a deep learning model for segmentation on glioma MRI images and to compare pre-processing methods to enhance model performance. The study uses: image enhancement algorithms include Contrast-Limited Adaptive Histogram Equalization (CLAHE), Gamma Correction (GC), Non-Local Mean Filter (NLMF), and Median and Wiener Filter (MWF); U-net, UNet++, and ThirdConv architectures; the BraTS2023 dataset; the Structural Similarity Index (SSIM) assesses image quality; and the Dice Similarity Coefficient (DSC) measures the similarity of tumor boundaries. Results indicate that the ThirdConv architecture trained on the normal dataset achieved a DSC value of 0.811. In the datasets enhanced using CLAHE, GC, and NLMF, the DSC values were 0.803 and 0.807 for MWF. In terms of image quality, SSIM values for CLAHE, GC, NLMF, and MWF were 0.916, 0.968, 1.000, and 0.936, respectively. In conclusion, image enhancement can add or reduce model performance. A model trained solely on an enhanced image dataset lacks significant added performance.

ประกาศคุณูปการ

ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงในความกรุณาของ ผู้ช่วยศาสตราจารย์ ดร.ฐิติพงศ์ แก้วเหล็ก ประธานที่ปรึกษาวิทยานิพนธ์ ที่ได้สละเวลาอันมีค่ามาเป็นที่ปรึกษา พร้อมทั้งให้คำแนะนำตลอดระยะเวลาในการทำวิทยานิพนธ์ฉบับนี้ และขอกราบขอบพระคุณคณะกรรมการสอบวิทยานิพนธ์ อันประกอบไปด้วย รองศาสตราจารย์ ดร.กิติวัฒน์ คำวัน ประธานที่ปรึกษาวิทยานิพนธ์ และ ผู้ช่วยศาสตราจารย์ ดร.นันทวัฒน์ อู่ดี กรรมการผู้ทรงคุณวุฒิ ที่ได้กรุณาให้คำแนะนำตลอดจนแก้ไขข้อบกพร่องของวิทยานิพนธ์ด้วยความเอาใจใส่ จนทำให้วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้อย่างสมบูรณ์และทรงคุณค่า

เหนือสิ่งอื่นใดขอกราบขอบพระคุณ บิดา มารดา ของผู้วิจัยที่ให้กำลังใจและให้ การสนับสนุนในทุกๆ ด้านอย่างดีที่สุดเสมอมา

คุณค่าและคุณประโยชน์อันพึงจะมีจากวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบและอุทิศแด่ผู้มีพระคุณทุกๆ ท่าน ผู้วิจัยหวังเป็นอย่างยิ่งว่า งานวิจัยนี้จะเป็นประโยชน์ต่อผู้ที่สนใจในเรื่องการเรียนรู้เชิงลึกรวมถึงการกำหนดขอบเขตของมะเร็งไกลโอมาไม่มากก็น้อย

กศตพรรษ นาคแนม

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1

บทนำ

ความเป็นมาและความความสำคัญของปัญหา

เนื้องอกระบบประสาทส่วนกลาง (Central nervous System tumor, CNS) เป็นกลุ่มเนื้องอกที่มีความหลากหลายรวมถึงเนื้องอกที่มีความร้ายแรงและไม่ร้ายแรง ซึ่งแบ่งออกเป็นเนื้องอกที่สมองและเนื้องอกที่ไขสันหลัง ในปี 2020 พบว่าทั่วโลกมีผู้ป่วยที่เป็นเนื้องอกสมอง 308,102 ราย แบ่งเป็นเพศชายจำนวน 168,346 ราย และ เพศหญิงจำนวน 139,756 ราย (1) ในประเทศสหรัฐอเมริกาปี 2023 มีผู้ป่วยที่เป็นเนื้องอกสมอง 24,810 ราย (2) ในประเทศไทยปี 2020 มีผู้ป่วยที่เป็นเนื้องอกสมอง 187,677 ราย แบ่งเป็นเพศชายจำนวน 91,846 ราย และ เพศหญิงจำนวน 95,931 ราย (3) เนื้องอกที่พบมากที่สุดใน CNS tumor ในผู้ใหญ่ คือ ไกลโอมา (Glioma) โดย Glioma แบ่งออกเป็น High grade glioma (HGG) และ Low grade glioma (LGG) ซึ่ง High grade glioma เป็นเนื้องอกที่แสดงการพยากรณ์โรคทางคลินิกที่มีความรุนแรง และมีการลุกลามอย่างรวดเร็ว มีอัตราการรอดชีวิตเฉลี่ย 2 ปี หรือน้อยกว่า ซึ่งแตกต่างจาก LGG ที่ทีการพยากรณ์ที่ดีกว่าและมีการลุกลามน้อยหรือไม่ลุกลาม ไม่ว่าจะเป็น HGG หรือ LGG จะปริมาตรของเนื้องอกที่คล้ายคลึงกัน เช่น Whole tumor, Tumor core, และ Active tumor region (4, 5) ปัจจัยเสี่ยงที่ทำให้เกิดเนื้องอกสมองเกี่ยวข้องกับความบกพร่องทางพันธุกรรม (Inherited Genetic) (6) รวมถึงการได้รับรังสีและได้รับสารเคมีต่าง ๆ ข้อบ่งชี้ทั่วไปของผู้ป่วยที่มีเนื้องอกสมองจะมีอาการปวดหัว คลื่นไส้ และอาเจียน กรณีข้อบ่งชี้เฉพาะอาการของผู้ป่วยจะขึ้นอยู่กับตำแหน่งของเนื้องอก ตัวอย่างเช่นเนื้องอกอยู่บริเวณสมองส่วนหน้าผู้ป่วยจะเกิดความยากลำบากในการวางแผน ทำกิจกรรมต่าง ๆ พฤติกรรม บุคลิกภาพ และทักษะทางสังคมเปลี่ยนแปลงไป ถ้าเนื้องอกเกิดบริเวณก้านสมองผู้ป่วยจะมีปัญหาเกี่ยวกับการประสานงาน พูดลำบาก มองเห็นภาพซ้อน เป็นต้น (7) เมื่อแพทย์สงสัยว่าผู้ป่วยมีเนื้องอกในสมองแพทย์สามารถตรวจสอบได้หลายวิธี เช่น การตรวจการกลายพันธุ์ของยีนส์ IDH1/2, 1p19q, หรือ TP53 เป็นต้น นอกจากนี้เนื้องอกสมองสามารถตรวจสอบได้จากภาพถ่ายต่าง ๆ ไม่ว่าจะเป็นภาพที่ได้จากเครื่องเอกซเรย์คอมพิวเตอร์ (Computed Tomography, CT) เครื่องถ่ายภาพรังสีโพสิตรอนร่วมกับภาพเอกซเรย์คอมพิวเตอร์ (Positron Emission Tomography/CT, PET/CT) รวมถึงการถ่ายภาพด้วยคลื่นแม่เหล็กไฟฟ้า (Magnetic Resonance Imaging, MRI) จากที่กล่าวในข้างต้น HGG หรือ LGG จะมีปริมาตรของเนื้องอกที่คล้ายคลึงกัน ซึ่งส่งผลให้การถ่ายภาพ MRI มีหลากหลายประเภทตามไปด้วย เนื่องจากการถ่ายภาพ

MRI แบบปกติ (T1-weighted, T1) ไม่สามารถบ่งบอกขอบเขตของเนื้องอกได้อย่างชัดเจน ดังนั้นจึงมีการถ่ายภาพเอ็มอาร์ไอชนิด T1-contrast (T1c), T2-weighted (T2), และ Fluid Attenuated Inversion Recovery (FLAIR) เพื่อบ่งบอกถึงบริเวณของ 3 พื้นที่ (Whole tumor, Tumor cone, และ Activate tumor region) ตามที่กล่าวมาได้อย่างชัดเจนซึ่งการใช้ภาพเอ็มอาร์ไอในการระบุขอบเขตของเนื้องอกอย่างแม่นยำมีความสำคัญอย่างมากในการใช้งานทางคลินิกหลากหลายอย่าง เช่น การตรวจหาเนื้องอก (Tumor detection) การกำหนดตำแหน่งของเนื้องอก (Tumor delineation) การวางแผนการรักษา (Treatment planning) ใช้เป็นภาพนำวิถี (Image guided) หรือติดตามอาการหรือการเติบโตของเนื้องอกของผู้ป่วย (Monitoring tumor growth) อย่างไรก็ตามการหาขอบเขตของเนื้องอกด้วยตนเองใช้เวลานาน (Time-consuming) เนื่องจากการกำหนดขอบเขตไม่ได้ทำบนภาพ ๆ เดียว และขึ้นอยู่กับประสบการณ์ของนักรังสีแพทย์ โดยอาจทำให้เกิดข้อผิดพลาดของมนุษย์ (Human error) ระหว่างการหาขอบเขตของเนื้องอกได้ ด้วยเหตุผลข้างต้นการหาขอบเขตของเนื้องอกแบบอัตโนมัติจึงมีความจำเป็นในการใช้งานเพื่อลดปัญหาดังกล่าว

ในปัจจุบันเทคโนโลยีมีการพัฒนาที่สูงขึ้น โดยเฉพาะด้านปัญญาประดิษฐ์ สามารถนำไปประยุกต์ใช้ได้ในหลากหลายสาขาวิชารวมถึงในทางการแพทย์ เช่น การใช้ปัญญาประดิษฐ์เพื่อตรวจสอบชนิดของโรคมะเร็ง คาดการณ์ตำแหน่งของโรคมะเร็ง การแบ่งส่วนอัตโนมัติในการแบ่งส่วนเนื้องอก จากที่กล่าวในข้างต้นมีการใช้ศาสตร์เกี่ยวกับการเรียนรู้เชิงลึกเข้ามาพัฒนาองค์ความรู้ต่าง ๆ มากมาย โดยการเรียนรู้เชิงลึก (Deep learning) เป็นส่วนหนึ่งของศาสตร์ปัญญาประดิษฐ์ที่มีความยืดหยุ่นในการทำงานมากกว่าปัญญาประดิษฐ์แบบการเรียนรู้ของเครื่อง (Machine learning) การเรียนรู้เชิงลึกสามารถค้นหาคุณสมบัติภายในข้อมูลภาพได้อัตโนมัติ ด้วยเหตุผลนี้การเรียนรู้เชิงลึกจึงมีความนิยมในการนำใช้ในการแบ่งส่วนภาพทางการแพทย์ จากงานวิจัยของ Lin M และคณะ ใช้ปัญญาประดิษฐ์แบบการเรียนรู้เชิงลึก มีการใช้สถาปัตยกรรม U-net ร่วมกับ Context block และ Fivefold cross-validation เพื่อขยายการรับสัญญาณของโครงข่ายประสาทเทียมและเพิ่มความแม่นยำในการแบ่งส่วน สามารถหาค่าสัมประสิทธิ์ความคล้ายคลึง (Disc Similarity Coefficient, DSC) ได้เท่ากับ 0.887 (5) ขณะที่งานวิจัยของ Thong Vo และคณะ ใช้ปัญญาประดิษฐ์แบบการเรียนรู้เชิงลึก มีการใช้สถาปัตยกรรม UVR-net สามารถหาค่า DSC ได้เท่ากับ 0.760 (8) ขณะที่งานวิจัยของ Jain Rahul และคณะ ใช้ปัญญาประดิษฐ์แบบการเรียนรู้เชิงลึก มีการใช้สถาปัตยกรรมที่เสนอในฝั่งของตัวเข้ารหัสจะมีเพียงหนึ่งชั้น ขณะที่ฝั่งของตัวถอดรหัสมีหลายชั้น จากนั้นจะรวมผลลัพธ์ที่ได้จากฝั่งตัวถอดรหัสเข้าด้วยกัน สามารถหาค่า DSC ได้เท่ากับ 0.731 (9) ซึ่งผลลัพธ์เหล่านี้ขึ้นอยู่กับหลายปัจจัยไม่ว่าจะเป็นสถาปัตยกรรมที่ใช้ รายละเอียดต่าง ๆ ที่เกี่ยวข้องกับการสร้างโมเดลการเรียนรู้เชิงลึก รวมไปถึงคุณภาพของข้อมูลภาพที่นำมาฝึกฝนให้กับโมเดลการเรียนรู้เชิงลึกด้วย จากงานของวิจัยของ Güneş AM และคณะ แสดงให้เห็นว่าคุณภาพของภาพส่งผลต่อคุณภาพของ

โมเดลการแบ่งส่วน โดยชุดข้อมูลที่มีสัญญาณรบกวน (Noise and artifact) เพิ่มขึ้น เมื่อนำไปฝึกฝนโมเดลเป็นจำนวนมากส่งผลให้โมเดลมีประสิทธิภาพเพิ่มขึ้น (10) ซึ่งตามความจริงเป็นจริงคุณภาพของภาพที่ดีควรทำให้โมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนมีประสิทธิภาพเพิ่มมากขึ้น ดังนั้นผู้วิจัยจึงมีความสนใจในศึกษาการเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก โดยมีวัตถุประสงค์เพื่อเปรียบเทียบผลของกระบวนการปรับปรุงสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองและพัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง

จุดมุ่งหมายของการวิจัย

1. เพื่อเปรียบเทียบผลของกระบวนการก่อนการประมวลผลภาพเพื่อเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง
2. เพื่อพัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง

ขอบเขตของงานวิจัย

เปรียบเทียบกระบวนการก่อนการประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยใช้อัลกอริทึมสำหรับการปรุงภาพทั้งหมด 4 เทคนิค ได้แก่ Contrast-Limited Adaptive Histogram Equalization (CLAHE), Gamma Correction (GC), Non-Local mean filter (NLMF), และ Median and Wiener filter (MWF) พร้อมทั้งใช้รูปสถาปัตยกรรม U-net, UNet++ และสถาปัตยกรรมที่ผู้วิจัยออกแบบ โดยใช้ชุดข้อมูลภาพ Brain Tumor Segmentation 2023 Dataset (BraTS2023) (4, 11-14) จากฐานข้อมูล synapse ซึ่งเป็นข้อมูลภาพเนื้องอกชนิดไกลโอมา ประกอบด้วยชุดข้อมูลสำหรับการฝึกฝน จำนวน 1,350 ชุดข้อมูล สำหรับการตรวจสอบจำนวน 1,050 ชุดข้อมูล และสำหรับการทดสอบจำนวน 600 ชุดข้อมูล โดยชุดข้อมูลสำหรับการฝึกฝนแบ่งเป็นภาพเอ็มอาร์ไอชนิด T1, T1c, T2, FLAIR, และภาพ Mask และการตรวจสอบคุณภาพของการปรับปรุงภาพจากจะประเมินค่า Structural Similarity Index (SSIM), Mean Squared Error (MSE), และ Peak Signal-to-Noise Ratio (PSNR) และเปรียบเทียบความเหมือนของขอบเขตก้อนมะเร็งด้วยค่า Dice similarity coefficient (DSC) และ Jaccard similarity coefficient (IoU) และประเมินประสิทธิภาพของโมเดลจากค่า Accuracy, Precision, และ Recall

สมมุติฐานของงานวิจัย

1. กระบวนการก่อนการประมวลผลภาพช่วยเพิ่มประสิทธิภาพของโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง

2. โมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองที่พัฒนาขึ้น สามารถกำหนดขอบเขตของมะเร็งไกลโอมาในสมองแบบอัตโนมัติได้ดี

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

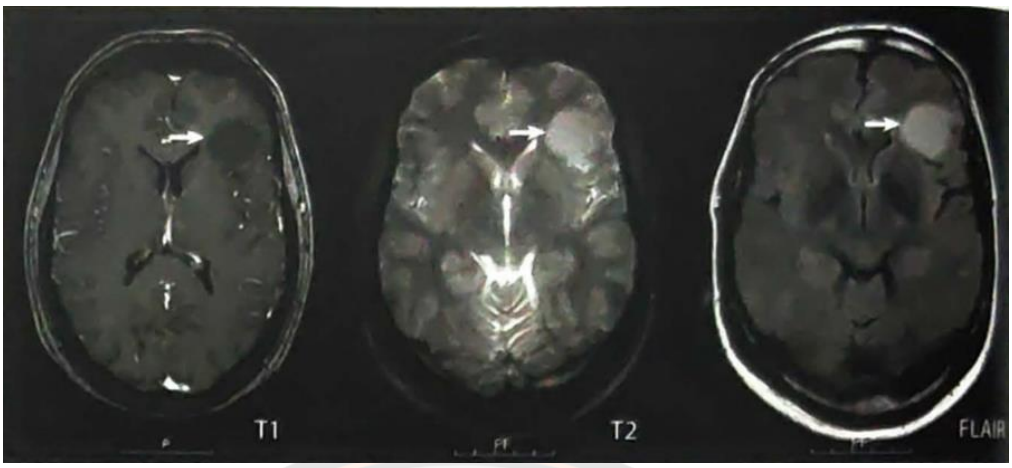
เอกสารที่เกี่ยวข้อง

1. ไกลโอมา (Glioma)

Glioma คือ เนื้องอกชนิดหนึ่งของเซลล์สมอง เรียกว่าเซลล์เกลีย (Glia cell) เป็นเนื้องอกพบมากที่สุดในมะเร็งระบบประสาทส่วนกลางในผู้ใหญ่ เป็นเนื้องอกชนิดที่ไม่ใช่มะเร็งและเนื้องอกชนิดมะเร็ง ซึ่งเกิดมาจากการได้รับความเสียหาย การกลายพันธุ์บริเวณเซลล์ห่อหุ้มประสาทต่าง ๆ เช่น เซลล์แอสโตรไซต์ (Astrocyte) หรือ โอลิโกเด็นโดรไซต์ (Oligodendrocyte) โดยเนื้องอกไกลโอมาส่วนใหญ่จะอยู่ในกลุ่มความรุนแรงสูง (High grade) มากกว่ากลุ่มความรุนแรงต่ำ (Low grade) อุบัติการณ์ในประเทศไทย พบผู้ป่วยที่เป็น CNS tumor โดยแบ่งเป็นเพศชาย 2.4 รายต่อประชากรไทย 1 แสนคน และในเพศหญิง 2.1 รายต่อประชากรไทย 1 แสนคน (15-17) ปัจจัยเสี่ยงที่ทำให้เกิดเนื้องอกไกลโอมาเกี่ยวข้องกับพันธุกรรม ผู้ป่วยที่เคยได้รับรังสีรักษาบริเวณสมอง โดยอาการแสดงของผู้ป่วยที่มีเนื้องอกไกลโอมาที่พบบ่อย ได้แก่ อาการชัก ปวดศีรษะ แขนขาอ่อนแรง คลื่นไส้และอาเจียน สำหรับการวินิจฉัยทำได้โดยการตรวจชิ้นเนื้อหรือพิจารณาจากภาพถ่ายทางการแพทย์ไม่จะเป็นภาพ MRI CT หรือ PET/CT เป็นต้น เพื่อจำแนกความรุนแรงของตัวโรคและใช้สำหรับการเลือกวิธีการรักษาที่เหมาะสมแก่ผู้ป่วย เนื้องอกไกลโอมา แบ่งเป็นหลายชนิด ซึ่งที่พบได้บ่อย ได้แก่ มะเร็งสมองแอสโตรไซต์โตมา (Astrocytoma) มะเร็งสมองโอลิโกเด็นโดรไกลโอมา (Oligodendroglioma) มะเร็งสมองไกลโอบลาสโตมา (Glioblastoma)

1.1 Astrocytoma

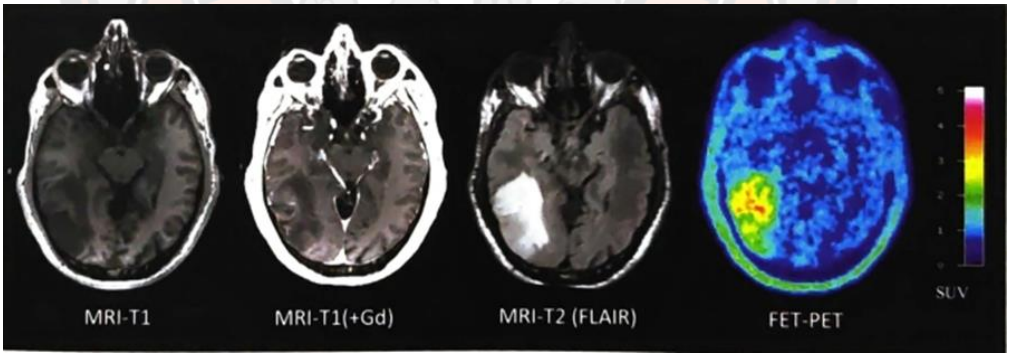
เป็นโรคที่เกิดมาจากเซลล์ Astrocyte สามารถพบได้ทั้งความรุนแรงต่ำและความรุนแรงสูง (WHO grade 1-4) การรักษาหลักของมะเร็งชนิดนี้ได้แก่การผ่าตัด ในกรณีที่โรคมีความรุนแรงสูงจำเป็นต้องดีรับการรักษาด้วยการฉายรังสีและยาเคมีบำบัดเพิ่มเติมหลังจากการผ่าตัด



ภาพ 1 ลักษณะของ astrocytoma บนภาพ MRI ชนิด T1 T2 และ FLAIR ตามลำดับ (18)

1.2 Oligodendroglioma

เป็นโรคที่เกิดจากเซลล์ Oligodendrocyte สามารถพบได้ทั้งความรุนแรงต่ำและความรุนแรงสูง (WHO grade 2, 3) การรักษาหลักของมะเร็งชนิดนี้ได้แก่การผ่าตัด ในกรณีที่โรคมีความรุนแรงสูงจำเป็นต้องดีรับการรักษาด้วยการฉายรังสีและยาเคมีบำบัดเพิ่มเติมหลังจากการผ่าตัด

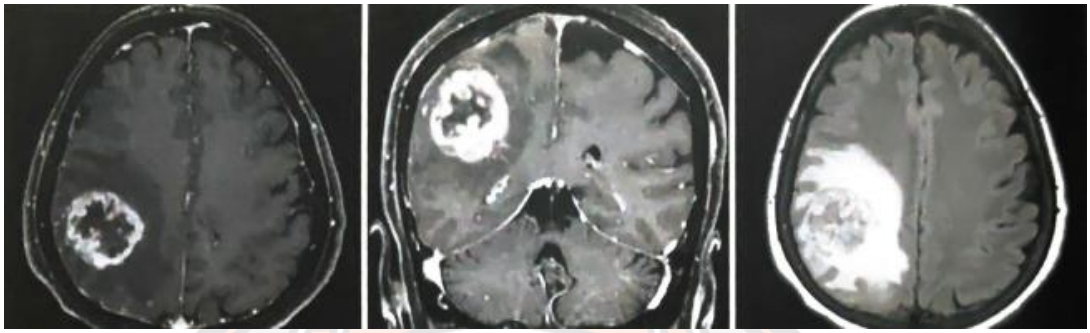


ภาพ 2 ลักษณะของ oligodendroglioma บนภาพ MRI ชนิด T1 T1Gd FLAIR และภาพ PET/CT ตามลำดับ (18)

1.3 Glioblastoma

เป็นโรคที่มีความรุนแรงสูงสุด (WHO grade 4) เป็นโรคที่พบได้สูงที่สุดของเนื้องอกสมอง เนื่องจากลักษณะของโรคจะรุกรานไปตามเนื้อเยื่อข้างเคียงทำให้ยากต่อการผ่าตัดดังนั้น

การรักษาด้วยการฉายรักษาและให้ยาเคมีบำบัดจึงเหมาะสมกว่าการผ่าตัด โดยส่วนใหญ่จะทำการรักษาด้วยการฉายรังสีพร้อมการให้ยาเคมีบำบัด (Concurrent chemoradiotherapy) และตามด้วยการให้ยาเคมีบำบัดหลังฉายเสร็จ (Adjuvant chemotherapy) (19)



ภาพ 3 ลักษณะของ glioblastoma บนภาพ MRI (18)

เนื้องอกในสมองสามารถเป็นได้ทั้งชนิดที่เป็นมะเร็งและไม่ใช่มะเร็ง และอาจมีขนาดตั้งแต่เล็กไปจนถึงใหญ่ อาการต่าง ๆ ได้แก่ อาการชัก ปวดศีรษะ แขนขาอ่อนแรง คลื่นไส้และอาเจียน ตัวเลือกการรักษาขึ้นอยู่กับความรุนแรงของโรค โดยการรักษาหลักคือการผ่าตัด และรักษาเสริมด้วยการฉายรังสีและเคมีบำบัด ปัจจัยที่ส่งผลต่อการเกิดเนื้องอกในสมองขึ้นอยู่กับกรรมพันธุ์ และการเคยได้รับรังสี โดยการตรวจพบและการรักษาประจำปี สามารถลดการเกิดโรคมะเร็ง และทีมผู้เชี่ยวชาญสามารถกำหนดแผนการรักษาที่ดีที่สุดสำหรับแต่ละบุคคลได้

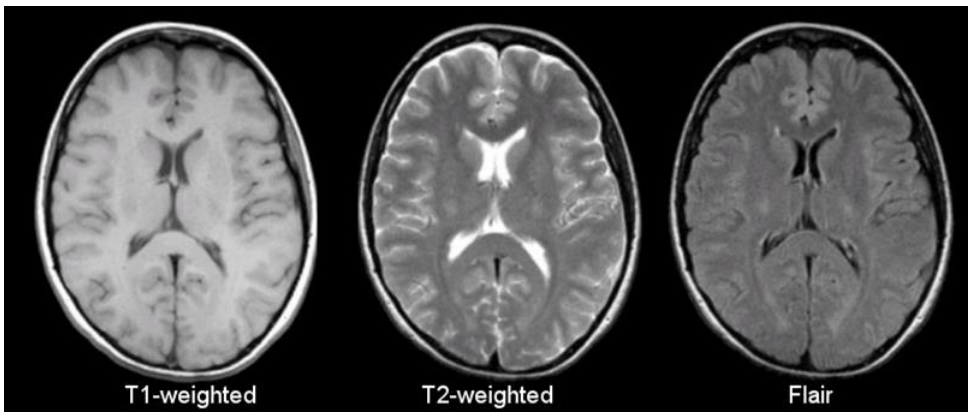
2. การถ่ายภาพด้วยคลื่นสนามแม่เหล็ก (MRI)

MRI เป็นเทคโนโลยีการถ่ายภาพที่ไม่ก่อให้เกิดอันตราย (Non-invasive) และเป็นเครื่องมือสำคัญที่สำคัญสำหรับการแพทย์ในปัจจุบัน มีประโยชน์ในการวินิจฉัยและติดตามโรคที่ส่งผลต่อเนื้อเยื่ออ่อน (Soft tissue) ของร่างกาย เช่น สมอง ไขสันหลัง และอวัยวะต่าง ๆ ซึ่งแตกต่างจากรังสีเอกซ์และการตรวจเอกซเรย์คอมพิวเตอร์ (CT) ซึ่งใช้รังสีไอออไนซ์ MRI ใช้สนามแม่เหล็กและคลื่นวิทยุเพื่อสร้างภาพที่มีรายละเอียดของร่างกายทำให้ MRI ปลอดภัยสำหรับผู้ป่วย โดยเฉพาะผู้ที่ต้องถ่ายภาพมากกว่าปกติ เช่น ผู้ที่มีภาวะทางระบบประสาทหรือมะเร็ง

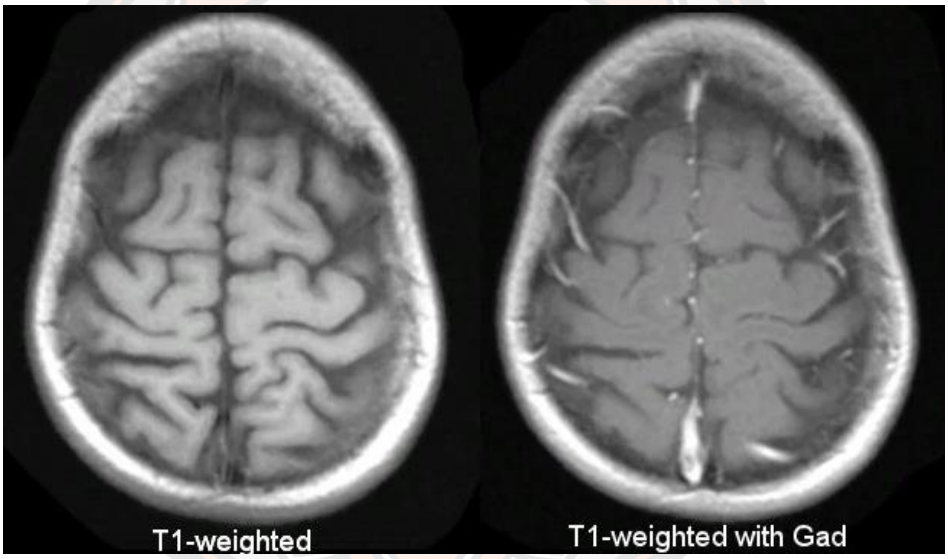
MRI มีประโยชน์ในการวินิจฉัยโรคเกี่ยวกับสมองและไขสันหลัง เนื่องจากสามารถสร้างภาพความละเอียดสูงของระบบประสาทได้ สามารถตรวจจับการเปลี่ยนแปลงของโครงสร้างสมอง เช่น เนื้องอก โรคเส้นเลือดสมองโป่งพอง (Aneurysms) และลิ่มเลือด ตลอดจนการเปลี่ยนแปลงของ

การทำงานของสมอง เช่น โรคอัลไซเมอร์ โรคปลอกประสาทเสื่อมแข็ง และโรคหลอดเลือดสมอง นอกเหนือจากการใช้งานด้านระบบประสาทแล้ว MRI ยังใช้เพื่อวินิจฉัยและติดตามสภาวะต่าง ๆ รวมถึงการบาดเจ็บของกล้ามเนื้อ โรคหัวใจและหลอดเลือด และมะเร็ง สามารถบ่งบอกข้อมูลเกี่ยวกับขนาด ตำแหน่ง และขอบเขตของเนื้องอก และสามารถใช้เพื่อเป็นแนวทางในการตรวจชิ้นเนื้อและขั้นตอนอื่น ๆ ได้ แม้ว่า MRI จะเป็นการถ่ายภาพที่ปลอดภัยและ Non-invasive แต่มีข้อจำกัดและความเสี่ยงบางประการที่เกี่ยวข้องกับเทคโนโลยีนี้ ผู้ป่วยที่มีอุปกรณ์ทางการแพทย์บางอย่าง เช่น เครื่องกระตุ้นหัวใจหรือประสาทหูเทียม อาจไม่สามารถเข้ารับการตรวจ MRI ได้ เนื่องจากอุปกรณ์มีส่วนประกอบของโลหะ นอกจากนี้ สนามแม่เหล็กยังทำให้เกิดความรู้สึกไม่สบายและโรคกลัวที่แคบสำหรับผู้ป่วยบางราย

ภาพที่ได้จากการถ่ายภาพ MRI มีหลายประเภท ซึ่งเป็นชุดของคลื่นความถี่วิทยุและการไล่ระดับของสนามแม่เหล็ก เพื่อสร้างความคมชัด (Contrast) ประเภทต่างๆ ในภาพ เพื่อเน้นคุณลักษณะต่างๆ ของเนื้อเยื่อและโครงสร้างในร่างกายที่สนใจ ซึ่งประเภทของภาพ MRI จะพิจารณาจากค่า Repetition time (TR) หรือ ระยะเวลาจากการให้ความถี่สั่นพ้อง (Resonance frequency, RF) ไปกระตุ้นครั้งแรกจนถึงการให้ RF กระตุ้นในครั้งถัดไป และ ค่า Echo time (TE) หรือ ระยะเวลาจากการให้ RF เข้าไปกระตุ้นครั้งแรกจนถึงเวลาที่ทำให้เกิดสัญญา โดยประเภทของภาพ MRI ที่พบบ่อย ได้แก่ 1. การถ่ายภาพแบบปกติ (T1-weighted, T1) โดยการถ่ายภาพประเภทนี้ให้รายละเอียดทางกายวิภาคของเนื้อเยื่อปกติได้ดี และมักใช้เพื่อตรวจหาความผิดปกติในเนื้อเยื่ออ่อน ซึ่งภาพ MRI ชนิด T1 ถูกสร้างขึ้นโดยใช้ TR ยาว และ TE สั้น ดังนั้นภาพ MRI ชนิด T1 มีลักษณะของภาพที่แสดงจะสว่างสำหรับไขมันและมืดสำหรับน้ำ 2. การถ่ายภาพที่ถ่วงน้ำหนัก T2 (T2-weighted, T2) โดยการถ่ายภาพที่ถ่วงน้ำหนัก T2 ใช้เพื่อเน้นความผิดปกติในโครงสร้างที่มีของไหล เช่น สมองและไขสันหลัง ภาพ MRI ชนิด T2 สร้างขึ้นโดยใช้ TR และ TE แบบยาว ซึ่งภาพ MRI ชนิด T2 มีลักษณะของภาพที่แสดงจะสว่างสำหรับน้ำและมืดสำหรับไขมัน 3. Fluid Attenuated Inversion Recovery (FLAIR) imaging โดย FLAIR เป็นประเภทหนึ่งของการถ่ายภาพ MRI ชนิด T2 ที่ยับยั้งสัญญาณจากของไหล ทำให้มองเห็นรอยโรคใกล้กับน้ำไขสันหลังได้ดีขึ้น การถ่ายภาพ FLAIR ใช้เวลาในการผกผัน TR และ TE ยาว



ภาพ 4 ตัวอย่างภาพถ่าย MRI ชนิด T1 T2 และ FLAIR ตามลำดับ (20)



ภาพ 5 ตัวอย่างภาพถ่าย MRI ชนิด T1 และ T1Gd (20)

MRI เป็นเทคโนโลยีการถ่ายภาพทางการแพทย์ขั้นสูงที่ให้ภาพที่มีรายละเอียดสูงของโครงสร้างภายในร่างกายโดยไม่ต้องใช้รังสีไอออไนซ์ แม้ว่าจะมีข้อดีหลายประการ เช่น การแสดงลักษณะของเนื้อเยื่ออ่อนที่เหนือกว่า CT Images แต่ก็มีข้อจำกัดบางประการ เช่น ค่าใช้จ่ายสูง ต้องใช้อุปกรณ์เฉพาะทางและบุคลากรที่ผ่านการฝึกอบรม ที่สำคัญผู้ป่วยที่มีอุปกรณ์หรือโลหะฝังอยู่ในร่างกายอาจไม่สามารถเข้ารับการตรวจ MRI ได้

3. ปัญญาประดิษฐ์ (Artificial Intelligence, AI)

ปัญญาประดิษฐ์เป็นศาสตร์แขนงหนึ่งที่เกี่ยวกับการศึกษาวิจัยและพัฒนาระบบคอมพิวเตอร์จากคุณลักษณะของข้อมูลที่นำมาสอนให้มีความสามารถในการจดจำ แยกแยะ และประมวลผลข้อความ ภาพ เสียง เช่น การสร้างหุ่นยนต์ สร้างสมองกล สร้างระบบประสาทรับรู้เลียนแบบมนุษย์ สร้างระบบการมองเห็น การช่วยตัดสินใจ คาดการณ์ ทำนาย หรือทำสิ่งต่าง ๆ ที่มีการตัดสินใจแทนมนุษย์ ฯลฯ

3.1 การเรียนรู้ของเครื่อง (Machine learning, ML)

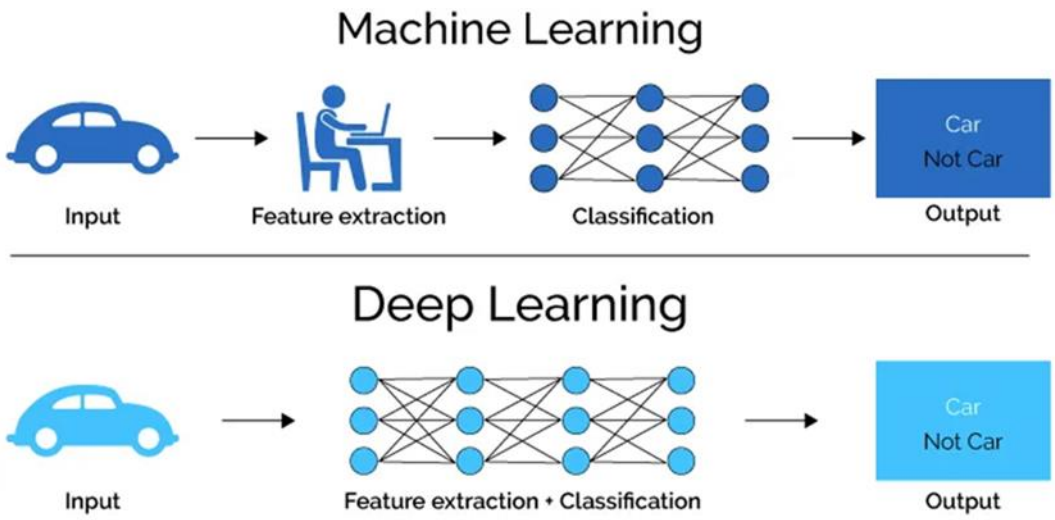
เป็นศาสตร์ย่อยของปัญญาประดิษฐ์ที่ทำให้คอมพิวเตอร์หรือสมองกลเกิดการเรียนรู้จากข้อมูลหรือกระบวนการเรียนรู้ด้วยเครื่อง แล้วนำความรู้นั้นมาใช้งาน วิเคราะห์คาดการณ์หรือทำสิ่งต่าง ๆ ให้กับปัญญาประดิษฐ์ เช่น แสดงข้อมูล แนะนำ ตัดสินใจ ควบคุมสิ่งต่าง ๆ ฯลฯ

3.2 การเรียนรู้เชิงลึก (Deep learning, DL)

เป็นศาสตร์ย่อยของการเรียนรู้ของเครื่อง ที่มีจุดประสงค์เดียวกับการเรียนรู้ของเครื่อง คือทำให้คอมพิวเตอร์เกิดการเรียนรู้ แล้วนำความรู้มาใช้งาน อย่างไรก็ตามการเรียนรู้เชิงลึกใช้วิธีการหรือเทคนิคที่มีลักษณะโครงข่ายประสาทเทียมที่มีความลึกหลายชั้น (Deep Neural Network, DNN) โดยมีการเลียนแบบจากการทำงานของโครงข่ายของสมองมนุษย์

3.3 ความแตกต่างระหว่างการเรียนรู้ของเครื่องและการเรียนรู้เชิงลึก

การเรียนรู้ของเครื่องมีความแตกต่างตรงที่ส่วนของโปรแกรมที่ทำการคำนวณประมวลผล การกำหนดเงื่อนไขของโปรแกรม การตัดสินใจต่าง ๆ ถูกสร้างโดยคอมพิวเตอร์ โดยผู้พัฒนาจะนำชุดข้อมูล (Dataset) ทั้งที่เป็นส่วน Data และ Output เข้าไปให้คอมพิวเตอร์เรียนรู้เพื่อให้คอมพิวเตอร์สร้างโมเดล (Model) หรือสร้างสมองขึ้นมา จากนั้นผู้พัฒนาเขียนโปรแกรมหลักเพิ่มเติม จะได้โปรแกรมที่สมบูรณ์ที่สามารถรับข้อมูลนำเข้า (Input) ประมวลผล และได้ผลลัพธ์ (Output) ออกมา ขณะที่การเรียนรู้เชิงลึกใช้โครงข่ายประสาทเทียมที่มีความลึกหลายชั้นเลียนแบบจากการทำงานของโครงข่ายของสมองมนุษย์ โดยทำการเรียนรู้และค้นหาคุณลักษณะเด่น (Feature) ด้วยเครือข่ายหลายชั้น ซึ่งมนุษย์ไม่จำเป็นต้องเพิ่ม Feature นั้น

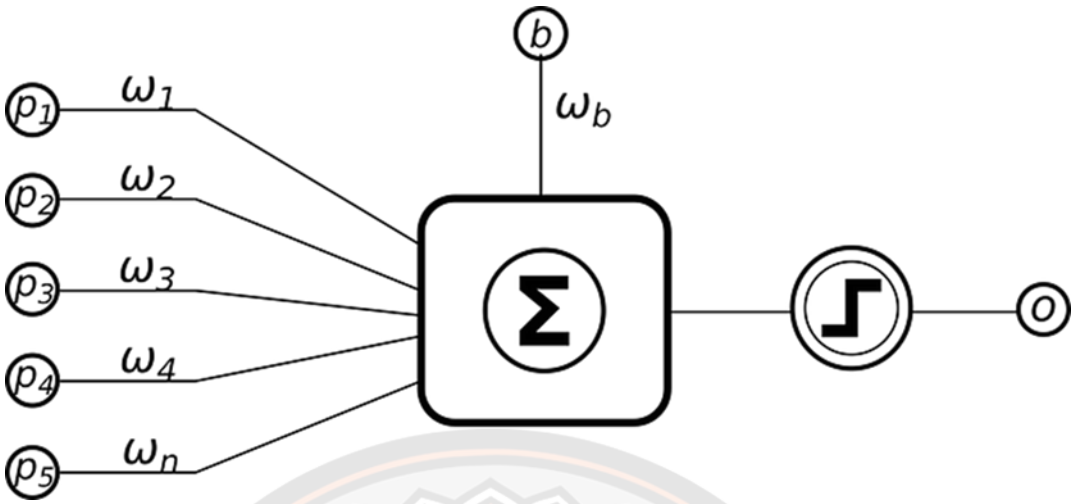


ภาพ 6 กระบวนกแรทำงานของ Machine learning และ Deep learning (21)

3.4 Artificial Neuron และ Perceptron

Artificial Neuron เป็นเซลล์ประสาทที่มนุษย์สร้างขึ้น โดยอาศัยหลักการหรือเลียนแบบการทำงานของเซลล์ประสาทจริง (Biological neuron) ซึ่งในทางคอมพิวเตอร์เรียกเซลล์ประสาทที่สร้างขึ้นว่า ประสาทเทียม (Artificial Neuron, AN) โดย AN เป็นโปรแกรมคอมพิวเตอร์ที่เขียนขึ้นทำหน้าที่คล้ายเซลล์ประสาทจริง คือรับ Input (x_1, x_2, ..., x_n) แล้วทำการประมวลผลหรือตัดสินหรือทำนาย (Predict) จากนั้นให้ผลลัพธ์ Output ออกมา โดยหน่วย Neuron นี้เรียกว่า เพอร์เซปทรอน (Perceptron) ซึ่งผลลัพธ์จะเป็น 1 หรือ 0 ซึ่งค่า Output ดังกล่าวอาจแทนผลว่าใช่หรือไม่

Perceptron สามารถทำงานประมวลผลได้ ตัดสินได้ จะมีส่วนประกอบที่ ทำหน้าที่สองส่วน คือ ผลรวมของข้อมูลนำเข้า (Sum) และ ฟังก์ชันการตัดสินใจ (Activation function) ในส่วนของ Input ที่รับค่าจาก Feature แต่ละตัวจะมีค่าน้ำหนัก (Weight, w_n) เป็นตัวกำหนดน้ำหนักของแต่ละ Input เพื่อกำหนดความสำคัญของ Input แต่ละตัว นอกจากนี้ยังมีค่า w_0 เรียกว่า Bias เป็นค่าช่วยปรับในการคำนวณตัดสิน



ภาพ 7 ลักษณะของ Perceptron (22)

3.5 ผลรวมของข้อมูลนำเข้า (Sum or Net Input function)

การรวมผลของ Perceptron ทำโดยนำค่า Input (x_n) คูณกับ Weight (w_n) ของแต่ละตัว จาก นั้นนำผลลัพธ์มารวมกัน โดยรวมค่า w_0 หรือ Bias ด้วย เมื่อได้ผลรวมข้อมูลจะถูกส่งไปตัดสินในขั้นต่อไป โดย Sum แสดงดังสมการ

$$s = w_0 + (w_1 x_1) + (w_2 x_2) + .. + (w_n x_n) \quad (1)$$

3.6 ฟังก์ชันตัดสินใจ (Activation function)

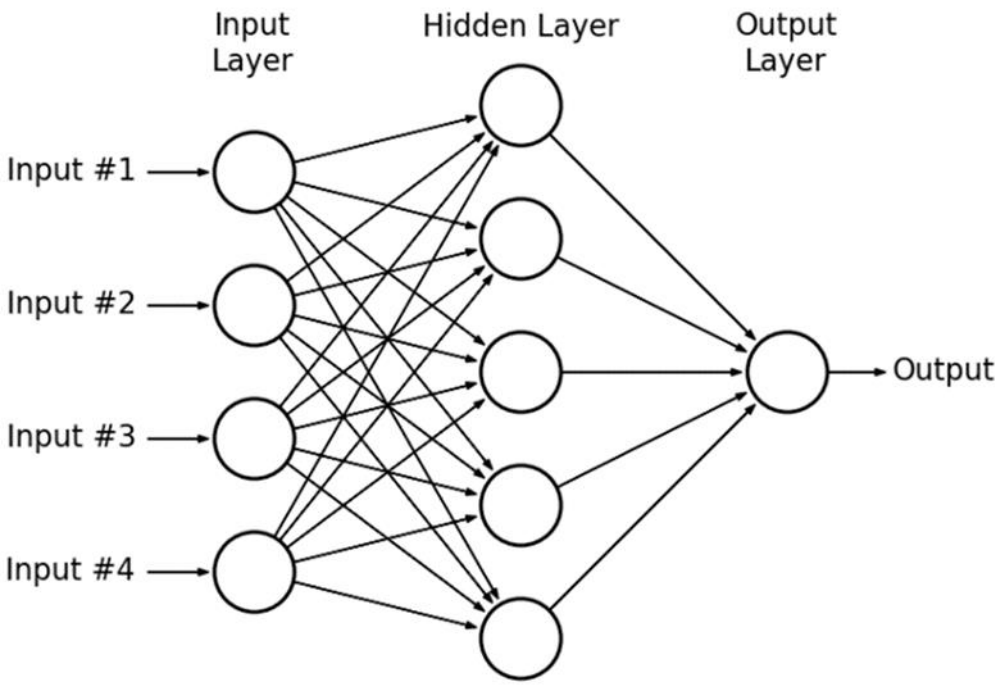
ค่าผลรวมหรือ Sum ก่อนหน้า จะนำมาประมวลผลตัดสิน โดยขั้นจะมีค่าตัวเลขที่ใช้เป็น ค่าเกณฑ์ตัดสิน ที่เรียกว่า Threshold ซึ่งค่านี้จะกำหนดให้ใช้ค่าเท่าใดก็ได้ ขึ้นอยู่กับการออกแบบ การทำงานในขั้นตอนนี้คือ นำค่าผลรวมที่ได้จาก Sum มาเปรียบเทียบกับค่า Threshold (t) ถ้าผลรวมมีค่าเท่ากับหรือสูงกว่าค่า Threshold การตัดสิน จะให้ผลลัพธ์ Output เป็น 1 แต่ถ้าค่า Sum น้อยกว่า Threshold ผลการตัดสินจะให้ Output 0 เช่น กำหนดให้ Threshold (t) เท่ากับ 0.5 ถ้าผลรวม s เท่ากับ 0.7 จะได้ผลลัพธ์ Output หรือ y เท่ากับ 1 ถ้าผลรวม s เท่ากับ -0.2 จะได้ผลลัพธ์ Output หรือ y = 0

3.7 Batch and Epoch size

Batch size คือการแบ่งข้อมูลไปคำนวณจำนวน n ครั้ง เนื่องจากการนำข้อมูลจำนวนมากเข้าสู่การคำนวณในครั้งเดียวทำให้หน่วยความจำประมวลผล (Random-Access Memory, RAM) ไม่เพียงพอต่อการรับข้อมูล และ Epoch size คือรอบในการฝึกฝนข้อมูล ซึ่งมีผลต่อความแม่นยำของโมเดล

3.8 เพอร์เซปทรอนหลายชั้น (Multi-Layer Perceptron, MLP)

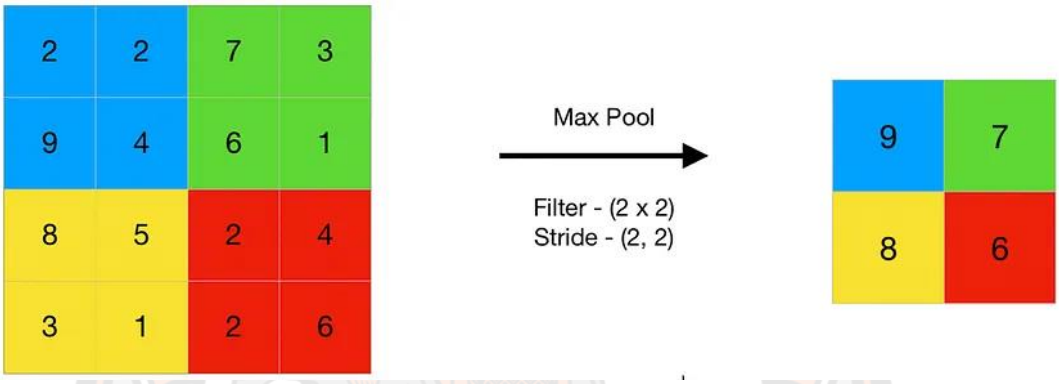
โครงสร้าง MLP ประกอบด้วยชั้นเลเยอร์ (Layer) ต่าง ๆ โดยแต่ละชั้นจะประกอบด้วยโหนด (Node) โดยทั้งหมดนี้ เรียกว่า Model โดยชั้นที่นำเข้าข้อมูล (Input Layer) เป็นส่วนที่รับ Input ซึ่งก็คือรับค่า Feature โดยแต่ละ Node ในชั้นนี้จะไม่มีตัวตัดสิน หรือ ไม่มี Activation function จะทำหน้าที่รับข้อมูลส่งเข้าไปประมวลผลต่อในชั้นประมวลผลที่ซ่อนอยู่ (Hidden Layer) เป็น Perceptron หรือ Neuron รับค่ามาจาก Input Layer ทำการรวมผลแล้วตัดสิน คล้ายกับ Perceptron แต่ผลการตัดสินหรือ Output ที่ได้จาก Hidden Layer นี้ ยังไม่ใช่เป็น Output สุดท้าย แต่จะส่งให้ Perceptron อีกชั้นประมวลผลต่อไป ซึ่งการที่มีชั้น Hidden Layer ทำให้เส้นตัดสินสามารถมีลักษณะเหมือนกับการผสมผสานช่วยให้สามารถตัดแบ่งหรือจำแนกกลุ่มข้อมูลที่ซับซ้อนได้ ชั้นผลลัพธ์ (Output Layer) เป็นส่วนที่รับข้อมูลต่อจาก Hidden Layer แล้วประมวลผล จากนั้นให้ค่าผลลัพธ์เอาต์พุตสุดท้าย กล่าวคือ Output Layer ทำหน้าที่รวมและตัดสินขั้นสุดท้าย



ภาพ 8 ลักษณะของ Multi-Layer Perceptron (23)

3.9 พูลลิ่ง (Pooling)

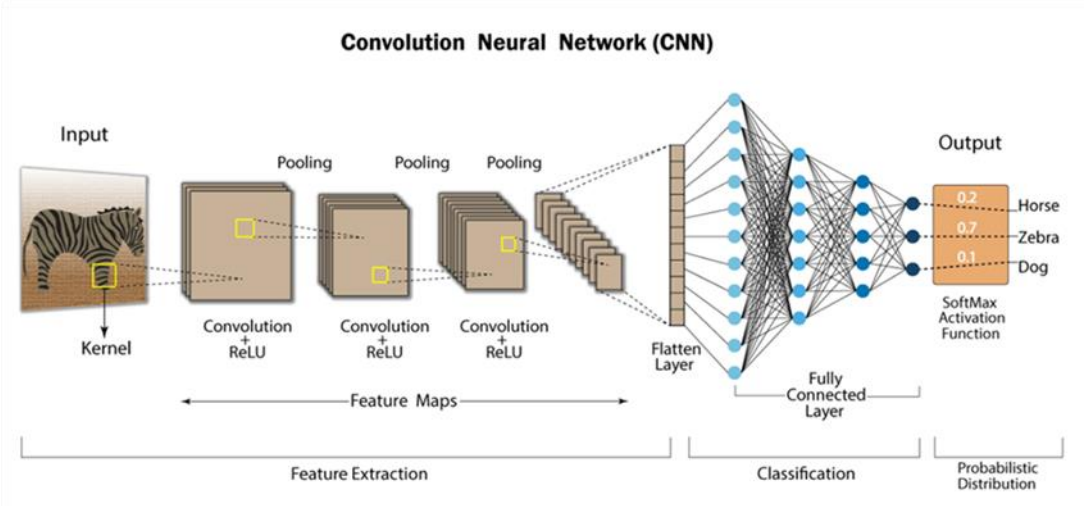
พูลลิ่งเป็นการย่อหรือลดขนาดรูปภาพ (Down Sampling) จึงช่วยในการลดมิติของคุณลักษณะของภาพ ลดจำนวนพารามิเตอร์ในการเรียนรู้ ลดปริมาณการคำนวณในระบบ เนื่องจากระบบไม่จำเป็นต้องใช้ภาพขนาดใหญ่ในการคำนวณก็สามารถแยกแยะได้ว่าวัตถุในภาพคือสิ่งใด ดังนั้น Pooling layer จะรวมคุณลักษณะของภาพ และเพิ่มประสิทธิภาพในการประมวลผลให้เร็วยิ่งขึ้น ซึ่งตัวอย่างของการทำ Pooling แสดงดังภาพ 9



ภาพ 9 Max pooling (24)

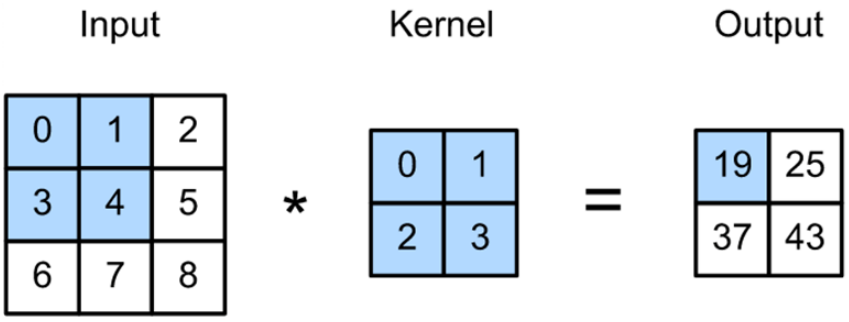
3.10 โครงข่ายประสาทแบบคอนโวลูชัน (Convolutional Neural Network)

Convolutional Neural Network หรือ CNN คือโครงข่ายประสาทเทียมที่มีแนวคิดมาจากการรับรู้ของระบบประสาทการมองเห็นของมนุษย์ที่มองวัตถุหรือภาพเป็นพื้นที่ย่อย ๆ ซึ่งการมองเห็นพื้นที่ย่อยของมนุษย์จะทำการแยกคุณลักษณะเด่นเอาไว้เพื่อประกอบการพิจารณาแล้วนำกลุ่มลักษณะเด่นของพื้นที่ย่อยมารวมกัน เพื่อให้ทราบรายละเอียดของวัตถุหรือภาพนั้น ซึ่ง CNN ประกอบด้วยองค์ประกอบหลัก 2 ส่วน คือ การค้นหาคุณลักษณะเด่น (Feature Extraction) เป็นขั้นตอนที่ทำการแยกคุณลักษณะเด่นของวัตถุที่อยู่ภายในภาพ และ โครงข่ายประสาทเทียม (Neural Network) เป็นส่วนที่ทำการจำแนกข้อมูล หรือ Classification ก่อนที่จะให้คำตอบ หรือ Output ออกมา โดย CNN มีลักษณะดังภาพ 10

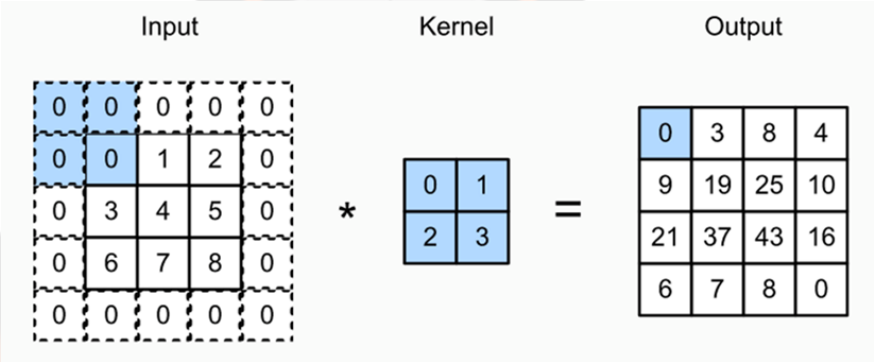


ภาพ 10 สถาปัตยกรรม CNN (25)

จะเห็นว่า CNN มีความลึกของชั้นหลายชั้น ซึ่งประกอบไปด้วยชั้น Input ซึ่งเป็นชั้นที่นำเข้าภาพเข้าสู่ระบบ โดยจะเป็นภาพสีเทา (Grayscale) หรือ ภาพสี เพื่อใช้สำหรับการฝึก ตรวจสอบ ทดสอบ และทำนาย ถัดมาคือชั้นคอนโวลูชัน (Convolution) เป็นชั้นที่นำข้อมูล Input มาผ่าน Kernel หรือ เมทริกซ์ข้อมูล ในลักษณะพิกเซลต่อพิกเซลดังภาพ 11 การทำ Convolution นั้นจะส่งผลให้มิติของภาพนั้นลดลง ซึ่งแก้ไขได้ด้วยการเพิ่มขอบเข้าไปก่อนการทำ Convolution เรียกว่า Padding ดังภาพ 12 ซึ่งระหว่างการทำ Convolution จะมีการกำหนดการเคลื่อนที่ของ Kernel โดยกำหนดจำนวนช่องที่เลื่อนไปในกระบวนการ Convolution แต่ละครั้ง (Stride) ตัวอย่างเช่น Stride = (1, 1) หมายความว่า Kernel เคลื่อนที่ในแนวระดับและแนวดิ่งครั้งละ 1 ช่อง ถัดมาคือชั้น Pooling เป็นชั้นที่ทำการลดขนาดของภาพลงครึ่งหนึ่ง ซึ่งส่งผลให้การคำนวณรวดเร็วขึ้น ขณะที่ยังรักษาคุณลักษณะเด่นของภาพไว้ จากนั้นภาพจะเข้าสู่ชั้น Flatten layer เพื่อแปลงข้อมูลให้อยู่ในลักษณะเมทริกซ์ 1 มิติ ก่อนเข้าสู่ชั้น Fully connected layer เพื่อทำการจำแนกข้อมูลก่อนให้ผลลัพธ์ Output ออกมา



ภาพ 11 การทำ Convolution (26)



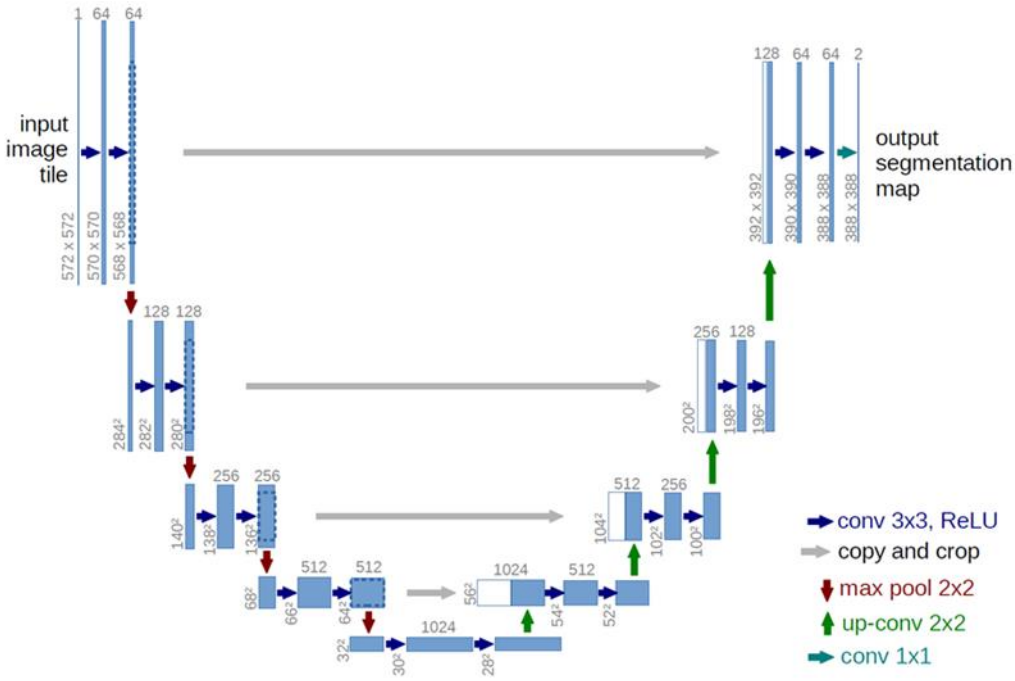
ภาพ 12 การทำ Padding (26)

4. สถาปัตยกรรม U-net (U-net architecture)

สถาปัตยกรรม U-Net เป็นโครงข่ายประสาทเทียมเต็มรูปแบบ (Fully Convolutional Neural Network) ที่ใช้กันทั่วไปสำหรับงานแบ่งส่วนภาพชีวการแพทย์ เปิดตัวครั้งแรกในปี 2015 โดย Ronneberger และคณะ และได้กลายเป็นตัวเลือกยอดนิยมสำหรับงานแบ่งส่วนจำนวนมากทั้งในการวิจัยและอุตสาหกรรม โดยรูปแบบสถาปัตยกรรมนี้ มีรูปแบบที่ย่อและขยายข้อมูลในแต่ละ Layers ลักษณะคล้ายรูปตัว U สถาปัตยกรรมแบ่งออกเป็นสองส่วน คือ ตัวเข้ารหัส (Encoder หรือ Contracting) และ ตัวถอดรหัส (Decoder หรือ Expansive)

ตัวเข้ารหัสประกอบด้วย Convolutional layers ซึ่งแต่ละเลเยอร์จะประกอบด้วย Activation function คือ Rectified linear unit หรือ ReLU ตามด้วยการดำเนินการดาวน์แซมปลิง (Down sampling) ทำให้ลดขนาดเชิงพื้นที่ของภาพในขณะที่เพิ่มจำนวน Feature map ซึ่งจะดึงคุณลักษณะระดับสูงออกจากรูปภาพที่นำเข้า ในแต่ละขั้นตอนการสุ่มตัวอย่าง จำนวนของ Feature map จะเพิ่มขึ้นเป็นสองเท่า ในขณะที่มิติเชิงพื้นที่จะลดลงครึ่งหนึ่ง

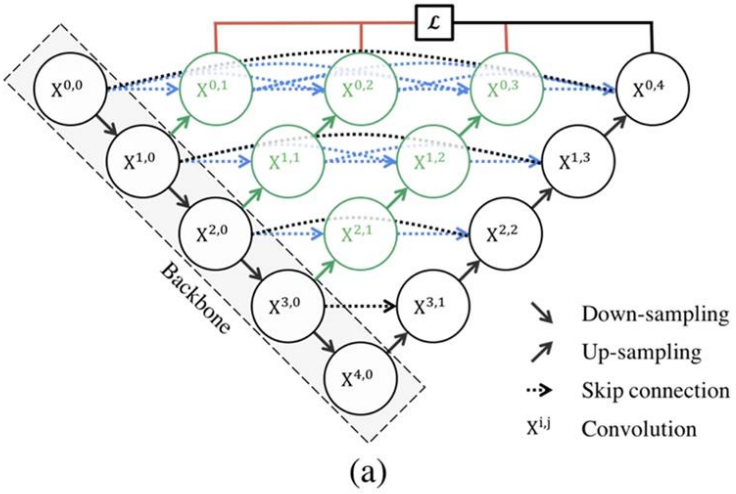
ตัวถอดรหัสเป็นส่วนครึ่งหลังของเครือข่ายและแสดง Feature ที่สร้างขึ้นโดยตัวเข้ารหัส และใช้การดำเนินการอัปแซมปลิง (Up sampling) และ Convolutional layers เพื่อแปลงขนาดของภาพกลับเป็นขนาดภาพดั้งเดิมในขณะที่รักษาข้อมูลที่มีความรายละเอียดสูงของภาพที่เรียนรู้โดยตัวเข้ารหัส โดยตัวถอดรหัสมีการเชื่อมต่อกับตัวเข้ารหัส ซึ่งช่วยให้โมเดลใช้ข้อมูลจากเลเยอร์ก่อนหน้าของตัวเข้ารหัสเพื่อปรับปรุงความแม่นยำในการแบ่งส่วน



ภาพ 13 ลักษณะของสถาปัตยกรรม U-net (27)

5. สถาปัตยกรรม UNet++ (UNet++ architecture)

UNet++ พัฒนามาจากสถาปัตยกรรม U-Net สำหรับการบางส่วนภาพ โดยปรับปรุงประสิทธิภาพของ U-Net เพื่อลดช่องว่าง (Gap) การเชื่อมต่อ หรือ Skip connections ระหว่างตัวเข้ารหัสและตัวถอดรหัส ซึ่งใน U-net ตัวเข้ารหัสและตัวถอดรหัสจะ Skip connections โดยตรง ทำให้การจับคู่ของ Feature map เกิดความคลาดเคลื่อน ใน UNet++ จะมี Dense อยู่ระหว่างเส้นทาง Skip connections ของตัวเข้ารหัสและตัวถอดรหัส ทำให้มั่นใจได้ว่า feature ของภาพไม่มีการสูญหายระหว่างทางแน่นอน ซึ่งมีส่วนช่วยในการปรับปรุงความแม่นยำของการแบ่งส่วน และการไล่ระดับสีของภาพ (Gradient flow)



ภาพ 14 ลักษณะของสถาปัตยกรรม UNet++ (28)

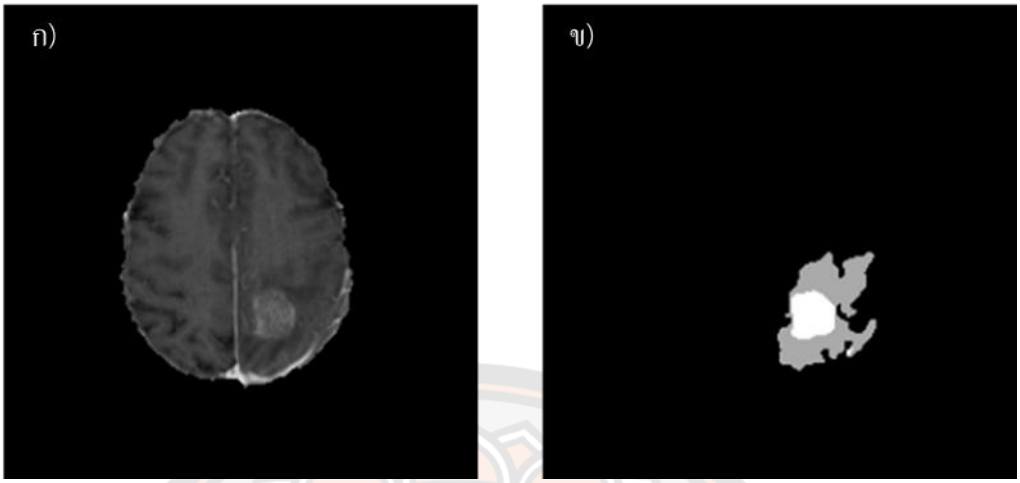
6. การแบ่งส่วนภาพ (Image segmentation)

การแบ่งส่วนรูปภาพ คือ กระบวนการแบ่งรูปภาพออกเป็นหลายส่วน ซึ่งแต่ละส่วนจะสอดคล้องกับวัตถุในภาพที่แตกต่างกัน โดยการแบ่งส่วนภาพเป็นงานที่สำคัญในด้าน Computer vision และการประมวลผลภาพ ซึ่งการแบ่งส่วนภาพถูกนำไปใช้ในแอพพลิเคชั่นต่าง ๆ รวมถึงการถ่ายภาพทางการแพทย์ รถยนต์ไร้คนขับ และวิทยาการหุ่นยนต์

การเรียนรู้เชิงลึกมีผลกระทบอย่างมากต่อการแบ่งส่วนภาพ โดย CNN เป็นเทคนิคหลักที่ใช้ในการแบ่งส่วนภาพ โดยมีสถาปัตยกรรมที่หลากหลายซึ่งออกแบบมาโดยเฉพาะสำหรับงานนี้ เช่น U-Net, SegNet และ Mask R-CNN

โดยทั่วไปสถาปัตยกรรมของ CNN เจะเกี่ยวข้องกับเฟรมเวิร์กตัวเข้ารหัสและตัวถอดรหัส โดยที่ตัวเข้ารหัสจะประมวลผลภาพเพื่อแยกคุณสมบัติที่เกี่ยวข้อง และตัวถอดรหัสจะจับคู่คุณสมบัติเหล่านี้กลับไปยังพิกเซลเพื่อสร้างภาพที่แบ่งส่วน การใช้การเชื่อมต่อระหว่างตัวเข้ารหัสและตัวถอดรหัส ในสถาปัตยกรรมทำให้ตัวถอดรหัสสามารถเข้าถึงข้อมูลคุณสมบัติระดับต่ำจากเลเยอร์ก่อนหน้าของตัวเข้ารหัส นอกจาก CNN แล้ว เทคนิคการเรียนรู้เชิงลึกอื่น ๆ เช่น โครงข่ายประสาทเทียมแบบเกิดซ้ำ (Recurrent neural networks , RNN) ถูกนำมาใช้เพื่อการแบ่งส่วนภาพเช่นกัน ตัวอย่างเช่น Fully Convolutional networks (FCN) ใช้ RNN เพื่อปรับแต่งการคาดคะเนการแบ่งส่วน ในขณะที่ Graph Convolutional networks (GCN) แบ่งกลุ่มรูปภาพโดยแสดงเป็นกราฟ

จากที่กล่าวมาข้างต้นการเรียนรู้เชิงลึกทำให้การแบ่งส่วนภาพที่ซับซ้อน ได้อย่างแม่นยำและมีประสิทธิภาพมากขึ้น แม้ในภาพที่มีคุณภาพแตกต่างกัน เช่น แสงที่แตกต่างกัน การบดบัง ฯลฯ



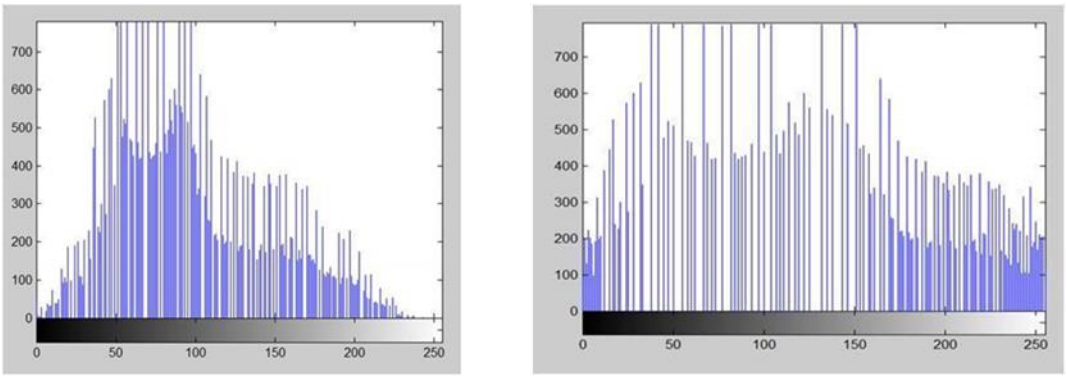
ภาพ 15 ก) แสดงภาพต้นฉบับ และ ข) แสดงภาพที่ผ่านกระบวนการการแบ่งส่วน

7. การปรับปรุงภาพ (Image enhancement)

การปรับปรุงภาพ (Image enhancement) เป็นกระบวนการที่ใช้ในการปรับคุณภาพของภาพเพื่อเพิ่มรายละเอียดหรือปรับโทนสีให้ตรงตามที่ต้องการ วิธีการปรับปรุงภาพที่ใช้จะแตกต่างกันไปตามความต้องการของผู้ใช้ โดยทั่วไปการปรับปรุงภาพมุ่งเน้นที่การปรับปรุงความสมดุลของภาพ เพิ่มความคมชัด ลดสัญญาณรบกวน ปรับความสว่างและความเข้มของภาพ เพื่อให้ภาพมีคุณภาพที่ดีขึ้นและสอดคล้องกับวัตถุประสงค์ที่ต้องการใช้ภาพนั้น ๆ ตามการประมวลผลภาพและการตีความข้อมูลต่าง ๆ ในแง่มุมที่แตกต่างกัน ซึ่งอัลกอริทึมที่ใช้ในการปรับปรุงภาพมีหลากหลายแบบในที่นี้จะยกตัวอย่างอัลกอริทึมที่ใช้ในงานวิจัย

7.1 Contrast-Limited Adaptive Histogram Equalization (CLAHE)

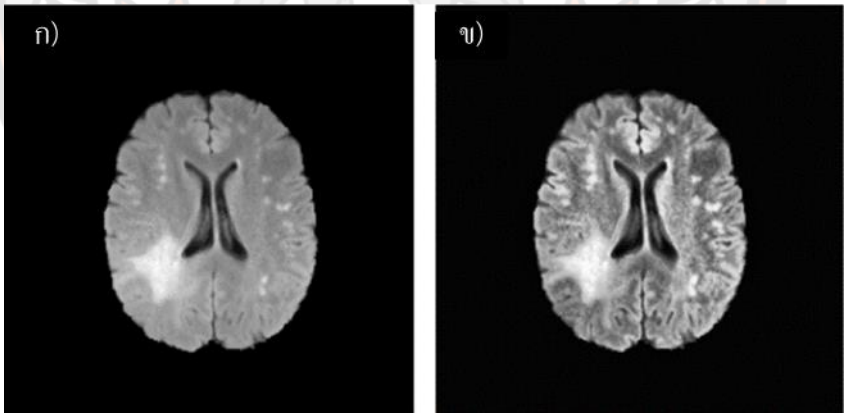
Histogram Equalization (HE) เป็นวิธีการประมวลผลภาพเพื่อขยายช่วงความเข้มของภาพ โดยปรับความแตกต่างของความเข้มระหว่างวัตถุสองชิ้นภายในภาพให้ใกล้เคียงกันจากภาพ 16 จะเห็นว่าภาพที่ความมืดเป็นส่วนใหญ่ฮิสโทแกรมจะรวมกันบริเวณด้านซ้ายของกราฟ และภาพที่มีสีความสว่างเป็นส่วนใหญ่ฮิสโตแกรมจะรวมกันบริเวณด้านขวาของกราฟ สำหรับภาพที่มีคอนทราส์ต่ำฮิสโทแกรมส่วนใหญ่จะรวมกันอยู่กึ่งกลางของกราฟ ซึ่ง HE จะทำให้ฮิสโทแกรมเหล่านี้ขยายช่วงออก เพื่อปรับปรุงคอนทราส์ของภาพให้ดีขึ้น (29-32) โดย HE ขึ้นอยู่กับ Cumulative distribution function (CDF) ในเทอมของ Probability density function ซึ่งเป็นผลรวมสะสมของความน่าจะเป็นทั้งหมดภายในภาพ ดังสมการที่ 2



ภาพ 16 ลักษณะของวิธีการประมวลผลภาพแบบ Histogram equalization (29)

$$H(i) = \sum_{i=0}^{k} P(r_k) = \frac{(L-1)}{N} \sum_{i=0}^{k} n_k \quad (2)$$

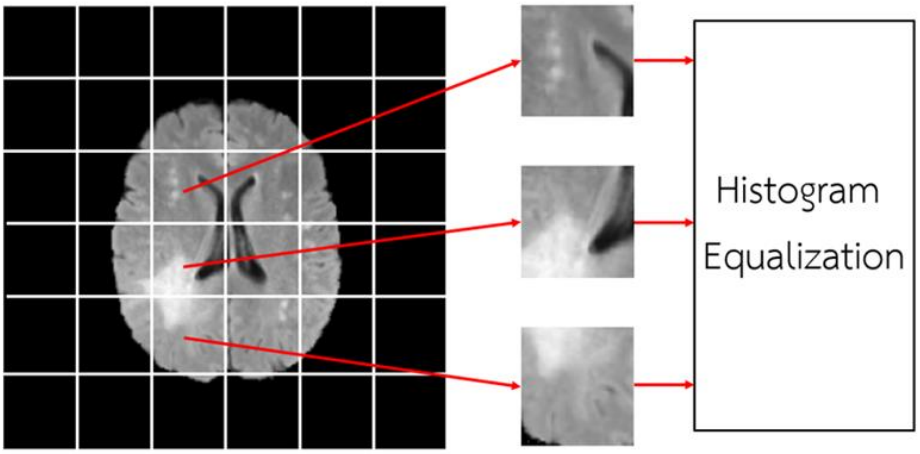
โดยที่ L คือ ระดับความเข้มสูงสุดของภาพ N คือ จำนวนพิกเซลทั้งหมดภายในภาพ r_k คือ ช่วงของระดับสีเทาตั้งแต่ 0 ถึง L – 1 n_k คือ จำนวนพิกเซลตามระดับสีเทา r_k และ H(j) คือ Histogram equalized



ภาพ 17 ก) แสดงภาพต้นฉบับ และ ข) แสดงภาพที่ผ่านการปรับปรุงภาพโดยใช้เทคนิค CLAHE

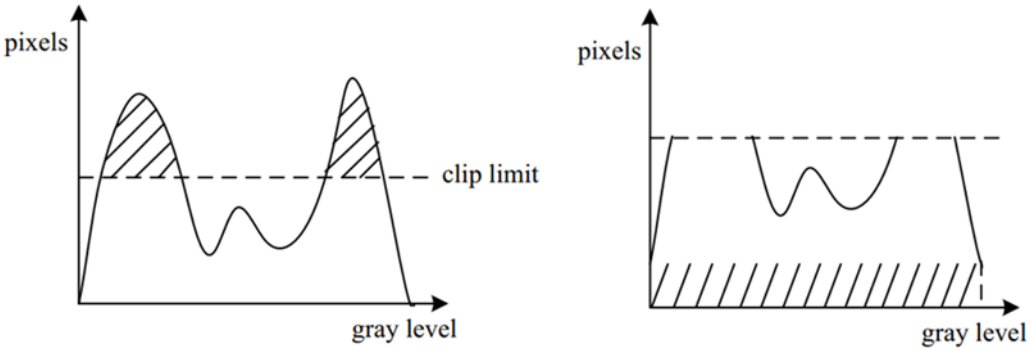
เนื่องจากเทคนิค HE พบข้อเสียบางอย่าง ได้แก่ ภาพสูญเสียสมดุลหรือเกิด Over-enhancement ในภาพที่มีความสว่างหรือความมืดมากเกินไป และ เกิดสัญญาณรบกวนที่พื้นหลังของภาพ สังเกตได้จากภาพ 17 ดังนั้นจึงมีการพัฒนาเทคนิค Adaptive Histogram Equalization (AHE) ขึ้น เพื่อลดสัญญาณรบกวนที่เกิดขึ้นบนพื้นหลังของภาพ โดยทำการแบ่งภาพออกเป็นพื้นที่

ย่อย หรือ ไทล์ (Tile) ตามผู้วิจัยกำหนด และนำพื้นที่ย่อยแต่ละพื้นที่มาปรับปรุงภาพด้วยเทคนิค HE ดังภาพ 18 จากนั้นนำพื้นที่ย่อยที่ได้รับการปรับปรุงทั้งหมดมารวมกันเป็นภาพเดียว และแก้ไขปัญหาเรื่องขอบที่ซ้อนทับเมื่อนำภาพมารวมกันด้วยวิธี Bilinear interpolation



ภาพ 18 การแบ่งภาพของเทคนิค Adaptive Histogram Equalization (AHE)

เนื่องจากเทคนิค AHE ไม่แก้ไขปัญหาเรื่อง Over-enhancement ของภาพ จึงมีการพัฒนาเทคนิค CLAHE ขึ้น โดยมี AHE และ HE เป็นพื้นฐาน โดย CLAHE นั้น จะใช้วิธีการจำกัดความสูงของฮิสโทแกรม (Clip Histogram) เพื่อลดสัญญาณรบกวนและ Over-enhancement ของคอนทราส์ โดยมีการกำหนดระดับในการกำจัดฮิสโทแกรม หรือ Clip limit เพื่อกระจายสัญญาณไปยังส่วนอื่น ๆ ของฮิสโทแกรม (33, 34) ซึ่งลักษณะการกำจัดฮิสโทแกรมแสดงดังภาพ 19



ภาพ 19 การกำจัดฮิสโทแกรมตามระดับที่กำหนด (33)

วิธีการทำงานของเทคนิค CLAHE สามารถพิจารณาได้จากสมการดังต่อไปนี้

$$N_{avg} = \frac{N_R \times N_C}{N_g} \quad (3)$$

โดยที่ N_R คือ จำนวนของ tile ในแถว N_C คือ จำนวนของ tile ในหลัก N_g คือ จำนวนของระดับสีเทาในแต่ละพื้นที่ และ N_{avg} คือ จำนวนของพิกเซลเฉลี่ย เมื่อทราบจำนวนของพิกเซลเฉลี่ยในแต่ละ Tile จะนำไปสู่การหาระดับจริงของการกำจัดฮิสโทแกรม (N_{CL}) โดยแสดงดังสมการที่ 4

$$N_{CL} = N_{cilp} \times N_{avg} \quad (4)$$

โดยที่ N_{clip} คือ ค่าระดับการกำจัดฮิสโทแกรม หรือ Clip limit ซึ่งเป็นค่าที่ผู้วิจัยกำหนด สำหรับค่าพิกเซลที่ถูกตัดจากอิสโทแกรมต่าง ๆ จะนำมารวมกันและเฉลี่ยค่าพิกเซลเพื่อเพิ่มให้กับระดับสีเทาอื่น ๆ ตามเงื่อนไขที่กำหนด ซึ่งผลรวมของค่าพิกเซลที่ถูกตัดและเฉลี่ยให้ระดับสีเทาอื่น ๆ (N_{acp}) แสดงดังสมการที่ 5 และเงื่อนไขของการเพิ่มพิกเซลในระดับสีเทาอื่น ๆ ของแต่ละฮิสโทแกรม แสดงดังสมการที่ 6

$$N_{acp} = \frac{N_{sumcp}}{N_g} \quad (5)$$

$$H(i)_{new} = \begin{cases} N_{CL}, & H(i) > N_{CL} \\ N_{CL}, & H(i) + N_{acp} \geq N_{CL} \\ H(i) + N_{acp}, & else \end{cases} \quad (6)$$

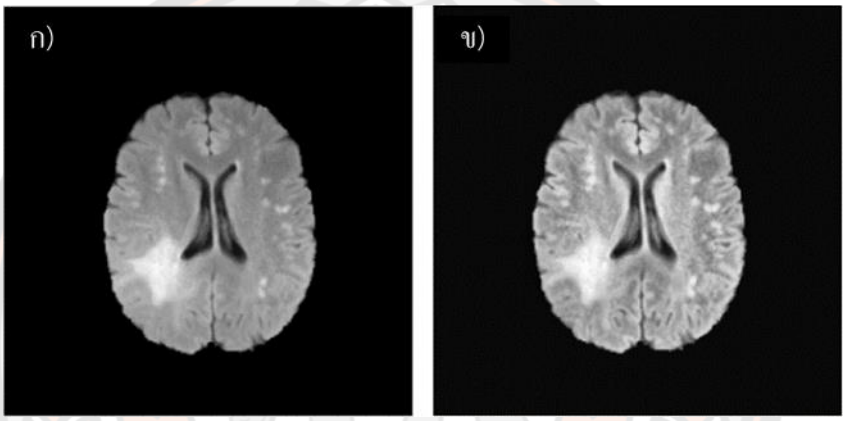
โดยที่ H(i) คือ จำนวนพิกเซลในแต่ละระดับสีเทาของฮิสโทแกรมเดิม และ $H(i)_{new}$ คือ จำนวนพิกเซลใหม่ในแต่ละระดับสีเทาที่ได้รับการเพิ่มพิกเซล สำหรับกรณีที่จำนวนพิกเซลของฮิสโทแกรมเดิมที่ได้รับการเพิ่มพิกเซล และมีค่าเกิน Clip limit จะทำการนำจำนวนพิกเซลที่เกินนั้น ไปเพิ่มให้กับตำแหน่งอื่น ๆ โดยเรียงจากระดับสีเทาต่ำสุดไปยังระดับสีเทาสูงสุด เมื่อคำนวณฮิสโทแกรมใหม่เสร็จ จะทำการเชื่อมต่อ Tiles แต่ละแผ่นเข้าด้วยกันผ่าน Bilinear interpolation เพื่อกำจัดขอบที่เชื่อมต่อระหว่าง Tiles

ข้อดีของเทคนิค CLAHE ช่วยเพิ่มคอนทราสต์เฉพาะที่และกำจัดสัญญาณรบกวนที่เกิดจากการขยายคอนทราสต์ที่ไม่สม่ำเสมอในภาพ MRI รวมถึงปรับปรุงโครงสร้างและรายละเอียดภายในภาพ MRI ให้มองเห็นได้ชัดมากขึ้น อย่างไรก็ตามเทคนิค CLAHE อาจมีการเพิ่มสัญญาณรบกวนในบริเวณที่มีความเข้มของภาพสม่ำเสมอ

7.2 Gamma Correction (GC)

GC เป็นเทคนิคสำหรับการประมวลผลภาพ โดยปรับค่าแกมมาตามลักษณะเฉพาะของภาพให้เหมาะสมกับความสว่างและความคมชัดในของภาพนั้น ๆ (35, 36) ซึ่ง GC แสดงตามสมการที่ 7 โดยข้อดีเทคนิค GC ปรับความสว่างโดยรวมของภาพในบริเวณที่มีความเข้มสูงและต่ำให้เสมอจากการกำหนดค่าพารามิเตอร์แกมมา ซึ่งมีความเสี่ยงให้เกิดการสูญเสียรายละเอียดของภาพในกรณีที่กำหนดพารามิเตอร์ไม่เหมาะสม

$$T(I) = I_{max}\left(\frac{I}{I_{max}}\right)^{\gamma} \tag{7}$$



ภาพ 20 ก) แสดงภาพต้นฉบับ และ ข) แสดงภาพที่ผ่านการปรับปรุงภาพโดยใช้เทคนิค GC

7.3 Non-local mean filter (NLMF)

NLMF คืออัลกอริทึมที่ใช้สำหรับกรองสัญญาณรบกวนออกจากภาพ ในขณะที่รักษาความคมชัดของบริเวณขอบของวัตถุภายในภาพ นอกจากนี้ยังสามารถปรับพื้นที่ภายในภาพให้มีความราบเรียบมากยิ่งขึ้น ซึ่ง NLMF จะคำนวณค่าเฉลี่ยของพิกเซลทั้งหมดภายในภาพ โดยพิจารณาความคล้ายคลึงระหว่างพิกเซล ผลลัพธ์ของค่าเฉลี่ยนั้นส่งผลให้เกิดการกำจัดสัญญาณรบกวน และทำให้พิกเซลของภาพมีค่าใกล้เคียงกัน (37-40) ซึ่ง NLMF แสดงตามสมการที่ 8

$$NL(i) = \sum_{j \in I} w(i,j)v(j) \tag{8}$$

ให้สัญญาณรบกวนภายในภาพ หรือ Noisy Image คือ $v = v(i) | \, i \in I$ ค่าโดยประมาณ หรือ Estimated value คือ NL(i) สำหรับพิกเซล i ที่ถ่วงน้ำหนักด้วยค่าเฉลี่ยความเข้มของพิกเซลทั้งหมด v(j) ภายในภาพ I โดยที่ w(i, j) คือค่าถ่วงน้ำหนักที่กำหนดค่าให้กับ v(j) ขึ้นอยู่กับความคล้ายคลึงระหว่างพิกเซล i และ j รวมไปถึงเงื่อนไขที่กำหนดดังนี้ $0 \leq w(i,j) \leq 1$

และ $\sum_j w(i,j) = 1$ ซึ่งความคล้ายคลึงระหว่างพิกเซล i และ j สามารถหาได้จากฟังก์ชันการลดลงของระยะทางที่ถ่วงน้ำหนักแบบยุคลิก (Weighted Euclidean distance) แสดงตามสมการที่ 9

$$E\left\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\right\|_{2,\sigma}^2 \tag{9}$$

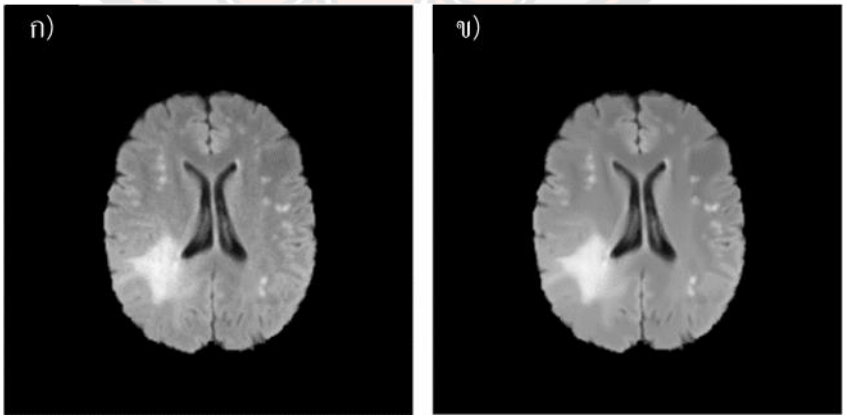
เมื่อ $\mathcal{N}_i$ และ $\mathcal{N}_j$ คือความเข้มของพื้นที่ใกล้เคียงที่มีศูนย์กลางอยู่ที่พิกเซล i และ j ขณะที่ σ คือค่าส่วนเบี่ยงเบนมาตรฐานของ Gaussian Kernel ซึ่งมีค่ามากกว่า 0 ทำให้ค่าถ่วงน้ำหนักเป็นไปตามสมการที่ 10

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{\left\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\right\|_{2,\sigma}^2}{h^2}} \tag{10}$$

เมื่อ Z(i) คือ Normalization constant ซึ่งหาได้จากสมการที่ 11 และ h คือขนาดของ Smoothing Kernel ซึ่งควบคุมการลดลงของฟังก์ชันเอกซ์โพเนนเชียลและการลดลงของค่าถ่วงน้ำหนักตามฟังก์ชันการลดลงของระยะทางที่ถ่วงน้ำหนักแบบยุคลิก ถ้าค่า h น้อยเกินไปส่งผลให้ไม่สามารถกำจัดสัญญาณรบกวน ในทางกลับกับถ้าค่า h มากเกินไปส่งผลให้ภาพเบลอ

$$Z(i) = \sum_j e^{-\frac{\left\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\right\|_{2,\sigma}^2}{h^2}} \tag{11}$$

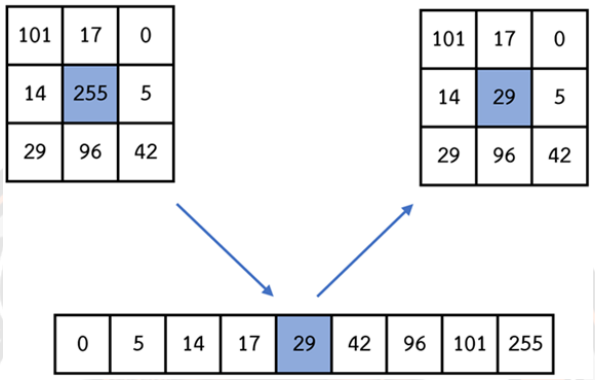
ข้อดีของเทคนิค NLMF คือ การลดสัญญาณรบกวนในขณะที่รักษารายละเอียดของภาพได้ดีมากกว่าเทคนิคอื่น ๆ อย่างไรก็ตามสมการในการคำนวณมีความซับซ้อน ดังนั้นการคำนวณจึงต้องใช้เวลาที่มากขึ้น



ภาพ 21 ก) แสดงภาพต้นฉบับ และ ข) แสดงภาพที่ผ่านการปรับปรุงภาพโดยใช้เทคนิค NLMF

7.4 Median and Wiener filter (MWF)

MF หรือตัวกรองข้อมูลโดยใช้ค่ามัธยฐาน เป็นตัวกรองที่ใช้ในการกำจัดสัญญาณรบกวนออกจากภาพโดยเฉพาะสัญญาณรบกวน Salt and pepper การทำงานของ MF นั้นจะนำค่าพิกเซลที่อยู่รอบตำแหน่งของพิกเซลที่สนใจมาจัดเรียงจากน้อยไปหามาก จากนั้นเลือกค่าที่อยู่กึ่งกลางไปใช้ (41) ดังภาพ 22 ซึ่งทำให้ภาพต้นฉบับไม่สูญเสียความคมชัด



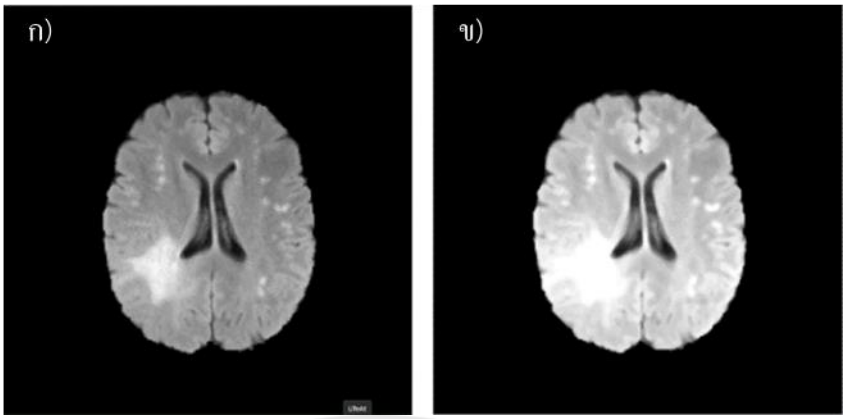
ภาพ 22 ลักษณะการทำงานของ Median filter

เนื่องจาก MF ไม่สามารถกำจัดสัญญาณรบกวนแบบกระตุ้นได้ (Impulse) เช่น สัญญาณรบกวนเกาส์เซียน ขณะที่การกรองด้วยเทคนิค WF สามารถกำจัดข้อด้อยในจุดนี้ได้ โดย WF เป็นตัวกรองที่ใช้ในการประมวลผลสัญญาณเพื่อลดสัญญาณรบกวนและเพิ่มคุณภาพของสัญญาณ โดยพยายามทำให้ค่า Mean square error ระหว่างภาพต้นฉบับกับภาพที่ได้รับการปรับปรุงมีค่าน้อยที่สุด ซึ่งการทำงานของ WF แบ่งเป็นสองส่วน คือ การกรองย้อนกลับ หรือ Inverse filtering และการปรับสัญญาณรบกวนให้เรียบ หรือ Nosie smoothing โดย WF แสดงได้ดังสมการที่ 12 ซึ่งอยู่ในรูปแบบของ Fourier domain (42)

$$W(f_1,f_2)=\frac{H^*(f_1,f_2)S_{xx}(f_1,f_2)}{|H(f_1,f_2)|^2S_{xx}(f_1,f_2)+S_{\eta\eta}(f_1,f_2)} \qquad (12)$$

โดยที่ $S_{xx}(f_1,\ f_2)$ และ $S_{\eta\eta}(f_1,\ f_2)$ คือ ความหนาแน่นสเปกตรัมของภาพต้นฉบับและสัญญาณรบกวน $H(f_1,\ f_2)$ คือ ตัวกรองเบลอ

ข้อดีของเทคนิค WF คือปรับสมดุลภาพ โดยการปรับสัญญาณรบกวนให้ราบเรียบมากขึ้นและรักษารายละเอียด ซึ่งพิจารณาจากคุณสมบัติทางสถิติของภาพและสัญญาณรบกวน อย่างไรก็ตามในทางปฏิบัติอาจเกิดความล่าช้าเนื่องจากมีการคำนวณการแปลงฟูริเยร์และผกผัน



ภาพ 23 ก) แสดงภาพต้นฉบับ และ ข) แสดงภาพที่ผ่านการปรับปรุงภาพโดยใช้เทคนิค MWF

8. เครื่องมือสำหรับการประเมินข้อมูล

8.1 Structural Similarity Index (SSIM)

ค่าที่บ่งบอกถึงคุณภาพของภาพ โดยเปรียบเทียบรูปร่างของภาพต้นฉบับและภาพที่ได้รับการปรับปรุง เพื่อประเมินความคล้ายคลึงของภาพทั้งสอง (43) โดยค่า SSIM อยู่ในช่วง -1 ถึง 1 ถ้าค่า SSIM ใกล้เคียง 1 แสดงถึงภาพที่มีความคล้ายคลึงกันและภาพที่ได้รับการปรับปรุงมีคุณภาพที่ดี ขณะที่ค่า SSIM ใกล้เตียง -1 แสดงถึงความคล้ายคลึงต่ำและภาพที่ได้รับการปรับปรุงมีคุณภาพต่ำ ซึ่งค่า SSIM หาได้จากสมการที่ 13

$$SSIM = i(x,y) \times c(x,y) \times s(x,y) \quad (13)$$

โดยที่ i คือ ดัชนีที่บ่งบอกถึงความสว่างที่คล้ายคลึงกัน, c คือ ดัชนีที่บ่งบอกถึงความคมชัดที่คล้ายคลึงกัน s คือ ดัชนีที่บ่งบอกถึงรูปร่างที่คล้ายคลึงกัน x คือ ภาพต้นฉบับ และ y คือ ภาพที่ได้รับการปรับปรุง

8.2 Mean squared error (MSE)

ค่าเฉลี่ยของความแตกต่างกำลังสองระหว่างภาพต้นฉบับและภาพที่ได้รับการปรับปรุง (44) ซึ่ง MSE แสดงดังตามสมการที่ 14 โดยค่า MSE เข้าใกล้ศูนย์บ่งบอกถึงจำนวนสัญญาณรบกวนระหว่างภาพต้นฉบับและภาพที่ได้รับการปรับปรุงมีค่าน้อย

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}\|f(i,j) - g(i,j)\|^2 \quad (14)$$

โดยที่ f คือ ค่าพิกเซลในภาพต้นฉบับ, g คือ ค่าพิกเซลในภาพที่ได้รับการปรับปรุง, m คือ จำนวนแถวพิกเซลของรูปภาพ, n คือ จำนวนหลักพิกเซลของรูปภาพ, I คือ ดัชนีของแถว และ j คือ ดัชนีของหลัก

8.3 Peak Signal-to-Noise Ratio (PSNR)

ค่าที่บ่งบอกถึงสัมประสิทธิ์ร้อยละของสัญญาณสูงสุดต่อสัญญาณรบกวน เพื่อประเมินระดับความแตกต่างของความเข้มของวัตถุภายในภาพและคุณภาพของภาพที่ปรับปรุง (45) ถ้าค่า PSNR สูงแสดงถึงภาพที่ได้รับการปรับปรุงมีคุณภาพที่ดี เนื่องจากสัญญาณสูงมีค่ามากกว่าสัญญาณรบกวน ในทางกลับกันค่า PSNR ต่ำแสดงถึงภาพที่ได้รับการปรับปรุงมีคุณภาพต่ำ ซึ่งค่า PSNR หาได้จากสมการที่ 15

$$PSNR = 10\log_{10}\frac{MAX^2}{MSE} \tag{15}$$

โดยที่ MAX คือค่าสัญญาณสูงสุดภายในภาพ และ MSE คือค่า Mean squared error หาได้จากสมการที่ 14

8.4 Dice Similarity Coefficient (DSC)

ค่าที่บ่งบอกความคล้ายคลึงระหว่างชุดข้อมูลสองชุด เพื่อประเมินความคล้ายคลึงของตำแหน่งวัตถุภายในภาพ โดยค่า DSC อยู่ในช่วง 0 ถึง 1 ในกรณีที่ค่า DSC เข้าใกล้ 0 แสดงถึงความคล้ายคลึงของชุดข้อมูลทั้งสองต่ำ ขณะที่ค่า DSC เข้าใกล้ 1 แสดงถึงค่าในชุดข้อมูลทั้งสองซ้อนทับกันอย่างสมบูรณ์ ในทางกลับกันค่า DSC เท่ากับ 0 แสดงถึงค่าในชุดข้อมูลทั้งสองไม่มีการซ้อนทับกัน ซึ่งค่า DSC หาได้จากสมการที่ 16

$$DSC = \frac{2|A \cap B|}{|A|+|B|} \tag{16}$$

โดยที่ A และ B คือชุดข้อมูลของ A และ B, $|A \cap B|$ คือ ส่วนที่ซ้ำของสองชุดข้อมูล

8.5 Jaccard Similarity Coefficient (IoU)

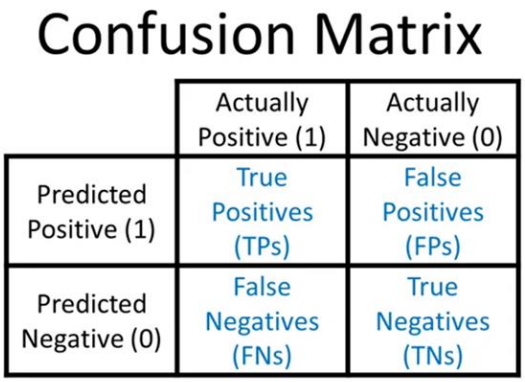
ค่าสัมประสิทธิ์ Jaccard หรือ Intersection over Union (IoU) เป็นการวัดอัตราส่วนของพื้นที่ทับซ้อนระหว่างการแบ่งส่วนข้อมูลที่ทำนายได้กับการแบ่งส่วนข้อมูลอ้างอิงต่อพื้นที่รวมของข้อมูลการแบ่งส่วนที่ทำนายได้กับข้อมูลการแบ่งส่วนที่ใช้อ้างอิง (46) ซึ่งค่า DSC หาได้จากสมการที่ 17

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{17}$$

โดยที่ A และ B คือชุดข้อมูลของ A และ B, $|A \cap B|$ คือ ส่วนที่ซ้ำของสองชุดข้อมูล, $|A \cup B|$

8.6 Confusion matrix

Confusion matrix คือ อุปกรณ์ที่ใช้วัดและประเมินความแม่นยำในการทำนายของโมเดล ซึ่งลักษณะของ Confusion matrix แสดงดังภาพ 24



ภาพ 24 ลักษณะของ Confusion Matrix(47)

True Positives (TP) คือ สิ่งที่ทำนายตรงกับสิ่งที่เกิดขึ้น ในกรณีที่ทำนายว่าสิ่งที่ทำนายเป็นจริงและสิ่งที่เกิดเป็นจริง ในที่นี้บ่งบอกถึงการทำนายขอบเขตของก้อนมะเร็งที่แบ่งส่วนได้อย่างถูกต้อง

True Negative (TN) คือ สิ่งที่ทำนายตรงกับสิ่งที่เกิดขึ้น ในกรณีที่ทำนายว่าสิ่งที่ทำนายเป็นเท็จและสิ่งที่เกิดขึ้นเป็นเท็จ ในที่นี้บ่งบอกถึงการทำนายขอบเขตของก้อนมะเร็งที่แบ่งส่วนมีส่วนหนึ่งถูกทำนายอย่างถูกต้องว่าเป็นพื้นหลัง

False Positive (FP) คือ สิ่งที่ทำนายไม่ตรงกับสิ่งที่เกิดขึ้น ในกรณีที่ทำนายว่าสิ่งที่ทำนายเป็นจริงแต่สิ่งที่เกิดเป็นเท็จ ในที่นี้บ่งบอกถึงการทำนายพื้นหลังอย่างไม่ถูกต้องว่าเป็นส่วนหนึ่งของขอบเขตของก้อนมะเร็งที่แบ่งส่วน

False Negative (FN) คือ สิ่งที่ทำนายไม่ตรงกับสิ่งที่เกิดขึ้น ในกรณีที่ทำนายว่าสิ่งที่ทำนายเป็นเท็จแต่สิ่งที่เกิดเป็นจริง ในที่นี้บ่งบอกถึงการทำนายขอบเขตของก้อนมะเร็งที่แบ่งส่วนมีส่วนหนึ่งถูกทำนายอย่างไม่ถูกต้องว่าเป็นพื้นหลัง

จากองค์ประกอบทั้ง 4 ส่วนที่กล่าวมาในข้างต้นสามารถนำมาหาค่าต่าง ๆ ไม่ว่าจะเป็นค่า Accuracy, Precision, และ Recall เพื่อประสิทธิภาพการทำนายของโมเดล โดยสามารถหาได้จากสมการที่ 18, 19, และ 20 ตามลำดับ

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (18)$$

โดยที่ Accuracy บอกถึงความแม่นยำของโมเดล หรือ จำนวนครั้งที่ทายถูกเทียบกับจำนวนครั้งที่ทายทั้งหมด

$$Precision = \frac{TP}{TP+FP} \quad (19)$$

โดยที่ Precision บอกถึงความเที่ยงตรงของข้อมูล หรือ จำนวนครั้งที่ทายว่าจริงแล้วถูกเทียบกับจำนวนครั้งที่ทายว่าจริงทั้งหมด

$$Recall = \frac{TP}{TP+FN} \quad (20)$$

โดยที่ Recall หรือ Sensitivity บอกถึงจำนวนครั้งที่ทายว่าจริงแล้วถูกเทียบกับจำนวนที่เกิดขึ้นจริงทั้งหมดในข้อมูล

ในขณะเดียวกันค่า DSC และค่า IoU สามารถเขียนได้ในรูปแบบของ confusion matrices ดังสมการที่ 21 และ 22

$$DSC = \frac{2TP}{2TP+FP+FN} \quad (21)$$

$$IoU = \frac{TP}{TP+FP+FN} \quad (22)$$

งานวิจัยที่เกี่ยวข้อง

Thong Vo และคณะ (2022) (8) นำเสนอสถาปัตยกรรมการเรียนรู้เชิงลึกเพื่อแบ่งส่วนโดยอัตโนมัติบนภาพเอ็มอาร์ไอสมองชนิด FLAIR, Post-contrast, และ Pre-contrast ซึ่งโมเดลมีชื่อว่า UVR-Net สถาปัตยกรรมที่นำเสนอนี้อิงตามสถาปัตยกรรม U-Net ผลการทดลองแสดงให้เห็นว่า UVR-Net ที่เสนอมีค่า DSC เท่ากับ 0.76 และค่า IoU เท่ากับ 0.89 เมื่อเทียบกับสถาปัตยกรรม U-Net แบบดั้งเดิมพบว่ามีค่า DSC มากกว่า 11% นอกจากนี้ ผู้วิจัยยังทำการวิเคราะห์ความไวสำหรับฟังก์ชันการสูญเสียในแบบจำลองที่เสนอ

Aboudi Fathia และคณะ (2022) (48) ได้แนะนำวิธีการสร้างโครงข่ายประสาทเทียมเชิงลึก โดยอิงตามสถาปัตยกรรม U-Net ที่สามารถแยกรอยโรคหลอดเลือดสมองตีบออกจากเนื้อเยื่อปกติ โดยใช้ภาพเอ็มอาร์ไอชนิด Diffusion-weighted imaging (DWI), T1, T2, และ FLAIR มีการใช้เทคนิค Fine-tuning ร่วมกับสถาปัตยกรรม U-Net ซึ่งผู้วิจัยใช้ชุดข้อมูลสาธารณะ ISLES 2015 โดยโมเดลที่ผู้วิจัยออกแบบมี DSC และความแม่นยำ เท่ากับ 55.77% และ 99.96% ตามลำดับ

Jain Rahul และคณะ (2023) (9) นำเสนอกระบวนการ Shared-encoder multi-class segmentation โดยสถาปัตยกรรมที่เสนอในฝั่งของตัวเข้ารหัสจะมีเพียงหนึ่งชั้น ขณะที่ฝั่งของตัวถอดรหัสมีหลายชั้น จากนั้นจะรวมผลลัพธ์ที่ได้จากฝั่งตัวถอดรหัสเข้าด้วยกัน ซึ่งผู้วิจัยใช้ชุดข้อมูลสาธารณะ BraTS20 และประเมินประสิทธิภาพเทียบกับแบบจำลอง 2D U-Net โดยสถาปัตยกรรมที่

ออกแบบมีค่า IoU และ DSC เท่ากับ 0.644 และ 0.731 ตามลำดับ ขณะที่สถาปัตยกรรม U-Net มีค่า IoU และ DSC เท่ากับ 0.604 และ 0.690 ตามลำดับ นอกจากนี้วิธีการที่ผู้วิจัยเสนอแสดงให้เห็นถึงประสิทธิภาพที่เพิ่มขึ้นอย่างเห็นได้ชัดในการแบ่งกลุ่มบนภาพเอ็มอาร์ไอชนิด T1-Gd

Lin M และคณะ (2021) (5) ได้พัฒนาการแบ่งส่วนอัตโนมัติของเนื้องอกในสมองในภาพ MR โดยใช้การเรียนรู้เชิงลึกและสถาปัตยกรรม U-net ร่วมกับ context block เพื่อขยายการรับสัญญาณของโครงข่ายประสาทเทียมและเพิ่มความแม่นยำในการแบ่งส่วน ผู้วิจัยมีการใช้ Fivefold cross-validation กับชุดข้อมูลฝึกฝน Brain Tumor Segmentation Challenge (BraTS) 2020 สำหรับชุดข้อมูล BraTS จะแบ่งมะเร็งออกเป็น 3 ส่วน ได้แก่ Whole tumor (WT), Tumor core (TC), และ Enhancing tumor (ET) โดยวิธีการสามารถประเมินความแม่นยำได้จากค่า Dice similarity coefficient (DSC) และ Hausdorff distance (HD) ซึ่งปริมาตรของก้อนมะเร็งที่สร้างขึ้นจะนำไปเปรียบเทียบกับปริมาตรที่สร้างด้วย Bland–Altman plots และ Pearson analysis โดยผลทดสอบ พบว่าวิธีการที่พัฒนามีค่า DSC เท่ากับ 0.923 ± 0.047, 0.893 ± 0.176 และ 0.846 ± 0.165 ตามลำดับ และ HD95 มีค่าเท่ากับ 3.946 ± 7.041, 3.981 ± 6.670 และ 10.128 ± 51.136 มิลลิเมตร บนชุดข้อมูล WT, TC และ ET ตามลำดับ ซึ่งผลการทดลองแสดงให้เห็นว่าวิธีการที่พัฒนามีความแม่นยำในการแบ่งส่วนที่ดีกว่าอย่างมีนัยสำคัญ ($p < 0.05$) มากกว่าเครือข่าย CNN อีกสองเครือข่าย

S. S. Pasha และคณะ (2019) (49) ได้ทำการศึกษาการปรับปรุงภาพเอ็มอาร์ไอสมอง โดยใช้เทคนิค Histogram equalization (HE) ประกอบด้วย HE, Local-HE, Adaptive HE, และ Contrast-Limited AHE ซึ่งผู้วิจัยทำการประเมินคุณภาพของภาพโดยใช้ค่า Peak Signal to Noise Ratio (PSNR), Signal to Noise (SNR), Structure similarity (SSIM), และ Mean square error (MSE) จากผลการศึกษาผู้วิจัยพบว่าเทคนิค HE มีส่วนช่วยในการเพิ่มความสว่างของภาพ สำหรับ Local-HE นั้นให้ผลลัพธ์ที่ดีกว่า AHE และ CLAHE ช่วยเพิ่มคอนทราสต์พร้อมทั้งลบสัญญาณรบกวนให้กับภาพ

M. Sahnoun และคณะ (2018) (50) ได้ทำการศึกษากระบวนการก่อนประมวลผลภาพบนภาพเอ็มอาร์ไอชนิด T1-w, T2-w, และ T2-flair ของผู้ป่วยที่มีพยาธิสภาพของเส้นเลือดตีบหลายเส้น โดยเปรียบเทียบเทคนิค Traditional Gamma Correction (TGC) และ Adaptive Gamma Correction (AGC) เพื่อให้แยกความแตกต่างระหว่าเนื้อเยื่อปกติกับเนื้อเยื่อที่ได้รับผลกระทบจากโรคได้ดีขึ้น ซึ่งผู้วิจัยทำการประเมินคุณภาพของภาพโดยใช้ค่า Peak Signal to Noise Ratio (PSNR), Absolute Mean Brightness Error (AMBE), Structure similarity (SSIM), Entropy, Entropy Measure Enhancement (EME), และ Quality-aware Relative Contrast Measure (QRCM) จากผลการศึกษาผู้วิจัยพบว่าสำหรับค่า PSNR, AMBE, SSIM และ EME ของภาพที่ปรับปรุงโดย

เทคนิค TGC โดยเฉพาะที่ค่า γ เท่ากับ 1.1 ให้คุณภาพของภาพที่ดีกว่าเมื่อเทียบกับภาพที่ปรับปรุงโดยเทคนิค AGC

L. Chandrashekar และ A. Sreedevi (2017) (39) ได้ทำการศึกษาแบบจำลองของสัญญาณรบกวนและเทคนิคการกำจัดสัญญาณรบกวนแบบไม่เชิงเส้นบนภาพเอ็มอาร์ไอ โดยเทคนิคแบบไม่เชิงเส้นในการศึกษาประกอบด้วย Anisotropic Filter, Bilateral Filter และ Non Local means filter ซึ่งเทคนิคตามที่กล่าวมาได้รับการประเมินจากค่า PSNR, Entropy, MSE, SSIM และเวลาทำงานโดยเฉลี่ย จากผลการศึกษาผู้วิจัยพบว่าเทคนิค Non Local means filter ให้ผลลัพธ์ของค่า PSNR, MSE and SSIM ดีกว่าบนภาพเอ็มอาร์ไอที่มีสัญญาณรบกวนชนิด Gaussian

A. Min และ Z. M. Kyu (2017) (41) ได้ทำการศึกษาการรวมผลลัพธ์ของการปรับปรุงภาพการจัดกลุ่มแบบ Adaptive k-means clustering และการดำเนินการทางสัณฐานสิทยาสำหรับการแบ่งส่วนเนื้องอกบนภาพเอ็มอาร์ไอสมอง สำหรับการปรับปรุงภาพผู้วิจัยใช้ Median filter (MF) และ Wiener filter (WF) และประเมินคุณภาพของการปรับปรุงภาพโดยใช้ค่า MSE และ PSNR สำหรับผลลัพธ์ของการแบ่งส่วนประเมินผลโดยค่า Sensitivity, Specific, Precision, และ Accuracy พร้อมทั้งเปรียบเทียบเวลาระหว่างเทคนิค Adaptive K-means clustering (AKM) และเทคนิค adaptive K-means clustering ที่ผ่านการดำเนินการทางสัณฐานวิทยา (AKMM) จากผลการศึกษาผู้วิจัยพบว่าการรวมกันของเทคนิค MF และ WF มีประสิทธิภาพมากกว่าการใช้เทคนิคใดเทคนิคหนึ่ง และการแบ่งส่วนเนื้องอกบนภาพเอ็มอาร์ไอชนิด T2-flair มีค่า Sensitivity, Specific, Precision, และ Accuracy เท่ากับ 85.41%, 98.90%, 78.30% และ 98.23% ตามลำดับ

บทที่ 3

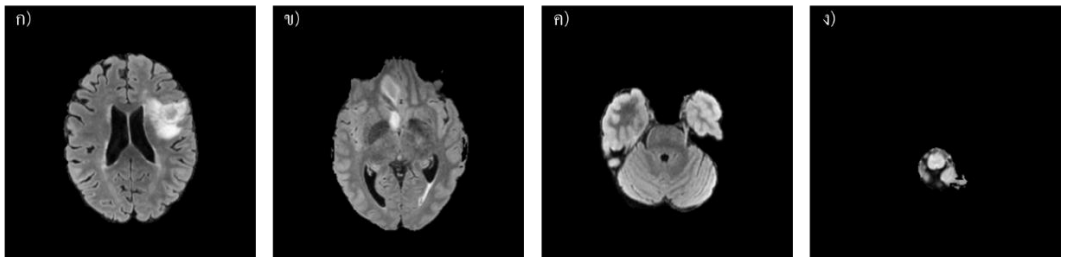
วิธีดำเนินงานวิจัย

ประชากรและกลุ่มตัวอย่าง

การศึกษานี้ใช้ชุดข้อมูลภาพเอ็มอาร์ไอ ซึ่งประกอบด้วยภาพเอ็มอาร์ไอ และข้อมูลการกำหนดขอบเขต จากฐานข้อมูลสาธารณะออนไลน์ Synapse ในชุดข้อมูลภาพ Brain Tumor Segmentation 2023 Dataset (BraTS2023) ซึ่งเป็นข้อมูลภาพเนื้องอกชนิดไกลโอมา (Glioma) สำหรับการฝึกฝนระบบจำนวน 1,350 ชุด สำหรับการตรวจสอบระบบจำนวน 1,050 ชุด และสำหรับการทดสอบระบบจำนวน 600 ชุด โดยในแต่ละชุดข้อมูลประกอบด้วยภาพเอ็มอาร์ไอชนิด T1, T1c, T2, FLAIR และภาพ Mask

เกณฑ์การคัดเข้าและคัดออกของชุดข้อมูล

1. เกณฑ์การคัดเข้า
   1.1 ภาพ MRI ที่มีรอยโรคและภาพที่วาดขอบเขตของรอยโรคหนึ่งรอย
   1.2 ภาพ MRI และภาพที่วาดขอบเขตของรอยโรคในแนวระนาบ (Axial)
   1.3 ภาพที่วาดขอบเขตของรอยโรคได้รับการวาดขอบเขตของรอยโรคจากผู้เชี่ยวชาญในสาขาที่เกี่ยวข้อง
   1.4 ภาพ MRI และภาพที่วาดขอบเขตของรอยโรคที่มีขอบเขตของสมองที่สมบูรณ์
2. เกณฑ์การคัดออก
   2.1 ภาพ MRI และภาพที่วาดขอบเขตรอยโรคที่ว่างเปล่า
   2.2 ภาพ MRI และภาพที่วาดขอบเขตรอยโรคที่ไม่มีรอยโรค



ภาพ 25 ก) และ ข) แสดงภาพสมองที่สมบุรณ์ และ ค) และ ง) แสดงภาพสมองที่ไม่สมบูรณ์

เครื่องมือที่ใช้ในการศึกษา

1. Notebook โดยมีรายละเอียดดังนี้ Intel(R) Core(TM) i5-11400H, Ram 32 GB, และ NVIDIA GeForce RTX 3050 laptop
2. Visual Studio Code สำหรับใช้เขียนโปรแกรม
3. Python 3.10.7
4. Library ต่าง ๆ ที่เกี่ยวข้อง

ขั้นตอนการดำเนินงานวิจัย

งานวิจัยนี้ศึกษาผลการเปรียบเทียบกระบวนการก่อนการประมวลผลภาพเพื่อเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง และ พัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยวิธีดำเนินการประกอบด้วย การเตรียมข้อมูล Image และ Mask, การออกแบบโมเดล, การเก็บรวบรวมข้อมูล, และการวิเคราะข้อมูล

1. การเตรียมข้อมูล Image และ Mask

แปลงไฟล์ภาพจากนามสกุล nii เป็นไฟล์ภาพนามสกุล png โดยเลือกเฉพาะภาพเอ็มอาร์ไอในแนวระนาบ (Axial) ของภาพเอ็มอาร์ไอทั้ง 4 ชนิด และมีเงื่อนไขว่า Mask ในแต่ละภาพต้องมีขอบเขตของก้อนมะเร็งมากกว่า 1% เมื่อเทียบกับข้อมูลทั้งหมดภายใน Mask ซึ่งผลลัพธ์ที่จะได้แบ่งเป็นภาพในแนวระนาบทั้งหมดชนิดละ 54,031 ภาพ จากนั้นทำการสุ่มข้อมูลภาพจำนวน 3,000 ภาพ พร้อมทั้งแปลงข้อมูล Image และ Mask ให้อยู่ในรูป Numpy array หรือไฟล์นามสกุล npy โดยมีการแปลงขนาดของภาพร่วมถึงแปลงข้อมูลภาพให้อยู่ในรูปแบบภาพสีเทา หรือ Grayscale ดังนั้นข้อมูล Image และ Mask ที่ได้จะเท่ากับ 128 × 128 × 1 ในขั้นตอนสุดท้ายสำหรับ Image จะทำการรวมภาพเอ็มอาร์ไอทั้ง 4 ชนิด เข้าด้วยกัน เพื่อสร้างชุดข้อมูลใหม่ ซึ่งมีขนาดเท่ากับ 128 × 128 × 4

แบ่งข้อมูล Image และ Mask ทั้ง 3,000 ชุด เป็น Train set จำนวน 1,350 ชุด และ Validation set จำนวน 1,050 ชุด และ Test set จำนวน 600 ชุด โดยในแต่ละชุดข้อมูลจะมี Mask หรือขอบเขตของก้อนมะเร็งตามจำนวนชุดข้อมูล

2. การปรับปรุงคุณภาพของภาพ

การปรับปรุงสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งบนภาพเอ็มอาร์ไอสมองของการศึกษานี้มีการใช้เทคนิคสำหรับการปรับปรุงภาพ 4 เทคนิค ได้แก่ CLAHE, GC, NLMF, และ MWF โดยกระบวนการปรับปรุงคุณภาพของภาพดังนี้

2.1 เตรียมข้อมูลตามที่กล่าวในหัวข้อการเตรียมข้อมูล Image และ Mask แต่ในขั้นตอนก่อนการรวมข้อมูลภาพเอ็มอาร์ไอทั้ง 4 ชนิด จะทำเพิ่มการเพิ่มสัญญาณเกาสเซียนทั้งหมด 2 ระดับ (σ = 25 และ 30) จากนั้นปรับปรุงคุณภาพของภาพด้วยเทคนิคต่าง ๆ ก่อนทำการรวมภาพเข้าด้วยกัน

2.2 การปรุงปรุงคุณภาพของภาพในเทคนิคต่าง ๆ มีการกำหนดพารามิเตอร์ดังนี้

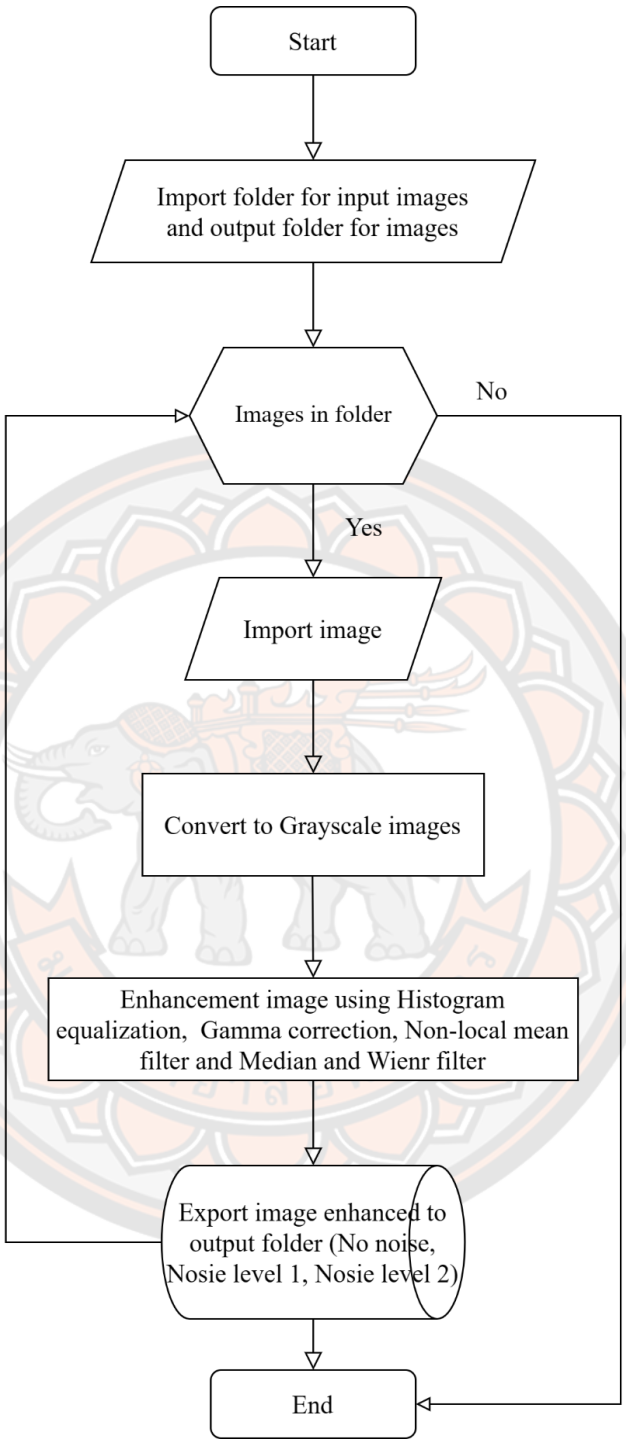
2.2.1 สำหรับเทคนิค CLAHE ทำการกำหนดขนาดของ Tiles ในช่วงขนาด 2 × 2 ถึง 8 × 8 โดยเพิ่มขึ้นทีละ 2 และกำหนดค่า Clip limit เท่ากับ 0.1, 0.2, และ 0.3

2.2.2 สำหรับเทคนิค GC ทำการกำหนดค่า Gamma ในช่วงขนาด 0.3 ถึง 1.9 โดยเพิ่มขึ้นทีละ 2

2.2.3 สำหรับเทคนิค NLMF ทำการกำหนดค่า h หรือ Smoothing kernel เท่ากับ 1 และ 2 ร่วมถึงกำหนดค่าขนาดของ Patch ในช่วงขนาด 2 × 2 ถึง 8 × 8 โดยเพิ่มขึ้นทีละ 2 และกำหนดขนาดของ Window เท่ากับ 10, 20, และ 30

2.2.4 สำหรับเทคนิค MF และ WF ทำการกำหนดขนาด Kernel ของ MF เท่ากับ 3 × 3, 5 × 5, และ 7 × 7 เป็นต้นไป จากนั้นระบบจะทำการส่งภาพที่ผ่านตัวกรอง MF เพื่อประมวลผลภาพผ่านตัวกรอง WF

2.3 เมื่อทำการปรับปรุงภาพเสร็จสิ้นทั้ง 4 เทคนิค โปรแกรมจะทำการส่งออกไฟล์ภาพไปยังพื้นที่จัดเก็บพร้อมทั้งชื่อไฟล์ตามเทคนิคการประมวลผลภาพ และแยกโฟลเดอร์จัดเก็บตามระดับของสัญญาณรบกวน จากนั้นนำภาพที่ได้รับการปรับปรุงมาเปรียบเทียบกับภาพต้นฉบับ เพื่อประเมินผลหาค่าพารามิเตอร์ที่ให้ผลลัพธ์ที่ดีที่สุดและนำผลประเมินเหล่านั้นไปเปรียบเทียบกับผลประเมินของโมเดลต่อไป ซึ่งแผนผังการปรับปรุงคุณภาพของภาพเป็นไปตามภาพ 26



ภาพ 26 แผนผังกระบวนการการปรับปรุงคุณภาพของภาพ

3. การออกแบบโมเดล

การพัฒนาโมเดลการเรียนรู้เชิงลึกของการศึกษานี้มีการพัฒนาบนพื้นฐานของสถาปัตยกรรมโครงข่าย U-Net และ UNet++ โดยมีเงื่อนไขคือโมเดลที่ออกแบบสามารถทำงานได้เร็ว ขณะเดียวกันยังคงรักษาความแม่นยำสูงในการทำนาย ซึ่งผู้วิจัยมีการออกโมเดลทั้งหมด 4 โมเดล และเลือกใช้เพียงหนึ่งโมเดลที่มีประสิทธิภาพมากกว่าโมเดลที่ใช้สถาปัตยกรรมโครงข่าย U-net และ UNet++ โดยรายละเอียดของแต่ละโมเดลแสดงดังนี้

3.1 โมเดลที่ใช้สถาปัตยกรรม U-net

โมเดลที่ใช้สถาปัตยกรรม U-net แบ่งออกเป็นสองส่วน คือ ตัวเข้ารหัสและตัวถอดรหัส ดังภาพ 13 โดยส่วนตัวเข้ารหัสมีการดำเนินการ Convolution ขนาด 3 × 3 จากนั้นจะดำเนินการผ่าน Batch Normalization และผ่าน Activation function (Rectified Linear Unit, ReLU) จากนั้นจะมีการส่งข้อมูลทั้งหมด 2 เส้นทาง โดยเส้นทางตามแนวดิ่ง จะส่งข้อมูลภาพผ่านกระบวนการ Max pooling ขนาด 2 × 2 เพื่อส่งข้อมูลไปยังชั้นถัดไป ขณะที่เส้นทางตามแนวราบจะมีการส่งข้อมูลไปยังฝั่งตัวถอดรหัส สำหรับตัวถอดรหัส มีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการจากส่วนที่อยู่ฝั่งตัวเข้ารหัส เข้ากับข้อมูลภาพที่ผ่านการ Up sampling จากชั้นที่อยู่ก่อนหน้า ซึ่งในแต่ละชั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 จากนั้นจะดำเนินการผ่าน Batch Normalization และผ่าน Activation function (Rectified Linear Unit, ReLU) และในชั้นสุดท้ายของตัวถอดรหัสจะมีการผ่านกระบวนการ Convolution ขนาด 1 × 1 Batch Normalization และ Activation function (Sigmoid) ก่อนให้ผลลัพธ์

3.2 โมเดลที่ใช้สถาปัตยกรรม UNet++

โมเดลที่ใช้สถาปัตยกรรม UNet++ แบ่งออกเป็นสามส่วน คือ ตัวเข้ารหัส ตัวถอดรหัส และตัวดำเนินการระหว่างตัวเข้ารหัสและตัวถอดรหัส ดังภาพ 14 โดยโดยส่วนตัวเข้ารหัสมีการดำเนินการ Convolution ขนาด 3 × 3 จากนั้นจะดำเนินการผ่าน Batch Normalization และผ่าน Activation function (Rectified Linear Unit, ReLU) จากนั้นจะมีการส่งข้อมูลทั้งหมด 2 เส้นทาง โดยเส้นทางตามแนวดิ่ง จะส่งข้อมูลภาพผ่านกระบวนการ Max pooling ขนาด 2 × 2 เพื่อส่งข้อมูลไปยังชั้นถัดไป ขณะที่เส้นทางตามแนวราบจะมีการส่งข้อมูลภาพผ่านกระบวนการที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส

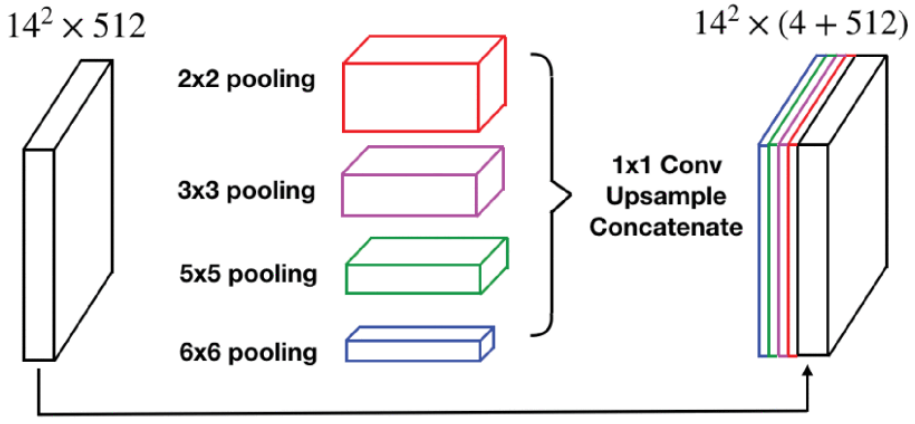
ในส่วนระหว่างตัวเข้ารหัสและตัวถอดรหัส จะมีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการ Convolution ขนาด 3 × 3 จากตัวเข้ารหัสที่อยู่ชั้นเดียวกัน จากนั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 ผ่าน Batch Normalization และ ReLU เพื่อไปรวมกับข้อมูลภาพที่ผ่านกระบวนการ Convolution ขนาด 3 × 3 และมีการ Up sampling มาจากชั้นถัดไป

จากนั้นจะส่งข้อมูลภาพไปเชื่อมต่อกับตัวถอดรหัสที่อยู่ในชั้นเดียวกันต่อไป โดยชั้นที่ 1 จะมีการดำเนินการทั้งหมด 3 ครั้ง ชั้นที่ 2 มีการดำเนินการทั้งหมด 2 ครั้ง และชั้นที่ 3 มีการดำเนินการทั้งหมด 1 ครั้ง ก่อนที่จะส่งข้อมูลไปยังตัวถอดรหัสที่อยู่ในแต่ละชั้น

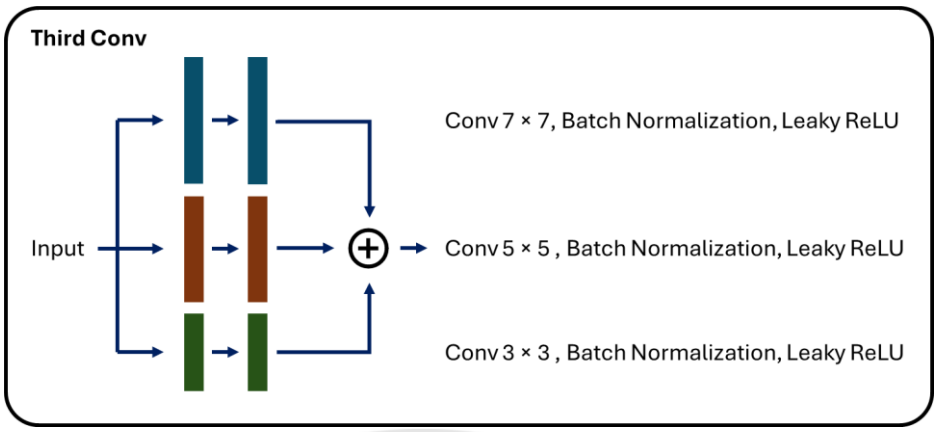
ในส่วนของตัวถอดรหัส มีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการจากส่วนที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส เข้ากับข้อมูลภาพที่ผ่านการ Up sampling จากชั้นที่อยู่ก่อนหน้า ซึ่งในแต่ละชั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 แต่ในชั้นสุดท้ายของตัวถอดรหัสจะมีการผ่านกระบวนการ Convolution ขนาด 1 × 1, Batch Normalization และ Activation function (Sigmoid) ก่อนจะนำออกภาพเพื่อไปใช้งาน

3.3 โมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบ 1

โมเดลที่ 1 มีแนวคิดจาก Residual Multi-Kernel Pooling ของ Z. Gu และคณะ (51) ที่มีการคำนวณ Kernel โดยใช้ Pooling ในหลายระดับก่อนนำมารวมกันเป็นหนึ่ง Kernel ดังภาพ 27 สำหรับในงานวิจัยนี้ผู้วิจัยได้ปรับเปลี่ยนจากการใช้ Pooling หลายระดับเป็นการใช้ Convolution (Conv) ในหลายระดับแทน เรียกว่า Third Conv แสดงดังภาพ 28 โดยภายใน Third Conv ประกอบไปด้วยการดำเนินการ Convolution ขนาด 3 × 3, 5 × 5, และ 7 × 7 ซึ่ง Convolution แต่ละขนาด จะดำเนินการผ่าน Batch Normalization และผ่าน Activation function (Leaky Rectified Linear Unit, Leaky ReLU) จากนั้นนำ Convolution ทั้ง 3 ขนาดมารวมกัน

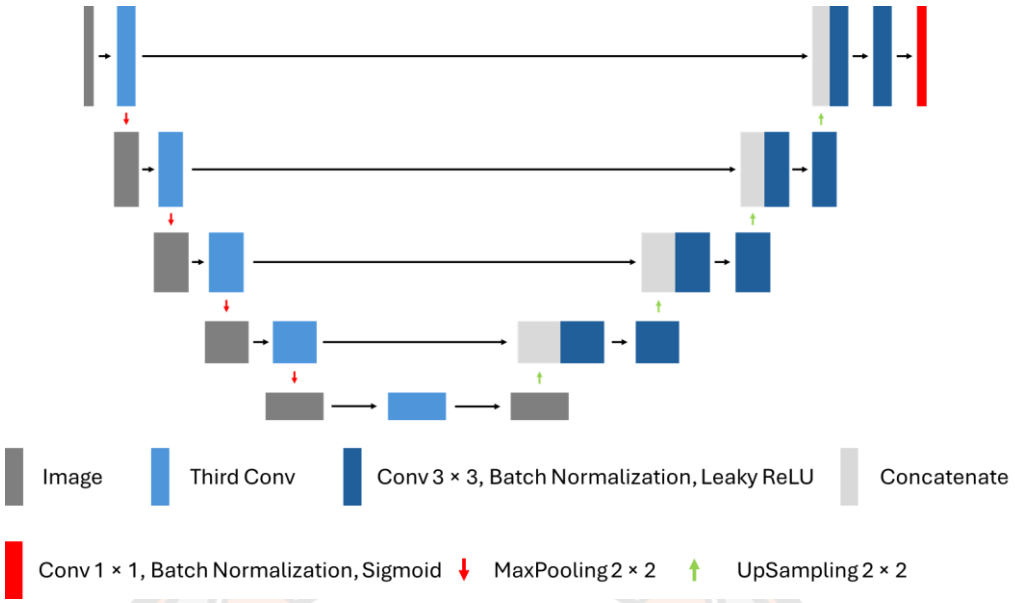


ภาพ 27 Residual Multi-Kernel Pooling ของ Z. Gu และคณะ (51)



ภาพ 28 Third Conv

ในส่วนของตัวเข้ารหัสและตัวถอดรหัสมีพื้นฐานมาจากสถาปัตยกรรม U-net โดยสถาปัตยกรรมแสดงดังภาพ 29 ส่วนของตัวเข้ารหัสนั้น ข้อมูลภาพจะผ่าน Third Conv จากนั้นจะมีการส่งข้อมูลทั้งหมด 2 เส้นทาง โดยเส้นทางตามแนวดิ่ง จะส่งข้อมูลภาพผ่านกระบวนการ Max pooling ขนาด 2 × 2 เพื่อส่งข้อมูลไปยังชั้นถัดไป ขณะที่เส้นทางตามแนวราบจะมีการส่งข้อมูลภาพไปยังฝั่งตัวถอดรหัส สำหรับตัวถอดรหัส มีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการจากส่วนที่อยู่ฝั่งตัวเข้ารหัส เข้ากับข้อมูลภาพที่ผ่านการ Up sampling จากชั้นที่อยู่ก่อนหน้า ซึ่งในแต่ละชั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 ทั้งหมด 2 ครั้ง ในชั้นสุดท้ายของตัวถอดรหัสจะมีการผ่านกระบวนการ Convolution ขนาด 1 × 1 Batch Normalization และ Activation function (Sigmoid) ก่อนให้ผลลัพธ์ออกมา



ภาพ 29 Architecture prototype I

3.4 โมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบ 2

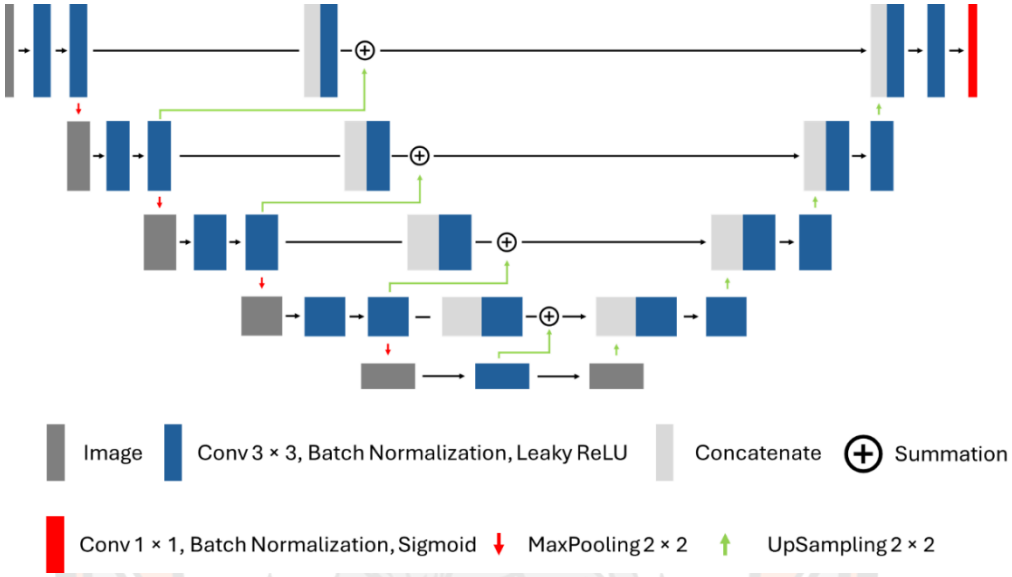
โมเดลที่ 2 แสดงดังภาพ 30 มีการพัฒนาบนพื้นฐานของสถาปัตยกรรมโครงข่าย U-Net และ UNet++ โดยโมเดลมีส่วนของตัวเข้ารหัสและถอดรหัสคล้ายกับสถาปัตยกรรม U-Net แต่มีการปรับปรุงในส่วนที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส ซึ่งแบ่งได้ทั้งหมด 3 ส่วน

ในส่วนของตัวเข้ารหัสนั้น ข้อมูลภาพจะผ่านกระบวนการ Convolution โดยใช้ Kernel ขนาด 3 × 3 ผ่าน Batch Normalization และผ่าน Activation function (Leaky ReLU) ทั้งหมด 2 ครั้ง จากนั้นจะมีการส่งข้อมูลทั้งหมด 2 เส้นทาง โดยเส้นทางตามแนวดิ่ง จะส่งข้อมูลภาพผ่านกระบวนการ Max pooling ขนาด 2 × 2 เพื่อส่งข้อมูลไปยังชั้นถัดไป ขณะที่เส้นทางตามแนวราบจะมีการส่งข้อมูลภาพผ่านกระบวนการที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส

ในส่วนระหว่างตัวเข้ารหัสและตัวถอดรหัส จะมีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการ Convolution โดยใช้ Kernel ขนาด 3 × 3 จากตัวเข้ารหัสที่อยู่ชั้นเดียวกัน จากนั้นข้อมูลภาพจะผ่านกระบวนการ Convolution โดยใช้ Kernel ขนาด 3 × 3 ผ่าน Batch Normalization และ Leaky ReLU เพื่อไปรวมกับข้อมูลภาพที่ผ่านกระบวนการ Convolution ขนาด 3 × 3 และมีการ Up sampling มาจากชั้นถัดไป จากนั้นจะส่งข้อมูลภาพไปเชื่อมต่อกับตัวถอดรหัสที่อยู่ในชั้นเดียวกันต่อไป

ในส่วนของตัวถอดรหัส มีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการจากส่วนที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส เข้ากับข้อมูลภาพที่ผ่านการ Up sampling จากชั้นที่อยู่ก่อนหน้า

ซึ่งในแต่ละชั้นข้อมูลภาพจะผ่านกระบวนการ Convolution โดยใช้ Kernel ขนาด 3 × 3 ทั้งหมด 2 ครั้ง แต่ในชั้นสุดท้ายของตัวถอดรหัสจะมีการผ่านกระบวนการ Convolution อีก 1 ครั้ง โดยผ่านกระบวนการ Convolution ขนาด 1 × 1, Batch Normalization และ Activation function (Sigmoid) ก่อนจะนำออกภาพเพื่อไปใช้งาน



ภาพ 30 Architecture prototype II

3.5 โมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบ 3

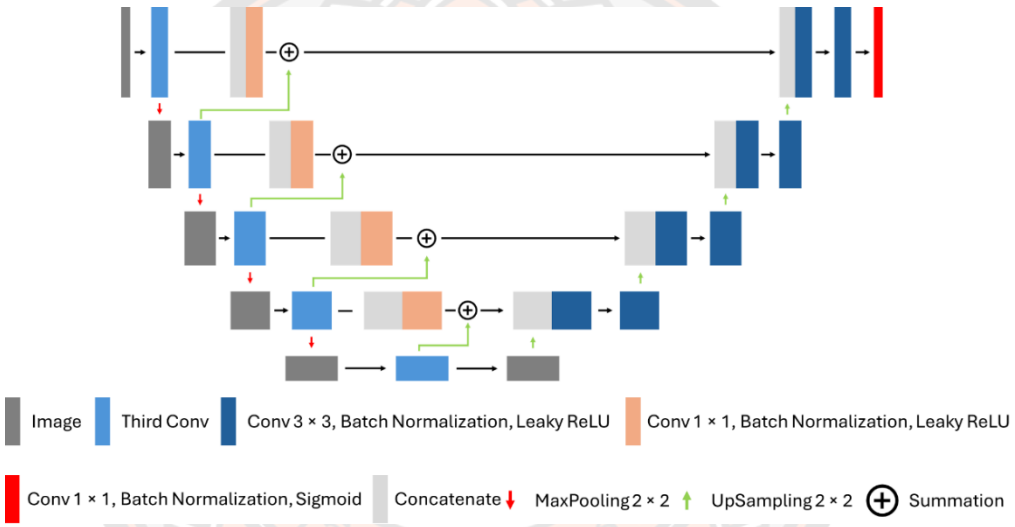
สำหรับโมเดลที่ 3 แสดงดังภาพ 31 โมเดลที่ 3 มีการรวม Third Conv จากโมเดลที่ 1 และ Convolution ที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัสของโมเดลที่ 2 เข้าด้วยกัน ส่วนประกอบอื่น ๆ มีพื้นฐานมาจากสถาปัตยกรรม U-net

ส่วนของตัวเข้ารหัสนั้น ข้อมูลภาพจะผ่าน Third Conv จากนั้นจะมีการส่งข้อมูลทั้งหมด 2 เส้นทาง โดยเส้นทางตามแนวดิ่ง จะส่งข้อมูลภาพผ่านกระบวนการ Max pooling ขนาด 2 × 2 เพื่อส่งข้อมูลไปยังชั้นถัดไป ขณะที่เส้นทางตามแนวราบจะมีการส่งข้อมูลภาพผ่านกระบวนการที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส

ส่วนระหว่างตัวเข้ารหัสและตัวถอดรหัส จะมีการเชื่อมข้อมูลภาพที่ผ่าน Third Conv จากตัวเข้ารหัสที่อยู่ชั้นเดียวกัน จากนั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 1 × 1 ผ่าน Batch Normalization และ Leaky ReLU เพื่อไปรวมกับข้อมูลภาพที่ผ่าน Third Conv

และมีการ Up sampling มาจากชั้นถัดไป จากนั้นจะส่งข้อมูลภาพไปเชื่อมต่อกับตัวถอดรหัสที่อยู่ในชั้นเดียวกันต่อไป

ส่วนของตัวถอดรหัส มีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการจากส่วนที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส เข้ากับข้อมูลภาพที่ผ่านการ up sampling จากชั้นที่อยู่ก่อนหน้า ซึ่งในแต่ละชั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 ทั้งหมด 2 ครั้ง แต่ในชั้นสุดท้ายของตัวถอดรหัสจะมีการผ่านกระบวนการ Convolution อีก 1 ครั้ง โดยผ่านกระบวนการ Convolution ขนาด 1 × 1, Batch Normalization และ Activation function (Sigmoid) ก่อนจะนำออกภาพเพื่อไปใช้งาน



ภาพ 31 Architecture prototype III

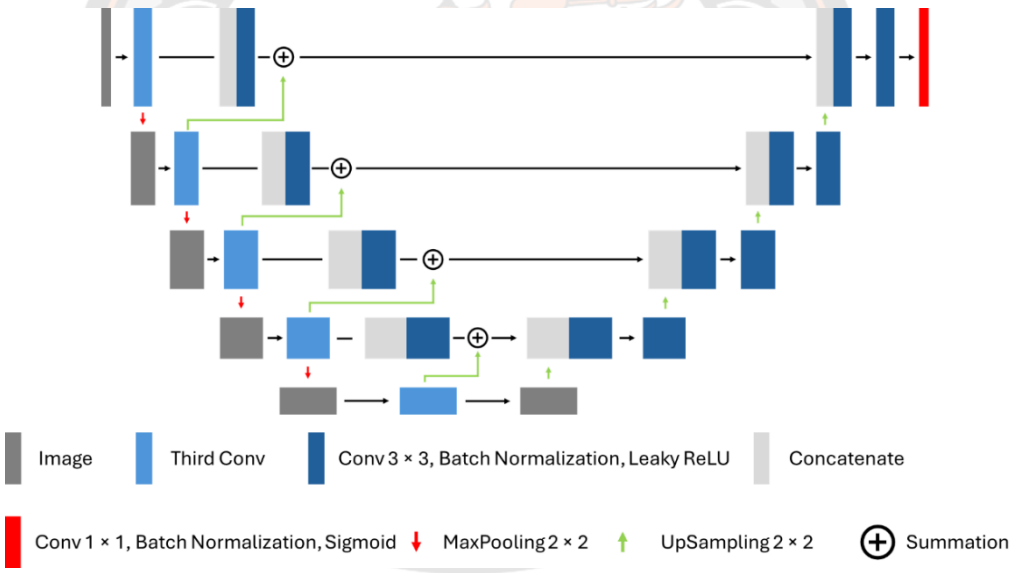
3.6 โมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบ 4

สำหรับโมเดลที่ 4 แสดงดังภาพ 32 โดยโมเดลที่ 4 เหมือนกับโมเดลที่ 3 แต่มีการเปลี่ยนขนาด Kernel ที่ดำเนินการผ่านกระบวนการ Convolution ที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส

ส่วนของตัวเข้ารหัสนั้น ข้อมูลภาพจะผ่าน Third Conv จากนั้นจะมีการส่งข้อมูลทั้งหมด 2 เส้นทาง โดยเส้นทางตามแนวดิ่ง จะส่งข้อมูลภาพผ่านกระบวนการ Max pooling ขนาด 2 × 2 เพื่อส่งข้อมูลไปยังชั้นถัดไป ขณะที่เส้นทางตามแนวราบจะมีการส่งข้อมูลภาพผ่านกระบวนการที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส

ส่วนระหว่างตัวเข้ารหัสและตัวถอดรหัส จะมีการเชื่อมข้อมูลภาพที่ผ่าน Third Conv จากตัวเข้ารหัสที่อยู่ชั้นเดียวกัน จากนั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 ผ่าน Batch Normalization และ Leaky ReLU เพื่อไปรวมกับข้อมูลภาพที่ผ่าน Third Conv และมีการ Up sampling มาจากชั้นถัดไป จากนั้นจะส่งข้อมูลภาพไปเชื่อมต่อกับตัวถอดรหัสที่อยู่ในชั้นเดียวกันต่อไป

ส่วนของตัวถอดรหัส มีการเชื่อมข้อมูลภาพที่ผ่านกระบวนการจากส่วนที่อยู่ระหว่างตัวเข้ารหัสและตัวถอดรหัส เข้ากับข้อมูลภาพที่ผ่านการ up sampling จากชั้นที่อยู่ก่อนหน้า ซึ่งในแต่ละชั้นข้อมูลภาพจะผ่านกระบวนการ Convolution ขนาด 3 × 3 ทั้งหมด 2 ครั้ง แต่ในชั้นสุดท้ายของตัวถอดรหัสจะมีการผ่านกระบวนการ Convolution อีก 1 ครั้ง โดยผ่านกระบวนการ Convolution ขนาด 1 × 1, Batch Normalization และ Activation function (Sigmoid) ก่อนจะนำออกภาพเพื่อไปใช้งาน



ภาพ 32 Architecture prototype IV

4. การเก็บรวบรวมข้อมูล

4.1 นำภาพเอ็มอาร์ไอทั้ง Train set, Validation set, และ Test set จำนวน 1,350, 1,050, และ 600 ชุด ตามลำดับ เพิ่มสัญญาณรบกวนเกาสเซียนสองระดับ และกระบวนการปรับปรุงภาพตามอัลกอริทึมทั้ง 4 แบบ ตามที่กล่าวในข้างต้น และทำการประเมินคุณภาพของการปรับปรุง

ภาพ ซึ่งขั้นตอนการปรับปรุงภาพแสดงดังภาพ 26 และขั้นตอนการประเมินผลของการปรับปรุงภาพแสดงดังภาพ 33

4.2 กำหนดพารามิเตอร์ที่เกี่ยวข้องและแบ่งการโหลดข้อมูลเข้าเนื่องจากไฟล์ที่นำมาใช้มีจำนวนมาก ซึ่งพารามิเตอร์ที่กำหนดมีดังนี้

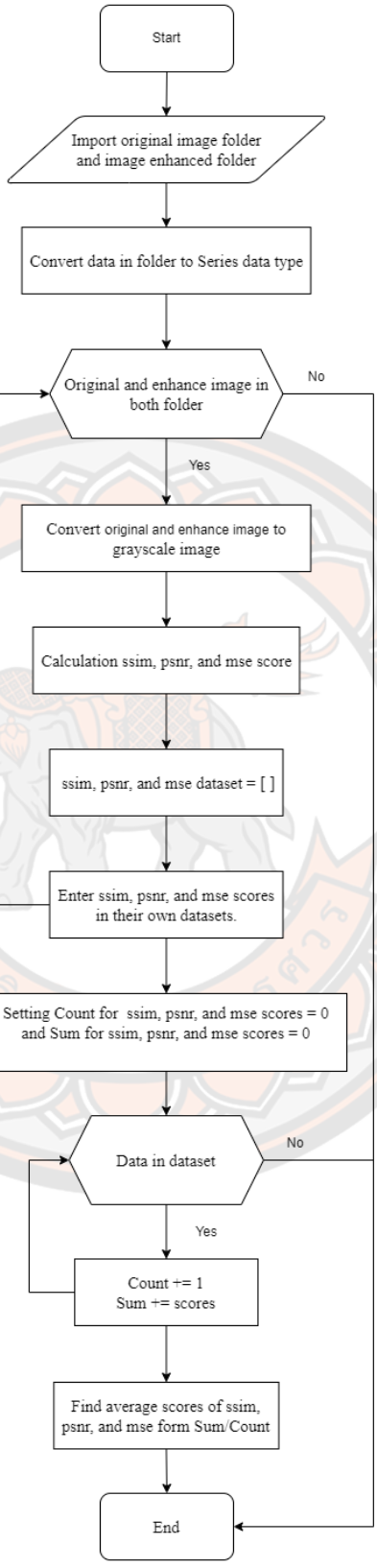
4.2.1 Target size หรือขนาดของภาพเท่ากับ 128 × 128 โดย Normalize ค่าพิกเซลจาก 0-255 เป็น 0 – 1

4.2.2 Batch size กำหนดให้มีค่าเท่ากับ 16

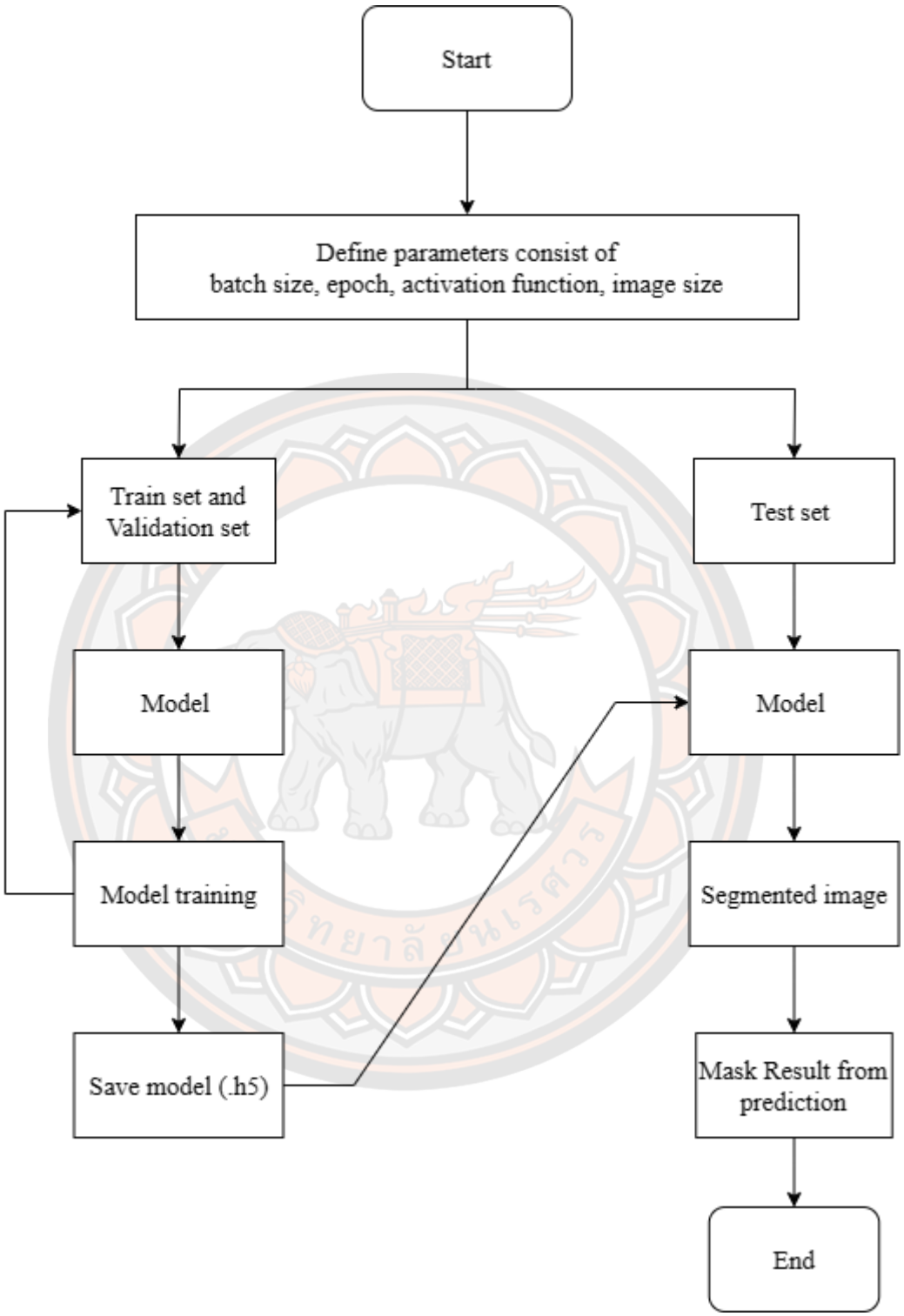
4.2.3 Epoch size กำหนดให้มีค่าเท่ากับ 100

4.2.4 Activation function กำหนดเป็น Leaky Rectified linear unit หรือ Leaky ReLU และ Sigmoid function

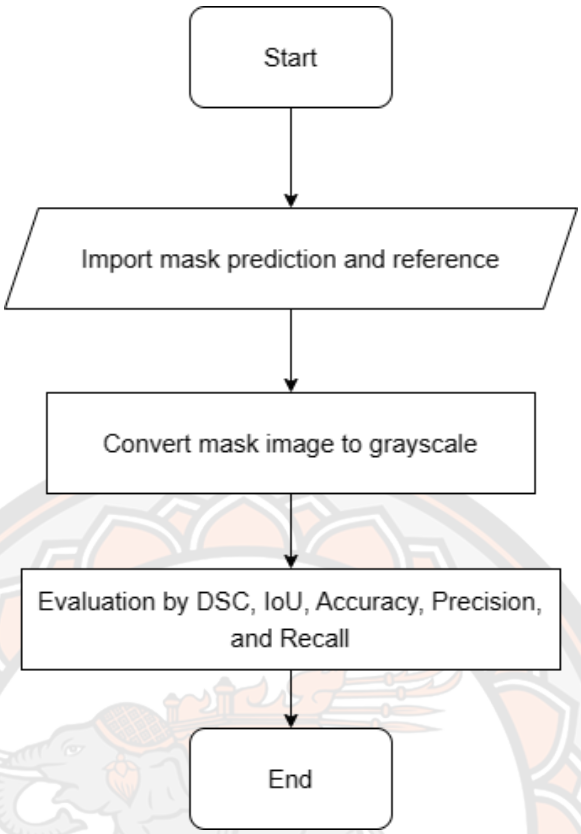
4.3 สร้างโมเดลจากสถาปัตยกรรม U-net, UNet++ และโมเดลที่ออกแบบ ตามลำดับ จากนั้นทำการ train โมเดลจากข้อมูลที่กล่าวมาในข้างต้น เมื่อเสร็จสิ้นการ Train ข้อมูลให้ทำการ Test โมเดล โดยใช้ชุดข้อมูลใน Test set ซึ่งแสดงเป็นแผนผังได้ดังภาพ 34 จากนั้นทำการประเมินโมเดลจากการคำนวณ DSC, IoU, Accuracy, Precision, และ Recall ระหว่างภาพ Mask ที่ได้จากโมเดลกับภาพ Mask ซึ่งแสดงเป็นแผนผังได้ดังภาพ 35



ภาพ 33 แผนผังการประเมินผลการปรับปรุงภาพ



ภาพ 34 แผนผังการ train และ test ของโมเดล



ภาพ 35 แผนผังการประเมินโมเดล

5. วิธีวิเคราะห์ข้อมูล

ในการศึกษานี้มีการประเมินคุณภาพของภาพ ประเมินความคล้ายคลึงของภาพ Mask ที่โมเดลทำนายกับ Mask อ้างอิง รวมถึงประเมินประสิทธิภาพของโมเดล ซึ่งการประเมินคุณภาพของภาพประเมินด้วยค่า Structural Similarity Index (SSIM), Mean squared error (MSE), และ Peak Signal-to-Noise Ratio (PSNR) โดยจะพิจารณาค่า SSIM เป็นอันดับแรก จากนั้นพิจารณาค่า MSE และ PSNR ตามลำดับ การประเมินความคล้ายคลึงของภาพ Mask ที่โมเดลทำนายกับ Mask อ้างอิง ประเมินด้วยค่า DSC และ IoU การประเมินประสิทธิภาพของโมเดลจะประเมินด้วยค่า Accuracy, Precision, และ Recall

เกณฑ์การเลือกพารามิเตอร์ในแต่ละเทคนิคการประเมินคุณภาพของภาพ จะพิจารณาจากผลการประเมินความคล้ายคลึงของภาพ Mask ที่โมเดลทำนายกับ Mask อ้างอิง โดยเลือกค่า DSC สูงสุด ปานกลาง และต่ำสุด ที่โมเดลสามารถทำนายได้ และการเลือกโมเดลที่ผู้วิจัยออกแบบ จะเลือกจากผลลัพธ์ของการประเมินที่มีค่าสูงสุดจากโมเดลที่ถูกฝึกฝนโดยใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบ ชุดข้อมูลที่ไม่ได้เพิ่มสัญญาณรบกวนและไม่มีการปรับปรุงคุณภาพของภาพ

บทที่ 4

ผลการศึกษา

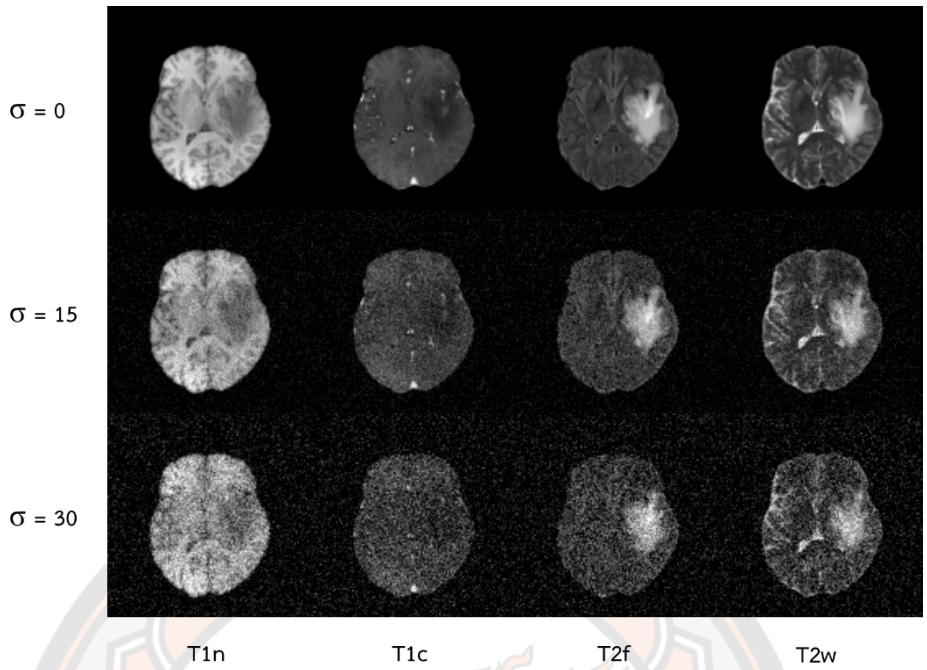
ผลการประเมินคุณภาพของภาพ

1. ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE

ผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค CLAHE พบว่าเมื่อพิจารณาค่า SSIM, MSE, และ PSNR ในทุกสัญญาณรบกวน จะเห็นว่าภาพที่ถูกปรับปรุงคุณภาพด้วยเทคนิค CLAHE โดยใช้พารามิเตอร์ Clip limt (Cl) เท่ากับ 0.1 และ Tile size (Ts) เท่ากับ 2 × 2 มีผลประเมินดีที่สุด ซึ่งในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวนมีค่า SSIM เท่ากับ 0.916 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 1 มีค่า SSIM เท่ากับ 0.989 และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 2 มีค่า SSIM เท่ากับ 0.997 สำหรับข้อมูลพารามิเตอร์อื่น ๆ แสดงดังตาราง 1 รวมถึงชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค CLAHE แสดงดังภาพ 36

ตาราง 1 แสดงผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค CLAHE

Gaussian noise (σ)	Parameter		SSIM	MSE	PSNR
	Cl	Ts			
0	0.1	2	0.916	3.262	43.883
15			0.989	3.222	43.962
30			0.997	2.428	44.944
0	0.2	8	0.711	378.208	22.605
15			0.693	671.432	20.064
30			0.724	903.239	18.911
0	0.3	6	0.740	115.234	28.313
15			0.843	147.261	27.087
30			0.873	181.343	25.763



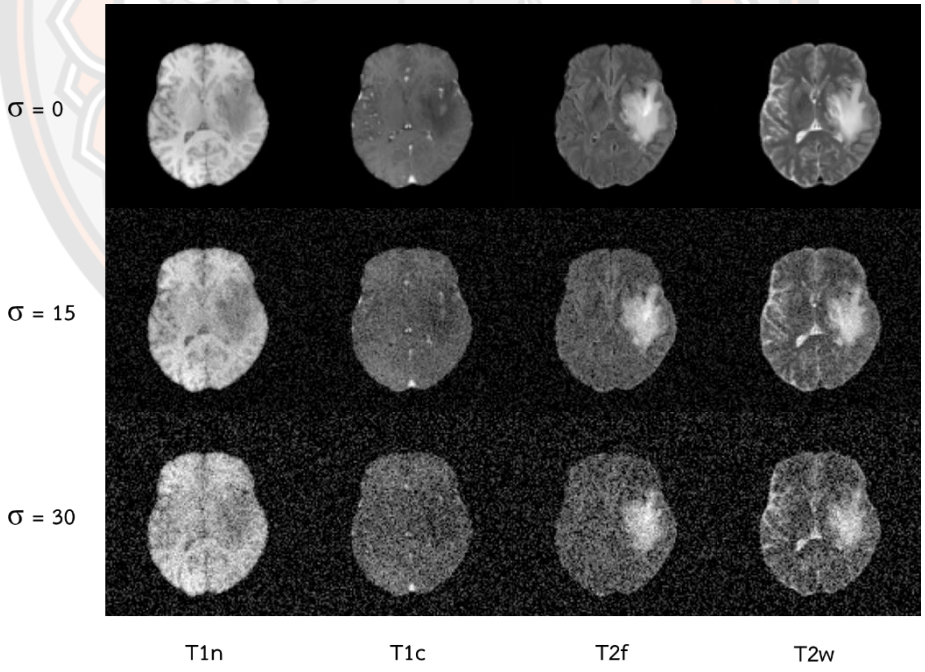
ภาพ 36 ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค CLAHE โดยใช้พารามิเตอร์ Cl เท่ากับ 0.1 และ Ts เท่ากับ 2 × 2

2. ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค GC

ผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค GC พบว่าเมื่อพิจารณาค่า SSIM, MSE, และ PSNR ในทุกสัญญาณรบกวน จะเห็นว่าภาพที่ถูกปรับปรุงคุณภาพด้วยเทคนิค GC โดยใช้พารามิเตอร์ Gamma เท่ากับ 0.7 มีผลประเมินดีที่สุด ซึ่งในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวนมีค่า SSIM เท่ากับ 0.968 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 1 มีค่า SSIM เท่ากับ 0.715 และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 2 มีค่า SSIM เท่ากับ 0.798 สำหรับข้อมูลพารามิเตอร์อื่น ๆ แสดงดังตาราง 2 รวมถึงชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค GC แสดงดังภาพ 37

ตาราง 2 แสดงผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค GC

Gaussian noise (σ)	Gamma Parameter	SSIM	MSE	PSNR
0	0.5	0.917	916.280	18.740
15		0.448	1447.554	16.588
30		0.553	1673.673	15.924
0	0.7	0.968	248.587	24.345
15		0.715	340.583	22.873
30		0.798	402.905	22.108
0	1.5	0.929	300.219	23.433
15		0.541	334.313	22.943
30		0.610	391.052	22.236



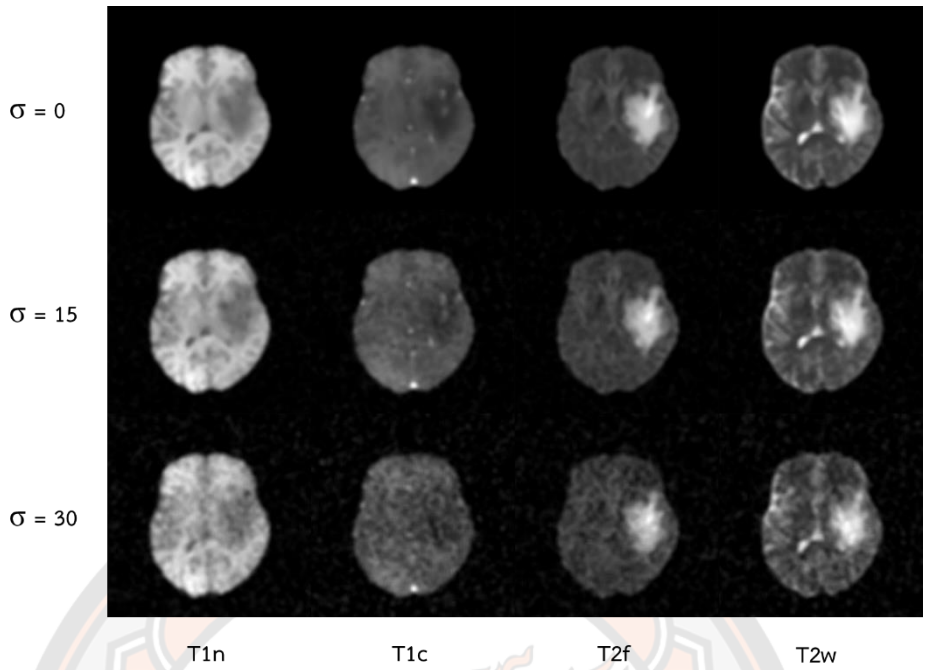
ภาพ 37 ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค GC โดยใช้พารามิเตอร์ Gamma เท่ากับ 0.7

3. ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค MWF

ผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค MWF พบว่าเมื่อพิจารณาค่า SSIM, MSE, และ PSNR ในทุกสัญญาณรบกวน จะเห็นว่าภาพที่ถูกปรับปรุงคุณภาพด้วยเทคนิค MWF โดยใช้พารามิเตอร์ Kernel เท่ากับ 3 × 3 มีผลประเมินดีที่สุด ซึ่งในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวนมีค่า SSIM เท่ากับ 0.936 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 1 มีค่า SSIM เท่ากับ 0.474 และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 2 มีค่า SSIM เท่ากับ 0.310 สำหรับข้อมูลพารามิเตอร์อื่น ๆ แสดงดังตาราง 3 รวมถึงชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค MWF แสดงดังภาพ 38

ตาราง 3 แสดงผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค MWF

Gaussian noise (σ)	Kernel Parameter	SSIM	MSE	PSNR
0	3	0.936	50.027	31.645
15		0.474	160.217	26.138
30		0.310	479.833	21.329
0	5	0.864	114.667	28.183
15		0.335	236.061	24.538
30		0.188	598.196	20.387
0	7	0.811	170.944	26.521
15		0.272	293.314	23.665
30		0.143	667.119	19.930



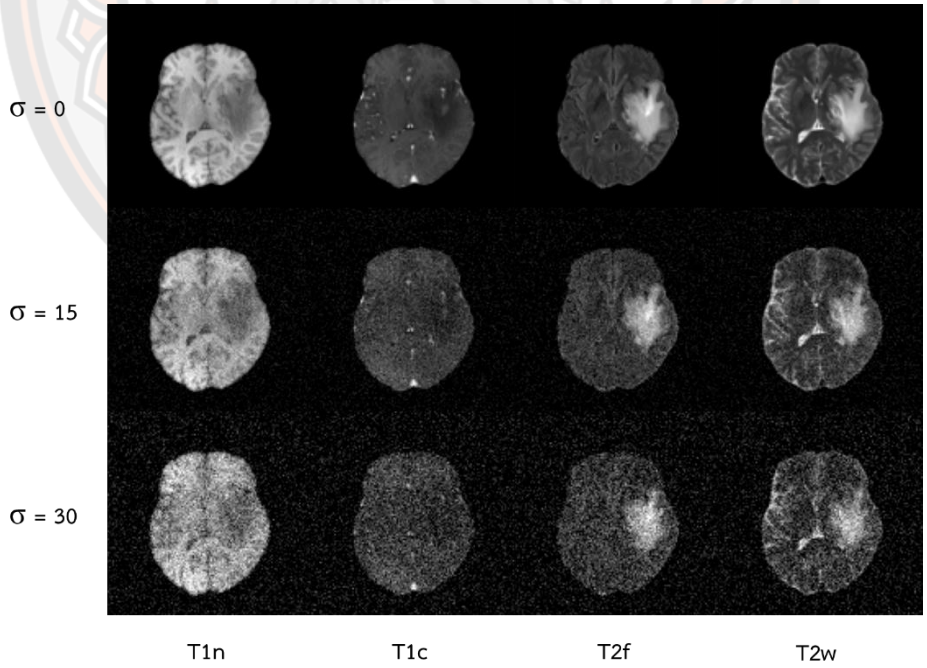
ภาพ 38 ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค MWF โดยใช้พารามิเตอร์ Kernel เท่ากับ 3 × 3

4. ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค NLMF

ผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค NLMF พบว่าเมื่อพิจารณาค่า SSIM, MSE, และ PSNR ในทุกสัญญาณรบกวน จะเห็นว่าภาพที่ถูกปรับปรุงคุณภาพด้วยเทคนิค NLMF โดยใช้พารามิเตอร์ Smooth เท่ากับ 1 Patch เท่ากับ 8 × 8 และ Window เท่ากับ 10 และพารามิเตอร์ Smooth เท่ากับ 2 Patch เท่ากับ 6 × 6 และ Window เท่ากับ 30 มีผลประเมินดีที่สุด โดยทั้งสองชุดพารามิเตอร์ ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวนมีค่า SSIM เท่ากับ 1.000 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 1 มีค่า SSIM เท่ากับ 1.000 และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับ 2 มีค่า SSIM เท่ากับ 1.000 อย่างไรก็ตามในชุดพารามิเตอร์ Smooth เท่ากับ 2 Patch เท่ากับ 6 × 6 และ Window เท่ากับ 30 มีสัญญาณรบกวนเล็กน้อย (MSE เท่ากับ 0.030) สำหรับข้อมูลพารามิเตอร์อื่น ๆ แสดงดังตาราง 4 รวมถึงชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค NLMF แสดงดังภาพ 39 สำหรับค่า PSNR ไม่สามารถคำนวณได้เนื่องจากค่า MSE มีค่าเท่ากับหรือใกล้เคียงศูนย์

ตาราง 4 แสดงผลลัพธ์การประเมินคุณภาพของภาพของชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพโดยใช้เทคนิค NLMF

Gaussian noise (σ)	Parameter			SSIM	MSE	PSNR
	Smooth	Patch	Window			
0	1	8	10	1.000	0.000	-
15				1.000	0.000	-
30				1.000	0.000	-
0	2	2	20	0.999	0.111	-
15				0.999	0.205	-
30				1.000	0.027	-
0	2	6	30	1.000	0.030	-
15				1.000	0.000	-
30				1.000	0.000	-



ภาพ 39 ชุดข้อมูลภาพที่ผ่านการปรับปรุงคุณภาพด้วยเทคนิค NLMF โดยใช้พารามิเตอร์ smooth เท่ากับ 1 patch เท่ากับ 8 และ window เท่ากับ 10

ผลการทดสอบโมเดลที่ออกแบบ

ผลลัพธ์การเปรียบเทียบความคล้ายคลึงของภาพ Mask ที่โมเดลทำนายกับ Mask อ้างอิง รวมถึงการประเมินประสิทธิภาพของโมเดลของชุดข้อมูลภาพที่ไม่ได้เพิ่มสัญญาณรบกวนและไม่ได้รับการปรับปรุงคุณภาพของภาพแสดงดังตาราง 5 พบว่า โมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม U-net, UNet++, Prototype 1, Prototype 2, Prototype 3, และ Prototype 4

มีค่า DSC เท่ากับ 0.807, 0.810, 0.806, 0.807, 0.807, และ 0.811 ตามลำดับ จะเห็นว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 มีค่า DSC สูงสุด

มีค่า IoU เท่ากับ 0.821, 0.837, 0.815, 0.820, 0.822, และ 0.834 ตามลำดับ จะเห็นว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 มีค่า IoU สูง ยกเว้นโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม UNet++

มีค่า Recall เท่ากับ 0.956, 0.958, 0.951, 0.951, 0.952, และ 0.961 ตามลำดับ จะเห็นว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 มีค่า Recall สูงสุด

มีค่า Precision เท่ากับ 0.855, 0.869, 0.851, 0.857, 0.859, และ 0.864 ตามลำดับ จะเห็นว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 มีค่า Precision สูง ยกเว้นโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม UNet++

มีค่า Accuracy เท่ากับ 0.994, 0.995, 0.994, 0.994, 0.995, และ 0.995 ตามลำดับ จะเห็นว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 มีค่า Accuracy เทียบเท่ากับ โมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม UNet++ และ Prototype 3

ตาราง 5 แสดงการเปรียบเทียบผลลัพธ์ในแต่ละโมเดล

Model	DSC	IoU	Recall	Precision	Accuracy
U-net	0.807	0.821	0.956	0.855	0.994
UNet++	0.810	0.837	0.958	0.869	0.995
Prototype 1	0.806	0.815	0.951	0.851	0.994
Prototype 2	0.807	0.820	0.951	0.857	0.994
Prototype 3	0.807	0.822	0.952	0.859	0.995
Prototype 4	0.811	0.834	0.961	0.864	0.995

จากผลลัพธ์ในข้างต้นจะเห็นว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 ให้ผลลัพธ์ที่ดีกว่าโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรมอื่น ๆ (Prototype 1-3) ที่ผู้วิจัยออกแบบ ดังนั้นผู้วิจัยจึงเลือกโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 สำหรับการเปรียบเทียบผลของการฝึกฝนโมเดลในชุดข้อมูลอื่น ๆ อย่างไรก็ตามสถาปัตยกรรม Prototype 4 มีค่า DSC ที่มากกว่าสถาปัตยกรรม UNet++ แต่กลับมีผลลัพธ์ค่า Precision ที่น้อยกว่า เนื่องจากโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม Prototype 4 ให้ผล True positives ที่มากกว่า บ่งบอกถึงการมีค่า Recall ที่สูงขึ้น และ False positives ที่มากเช่นกัน ซึ่งหมายความว่าโมเดลทำนายว่าจริงแต่ผลลัพธ์เป็นเท็จได้มากเช่นกัน ดังนั้นค่า Precision จึงลดลง ในทางกลับกันโมเดลที่ถูกฝึกฝนด้วยสถาปัตยกรรม UNet++ มีค่า Precision ที่สูงกว่า ซึ่งบ่งบอกถึงการหลีกเลี่ยงผลของ False positives ได้ดีกว่า ส่งผลให้ค่า IoU สูงขึ้น ในขณะเดียวกันกลับส่งผลให้ค่า True positives น้อยลง ทำให้ค่า Recall และ ค่า DSC น้อยลงตามไปด้วย

ผลการประเมินความคล้ายคลึงของภาพ Mask ที่โมเดลทำนายกับ Mask อ้างอิง และผลลัพธ์การประเมินประสิทธิภาพของโมเดล

1. ชุดข้อมูลการไม่มีการเพิ่มสัญญาณรบกวน และโมเดลใช้สถาปัตยกรรม U-net

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม U-net ที่ใช้ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.807 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.807, 0.805, และ 0.802 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.799, 0.809, และ 0.806 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.799, 0.793, และ 0.772 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.798, 0.806, และ 0.808 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 6 และภาพ 40 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 6 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม U-net ที่ใช้ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.807	0.821	0.956	0.855	0.994
CLAHE model1	Clip limit = 0.1 Tile = 2	0.807	0.897	0.954	0.857	0.995
CLAHE model2	Clip limit = 0.2 Tile = 8	0.805	0.893	0.948	0.857	0.994
CLAHE model3	Clip limit = 0.3 Tile = 6	0.802	0.887	0.945	0.849	0.994
GC model1	Gamma = 0.5	0.799	0.883	0.939	0.848	0.994
GC model2	Gamma = 0.7	0.809	0.898	0.956	0.858	0.995
GC model3	Gamma = 1.5	0.806	0.896	0.949	0.860	0.995
MWF model1	Kernel = 3	0.799	0.882	0.940	0.843	0.994
MWF model2	Kernel = 5	0.793	0.872	0.930	0.832	0.993
MWF model3	Kernel = 7	0.772	0.839	0.896	0.802	0.991
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.798	0.884	0.941	0.847	0.994
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.806	0.894	0.951	0.856	0.994
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.808	0.901	0.956	0.862	0.995

ภาพ 40 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรม U-net และชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

2. ชุดข้อมูลการเพิ่มสัญญาณรบกวนระดับที่ 1 และโมเดลใช้สถาปัตยกรรม U-net

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม U-net ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1 พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.800 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.800, 0.799, และ 0.801 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.800, 0.799, และ 0.798 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.791, 0.787, และ 0.780 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.798, 0.801, และ 0.801 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 7 และภาพ 41 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 7 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม U-net ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.800	0.885	0.941	0.846	0.994
CLAHE model1	Clip limit = 0.1 Tile = 2	0.800	0.884	0.940	0.846	0.994
CLAHE model2	Clip limit = 0.2 Tile = 8	0.799	0.881	0.934	0.847	0.994
CLAHE model3	Clip limit = 0.3 Tile = 6	0.801	0.884	0.940	0.847	0.994
GC model1	Gamma = 0.5	0.800	0.885	0.939	0.847	0.994
GC model2	Gamma = 0.7	0.799	0.883	0.937	0.847	0.994
GC model3	Gamma = 1.5	0.798	0.883	0.937	0.846	0.994
MWF model1	Kernel = 3	0.791	0.871	0.922	0.838	0.993
MWF model2	Kernel = 5	0.787	0.861	0.917	0.825	0.993
MWF model3	Kernel = 7	0.780	0.851	0.906	0.814	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.798	0.882	0.937	0.846	0.994
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.801	0.886	0.942	0.848	0.994
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.801	0.885	0.940	0.848	0.994

ภาพ 41 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรม U-net และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

3. ชุดข้อมูลการเพิ่มสัญญาณรบกวนระดับที่ 2 และโมเดลใช้สถาปัตยกรรม U-net

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม U-net ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2 พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.789 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.789, 0.781, และ 0.791 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.787, 0.793, และ 0.791 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.787, 0.780, และ 0.775 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.789, 0.788, และ 0.791 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 8 และภาพ 42 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 8 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม U-net ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.789	0.867	0.919	0.834	0.993
CLAHE model1	Clip limit = 0.1 Tile = 2	0.789	0.865	0.919	0.830	0.993
CLAHE model2	Clip limit = 0.2 Tile = 8	0.781	0.854	0.908	0.821	0.992
CLAHE model3	Clip limit = 0.3 Tile = 6	0.791	0.869	0.921	0.835	0.993
GC model1	Gamma = 0.5	0.787	0.862	0.913	0.829	0.993
GC model2	Gamma = 0.7	0.793	0.870	0.923	0.835	0.993
GC model3	Gamma = 1.5	0.791	0.868	0.921	0.833	0.993
MWF model1	Kernel = 3	0.787	0.865	0.915	0.833	0.993
MWF model2	Kernel = 5	0.780	0.854	0.902	0.825	0.992
MWF model3	Kernel = 7	0.775	0.842	0.897	0.805	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.789	0.866	0.920	0.829	0.993
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.788	0.865	0.916	0.834	0.993
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.791	0.869	0.921	0.835	0.993

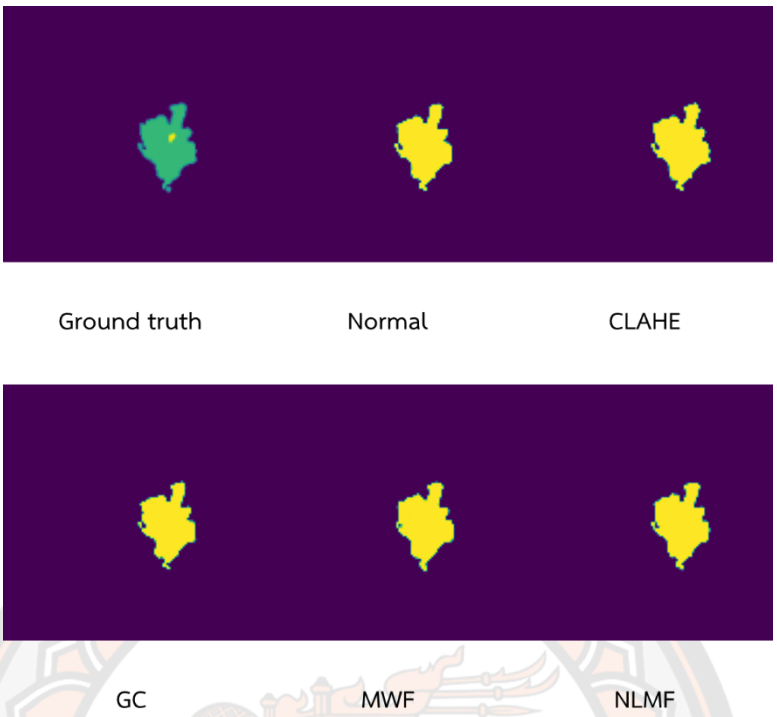
ภาพ 42 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรม U-net และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

4. ชุดข้อมูลการไม่มีการเพิ่มสัญญาณรบกวน และโมเดลใช้สถาปัตยกรรม UNet++

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ใช้ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.810 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.810, 0.808, และ 0.806 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.806, 0.803, และ 0.806 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.806, 0.797, และ 0.784 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.802, 0.810, และ 0.808 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 9 และภาพ 43 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 9 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ใช้ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.810	0.837	0.958	0.869	0.995
CLAHE model1	Clip limit = 0.1 Tile = 2	0.810	0.903	0.958	0.863	0.995
CLAHE model2	Clip limit = 0.2 Tile = 8	0.808	0.904	0.951	0.870	0.995
CLAHE model3	Clip limit = 0.3 Tile = 6	0.806	0.906	0.948	0.878	0.995
GC model1	Gamma = 0.5	0.806	0.896	0.949	0.861	0.995
GC model2	Gamma = 0.7	0.803	0.892	0.947	0.856	0.994
GC model3	Gamma = 1.5	0.806	0.894	0.952	0.856	0.994
MWF model1	Kernel = 3	0.806	0.897	0.949	0.860	0.995
MWF model2	Kernel = 5	0.797	0.879	0.934	0.841	0.994
MWF model3	Kernel = 7	0.784	0.860	0.915	0.822	0.993
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.802	0.890	0.945	0.853	0.994
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.810	0.902	0.956	0.865	0.995
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.808	0.903	0.953	0.868	0.995



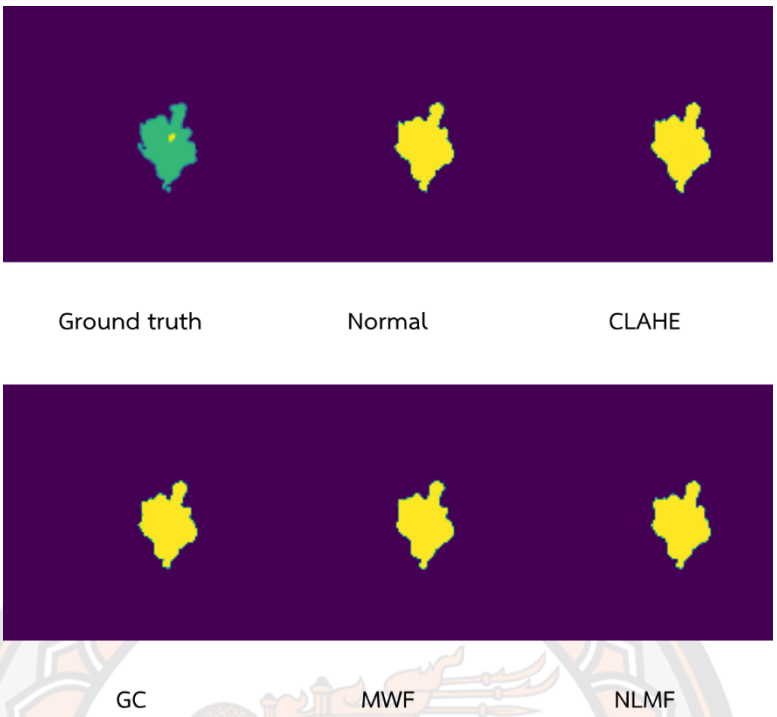
ภาพ 43 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรม U-net++ และชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

5. ชุดข้อมูลการเพิ่มสัญญาณรบกวนระดับที่ 1 และโมเดลใช้สถาปัตยกรรม UNet++

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1 พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.803 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.803, 0.798, และ 0.804 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.797, 0.802, และ 0.801 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.797, 0.789, และ 0.781 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.801, 0.796, และ 0.802 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 10 และภาพ 44 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 10 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.803	0.888	0.943	0.852	0.994
CLAHE model1	Clip limit = 0.1 Tile = 2	0.803	0.891	0.941	0.855	0.994
CLAHE model2	Clip limit = 0.2 Tile = 8	0.798	0.883	0.937	0.848	0.994
CLAHE model3	Clip limit = 0.3 Tile = 6	0.804	0.894	0.943	0.860	0.995
GC model1	Gamma = 0.5	0.797	0.877	0.931	0.841	0.993
GC model2	Gamma = 0.7	0.802	0.888	0.941	0.851	0.994
GC model3	Gamma = 1.5	0.801	0.884	0.940	0.847	0.994
MWF model1	Kernel = 3	0.797	0.881	0.932	0.848	0.994
MWF model2	Kernel = 5	0.789	0.865	0.915	0.835	0.993
MWF model3	Kernel = 7	0.781	0.852	0.904	0.817	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.801	0.885	0.938	0.849	0.994
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.796	0.880	0.932	0.846	0.994
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.802	0.886	0.939	0.851	0.994



ภาพ 44 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรม U-net++ และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

6. ชุดข้อมูลการเพิ่มสัญญาณรบกวนระดับที่ 2 และโมเดลใช้สถาปัตยกรรม UNet++

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2 พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.792 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.791, 0.782, และ 0.792 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.790, 0.790, และ 0.793 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.789, 0.781, และ 0.775 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.793, 0.790, และ 0.790 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 11 และภาพ 45 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 11 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.792	0.872	0.925	0.837	0.993
CLAHE model1	Clip limit = 0.1 Tile = 2	0.791	0.871	0.922	0.838	0.993
CLAHE model2	Clip limit = 0.2 Tile = 8	0.782	0.856	0.909	0.823	0.993
CLAHE model3	Clip limit = 0.3 Tile = 6	0.792	0.873	0.921	0.840	0.993
GC model1	Gamma = 0.5	0.790	0.867	0.919	0.833	0.993
GC model2	Gamma = 0.7	0.790	0.868	0.921	0.835	0.993
GC model3	Gamma = 1.5	0.793	0.870	0.922	0.836	0.993
MWF model1	Kernel = 3	0.789	0.867	0.918	0.834	0.993
MWF model2	Kernel = 5	0.781	0.853	0.906	0.819	0.992
MWF model3	Kernel = 7	0.775	0.844	0.897	0.810	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.793	0.874	0.923	0.841	0.993
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.790	0.870	0.921	0.836	0.993
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.790	0.868	0.919	0.836	0.993

ภาพ 45 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรม U-net++ และชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

7. ชุดข้อมูลการไม่มีการเพิ่มสัญญาณรบกวน และโมเดลใช้สถาปัตยกรรม Prototype

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม Prototype ที่ใช้ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.811 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.803, 0.807, และ 0.803 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.809, 0.803, และ 0.805 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.807, 0.795, และ 0.782 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.803, 0.804, และ 0.805 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 12 และภาพ 46 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 12 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม Prototype ที่ใช้ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.811	0.904	0.961	0.864	0.995
CLAHE model1	Clip limit = 0.1 Tile = 2	0.803	0.889	0.947	0.850	0.994
CLAHE model2	Clip limit = 0.2 Tile = 8	0.807	0.895	0.952	0.856	0.994
CLAHE model3	Clip limit = 0.3 Tile = 6	0.803	0.889	0.946	0.851	0.994
GC model1	Gamma = 0.5	0.809	0.898	0.956	0.858	0.995
GC model2	Gamma = 0.7	0.803	0.889	0.945	0.852	0.994
GC model3	Gamma = 1.5	0.805	0.893	0.949	0.854	0.994
MWF model1	Kernel = 3	0.807	0.896	0.951	0.858	0.995
MWF model2	Kernel = 5	0.795	0.877	0.934	0.838	0.994
MWF model3	Kernel = 7	0.782	0.856	0.911	0.818	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.803	0.890	0.949	0.850	0.994
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.804	0.893	0.947	0.856	0.994
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.805	0.892	0.950	0.852	0.994

ภาพ 46 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบและชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

8. ชุดข้อมูลการเพิ่มสัญญาณรบกวนระดับที่ 1 และโมเดลใช้สถาปัตยกรรม Prototype

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม Prototype ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1 พบว่าโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.798 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.798, 0.797, และ 0.802 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.798, 0.796, และ 0.802 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.798, 0.780, และ 0.778 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.802, 0.802, และ 0.797 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 13 และภาพ 47 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 13 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม Prototype ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.798	0.880	0.934	0.845	0.994
CLAHE model1	Clip limit = 0.1 Tile = 2	0.798	0.880	0.935	0.842	0.994
CLAHE model2	Clip limit = 0.2 Tile = 8	0.797	0.876	0.931	0.841	0.993
CLAHE model3	Clip limit = 0.3 Tile = 6	0.802	0.886	0.943	0.847	0.994
GC model1	Gamma = 0.5	0.798	0.879	0.935	0.841	0.994
GC model2	Gamma = 0.7	0.796	0.877	0.932	0.840	0.993
GC model3	Gamma = 1.5	0.802	0.887	0.941	0.850	0.994
MWF model1	Kernel = 3	0.798	0.881	0.936	0.843	0.994
MWF model2	Kernel = 5	0.780	0.856	0.909	0.821	0.993
MWF model3	Kernel = 7	0.778	0.849	0.903	0.812	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.802	0.886	0.941	0.849	0.994
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.802	0.886	0.942	0.849	0.994
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.797	0.879	0.931	0.845	0.994

ภาพ 47 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบและชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

9. ชุดข้อมูลการเพิ่มสัญญาณรบกวนระดับที่ 2 และโมเดลใช้สถาปัตยกรรม Prototype

ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม Prototype ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2 พบว่า โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ไม่ได้รับการปรับปรุงคุณภาพของภาพให้ค่า DSC เท่ากับ 0.786 โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค CLAHE โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.789, 0.785, และ 0.795 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค GC โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.792, 0.788, และ 0.789 ตามลำดับ โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค MWF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.787, 0.778, และ 0.776 ตามลำดับ และโมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลภาพที่ได้กับการปรุงภาพด้วยเทคนิค NLMF โมเดลที่ 1, โมเดลที่ 2, และโมเดลที่ 3 มีค่า DSC เท่ากับ 0.793, 0.788, และ 0.792 ตามลำดับ สำหรับค่าอื่น ๆ แสดงดังตาราง 14 และภาพ 48 แสดง Mask ที่โมเดลทำนาย ในแต่ละเทคนิคการปรับปรุงคุณภาพของภาพที่มีค่า DSC สูงสุด

ตาราง 14 ผลลัพธ์การประเมินความคล้ายคลึงและผลลัพธ์การประเมินประสิทธิภาพของโมเดลที่ใช้สถาปัตยกรรม Prototype ที่ใช้ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Model	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.786	0.862	0.915	0.827	0.993
CLAHE model1	Clip limit = 0.1 Tile = 2	0.789	0.865	0.918	0.830	0.993
CLAHE model2	Clip limit = 0.2 Tile = 8	0.785	0.858	0.914	0.822	0.993
CLAHE model3	Clip limit = 0.3 Tile = 6	0.795	0.873	0.927	0.836	0.993
GC model1	Gamma = 0.5	0.792	0.868	0.923	0.832	0.993
GC model2	Gamma = 0.7	0.788	0.863	0.916	0.828	0.993
GC model3	Gamma = 1.5	0.789	0.864	0.917	0.829	0.993
MWF model1	Kernel = 3	0.787	0.861	0.913	0.827	0.993
MWF model2	Kernel = 5	0.778	0.848	0.903	0.812	0.992
MWF model3	Kernel = 7	0.776	0.843	0.900	0.806	0.992
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.793	0.870	0.923	0.833	0.993
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.788	0.865	0.918	0.830	0.993
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.792	0.869	0.923	0.833	0.993

ภาพ 48 Mask ต้นฉบับกับ Mask ที่ทำนายของโมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบและชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

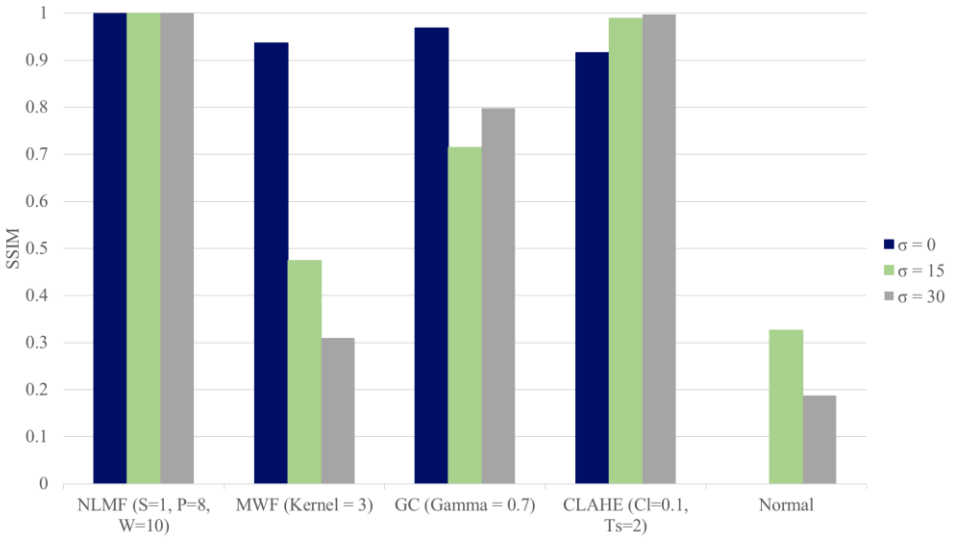
บทที่ 5

บทสรุป

งานวิจัยนี้พัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยใช้สถาปัตยกรรม U-net, UNet++, และสถาปัตยกรรมที่ผู้วิจัยออกแบบ รวมถึงศึกษาผลของกระบวนการก่อนการประมวลผลภาพเพื่อช่วยเพิ่มประสิทธิภาพของโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยใช้เทคนิคการปรับปรุงคุณภาพของภาพ 4 เทคนิค ได้แก่ Contrast-limited adaptive histogram equalization, Gamma correction, Median and Wiener filter, และ Non-local mean filter

อภิปรายผลของกระบวนการก่อนการประมวลผลภาพ

การศึกษาผลของกระบวนการก่อนการประมวลผลภาพเพื่อช่วยเพิ่มประสิทธิภาพของโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง พบว่า ในชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน เทคนิคที่สามารถเพิ่มคุณภาพของภาพได้ดีที่สุดคือ NLMF, GC, MWF, และ CLAHE ตามลำดับ สังเกตได้จากแท่งกราฟสีน้ำเงินในภาพ 49 และในชุดข้อมูลที่มีการเพิ่มสัญญาณรบกวน เทคนิคที่สามารถเพิ่มคุณภาพของภาพได้ดีที่สุดคือ NLMF, CLAHE, GC, และ MWF ตามลำดับ สังเกตได้จากแท่งกราฟสีเขียวและสีเทาในภาพ 49



ภาพ 49 การเปรียบเทียบค่า SSIM ในเทคนิคการเพิ่มคุณภาพ

เปรียบเทียบประสิทธิภาพของโมเดลกับคุณภาพของภาพที่ได้รับการปรังปรุง

คุณภาพของภาพส่งผลต่อประสิทธิภาพการทำงานของโมเดลการเรียนรู้เชิงลึก จากตาราง 15 เมื่อเปรียบเทียบเทคนิคเดียวกัน โมเดลที่ใช้สถาปัตยกรรม U-net, UNet++, และสถาปัตยกรรมที่ผู้วิจัยออกแบบ และถูกฝึกฝนด้วยเทคนิค CLAHE ด้วยพารามิเตอร์ Clip limt เท่ากับ 0.2 และ Tile size เท่ากับ 8 × 8 (CLAHE model2) มีค่า DSC เท่ากับ 0.805, 0.808, และ 0.807 ตามลำดับ และมีค่า SSIM เท่ากับ 0.711 เมื่อเทียบกับโมเดลที่ถูกฝึกฝนด้วยเทคนิคเดียวกัน (CLAHE model3) แต่ปรับพารามิเตอร์ (Clip limt เท่ากับ 0.3 และ Tile size เท่ากับ 6 × 6) มีค่า DSC เท่ากับ 0.802, 0.806, และ 0.803 ตามลำดับ และมีค่า SSIM เท่ากับ 0.740 จะเห็นว่าคุณภาพของภาพที่ใช้ฝึกฝนโมเดล CLAHE model3 มีคุณภาพของภาพที่ดีกว่า แต่กลับทำให้ประสิทธิของโมเดลลดลง นอกจากนี้โมเดลที่ถูกฝึกฝนด้วยชุดข้อมูลที่มีคุณภาพของภาพเท่ากัน ส่งผลให้ประสิทธิของโมเดลแตกต่าง สังเกตได้จากโมเดล NLMF model1 กับ NLMF model3 จากผลดังกล่าวจะเห็นว่าคุณภาพของภาพที่ด้อยกว่าอาจทำให้ประสิทธิภาพของโมเดลดีขึ้น ขณะที่การใช้ชุดข้อมูลที่มีคุณภาพภาพเทียบเท่ากับ แต่มีการใช้เทคนิคการเพิ่มคุณภาพที่ต่างกัน ส่งผลให้ประสิทธิภาพของโมเดลแตกต่างกันตามไปด้วย สังเกตได้จาก CLAHE model1 และ GC model1

ตาราง 15 การเปรียบเทียบผลลัพธ์ของโมเดลการเรียนรู้เชิงลึกและการปรับปรุงคุณภาพของภาพ (ชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน)

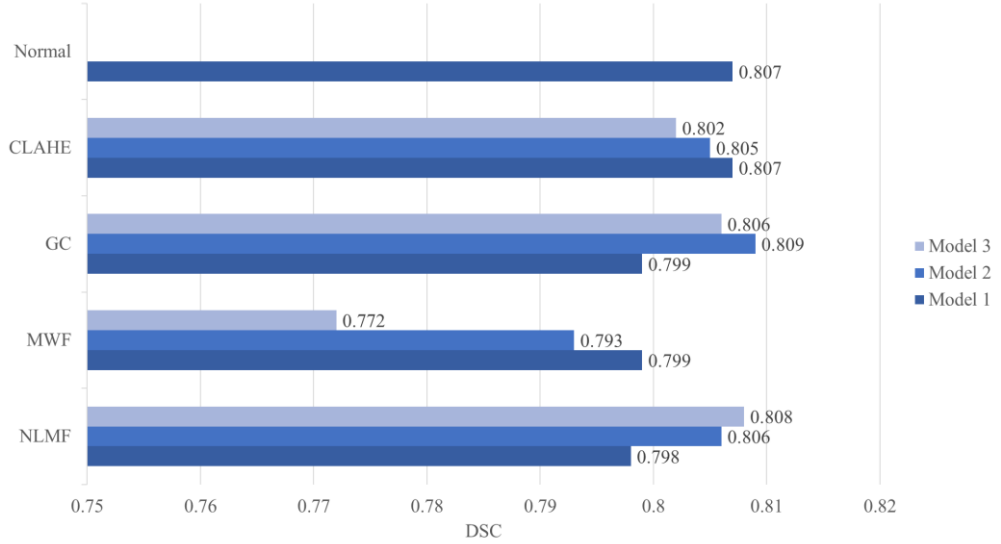
Enhanced dataset model	Parameters	DSC of U-net architecture	DSC of UNet++ architecture	DSC of Prototype architecture	SSIM
Normal	-	0.807	0.810	0.811	-
CLAHE model1	Clip limit = 0.1 Tile = 2	0.807	0.810	0.803	0.916
CLAHE model2	Clip limit = 0.2 Tile = 8	0.805	0.808	0.807	0.711
CLAHE model3	Clip limit = 0.3	0.802	0.806	0.803	0.740

Enhanced dataset model	Parameters	DSC of U-net architecture	DSC of UNet++ architecture	DSC of Prototype architecture	SSIM
	Tile = 6				
GC model1	Gamma = 0.5	0.799	0.806	0.809	0.917
GC model2	Gamma = 0.7	0.809	0.803	0.803	0.968
GC model3	Gamma = 1.5	0.806	0.806	0.805	0.929
MWF model1	Kernel = 3	0.799	0.806	0.807	0.936
MWF model2	Kernel = 5	0.793	0.797	0.795	0.864
MWF model3	Kernel = 7	0.772	0.784	0.782	0.811
NLMF model1	Smooth = 1 Patch = 8 Window = 10	0.798	0.802	0.803	1.000
NLMF model2	Smooth = 2 Patch = 2 Window = 20	0.806	0.810	0.804	0.999
NLMF model3	Smooth = 2 Patch = 6 Window = 30	0.808	0.808	0.805	1.000

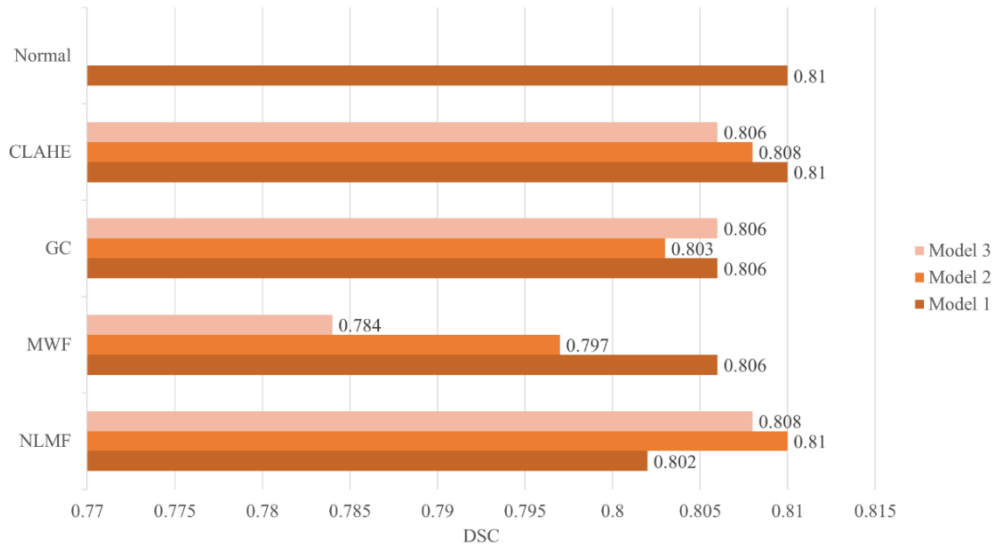
อภิปรายผลการพัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพ

จากงานวิจัยพัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองพบว่า เมื่อเปรียบเทียบโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยใช้ชุดข้อมูลที่ไม่ผ่านการเพิ่มสัญญาณรบกวนและไม่มีการเพิ่มคุณภาพของ โดยโมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกแบบลำดับที่ 4 ให้ผลลัพธ์การประเมินที่ดีกว่าสถาปัตยกรรมที่ผู้วิจัยออกแบบลำดับที่ 1 ถึง 3 สำหรับทุกค่าการประเมิน รวมถึงให้ผลลัพธ์ที่ดีกว่าในโมเดลที่ใช้สถาปัตยกรรม U-net และ UNet++ ยกเว้นโมเดลที่ใช้สถาปัตยกรรม UNet++ ที่ให้ค่า precision ที่มากกว่า ซึ่งสังเกตได้จากตาราง 5 ดังนั้นสำหรับการศึกษานี้ผู้วิจัยจึงเลือกโมเดลที่ใช้สถาปัตยกรรมที่ผู้วิจัยออกลำดับที่ 4 ในการฝึกฝนโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง

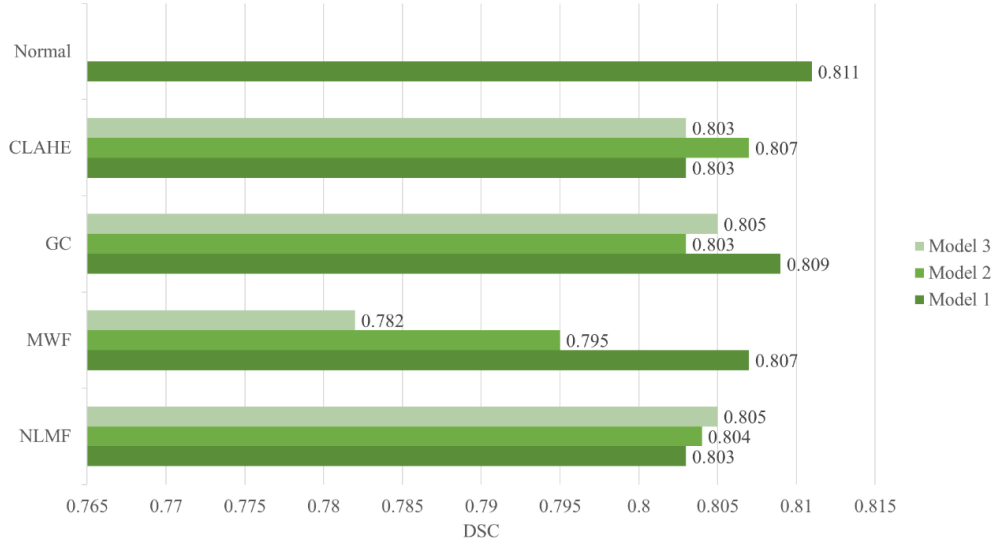
ผลลัพธ์ของโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง ที่ถูกฝึกฝนด้วยชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน พบว่า โมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net มีค่า DSC เท่ากับ 0.807 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค GC model2 และ NLMF model3 มีค่า DSC ที่มากกว่า ขณะที่เทคนิค CLAHE model1 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 50 โมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ ในชุดข้อมูลที่ไม่ผ่านการปรับปรุงคุณภาพของภาพมีค่า DSC เท่ากับ 0.810 ขณะที่ไม่มีชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพให้ค่า DSC ที่มากกว่า อย่างไรก็ตามมีชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE model1 และ NLMF model2 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 51 โมเดลที่ฝึกฝนด้วยสถาปัตยกรรมที่ผู้วิจัยออกแบบ ในชุดข้อมูลที่ไม่ผ่านการปรับปรุงคุณภาพของภาพมีค่า DSC เท่ากับ 0.811 ไม่มีชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพที่ให้ค่า DSC ที่มากว่าหรือเทียบเท่าสังเกตได้จากภาพ 52 โดยภาพผลการทำนาย Mask ของโมเดลแสดงดังภาพ 53



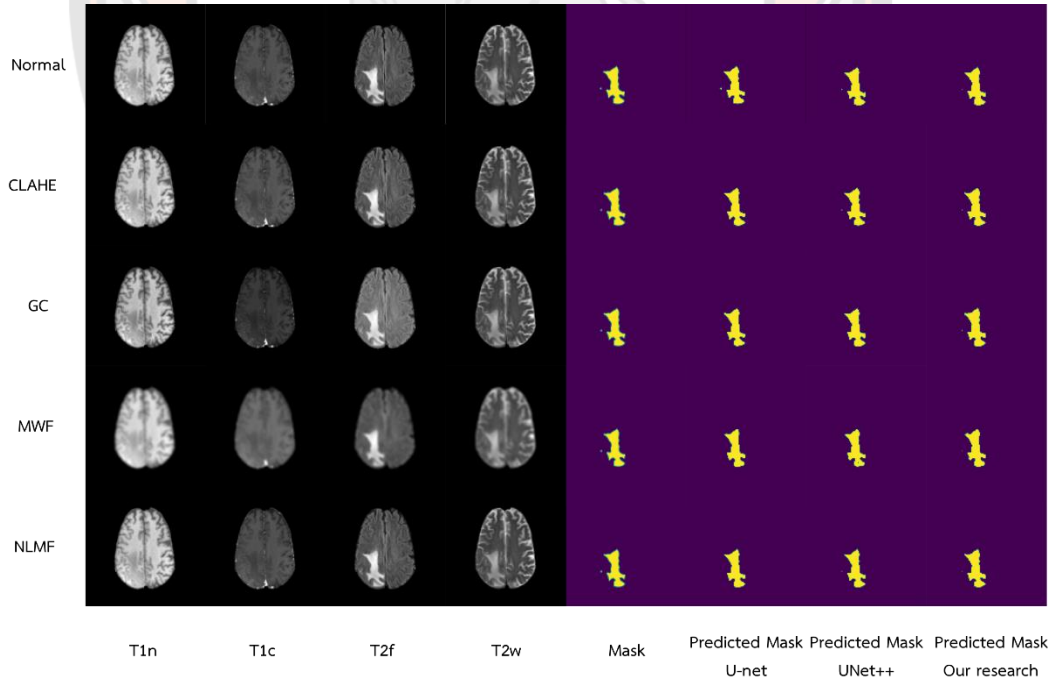
ภาพ 50 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net และชุดข้อมูลที่ไม่ผ่านการเพิ่มสัญญาณรบกวน



ภาพ 51 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ และชุดข้อมูลที่ไม่ผ่านการเพิ่มสัญญาณรบกวน

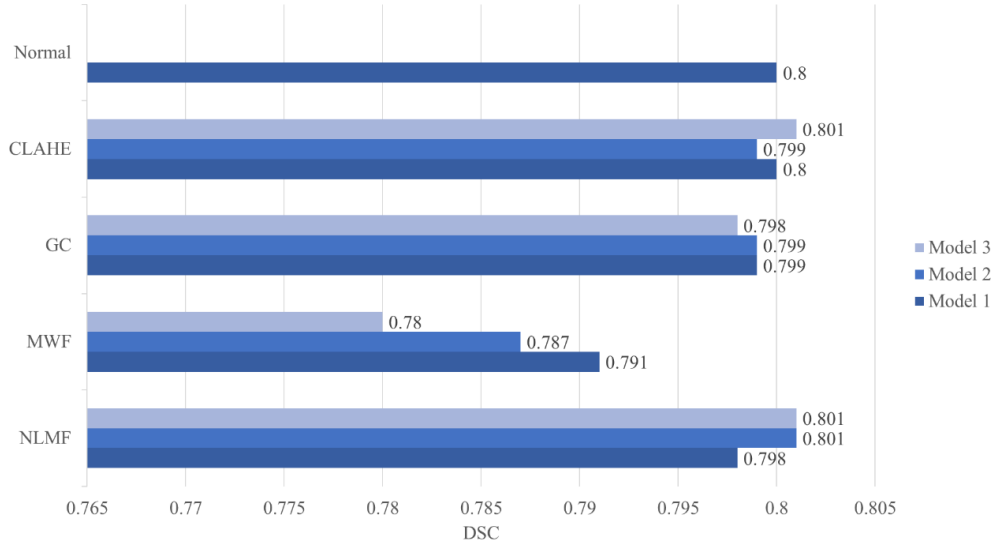


ภาพ 52 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรมที่ผู้วิจัยออกแบบและชุดข้อมูลที่ไม่ผ่านการเพิ่มสัญญาณรบกวน

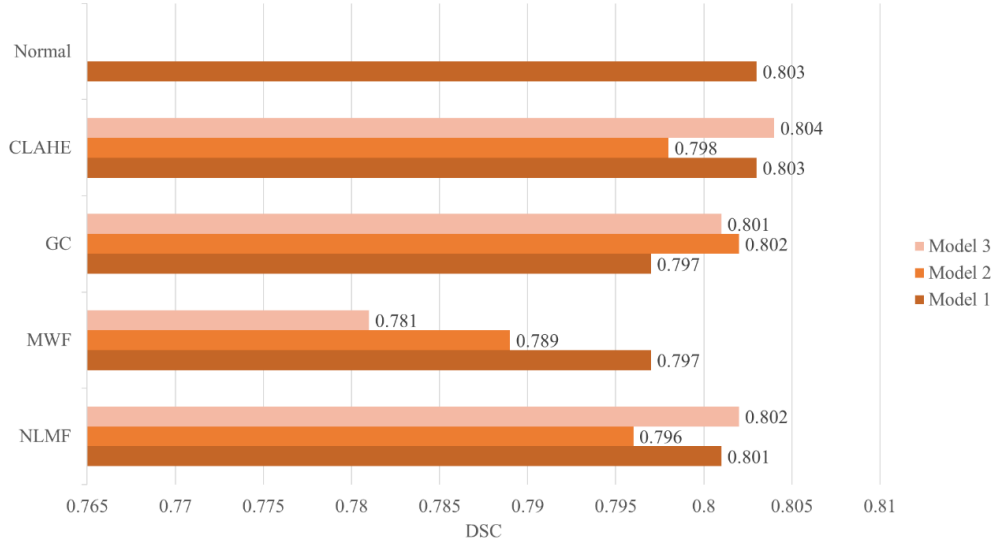


ภาพ 53 ภาพ Mask ที่โมเดลทำนายของโมเดลที่ฝึกฝนด้วยชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวน

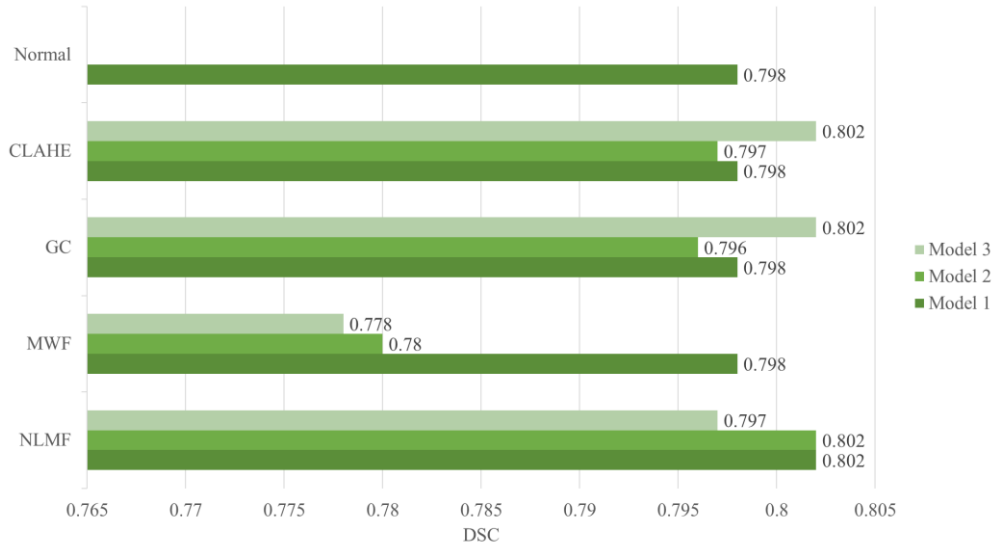
ผลลัพธ์ของโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง ที่ถูกฝึกฝนด้วยชุดข้อมูลที่ไเพิ่มสัญญาณรบกวนระดับที่ 1 พบว่า โมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net มีค่า DSC เท่ากับ 0.800 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE model3, NLMF model2 และ NLMF model3 มีค่า DSC ที่มากกว่า ขณะที่เทคนิค CLAHE model1 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 54 โมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ มีค่า DSC เท่ากับ 0.803 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE model3 มีค่า DSC ที่มากกว่า ขณะที่เทคนิค CLAHE model1 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 55 โมเดลที่ฝึกฝนด้วยสถาปัตยกรรมที่ผู้วิจัยออกแบบ ในชุดข้อมูลที่ไม่ผ่านการปรับปรุงคุณภาพของภาพมีค่า DSC เท่ากับ 0.798 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE model3, GC model3, NLMF model1, และ NLMF model2 มีค่า DSC ที่มากกว่า ขณะที่เทคนิค CLAHE model1, GC model1, และ MWF model1 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 56 โดยภาพผลการทำนาย Mask ของโมเดลแสดงดังภาพ 57



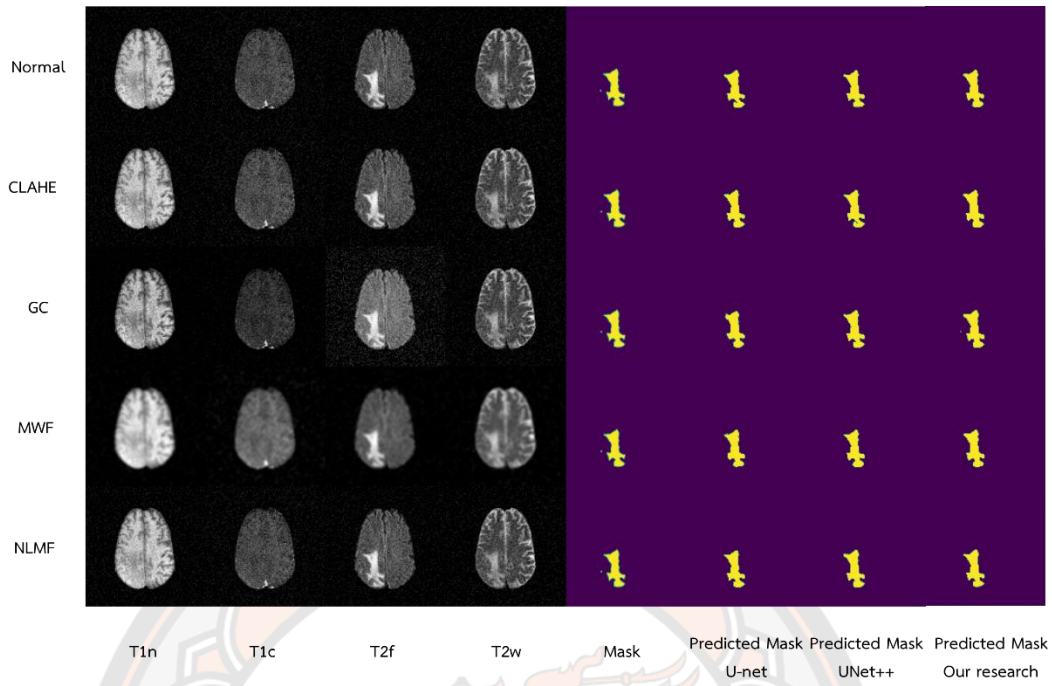
ภาพ 54 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net และชุดข้อมูลที่มีเพิ่มสัญญาณรบกวนระดับที่ 1



ภาพ 55 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ และชุดข้อมูลที่มีเพิ่มสัญญาณรบกวนระดับที่ 1

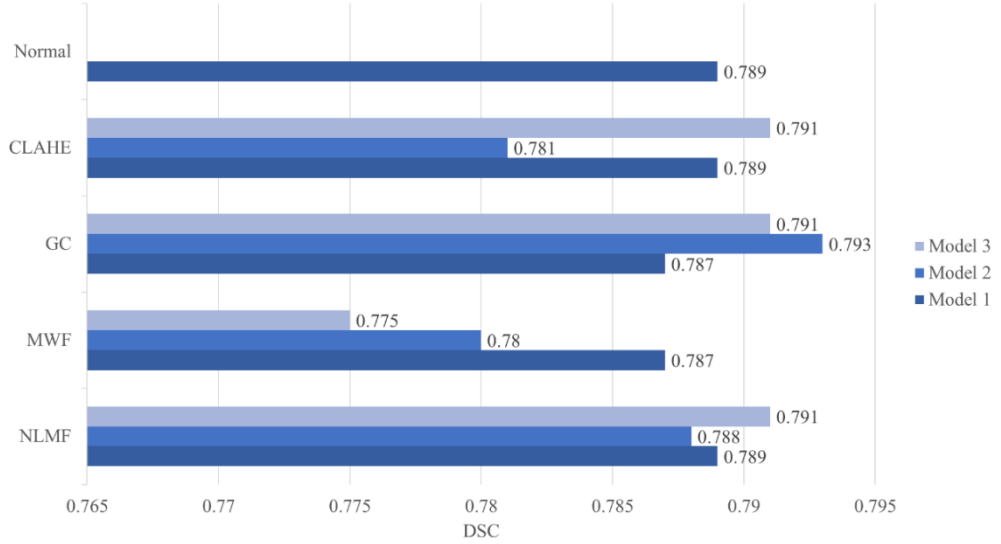


ภาพ 56 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรมที่ผู้วิจัยออกแบบและชุดข้อมูลที่มีเพิ่มสัญญาณรบกวนระดับที่ 1

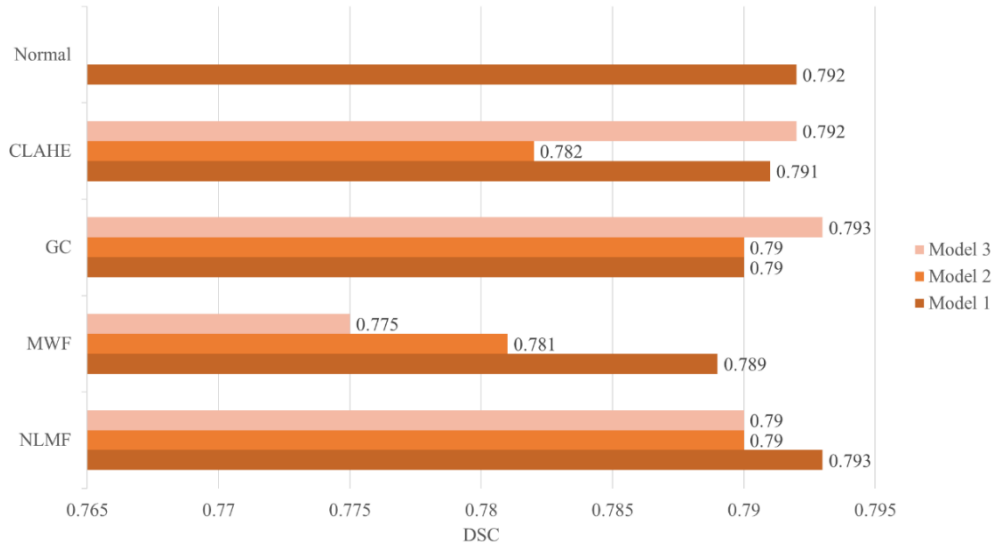


ภาพ 57 ภาพ Mask ที่โมเดลทำนายของโมเดลที่ฝึกฝนด้วยชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

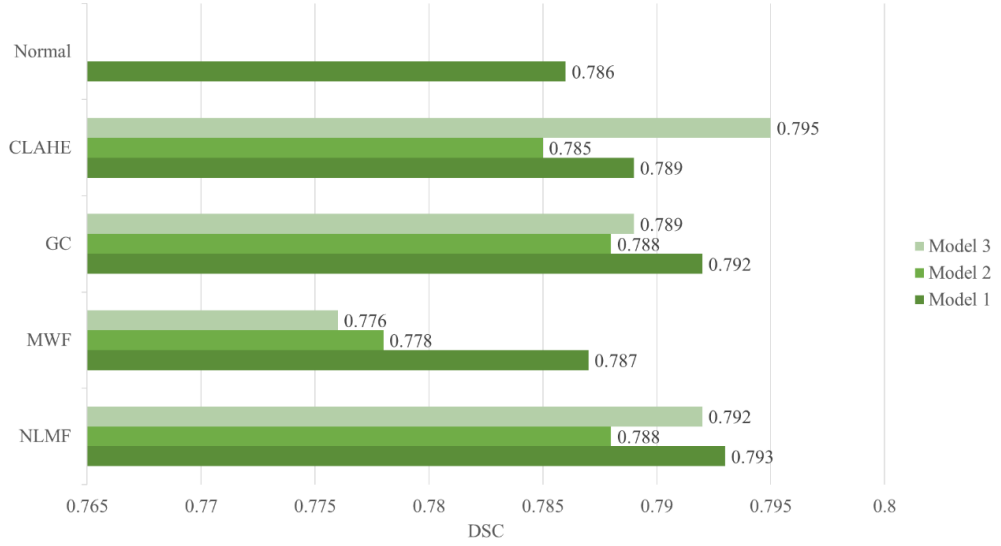
ผลลัพธ์ของโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง ที่ถูกฝึกฝนด้วยชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2 พบว่า โมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net มีค่า DSC เท่ากับ 0.789 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE model3, GC model2, GC model3, และ NLMF model3 มีค่า DSC ที่มากกว่า ขณะที่เทคนิค CLAHE model1 และ NLMF model1 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 58 โมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ มีค่า DSC เท่ากับ 0.792 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค GC model3 และ NLMF model1 มีค่า DSC ที่มากกว่าขณะที่เทคนิค CLAHE model3 ให้ค่า DSC ที่เท่ากัน สังเกตได้จากภาพ 59 โมเดลที่ฝึกฝนด้วยสถาปัตยกรรมที่ผู้วิจัยออกแบบ ในชุดข้อมูลที่ไม่ผ่านการปรับปรุงคุณภาพของภาพมีค่า DSC เท่ากับ 0.786 ขณะที่ชุดข้อมูลที่ผ่านการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE model1, CLAHE model3, GC ทุกโมเดล, MWF model1, และ NLMF ทุกโมเดล มีค่า DSC ที่มากกว่า สังเกตได้จากภาพ 60 โดยภาพผลการทำนาย Mask ของโมเดลแสดงดังภาพ 61



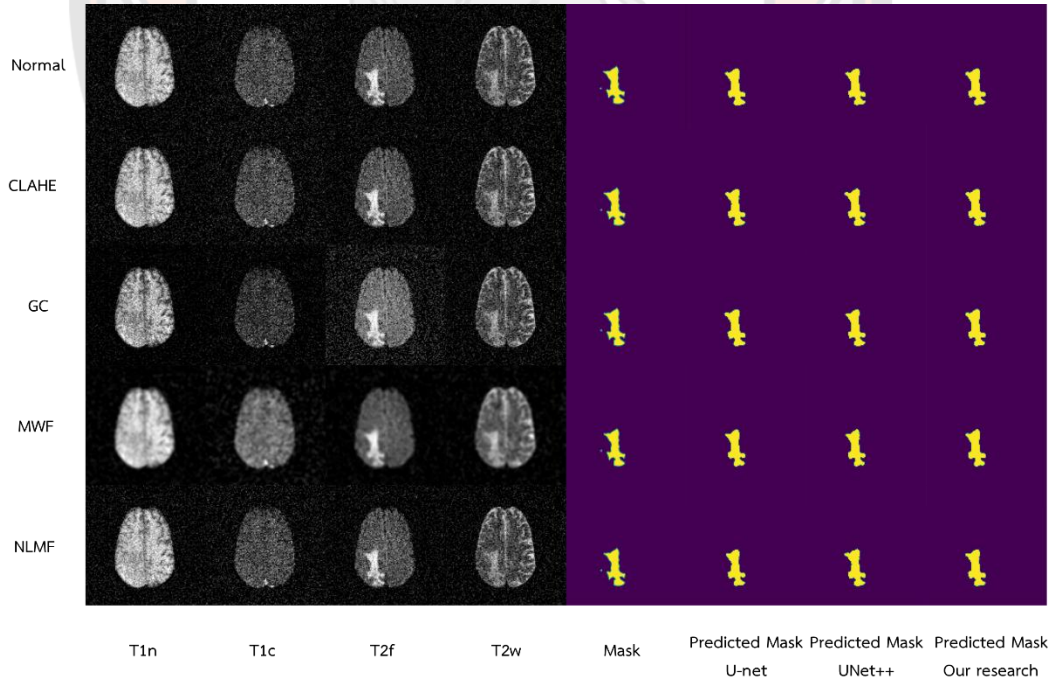
ภาพ 58 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net และชุดข้อมูลที่มีเพิ่มสัญญาณรบกวนระดับที่ 2



ภาพ 59 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ และชุดข้อมูลที่มีเพิ่มสัญญาณรบกวนระดับที่ 2



ภาพ 60 ค่า DSC ของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรมที่ผู้วิจัยออกแบบและชุดข้อมูลที่มีเพิ่มสัญญาณรบกวนระดับที่ 2



ภาพ 61 ภาพ Mask ที่โมเดลทำนายของโมเดลที่ฝึกฝนด้วยชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

อภิปรายผลการเปรียบเทียบประสิทธิภาพของโมเดลกับงานวิจัยก่อนหน้า

ผลลัพธ์การเปรียบเทียบโมเดลสำหรับการแบ่งข้อมูลภาพที่ฝึกฝนด้วยชุดข้อมูลที่ผ่านเทคนิคการปรับปรุงคุณภาพกับงานวิจัยก่อนหน้าที่ศึกษาการแบ่งส่วนข้อมูลของมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง สังเกตได้จากตาราง 16 จะเห็นว่าค่า DSC ของงานวิจัยนี้ในชุดข้อมูลที่ไม่ผ่านการเพิ่มสัญญาณรบกวนให้ผลลัพธ์ DSC สูงสุดเท่ากับ 0.811 (ThirdConvNet) และต่ำสุดเท่ากับ 0.803 (ThirdConvNet + CLAHE, ThirdConvNet + GC, และ ThirdConvNet + NLMF) ซึ่งให้ผลลัพธ์น้อยกว่างานวิจัยของ Ghosh. S (52), Lin M. (5), Al Nasim MA (53), Manasa K. (54), และ Yan C. (55) ซึ่งแสดงให้เห็นว่าโมเดลที่เพิ่มเทคนิคการปรับคุณภาพของภาพเพียงอย่างเดียวไม่เพียงที่จะเพิ่มประสิทธิภาพการทำงานของโมเดล รวมถึงสถาปัตยกรรมที่ผู้วิจัยออกแบบเพิ่มประสิทธิภาพการทำงานให้กับโมเดลได้น้อยกว่างานวิจัยที่กล่าวมาให้ข้างต้น อย่างไรตามมีงานวิจัยก่อนหน้าที่ให้ผลลัพธ์ค่า DSC น้อยกว่างานวิจัยนี้ ซึ่งได้แก่งานวิจัยของ Thong Vo. (8), Fathia A. (48), และ Rahul J. ที่มีค่า DSC เท่ากับ 0.760 และ 0.558 ตามลำดับ เมื่อเปรียบเทียบค่า accuracy ให้ผลลัพธ์ที่ไม่แตกต่างกัน ซึ่งมีค่าร้อยละ 99 รวมถึงค่า recall ที่มีค่ามากกว่าร้อยละ 90 เมื่อเทียบกับงานวิจัยอื่น ๆ ที่แสดงซึ่งมีค่าใกล้เคียงกัน

ตาราง 16 การเปรียบเทียบประสิทธิภาพของโมเดลกับงานวิจัยก่อนหน้า

Autor	Image	Network	Recall	Precision	Accuracy	DSC	IoU
Ghosh S. (52) 2021	TCGA-LGG	U-Net + ResNeXt50	-	-	0.996	0.932	0.890
		U-Net + FPN	-	-	0.993	0.899	0.860
Lin M. (5) 2021	BraTS 2020	U-Net+content block	-	-	-	0.887	-
Al Nasim MA (53) 2022	BraTS 2019	U-Net + empirical analysis	0.997	-	0.998	0.920	0.913
Thong Vo. (8) 2022	Mixed from 5 institut	UVR-Net	-	-	0.990	0.760	0.890

Autor	Image	Network	Recall	Precision	Accuracy	DSC	IoU
	ions						
Fathia A. (48) 2022	ISLES 2015	U-Net + Fine tuning	-	0.997	0.999	0.558	-
Manasa K. (54) 2022	BraTS 2018	U-Net + Zernile Moments	0.877	0.810	-	0.852	-
Yan C. (55) 2022	BraTS 2018	SEResU-Net	0.923	-	-	0.911	-
Rahul J. (9) 2023	BraTS 2020	SEMC-Net	-	-	-	0.731	0.644
Our proposed	BraTS 2023	U-Net	0.945	0.857	0.995	0.807	0.821
		U-Net + CLAHE	0.954	0.857	0.995	0.807	0.897
		U-Net + GC	0.956	0.858	0.995	0.809	0.898
		U-Net + NLMF	0.941	0.847	0.994	0.798	0.884
		U-Net + MWF	0.940	0.843	0.994	0.799	0.882
		UNet++	0.958	0.869	0.995	0.810	0.837
		UNet++ + CLAHE	0.958	0.863	0.995	0.810	0.903
		UNet++ + GC	0.947	0.856	0.994	0.803	0.892
		UNet++ + NLMF	0.945	0.853	0.994	0.802	0.890
		UNet++ + MWF	0.949	0.860	0.995	0.806	0.897
		ThirdConv Net	0.961	0.864	0.995	0.811	0.904

Autor	Image	Network	Recall	Precision	Accuracy	DSC	IoU
		ThirdConv Net + CLAHE	0.947	0.850	0.994	0.803	0.889
		ThirdConv Net + GC	0.945	0.852	0.994	0.803	0.889
		ThirdConv Net + NLMF	0.949	0.850	0.994	0.803	0.890
		ThirdConv Net + MWF	0.951	0.858	0.995	0.807	0.896

ข้อจำกัดของงานวิจัย

งานวิจัยนี้ศึกษาการเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก มีข้อจำกัดดังนี้ ข้อที่หนึ่งการใช้ชุดข้อมูลภาพในสองมิติอาจลดคุณภาพของภาพในการฝึกฝนโมเดล เนื่องจากชุดข้อมูลภาพต้นฉบับออกแบบในลักษณะสามมิติ ข้อที่สองคอมพิวเตอร์ที่ใช้ในการศึกษานี้มีประสิทธิภาพไม่เพียงพอสำหรับการฝึกฝนโมเดลด้วยชุดข้อมูลสามมิติ ข้อสุดท้ายควรแยกประเภทของภาพเอ็มอาร์ไอเนื่องจากลักษณะเฉพาะของภาพเอ็มอาร์ไอแต่ละประเภทแตกต่างกัน อย่างไรก็ตามในงานวิจัยนี้มีการใช้พารามิเตอร์เดียวสำหรับการปรับปรุงคุณภาพของภาพในภาพเอ็มอาร์ไอทุกประเภท เนื่องจาก Mask ที่ใช้ในการฝึกฝนโมเดลถูกออกแบบมาสำหรับการใช้ในภาพเอ็มอาร์ไอทั้งสี่ประเภทพร้อมกัน

สรุปผลการวิจัย

งานวิจัยนี้ศึกษาการเปรียบเทียบกระบวนการก่อนประมวลผลภาพสำหรับเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมองโดยใช้การเรียนรู้เชิงลึก เพื่อพัฒนาโมเดลการเรียนรู้เชิงลึกสำหรับการแบ่งส่วนภาพมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยใช้สถาปัตยกรรม U-net, UNet++, และสถาปัตยกรรมที่ผู้วิจัยออกแบบ รวมถึงศึกษาผลของกระบวนการก่อนการประมวลผลภาพเพื่อช่วยเพิ่มประสิทธิภาพของโมเดลการเรียนรู้เชิงลึกสำหรับ

การแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง โดยใช้เทคนิคการปรับปรุงคุณภาพของภาพ 4 เทคนิค ได้แก่ Contrast-limited adaptive histogram equalization, Gamma correction, Median and Wiener filter, และ Non-local mean filter พบว่า ผลของกระบวนการก่อนการประมวลผลภาพช่วยเพิ่มคุณภาพของภาพได้ดี โดยเฉพาะเทคนิค Non-local mean filter สามารถเพิ่มคุณภาพของภาพทั้งในชุดข้อมูลที่ไม่มีการเพิ่มสัญญาณรบกวนและมีการเพิ่มสัญญาณรบกวน เทคนิค Gamma correction และ เทคนิค Median and Wiener filter สามารถเพิ่มคุณภาพของภาพได้ดีในชุดข้อมูลปกติ อย่างไรก็ตามเมื่อใช้เทคนิคทั้งสองในชุดข้อมูลที่มีการเพิ่มสัญญาณรบกวนกลับทำให้คุณภาพของภาพลดลงกว่าการใช้เทคนิค Contrast-limited adaptive histogram equalization แสดงให้เห็นว่าเทคนิค Contrast-limited adaptive histogram equalization สามารถกำจัดสัญญาณรบกวนในชุดข้อมูลภาพได้ดีกว่า สำหรับผลของกระบวนการก่อนการประมวลผลภาพช่วยเพิ่มประสิทธิภาพของการแบ่งข้อมูลภาพของก้อนมะเร็งไกลโอมาบนภาพเอ็มอาร์ไอสมอง เทคนิค Non-local mean filter, Contrast-limited adaptive histogram equalization, และ Gamma correction สามารถเพิ่มประสิทธิภาพให้กับโมเดลได้ ขณะที่เทคนิค Median and Wiener filter กลับทำให้ประสิทธิภาพของโมเดลลดลงอย่างเห็นได้ชัด จากที่กล่าวมาจะเห็นว่าการใช้ชุดข้อมูลที่ได้รับการเพิ่มคุณภาพด้วยเทคนิคต่าง ๆ สามารถเพิ่มหรือลดประสิทธิภาพให้กับโมเดลได้ ทั้งนี้ขึ้นอยู่กับการกำหนดพารามิเตอร์ในแต่ละเทคนิครวมถึงสถาปัตยกรรมที่ใช้

ภาคผนวก

ภาคผนวก ก ผลลัพธ์การประเมินคุณภาพของภาพ

1. เทคนิค Contrast-Limited Adaptive Histogram Equalization (CLAHE)

ตาราง 17 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค CLAHE ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Cliplimit	Tilesize	SSIM	MSE	PSNR
0.1	2	0.916	3.262	43.883
0.1	4	0.911	27.442	34.761
0.1	6	0.740	115.234	28.313
0.1	8	0.711	378.208	22.605
0.2	2	0.915	8.322	39.904
0.2	4	0.911	27.442	34.761
0.2	6	0.740	115.234	28.313
0.2	8	0.711	378.208	22.605
0.3	2	0.914	12.386	38.169
0.3	4	0.911	27.442	34.761
0.3	6	0.740	115.234	28.313
0.3	8	0.711	378.208	22.605

ตาราง 18 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค CLAHE ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Cliplimit	Tilesize	SSIM	MSE	PSNR
0.1	2	0.989	3.222	43.962
0.1	4	0.948	39.859	32.945
0.1	6	0.843	147.261	27.087
0.1	8	0.693	671.432	20.064
0.2	2	0.980	15.031	37.132

Cliplimit	Tilesize	SSIM	MSE	PSNR
0.2	4	0.948	39.859	32.945
0.2	6	0.843	147.261	27.087
0.2	8	0.693	671.432	20.064
0.3	2	0.973	23.506	35.350
0.3	4	0.948	39.859	32.945
0.3	6	0.843	147.261	27.087
0.3	8	0.693	671.432	20.064

ตาราง 19 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค CLAHE ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Cliplimit	Tilesize	SSIM	MSE	PSNR
0.1	2	0.997	2.428	44.944
0.1	4	0.962	42.778	32.133
0.1	6	0.873	181.343	25.763
0.1	8	0.724	903.239	18.911
0.2	2	0.990	13.721	37.894
0.2	4	0.962	42.778	32.133
0.2	6	0.873	181.343	25.763
0.2	8	0.724	903.239	18.911
0.3	2	0.985	23.793	35.261
0.3	4	0.962	42.778	32.133
0.3	6	0.873	181.343	25.763
0.3	8	0.724	903.239	18.911

2. เทคนิค Gamma Correction (GC)

ตาราง 20 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค GC ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Gamma	SSIM	MSE	PSNR
0.3	0.848	2444.457	14.560
0.5	0.917	916.280	18.740
0.7	0.968	248.587	24.345
0.9	0.997	19.392	35.379
1.3	0.970	134.150	26.932
1.5	0.929	300.219	23.433
1.7	0.886	486.790	21.344
1.9	0.845	675.590	19.938

ตาราง 21 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค GC ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Gamma	SSIM	MSE	PSNR
0.3	0.276	4785.396	11.385
0.5	0.448	1447.554	16.588
0.7	0.715	340.583	22.873
0.9	0.967	23.989	34.390
1.3	0.739	154.211	26.298
1.5	0.541	334.313	22.943
1.7	0.409	530.594	20.948
1.9	0.322	725.475	19.606

ตาราง 22 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค GC ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Gamma	SSIM	MSE	PSNR
0.3	0.341	5256.337	10.954
0.5	0.553	1673.673	15.924
0.7	0.798	402.905	22.108
0.9	0.978	28.728	33.574
1.3	0.806	181.466	25.567
1.5	0.610	391.052	22.236
1.7	0.452	616.293	20.267
1.9	0.340	836.263	18.951

3. เทคนิค Median and Wiener filters (MWF)

ตาราง 23 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค MWF ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Kernel	SSIM	MSE	PSNR
3 × 3	0.936	50.027	31.645
5 × 5	0.864	114.667	28.183
7 × 7	0.811	170.944	26.521

ตาราง 24 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค MWF ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Kernel	SSIM	MSE	PSNR
3 × 3	0.474	160.217	26.138
5 × 5	0.335	236.061	24.538
7 × 7	0.272	293.314	23.665

ตาราง 25 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค MWF ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Kernel	SSIM	MSE	PSNR
3 × 3	0.310	479.833	21.329
5 × 5	0.188	598.196	20.387
7 × 7	0.143	667.119	19.930

4. เทคนิค Non-local mean filer (NLMF)

ตาราง 26 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค NLMF ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Smooth	Patch	Window	SSIM	MSE	PSNR
1	2	10	1.000	0.007	-
1	2	20	1.000	0.009	-
1	2	30	1.000	0.011	-
1	4	10	1.000	0.001	-
1	4	20	1.000	0.002	-
1	4	30	1.000	0.002	-
1	6	10	1.000	0.001	-
1	6	20	1.000	0.001	-
1	6	30	1.000	0.001	-
1	8	10	1.000	0.000	-
1	8	20	1.000	0.000	-
1	8	30	1.000	0.000	-
1	10	10	1.000	0.000	-
1	10	20	1.000	0.000	-

Smooth	Patch	Window	SSIM	MSE	PSNR
1	10	30	1.000	0.000	-
2	2	10	0.999	0.082	-
2	2	20	0.999	0.111	-
2	2	30	0.999	0.132	-
2	4	10	1.000	0.040	-
2	4	20	0.999	0.051	-
2	4	30	0.999	0.059	-
2	6	10	1.000	0.022	-
2	6	20	1.000	0.027	-
2	6	30	1.000	0.030	-
2	8	10	1.000	0.013	-
2	8	20	1.000	0.016	-
2	8	30	1.000	0.017	-
2	10	10	1.000	0.007	-
2	10	20	1.000	0.008	-
2	10	30	1.000	0.009	-

ตาราง 27 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค NLMF ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Smooth	Patch	Window	SSIM	MSE	PSNR
1	2	10	1.000	0.008	-
1	2	20	1.000	0.018	-
1	2	30	1.000	0.029	-
1	4	10	1.000	0.000	-
1	4	20	1.000	0.000	-
1	4	30	1.000	0.000	-
1	6	10	1.000	0.000	-

Smooth	Patch	Window	SSIM	MSE	PSNR
1	6	20	1.000	0.000	-
1	6	30	1.000	0.000	-
1	8	10	1.000	0.000	-
1	8	20	1.000	0.000	-
1	8	30	1.000	0.000	-
1	10	10	1.000	0.000	-
1	10	20	1.000	0.000	-
1	10	30	1.000	0.000	-
2	2	10	1.000	0.093	-
2	2	20	0.999	0.205	-
2	2	30	0.998	0.305	-
2	4	10	1.000	0.001	-
2	4	20	1.000	0.001	-
2	4	30	1.000	0.002	-
2	6	10	1.000	0.000	-
2	6	20	1.000	0.000	-
2	6	30	1.000	0.000	-
2	8	10	1.000	0.000	-
2	8	20	1.000	0.000	-
2	8	30	1.000	0.000	-
2	10	10	1.000	0.000	-
2	10	20	1.000	0.000	-
2	10	30	1.000	0.000	-

ตาราง 28 แสดงผลลัพธ์การประเมินคุณภาพของภาพโดยใช้เทคนิค NLMF ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Smooth	Patch	Window	SSIM	MSE	PSNR
1	2	10	1.000	0.001	-
1	2	20	1.000	0.003	-
1	2	30	1.000	0.006	-
1	4	10	1.000	0.000	-
1	4	20	1.000	0.000	-
1	4	30	1.000	0.000	-
1	6	10	1.000	0.000	-
1	6	20	1.000	0.000	-
1	6	30	1.000	0.000	-
1	8	10	1.000	0.000	-
1	8	20	1.000	0.000	-
1	8	30	1.000	0.000	-
1	10	10	1.000	0.000	-
1	10	20	1.000	0.000	-
1	10	30	1.000	0.000	-
2	2	10	1.000	0.011	-
2	2	20	1.000	0.027	-
2	2	30	1.000	0.045	-
2	4	10	1.000	0.000	-
2	4	20	1.000	0.000	-
2	4	30	1.000	0.000	-
2	6	10	1.000	0.000	-
2	6	20	1.000	0.000	-
2	6	30	1.000	0.000	-
2	8	10	1.000	0.000	-

Smooth	Patch	Window	SSIM	MSE	PSNR
2	8	20	1.000	0.000	-
2	8	30	1.000	0.000	-
2	10	10	1.000	0.000	-
2	10	20	1.000	0.000	-
2	10	30	1.000	0.000	-

ภาคผนวก ข ผลลัพธ์การประเมินประสิทธิของโมเดล

1. สถาปัตยกรรม U-net

1.1 ชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

ตาราง 29 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.807	0.896	0.956	0.855	0.994
CLAHE	cl0.1_ts2	0.807	0.897	0.954	0.857	0.995
CLAHE	cl0.1_ts4	0.802	0.888	0.945	0.850	0.994
CLAHE	cl0.1_ts6	0.802	0.887	0.944	0.849	0.994
CLAHE	cl0.1_ts8	0.807	0.896	0.952	0.858	0.994
CLAHE	cl0.2_ts2	0.807	0.897	0.955	0.857	0.995
CLAHE	cl0.2_ts4	0.807	0.896	0.954	0.856	0.994
CLAHE	cl0.2_ts6	0.804	0.894	0.949	0.857	0.994
CLAHE	cl0.2_ts8	0.805	0.893	0.948	0.857	0.994
CLAHE	cl0.3_ts2	0.805	0.894	0.950	0.856	0.994
CLAHE	cl0.3_ts4	0.806	0.896	0.951	0.859	0.994
CLAHE	cl0.3_ts6	0.802	0.887	0.945	0.849	0.994
CLAHE	cl0.3_ts8	0.805	0.894	0.948	0.858	0.994
GC	g0.3	0.800	0.884	0.941	0.847	0.994
GC	g0.5	0.799	0.883	0.939	0.848	0.994
GC	g0.7	0.809	0.898	0.956	0.858	0.995
GC	g0.9	0.807	0.896	0.955	0.856	0.994
GC	g1.3	0.804	0.890	0.947	0.852	0.994
GC	g1.5	0.806	0.896	0.949	0.860	0.995
GC	g1.7	0.808	0.895	0.953	0.856	0.994

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
GC	g1.9	0.806	0.894	0.950	0.856	0.994
MWF	k3	0.799	0.882	0.940	0.843	0.994
MWF	k5	0.793	0.872	0.930	0.832	0.993
MWF	k7	0.772	0.839	0.896	0.802	0.991
NLMF	s1_p2_w10	0.802	0.888	0.945	0.850	0.994
NLMF	s1_p2_w20	0.807	0.897	0.952	0.860	0.995
NLMF	s1_p2_w30	0.807	0.898	0.954	0.858	0.995
NLMF	s1_p4_w10	0.807	0.895	0.951	0.857	0.994
NLMF	s1_p4_w20	0.803	0.890	0.946	0.854	0.994
NLMF	s1_p4_w30	0.806	0.897	0.953	0.859	0.995
NLMF	s1_p6_w10	0.808	0.896	0.952	0.858	0.994
NLMF	s1_p6_w20	0.803	0.890	0.945	0.853	0.994
NLMF	s1_p6_w30	0.806	0.894	0.953	0.854	0.994
NLMF	s1_p8_w10	0.798	0.884	0.941	0.847	0.994
NLMF	s1_p8_w20	0.807	0.895	0.951	0.857	0.994
NLMF	s1_p8_w30	0.804	0.896	0.946	0.863	0.995
NLMF	s2_p2_w10	0.808	0.900	0.954	0.862	0.995
NLMF	s2_p2_w20	0.806	0.894	0.951	0.856	0.994
NLMF	s2_p2_w30	0.800	0.886	0.943	0.848	0.994
NLMF	s2_p4_w10	0.803	0.888	0.948	0.849	0.994
NLMF	s2_p4_w20	0.800	0.889	0.944	0.852	0.994
NLMF	s2_p4_w30	0.798	0.882	0.940	0.844	0.994
NLMF	s2_p6_w10	0.807	0.897	0.953	0.859	0.995
NLMF	s2_p6_w20	0.803	0.891	0.945	0.855	0.994
NLMF	s2_p6_w30	0.808	0.901	0.956	0.862	0.995
NLMF	s2_p8_w10	0.804	0.892	0.949	0.854	0.994
NLMF	s2_p8_w20	0.801	0.888	0.943	0.852	0.994
NLMF	s2_p8_w30	0.807	0.897	0.952	0.860	0.995

1.2 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

ตาราง 30 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.800	0.885	0.804	0.941	0.846
CLAHE	cl0.1_ts2	0.800	0.884	0.940	0.846	0.994
CLAHE	cl0.1_ts4	0.801	0.885	0.940	0.848	0.994
CLAHE	cl0.1_ts6	0.796	0.878	0.931	0.842	0.994
CLAHE	cl0.1_ts8	0.796	0.879	0.932	0.845	0.994
CLAHE	cl0.2_ts2	0.801	0.886	0.941	0.850	0.994
CLAHE	cl0.2_ts4	0.798	0.881	0.936	0.846	0.994
CLAHE	cl0.2_ts6	0.798	0.883	0.933	0.851	0.994
CLAHE	cl0.2_ts8	0.799	0.881	0.934	0.847	0.994
CLAHE	cl0.3_ts2	0.800	0.886	0.939	0.850	0.994
CLAHE	cl0.3_ts4	0.797	0.882	0.935	0.846	0.994
CLAHE	cl0.3_ts6	0.801	0.884	0.940	0.847	0.994
CLAHE	cl0.3_ts8	0.796	0.877	0.931	0.842	0.994
GC	g0.3	0.797	0.879	0.933	0.843	0.994
GC	g0.5	0.800	0.885	0.939	0.847	0.994
GC	g0.7	0.799	0.883	0.937	0.847	0.994
GC	g0.9	0.800	0.885	0.940	0.848	0.994
GC	g1.3	0.801	0.885	0.939	0.848	0.994
GC	g1.5	0.798	0.883	0.937	0.846	0.994
GC	g1.7	0.801	0.885	0.939	0.848	0.994
GC	g1.9	0.800	0.886	0.940	0.850	0.994
MWF	k3	0.791	0.871	0.783	0.922	0.838

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
MWF	k5	0.787	0.861	0.768	0.917	0.825
MWF	k7	0.780	0.851	0.751	0.906	0.814
NLMF	s1_p2_w10	0.794	0.875	0.790	0.928	0.841
NLMF	s1_p2_w20	0.801	0.886	0.805	0.941	0.849
NLMF	s1_p2_w30	0.799	0.884	0.803	0.936	0.850
NLMF	s1_p4_w10	0.798	0.883	0.802	0.937	0.847
NLMF	s1_p4_w20	0.797	0.881	0.798	0.936	0.844
NLMF	s1_p4_w30	0.802	0.886	0.806	0.941	0.848
NLMF	s1_p6_w10	0.802	0.888	0.809	0.943	0.850
NLMF	s1_p6_w20	0.800	0.885	0.804	0.939	0.848
NLMF	s1_p6_w30	0.796	0.876	0.792	0.933	0.839
NLMF	s1_p8_w10	0.798	0.882	0.800	0.937	0.846
NLMF	s1_p8_w20	0.804	0.892	0.815	0.947	0.854
NLMF	s1_p8_w30	0.800	0.886	0.805	0.941	0.849
NLMF	s2_p2_w10	0.796	0.878	0.794	0.934	0.840
NLMF	s2_p2_w20	0.801	0.886	0.806	0.942	0.848
NLMF	s2_p2_w30	0.801	0.886	0.805	0.938	0.851
NLMF	s2_p4_w10	0.801	0.887	0.807	0.939	0.851
NLMF	s2_p4_w20	0.800	0.884	0.802	0.941	0.845
NLMF	s2_p4_w30	0.800	0.883	0.802	0.938	0.846
NLMF	s2_p6_w10	0.801	0.887	0.806	0.945	0.846
NLMF	s2_p6_w20	0.793	0.873	0.788	0.929	0.837
NLMF	s2_p6_w30	0.801	0.885	0.805	0.940	0.848
NLMF	s2_p8_w10	0.801	0.885	0.805	0.940	0.848
NLMF	s2_p8_w20	0.802	0.887	0.807	0.941	0.850
NLMF	s2_p8_w30	0.799	0.885	0.804	0.939	0.847

1.3 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

ตาราง 31 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม U-net ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.789	0.867	0.919	0.834	0.993
CLAHE	cl0.1_ts2	0.789	0.865	0.919	0.830	0.993
CLAHE	cl0.1_ts4	0.790	0.866	0.922	0.829	0.993
CLAHE	cl0.1_ts6	0.787	0.863	0.915	0.830	0.993
CLAHE	cl0.1_ts8	0.784	0.859	0.906	0.830	0.993
CLAHE	cl0.2_ts2	0.787	0.864	0.917	0.829	0.993
CLAHE	cl0.2_ts4	0.789	0.866	0.920	0.830	0.993
CLAHE	cl0.2_ts6	0.788	0.865	0.918	0.830	0.993
CLAHE	cl0.2_ts8	0.781	0.854	0.908	0.821	0.992
CLAHE	cl0.3_ts2	0.788	0.864	0.915	0.832	0.993
CLAHE	cl0.3_ts4	0.791	0.870	0.923	0.834	0.993
CLAHE	cl0.3_ts6	0.791	0.869	0.921	0.835	0.993
CLAHE	cl0.3_ts8	0.788	0.865	0.915	0.834	0.993
GC	g0.3	0.787	0.863	0.914	0.830	0.993
GC	g0.5	0.787	0.862	0.913	0.829	0.993
GC	g0.7	0.793	0.870	0.923	0.835	0.993
GC	g0.9	0.791	0.868	0.921	0.833	0.993
GC	g1.3	0.790	0.867	0.922	0.832	0.993
GC	g1.5	0.791	0.868	0.921	0.833	0.993
GC	g1.7	0.788	0.862	0.917	0.827	0.993
GC	g1.9	0.788	0.867	0.918	0.833	0.993

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
MWF	k3	0.787	0.865	0.915	0.833	0.993
MWF	k5	0.780	0.854	0.902	0.825	0.992
MWF	k7	0.775	0.842	0.897	0.805	0.992
NLMF	s1_p2_w10	0.787	0.863	0.917	0.828	0.993
NLMF	s1_p2_w20	0.792	0.872	0.921	0.839	0.993
NLMF	s1_p2_w30	0.789	0.865	0.918	0.830	0.993
NLMF	s1_p4_w10	0.790	0.869	0.921	0.835	0.993
NLMF	s1_p4_w20	0.791	0.870	0.923	0.835	0.993
NLMF	s1_p4_w30	0.790	0.868	0.920	0.832	0.993
NLMF	s1_p6_w10	0.789	0.866	0.917	0.833	0.993
NLMF	s1_p6_w20	0.790	0.865	0.922	0.828	0.993
NLMF	s1_p6_w30	0.790	0.868	0.919	0.834	0.993
NLMF	s1_p8_w10	0.789	0.866	0.920	0.829	0.993
NLMF	s1_p8_w20	0.791	0.868	0.921	0.834	0.993
NLMF	s1_p8_w30	0.787	0.862	0.919	0.824	0.993
NLMF	s2_p2_w10	0.791	0.869	0.922	0.834	0.993
NLMF	s2_p2_w20	0.788	0.865	0.916	0.834	0.993
NLMF	s2_p2_w30	0.788	0.866	0.919	0.832	0.993
NLMF	s2_p4_w10	0.790	0.867	0.920	0.833	0.993
NLMF	s2_p4_w20	0.789	0.869	0.921	0.833	0.993
NLMF	s2_p4_w30	0.790	0.866	0.922	0.830	0.993
NLMF	s2_p6_w10	0.789	0.865	0.919	0.830	0.993
NLMF	s2_p6_w20	0.785	0.860	0.914	0.826	0.993
NLMF	s2_p6_w30	0.791	0.869	0.921	0.835	0.993
NLMF	s2_p8_w10	0.790	0.867	0.919	0.832	0.993
NLMF	s2_p8_w20	0.790	0.867	0.920	0.833	0.993
NLMF	s2_p8_w30	0.791	0.869	0.920	0.836	0.993

2. สถาปัตยกรรม UNet++

2.1 ชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

ตาราง 32 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.810	0.906	0.958	0.869	0.995
CLAHE	cl0.1_ts2	0.810	0.903	0.958	0.863	0.995
CLAHE	cl0.1_ts4	0.801	0.888	0.943	0.850	0.994
CLAHE	cl0.1_ts6	0.808	0.906	0.950	0.875	0.995
CLAHE	cl0.1_ts8	0.807	0.898	0.953	0.861	0.995
CLAHE	cl0.2_ts2	0.809	0.908	0.955	0.875	0.995
CLAHE	cl0.2_ts4	0.806	0.895	0.951	0.857	0.994
CLAHE	cl0.2_ts6	0.807	0.899	0.951	0.865	0.995
CLAHE	cl0.2_ts8	0.808	0.904	0.951	0.870	0.995
CLAHE	cl0.3_ts2	0.807	0.898	0.954	0.860	0.995
CLAHE	cl0.3_ts4	0.802	0.890	0.942	0.857	0.994
CLAHE	cl0.3_ts6	0.806	0.906	0.948	0.878	0.995
CLAHE	cl0.3_ts8	0.808	0.899	0.955	0.860	0.995
GC	g0.3	0.800	0.886	0.942	0.849	0.994
GC	g0.5	0.806	0.896	0.949	0.861	0.995
GC	g0.7	0.803	0.892	0.947	0.856	0.994
GC	g0.9	0.798	0.890	0.932	0.861	0.994
GC	g1.3	0.796	0.880	0.931	0.847	0.994
GC	g1.5	0.806	0.894	0.952	0.856	0.994
GC	g1.7	0.797	0.889	0.931	0.862	0.994
GC	g1.9	0.799	0.895	0.933	0.869	0.995
MWF	k3	0.806	0.897	0.949	0.860	0.995

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
MWF	k5	0.797	0.879	0.934	0.841	0.994
MWF	k7	0.784	0.860	0.915	0.822	0.993
NLMF	s1_p2_w10	0.806	0.899	0.952	0.863	0.995
NLMF	s1_p2_w20	0.804	0.893	0.949	0.856	0.994
NLMF	s1_p2_w30	0.808	0.902	0.953	0.867	0.995
NLMF	s1_p8_w20	0.802	0.890	0.945	0.853	0.994
NLMF	s2_p2_w20	0.810	0.902	0.956	0.865	0.995
NLMF	s2_p6_w30	0.808	0.903	0.953	0.868	0.995

2.2 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

ตาราง 33 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ ใน ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.803	0.888	0.810	0.943	0.852
CLAHE	cl0.1_ts2	0.803	0.891	0.812	0.941	0.855
CLAHE	cl0.1_ts4	0.804	0.891	0.814	0.944	0.856
CLAHE	cl0.1_ts6	0.797	0.879	0.796	0.933	0.843
CLAHE	cl0.1_ts8	0.798	0.879	0.796	0.933	0.844
CLAHE	cl0.2_ts2	0.797	0.881	0.798	0.934	0.845
CLAHE	cl0.2_ts4	0.802	0.888	0.809	0.944	0.850
CLAHE	cl0.2_ts6	0.801	0.886	0.806	0.940	0.850
CLAHE	cl0.2_ts8	0.798	0.883	0.802	0.937	0.848
CLAHE	cl0.3_ts2	0.802	0.890	0.811	0.941	0.855
CLAHE	cl0.3_ts4	0.798	0.881	0.799	0.934	0.846
CLAHE	cl0.3_ts6	0.804	0.894	0.818	0.943	0.860

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
CLAHE	cl0.3_ts8	0.794	0.875	0.789	0.927	0.841
GC	g0.3	0.781	0.854	0.761	0.907	0.823
GC	g0.5	0.797	0.877	0.793	0.931	0.841
GC	g0.7	0.802	0.888	0.808	0.941	0.851
GC	g0.9	0.794	0.875	0.789	0.924	0.843
GC	g1.3	0.782	0.861	0.768	0.907	0.832
GC	g1.5	0.801	0.884	0.803	0.940	0.847
GC	g1.7	0.778	0.850	0.754	0.903	0.818
GC	g1.9	0.779	0.856	0.762	0.902	0.829
MWF	k3	0.797	0.881	0.798	0.932	0.848
MWF	k5	0.789	0.865	0.774	0.915	0.835
MWF	k7	0.781	0.852	0.753	0.904	0.817
NLMF	s1_p2_w10	0.796	0.882	0.799	0.932	0.850
NLMF	s1_p2_w20	0.796	0.879	0.796	0.931	0.844
NLMF	s1_p2_w30	0.801	0.885	0.804	0.938	0.849
NLMF	s1_p8_w20	0.796	0.880	0.796	0.932	0.846
NLMF	s2_p2_w20	0.802	0.886	0.806	0.939	0.851
NLMF	s2_p6_w30	0.803	0.891	0.812	0.941	0.855

2.3 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

ตาราง 34 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม UNet++ ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.792	0.872	0.925	0.837	0.993
CLAHE	cl0.1_ts2	0.791	0.871	0.922	0.838	0.993
CLAHE	cl0.1_ts4	0.792	0.872	0.925	0.836	0.993

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
CLAHE	cl0.1_ts6	0.792	0.872	0.925	0.837	0.993
CLAHE	cl0.1_ts8	0.783	0.858	0.914	0.822	0.993
CLAHE	cl0.2_ts2	0.790	0.869	0.920	0.835	0.993
CLAHE	cl0.2_ts4	0.793	0.873	0.923	0.839	0.993
CLAHE	cl0.2_ts6	0.792	0.874	0.925	0.840	0.993
CLAHE	cl0.2_ts8	0.782	0.856	0.909	0.823	0.993
CLAHE	cl0.3_ts2	0.792	0.872	0.924	0.836	0.993
CLAHE	cl0.3_ts4	0.789	0.868	0.919	0.835	0.993
CLAHE	cl0.3_ts6	0.792	0.873	0.921	0.840	0.993
CLAHE	cl0.3_ts8	0.784	0.860	0.910	0.829	0.993
GC	g0.3	0.778	0.846	0.897	0.816	0.992
GC	g0.5	0.790	0.867	0.919	0.833	0.993
GC	g0.7	0.790	0.868	0.921	0.835	0.993
GC	g0.9	0.784	0.858	0.906	0.829	0.993
GC	g1.3	0.756	0.818	0.870	0.788	0.991
GC	g1.5	0.793	0.870	0.922	0.836	0.993
GC	g1.7	0.760	0.821	0.874	0.795	0.991
GC	g1.9	0.764	0.839	0.877	0.818	0.992
MWF	k3	0.789	0.867	0.918	0.834	0.993
MWF	k5	0.781	0.853	0.906	0.819	0.992
MWF	k7	0.775	0.844	0.897	0.810	0.992
NLMF	s1_p2_w10	0.790	0.869	0.922	0.834	0.993
NLMF	s1_p2_w20	0.789	0.868	0.917	0.837	0.993
NLMF	s1_p2_w30	0.793	0.874	0.923	0.841	0.993
NLMF	s1_p8_w20	0.790	0.870	0.921	0.836	0.993
NLMF	s2_p2_w20	0.790	0.868	0.919	0.836	0.993
NLMF	s2_p6_w30	0.791	0.871	0.922	0.838	0.993

3. สถาปัตยกรรม ThirdConv

3.1 ชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

ตาราง 35 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม ThirdConv ในชุดข้อมูลที่ไม่เพิ่มสัญญาณรบกวน

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.811	0.904	0.834	0.961	0.864
CLAHE	cl0.1_ts2	0.803	0.889	0.947	0.850	0.994
CLAHE	cl0.2_ts8	0.807	0.895	0.952	0.856	0.994
CLAHE	cl0.3_ts6	0.803	0.889	0.946	0.851	0.994
GC	g0.5	0.809	0.898	0.956	0.858	0.995
GC	g0.7	0.803	0.889	0.945	0.852	0.994
GC	g1.5	0.805	0.893	0.949	0.854	0.994
MWF	k3	0.807	0.896	0.951	0.858	0.995
MWF	k5	0.795	0.877	0.934	0.838	0.994
MWF	k7	0.782	0.856	0.911	0.818	0.992
NLMF	s1_p8_w20	0.808	0.900	0.955	0.862	0.995
NLMF	s2_p2_w20	0.804	0.893	0.947	0.856	0.994
NLMF	s2_p6_w30	0.805	0.892	0.950	0.852	0.994

3.2 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

ตาราง 36 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม ThirdConv ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 1

Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.798	0.880	0.934	0.845	0.994
CLAHE	cl0.1_ts2	0.798	0.880	0.935	0.842	0.994
CLAHE	cl0.2_ts8	0.797	0.876	0.931	0.841	0.993
CLAHE	cl0.3_ts6	0.802	0.886	0.943	0.847	0.994
GC	g0.5	0.798	0.879	0.935	0.841	0.994
GC	g0.7	0.796	0.877	0.932	0.840	0.993
GC	g1.5	0.802	0.887	0.941	0.850	0.994
MWF	k3	0.798	0.881	0.936	0.843	0.994
MWF	k5	0.780	0.856	0.909	0.821	0.993
MWF	k7	0.778	0.849	0.903	0.812	0.992
NLMF	s1_p8_w20	0.802	0.886	0.939	0.850	0.994
NLMF	s2_p2_w20	0.802	0.886	0.942	0.849	0.994
NLMF	s2_p6_w30	0.797	0.879	0.931	0.845	0.994

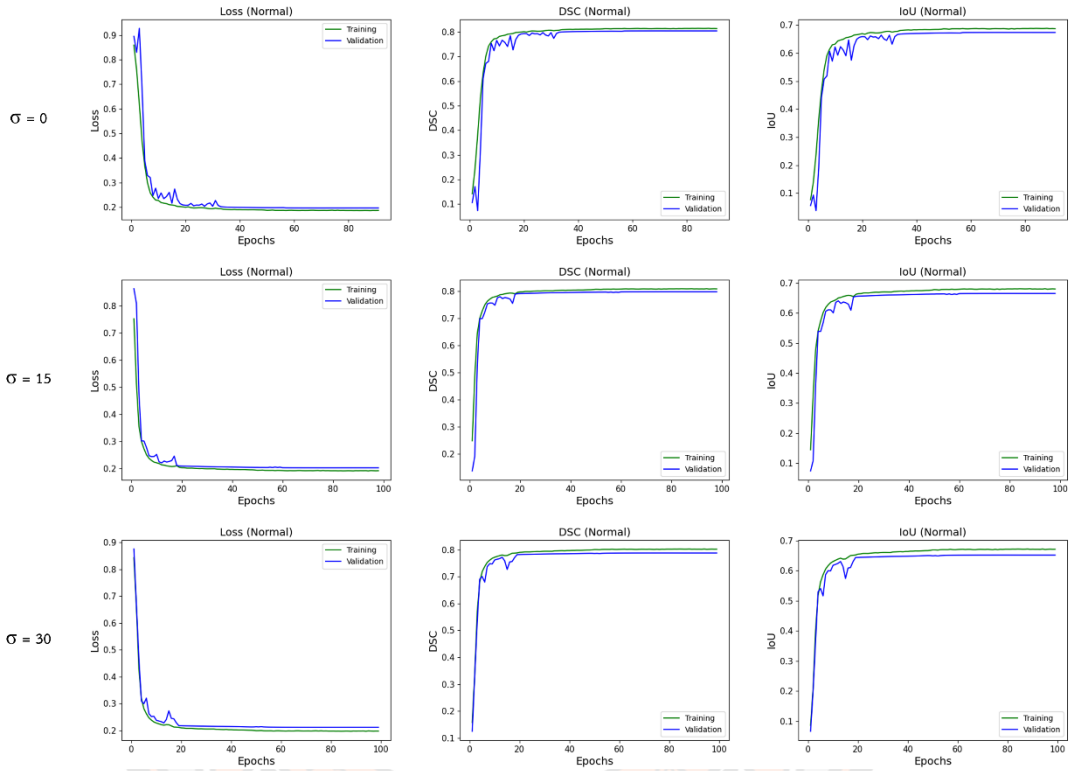
3.3 ชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

ตาราง 37 แสดงผลลัพธ์การประเมินประสิทธิของโมเดลที่ฝึกฝนด้วยสถาปัตยกรรม ThirdConv ในชุดข้อมูลที่เพิ่มสัญญาณรบกวนระดับที่ 2

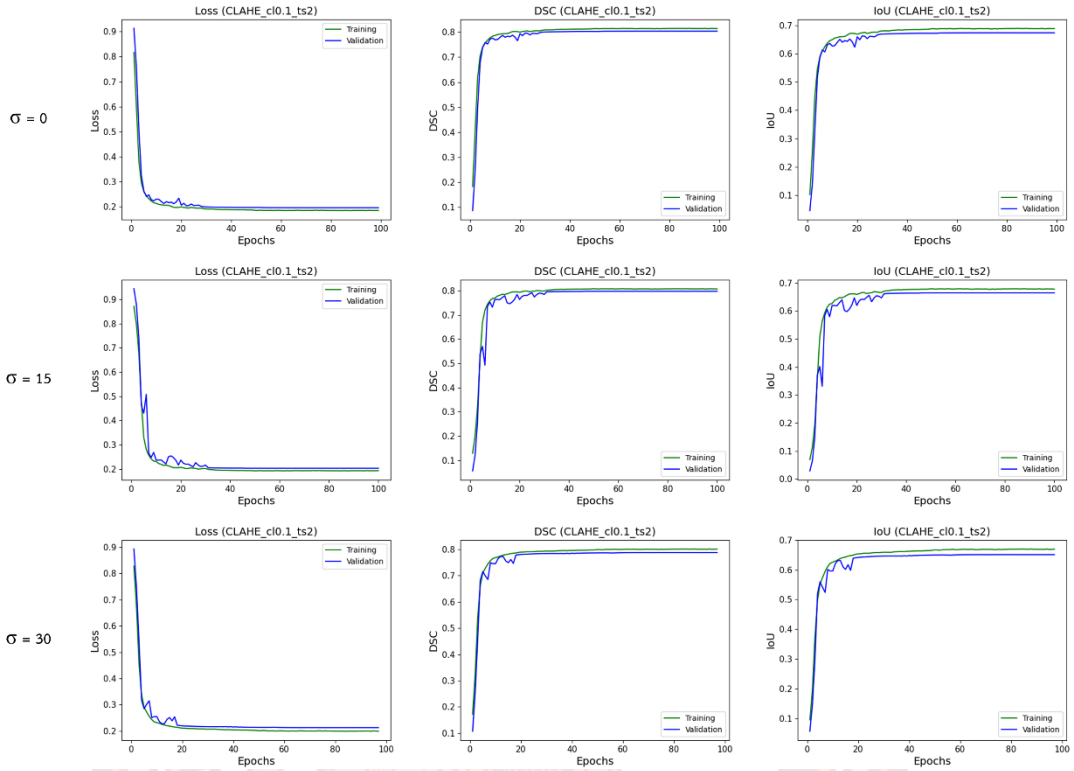
Tech	Parameter	DSC	IoU	Recall	Precision	Accuracy
Normal	-	0.786	0.862	0.915	0.827	0.993
CLAHE	cl0.1_ts2	0.789	0.865	0.918	0.830	0.993
CLAHE	cl0.2_ts8	0.785	0.858	0.914	0.822	0.993
CLAHE	cl0.3_ts6	0.795	0.873	0.927	0.836	0.993
GC	g0.5	0.792	0.868	0.923	0.832	0.993
GC	g0.7	0.788	0.863	0.916	0.828	0.993
GC	g1.5	0.789	0.864	0.917	0.829	0.993
MWF	k3	0.787	0.861	0.913	0.827	0.993
MWF	k5	0.778	0.848	0.903	0.812	0.992
MWF	k7	0.776	0.843	0.900	0.806	0.992
NLMF	s1_p8_w20	0.791	0.866	0.919	0.830	0.993
NLMF	s2_p2_w20	0.788	0.865	0.918	0.830	0.993
NLMF	s2_p6_w30	0.792	0.869	0.923	0.833	0.993

ภาคผนวก ค กราฟความสัมพันธ์ระหว่าง Training set และ Validation set

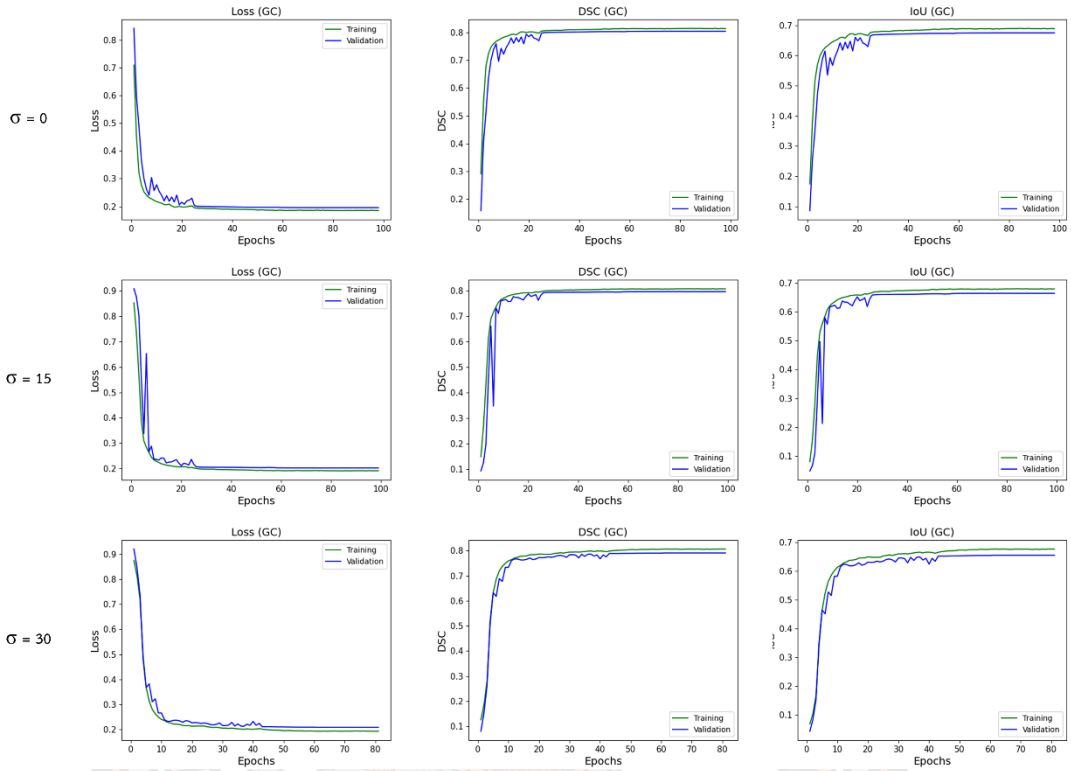
1. สถาปัตยกรรม U-net



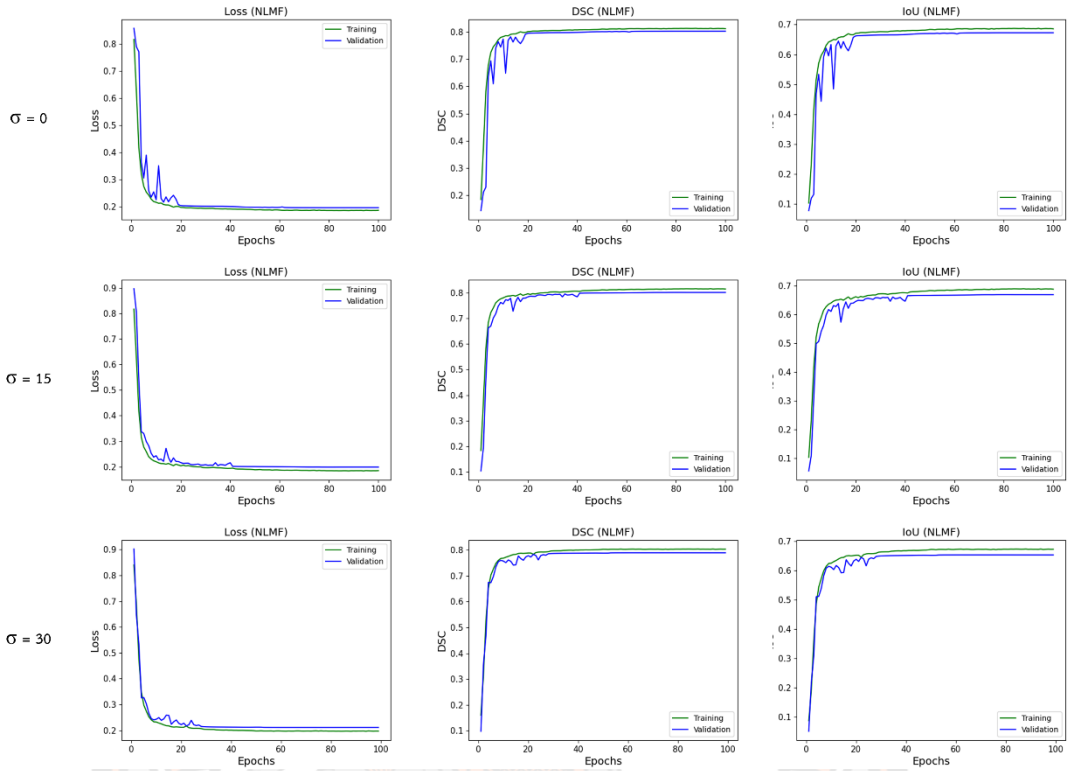
ภาพ 62 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่ไม่มีการปรับปรุงคุณภาพของภาพ



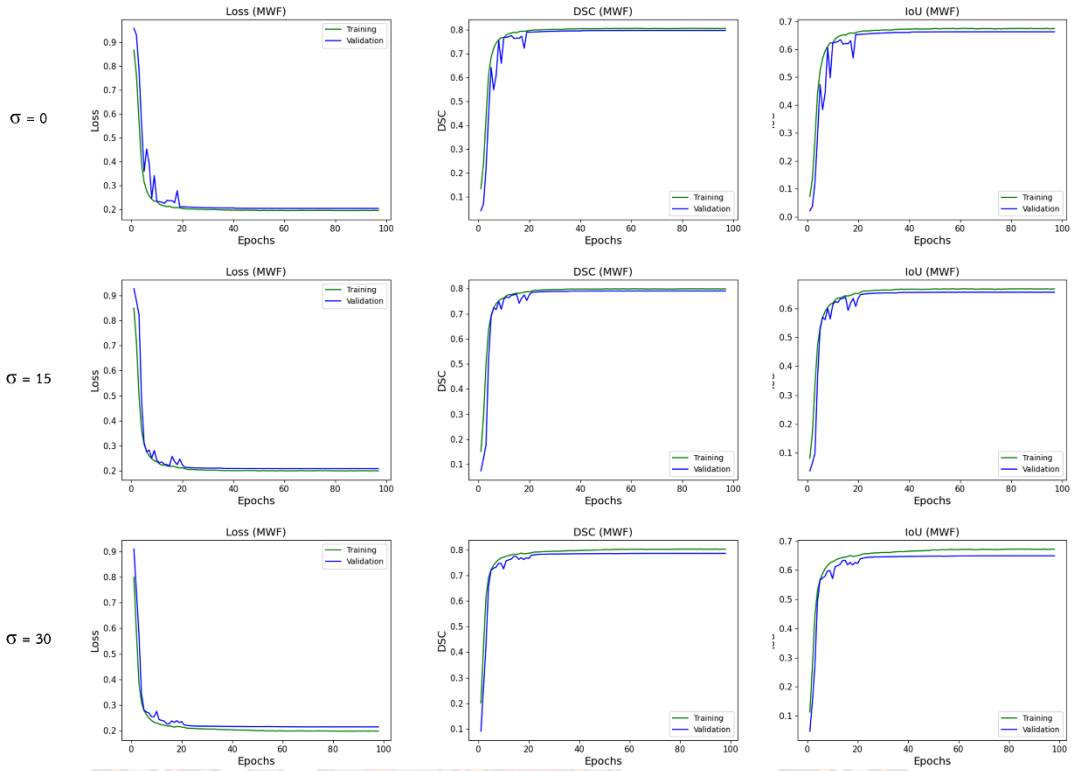
ภาพ 63 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE โดยใช้พารามิเตอร์ cliplimit = 0.1 และ tile = 2



ภาพ 64 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค GC โดยใช้พารามิเตอร์ gamma = 0.7

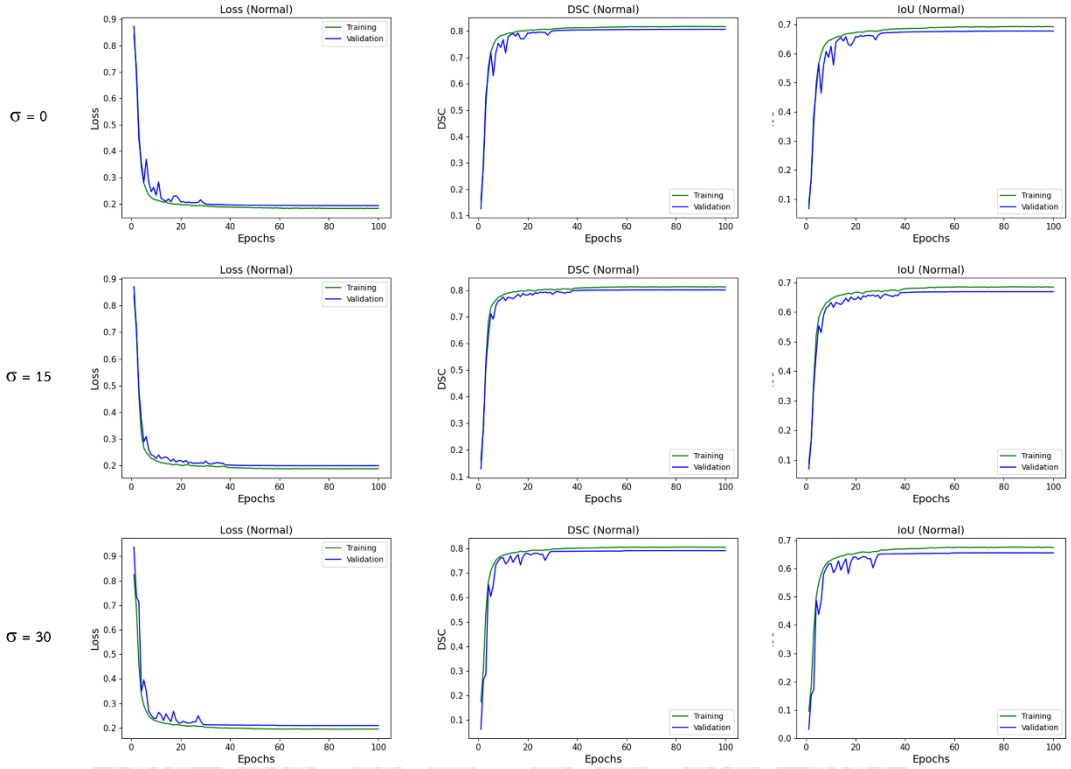


ภาพ 65 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค NLMF โดยใช้พารามิเตอร์ smooth = 1, patch = 8, และ window = 20

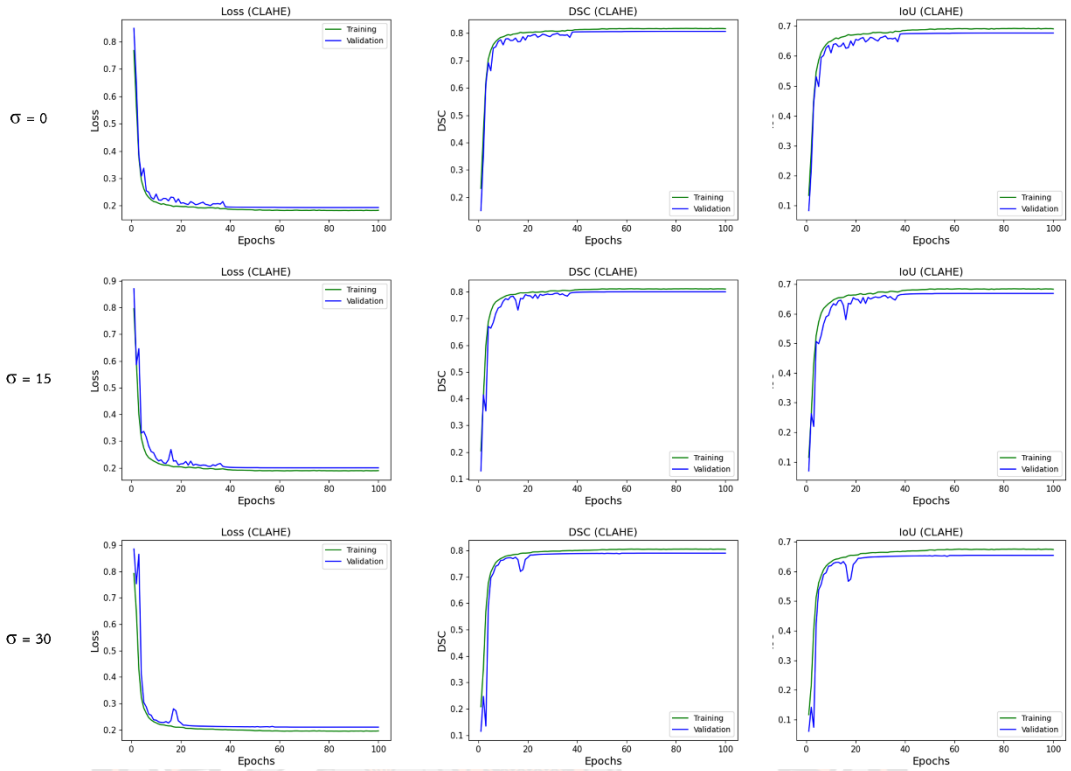


ภาพ 66 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค MWF โดยใช้พารามิเตอร์ kernel = 3

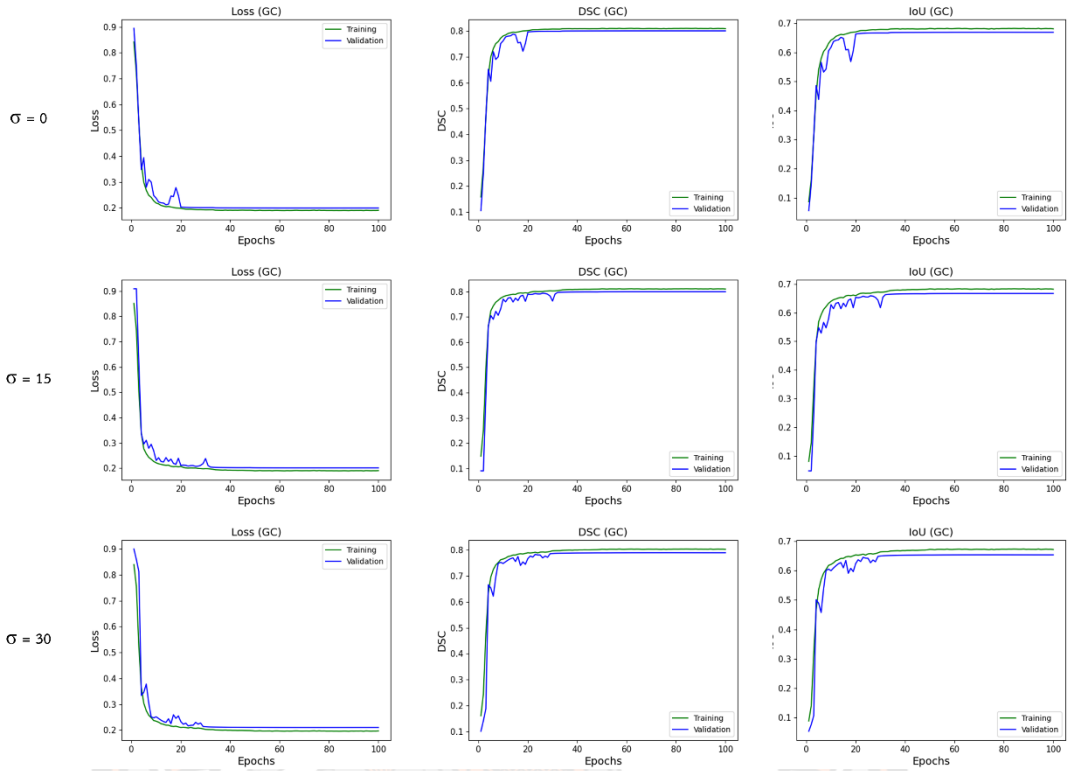
2. สถาปัตยกรรม UNet++



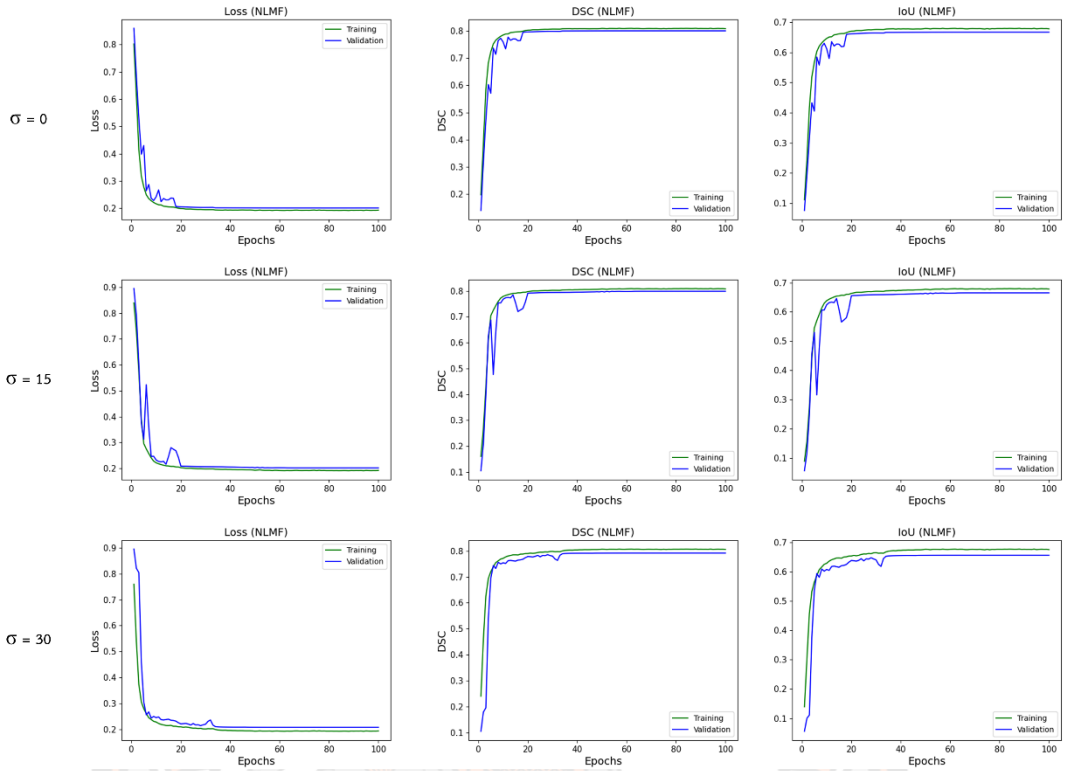
ภาพ 67 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่ไม่มีการปรับปรุงคุณภาพของภาพ



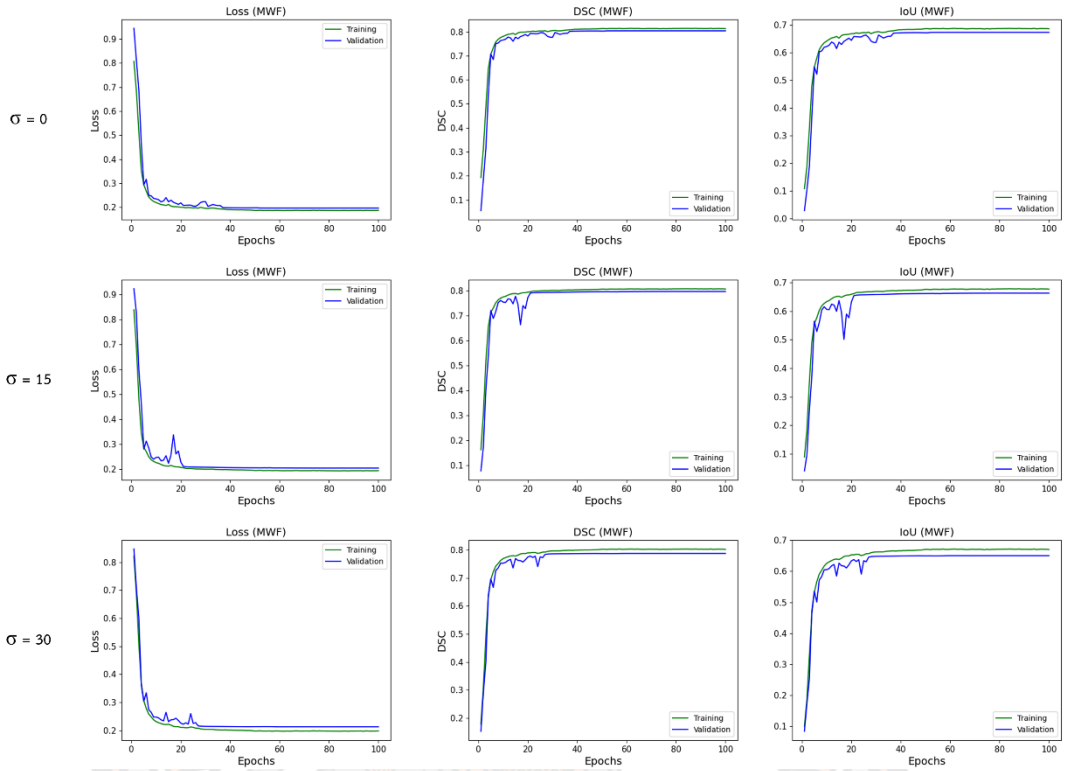
ภาพ 68 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE โดยใช้พารามิเตอร์ cliplimit = 0.1 และ tile = 2



ภาพ 69 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค GC โดยใช้พารามิเตอร์ gamma = 0.7

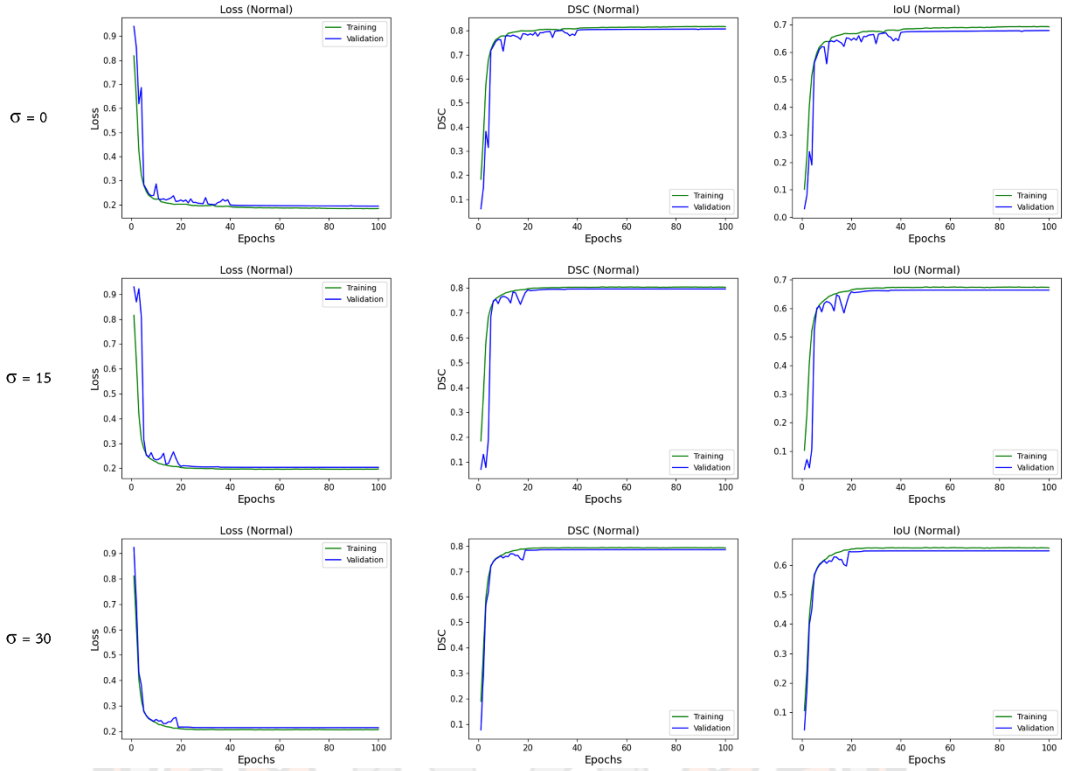


ภาพ 70 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค NLMF โดยใช้พารามิเตอร์ smooth = 1, patch = 8, และ window = 20

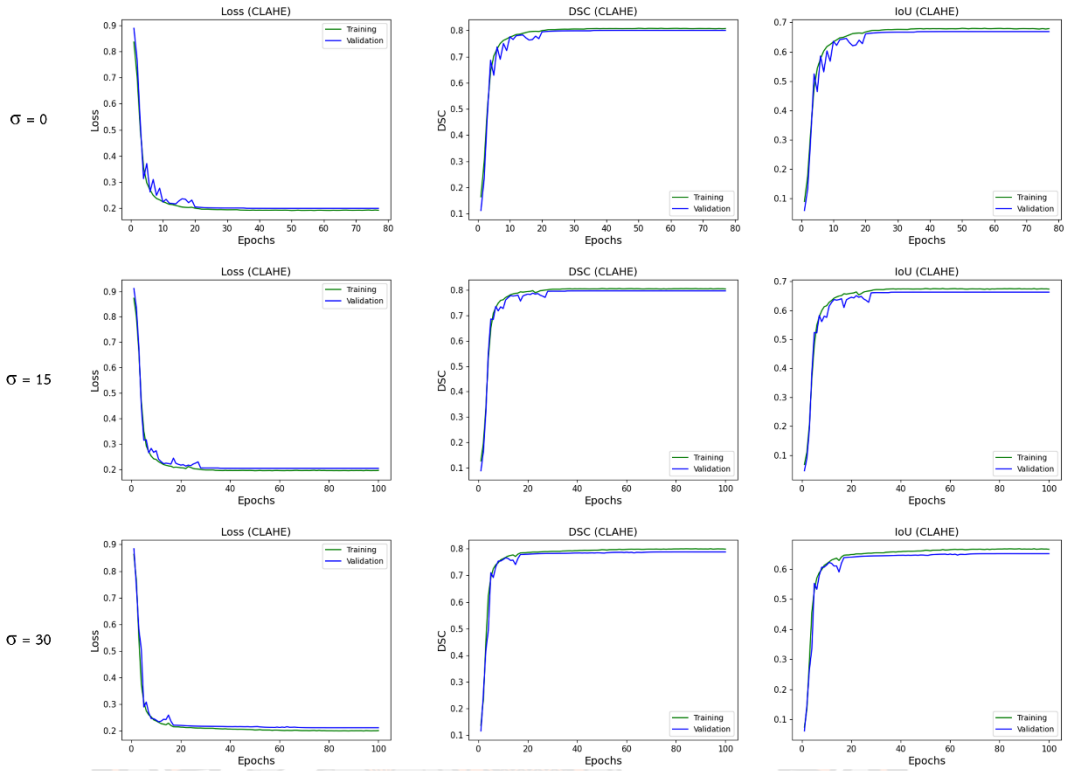


ภาพ 71 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค MWF โดยใช้พารามิเตอร์ kernel = 3

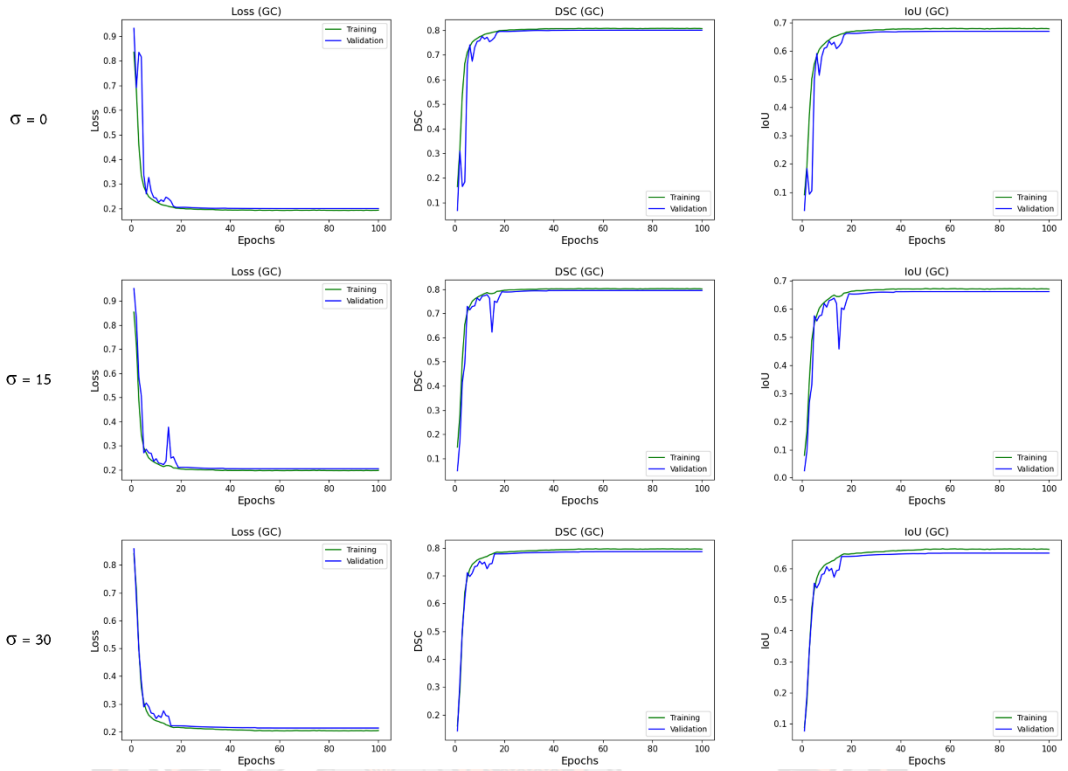
3. สถาปัตยกรรม Third Conv



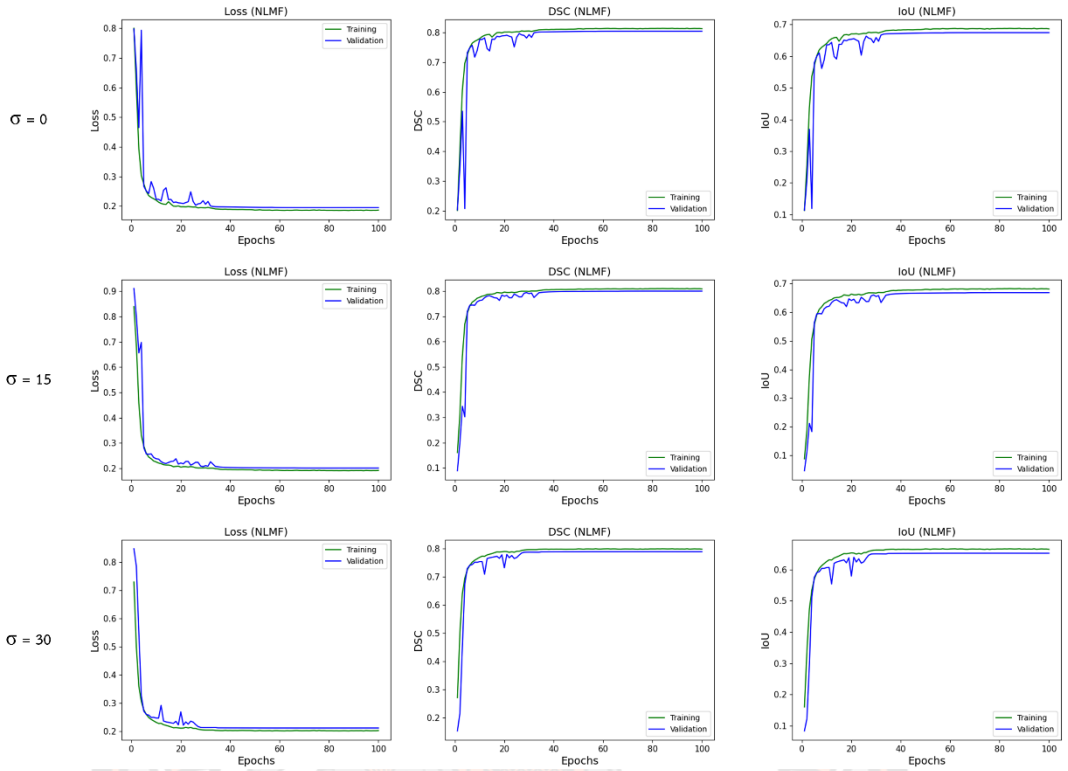
ภาพ 72 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่ไม่มีการปรับปรุงคุณภาพของภาพ



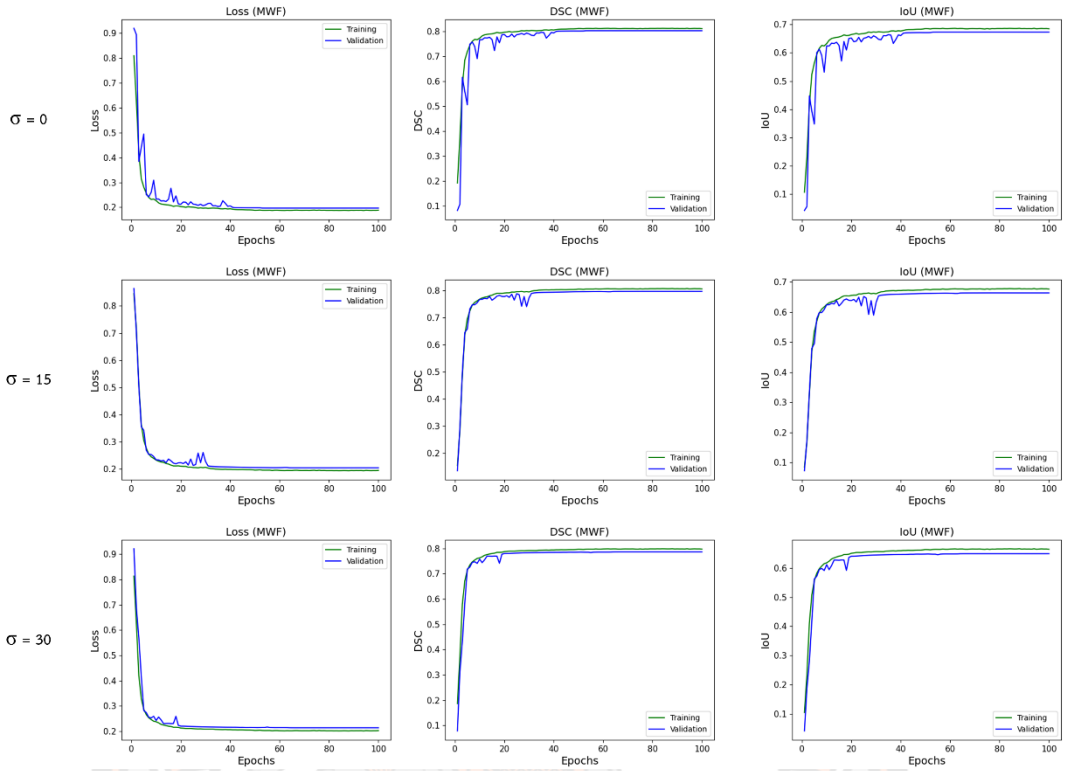
ภาพ 73 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE โดยใช้พารามิเตอร์ cliplimit = 0.1 และ tile = 2



ภาพ 74 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค GC โดยใช้พารามิเตอร์ gamma = 0.7



ภาพ 75 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค NLMF โดยใช้พารามิเตอร์ smooth = 1, patch = 8, และ window = 20



ภาพ 76 แสดงกราฟความสัมพันธ์ของ Loss, DSC, และ IoU ระหว่าง Training set และ Validation set ในชุดข้อมูลที่มีการปรับปรุงคุณภาพของภาพด้วยเทคนิค MWF โดยใช้พารามิเตอร์ kernel = 3

ภาคผนวก ง ภาษาเทียม

1. โค้ดสำหรับการแปลงไฟล์นามสกุล .nii เป็น .png

```
BEGIN Main
    GET T1, T1c, T2, FLAIR, and Mask images path for input and output
    Initialize T1, T1c, T2, FLAIR, and Mask images input path variable to list
    Initialize T1, T1c, T2, FLAIR, and Mask images output path variable to list
    FOR objects in mask path
        GET .nii image files
        COMPUTE number of slices from shape of index number 2 of .nii image files #axial plane
        FOR loop in number of slices
            COMPUTE data divided by 255 for normalize
            COMPUTE convert data normalized to uint8 type
            IF add Gaussian noise THEN
                COMPUTE gaussian noise in images that get images and sigma values
                Save .png image files
            ELSE
                Save .png image files
            ENDIF
        FOREND
    FOREND
    END Main

    BEGIN gaussian noise
        Initialize “col” and “row” variable to shape of image
        Initialize “gauss” variable to random noise that determines mean equal to 0, sigma, and number of shapes of column and row
        image noised = image plus gauss
        Return image noised
    END gaussian noise
```

2. โค้ดสำหรับปรับปรุงคุณภาพของภาพด้วยเทคนิค CLAHE

BEGIN Main

GET T1, T1c, T2, FLAIR, and Mask images path for input and output

Initialize “Cl”, “Ts”, “psnr_values_set”, “ssim_values_set”, “mse_values_set”, “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set” variable to empty list

FOR Loop 1 to 11 #Cliplimt in range 0.1 – 1.0

Append divided Loop by 10 to “Cl” list

ENDFOR

FOR Loop 2 to 11 #Tile size in range 2 – 10

Append Loop to “Ts” list

ENDFOR

FOR Order of Cliplimit in “Cl”

DISPLAY number of loop

FOR Order of Tile size in “Ts”

FOR Order of files in MRI image path

Get MRI images and convert RGB image to grayscale image

Resize images

Image enhancements using CLAHE functions that get image resized, Order of Cliplimit, and Order of Tile size

COMPUST peak signal to noise ratio values that get image resized and image enhanced

Append peak signal to noise ratio values to “psnr_values_set”

COMPUST structural similarity index values that get image resized and image enhanced

Append structural similarity index values to “ssim_values_set”

COMPUST mean squared error values that get image resized and image enhanced

Append mean squared error values to “mse_values_set”

ENDFOR

COMPUST Average of psnr, ssim, and mse values

Append Average of psnr, ssim, and mse values into “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”, respectively.

Clear values in “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”

ENDFOR

ENDFOR

Save the excel file that get order of cliplimit, order of tilesize, psnr, ssim, and mse

DISPLAY Finish

END Main

BEGIN CLAHE

Call function createCLAHE in OpenCV get into “clahe”

Apply “clahe” function and image

Return “clahe”

END CLAHE

3. โค้ดสำหรับปรับปรุงคุณภาพของภาพด้วยเทคนิค GC

BEGIN Main

GET T1, T1c, T2, FLAIR, and Mask images path for input and output

Initialize “Gm”, “psnr_values_set”, “ssim_values_set”, “mse_values_set”, “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set” variable to empty list

FOR Loop 1 to 21 #Gamma in range 0.1 – 1.0

Append divided Loop by 10 to “Gm” list

ENDFOR

FOR Order of Gamma in “Gm”

DISPLAY number of loop

FOR Order of files in MRI image path

Get MRI images and convert RGB image to grayscale image

Resize images

Image enhancements using GC functions that get image resized and Order of Gamma

COMPUST peak signal to noise ratio values that get image resized and image enhanced

Append peak signal to noise ratio values to “psnr_values_set”

COMPUST structural similarity index values that get image resized and image enhanced

Append structural similarity index values to “ssim_values_set”

COMPUST mean squared error values that get image resized and image enhanced

Append mean squared error values to “mse_values_set”

ENDFOR

COMPUST Average of psnr, ssim, and mse values

Append Average of psnr, ssim, and mse values into “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”, respectively.

Clear values in “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”

ENDFOR

Save the excel file that get order of gamma, psnr, ssim, and mse

DISPLAY Finish

END Main

BEGIN GC

gc = image divided by 255.0 to the power of gamma values, then multiplied by 255.0

Return gc

END GC

4. โค้ดสำหรับปรับปรุงคุณภาพของภาพด้วยเทคนิค NLMF

BEGIN Main

GET T1, T1c, T2, FLAIR, and Mask images path for input and output

Initialize “Smooth”, “Patchsize”, “Windowsize”, “psnr_values_set”, “ssim_values_set”, “mse_values_set”, “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set” variable to empty list

FOR Loop 1 to 6 #Smooth values in range 1 – 10

Append Loop to “Smooth” list

ENDFOR

FOR Loop 1 to 11 #Patch size in range 1 – 10

Append Loop to “Patchsize” list

ENDFOR

FOR Loop 10 to 31 that increase 5 steps #Window size in range 10 – 30

Append Loop to “Windowsize” list

ENDFOR

FOR Order of Smooth in “Smooth”

DISPLAY number of loop

FOR Order of Patch in “Patchsize”

FOR Order of Window in “Windowsize”

FOR Order of files in MRI image path

Get MRI images and convert RGB image to grayscale image

Resize images

Image enhancements using NLMF functions that get image resized, Order of Smooth, Order of Patch and Order of Window

COMPUST peak signal to noise ratio values that get image resized and image enhanced

Append peak signal to noise ratio values to “psnr_values_set”

COMPUST structural similarity index values that get image resized and image enhanced

Append structural similarity index values to “ssim_values_set”

COMPUST mean squared error values that get image resized and image enhanced

Append mean squared error values to “mse_values_set”

ENDFOR

COMPUST Average of psnr, ssim, and mse values

Append Average of psnr, ssim, and mse values into “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”, respectively.

Clear values in “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”

ENDFOR

ENDFOR

ENDFOR

Save the excel file that get order of smooth, order of patch, order of window, psnr, ssim, and mse

DISPLAY Finish

END Main

BEGIN NLMF

Call function fastNlMeansDenoising in OpenCV get into “nlmf”

Return “nlmf”

END NLMF

5. โค้ดสำหรับปรับปรุงคุณภาพของภาพด้วยเทคนิค MWF

BEGIN Main

GET T1, T1c, T2, FLAIR, and Mask images path for input and output

Initialize “Kn”, “psnr_values_set”, “ssim_values_set”, “mse_values_set”, “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set” variable to empty list

FOR Loop 3 to 8 that increase 2 steps #Kernel in range 3 – 7

Append Loop to “Kn” list

ENDFOR

FOR Order of Kernel in “Kn”

DISPLAY number of loop

FOR Order of files in MRI image path

Get MRI images and convert RGB image to grayscale image

Resize images

Image enhancements using MWF functions that get image resized and Order of Kernel

COMPUST peak signal to noise ratio values that get image resized and image enhanced

Append peak signal to noise ratio values to “psnr_values_set”

COMPUST structural similarity index values that get image resized and image enhanced

Append structural similarity index values to “ssim_values_set”

COMPUST mean squared error values that get image resized and image enhanced

Append mean squared error values to “mse_values_set”

ENDFOR

COMPUST Average of psnr, ssim, and mse values

Append Average of psnr, ssim, and mse values into “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”, respectively.

Clear values in “psnr_avg_set”, “ssim_avg_set”, and “mse_avg_set”

ENDFOR

Save the excel file that get order of kernel, psnr, ssim, and mse

DISPLAY Finish

END Main

BEGIN MWF

Call function medianBlur in OpenCV get into “MF” that get image and kernel values

Call function xphoto.createSimpleWB.balanceWhite in OpenCV get into “WF” that get “MF”

Return WF

END MWF

6. โค้ดสำหรับการแปลงไฟล์นามสกุล .png เป็น .npy

BEGIN Main

GET T1, T1c, T2, FLAIR, and Mask images path for input and output

FOR objects in number of mask path

GET .png image files

COMPUTE convert RGB image to grayscale image

COMPUTE resize image to 128 × 128

COMPUTE image data divided by 255 for normalize

COMPUTE convert data normalized to float32 type

COMPUTE expand the shape of an array of mask image to 1 dimention

Call function stack in Numpy that get T1, T1c, T2, and FLAIR images for stack data, then get data into “fuse_image”

COMPUTE find the unique elements of an array in mask using unique functions in Numpy library that get data into “val” and “counts”

IF one minus (index number 0 of counts divided by summation of counts) has more value than 0.01

DISPLAY Save

Save .npy fuse image

Save .npy mask image

ENDIF

ENDFOR

END Main

7. โค้ดสำหรับการฝึกฝนโมเดลการเรียนรู้เชิงลึกที่ใช้สถาปัตยกรรม U-net

BEGIN Train

Initialize “seed” variable to 42 both Numpy and Tenserflow libraries.

Initialize “name_model” variable to U_net

Initialize “name_enhance” variable to enhance name

Initialize “name_noise” variable to noise levels

Initialize “batch_size” variable to 16

Initialize “epochs” variable to 100

Initialize “learning rate” variable to 0.001

GET model path, train_img, val_img, , train_mask, and val_mask

COMPUTE get the list of train_img, and val_img in the specified directory and get the data to train_img_list, and val_img, respectively.

COMPUTE get the list of train_mask, and val_mask in the specified directory and get the data to train_mask_list, and val_mask_list, respectively.

GET image from ImageLoader that input train_img, train_img_list, train_mask, train_mask_list, and batch_size, then get into train_img_data

GET image from ImageLoader that input val_img, val_img_list, val_mask, val_mask_list, and batch_size, then get into val_img_data

epoch step train = number of train_img_list floor divided by batch_size

epoch step validation = number of val_img_list floor divided by batch_size

COMPUTE build_unet

COMPUTE compile the model that determines loss functions is dice loss, Optimizer is Adam that get the learning rate, and metrics is dice_coef.

Call functions “callbacks” for use ModelCheckpoint, ReduceLROnPlateau, EarlyStopping, and CSVLogger in Tensorflow library

COMPUTE fit of model that input values train_img_data, epoch step train, epoch, val_img_data, epoch step validation, and callbacks

Save the model

END Train

```
BEGIN ImageLoader
    Initialize “num” variable to number of image list
    WHILE True
        Initialize “batch_start” variable to 0
        Initialize “batch_end” equal to batch_size
        WHILE less batch_start than “num”
            limit = minimum of batch_end and “num”
            X = load_img that get img_dir, img_list(batch_start:limit)
            Y = load_img that get mask_dir, mask_list(batch_start:limit)
            YIELD X and Y
            add batch_start to batch_size
            add batch_end to batch_size
        ENDWHILE
    ENDWHILE
END ImageLoder

BEGIN load_img
    Initialize “images” variable to empty set
    FOR enumerate of index and image name in img_list
        IF File extension as npy
            image = load image that get img_dir plus image name
            Append image into “images”
        ENDIF
    ENDFOR
    images = convert “images” to array
    Return “images”
END load_img

BEGIN build_unet
    Initialize “inputs” variable to input shape using Input function
```

Initialize “e1” and “p1” variables to encoder_block that get data “inputs” and number of filters equal to 16

Initialize “e2” and “p2” variables to encoder_block that get data “p1” and number of filters equal to 32

Initialize “e3” and “p3” variables to encoder_block that get data “p2” and number of filters equal to 64

Initialize “e4” and “p4” variables to encoder_block that get data “p3” and number of filters equal to 64

Initialize “b1” variable to conv_block that get data “p4” and number of filters equal to 256

Initialize “d1” variable to decoder_block that get data “b1”, “e4”, and number of filters equal to 128

Initialize “d2” variable to decoder_block that get data “d1”, “e3”, and number of filters equal to 64

Initialize “d3” variable to decoder_block that get data “d2”, “e2”, and number of filters equal to 32

Initialize “d4” variable to decoder_block that get data “d3”, “e1”, and number of filters equal to 16

outputs = 2D convolution that get filters equal to 1, kernel size equal to 1, padding is same, activation function is sigmoid, and “d4”

COMPUTE model that get “inputs” and “outputs”

Return model

END build_unet

BEGIN encorder_block

Initialize “x” variable to conv_block that get data “inputs” and number of filters

Initialize “p” variable to 2 × 2 Max pooling and get the “x” values to compute

Return “x” and “p”

END encorder_block

BEGIN decorder_block

Initialize “x” variable to 2D convolution transpose that get data “inputs”, number of filters, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x” variable to concatenate between “x” and “skip_connect”

Initialize “x” variable to conv_block that get data “x” and number of filters

Return “x”

END decorder_block

BEGIN conv_block

Initialize “x” variable to 2D convolution that get data “inputs”, number of filters, 3 × 3 kernel size, and padding is same.

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to ReLU activation that get data “x”

Initialize “x” variable to 2D convolution that get data “x”, number of filters, 3 × 3 kernel size, and padding is same.

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to ReLU activation that get data “x”

Return “x”

END conv_block

BEGIN dice_coef

Initialize “smooth” variable to 1.0

COMPUTE reference data into 1D using flatten function

COMPUTE prediction data into 1D using flatten function

intersection = computes the sum of elements across dimensions of reference data and prediction data

Return (2 times intersection plus smooth) divided by (computes the sum of elements across dimensions of reference data plus computes the sum of elements across dimensions of prediction data plus smooth)

END dice_coef

BEGIN dice_loss

Return 1 minus dice_coef

END dice_loss

BEGIN Test

Initialize “seed” variable to 42 both Numpy and Tenserflow libraries.

Initialize “name_model” variable to U_net

Initialize “name_enhance” variable to enhance name

Initialize “name_noise” variable to noise levels

Initialize “batch_size” variable to 16

Initialize “epochs” variable to 100

GET model path, csv path, test_img and test_mask

Initialize “my_model” variable to load model from model path

Initialize “Score” variable to empty list

Initialize “counts” variable to 0

FOR i and j in zip that get test_img and test_mask

test_img = load test image from index i

test_mask = load test mask from index j

test_image = expand the shape of an array of test_img that setting axis equal to 0

pred = prediction of model using predict function that get test_image and setting index equal to 0

pred = using squeeze function to convert data to 1 dimension

COMPUST create loss, dsc, and IoU graph from loss_graph

COMPUST create graph for compare MRI images and mask predicted from create_graph

add counts to 1

test_mask = (more test_mask than 0.5) and use flatten to 1 dimension

pred = (more pred than 0.5) and use flatten to 1 dimension

Initialize “Dice_value” variable to calculate dsc score

Initialize “jac_value” variable to calculate jaccard score

Initialize “recall_value” variable to calculate recall score

Initialize “precision_value” variable to calculate precision score

Initialize “accuracy_value” variable to calculate accuracy score

IF precision_value not equal to 0

Append “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” into “Score” list

ENDIF

ENDFOR

score = list of s(1:) FOR data in Score list ENDFOR

COMPUTE Average of “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” in score lists

DISPLAY “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value”

Save Average of “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” to excel file

END Test

BEGIN loss_graph

loss = get the loss history from model

val_loss = get the validation loss history from model

epoch = 1 to number of loss plus 1

DETERMINE plot loss data and label line of graph is green color

DETERMINE plot validation loss data and label line of graph is blue color

DETERMINE graph name and use font size equal to 14

DETERMINE x axis name to Epochs and use font size equal to 14

DETERMINE y axis name to graph name and use font size equal to 14

Save graph

END loss_graph

BEGIN create_graph

Initialize “figsize” variable to 12 × 8

Plot the six graph and title determines consists of t1n Image, t1c Image, flair Image, t2w Image, reference mask, and prediction mask

Save graph

END create_graph

8. โค้ดสำหรับการฝึกฝนโมเดลการเรียนรู้เชิงลึกที่ใช้สถาปัตยกรรม UNet++

BEGIN Train

Initialize “seed” variable to 42 both Numpy and Tenserflow libraries.

Initialize “name_model” variable to UNetplusplus

Initialize “name_enhance” variable to enhance name

Initialize “name_noise” variable to noise levels

Initialize “batch_size” variable to 16

Initialize “epochs” variable to 100

Initialize “learning rate” variable to 0.001

GET model path, train_img, val_img, , train_mask, and val_mask

COMPUTE get the list of train_img, and val_img in the specified directory and get the data to train_img_list, and val_img, respectively.

COMPUTE get the list of train_mask, and val_mask in the specified directory and get the data to train_mask_list, and val_mask_list, respectively.

GET image from ImageLoader that input train_img, train_img_list, train_mask, train_mask_list, and batch_size, then get into train_img_data

GET image from ImageLoader that input val_img, val_img_list, val_mask, val_mask_list, and batch_size, then get into val_img_data

epoch step train = number of train_img_list floor divided by batch_size

epoch step validation = number of val_img_list floor divided by batch_size

COMPUTE build_unet_plus_plus

COMPUTE compile the model that determines loss functions is dice loss, Optimizer is Adam that get the learning rate, and metrics is dice_coef.

Call functions “callbacks” for use ModelCheckpoint, ReduceLROnPlateau, EarlyStopping, and CSVLogger in Tensorflow library

COMPUTE fit of model that input values train_img_data, epoch step train, epoch, val_img_data, epoch step validation, and callbacks

Save the model

END Train

BEGIN ImageLoader

Initialize “num” variable to number of image list

WHILE True

Initialize “batch_start” variable to 0

Initialize “batch_end” equal to batch_size

WHILE less batch_start than “num”

limit = minimum of batch_end and “num”

X = load_img that get img_dir, img_list(batch_start:limit)

Y = load_img that get mask_dir, mask_list(batch_start:limit)

YIELD X and Y

add batch_start to batch_size

add batch_end to batch_size

ENDWHILE

ENDWHILE

END ImageLoder

BEGIN load_img

Initialize “images” variable to empty set

FOR enumerate of index and image name in img_list

IF File extension as npy

image = load image that get img_dir plus image name

Append image into “images”

ENDIF

ENDFOR

images = convert “images” to array

Return “images”

END load_img

BEGIN build_unet_plus_plus

Initialize “inputs” variable to input shape using Input function

Initialize “x_00” variable to conv_block that get data “inputs” and number of filters equal to 16

Initialize “p_00” variable to 2 × 2 2D max pooling that get data “x_00”

Initialize “x_10” variable to conv_block that get data “p_00” and number of filters equal to 32

Initialize “p_10” variable to 2 × 2 2D max pooling that get data “x_10”

Initialize “x_20” variable to conv_block that get data “p_10” and number of filters equal to 64

Initialize “p_20” variable to 2 × 2 2D max pooling that get data “x_20”

Initialize “x_30” variable to conv_block that get data “p_20” and number of filters equal to 32

Initialize “p_30” variable to 2 × 2 2D max pooling that get data “x_30”

Initialize “x_40” variable to conv_block that get data “p_30” and number of filters equal to 256

Initialize “Up_31” variable to 2D convolution transpose that get data “x_40”, number of filters equal to 128, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_31” variable to concatenate between “Up_31” and “x_30”

Initialize “x_31” variable to conv_block that get data “x_31” and number of filters equal to 128

Initialize “Up_21” variable to 2D convolution transpose that get data “x_30”, number of filters equal to 64, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_21” variable to concatenate between “Up_21” and “x_20”

Initialize “x_21” variable to conv_block that get data “x_21” and number of filters equal to 64

Initialize “Up_11” variable to 2D convolution transpose that get data “x_20”, number of filters equal to 32, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_11” variable to concatenate between “Up_11” and “x_10”

Initialize “x_11” variable to conv_block that get data “x_11” and number of filters equal to 32

Initialize “Up_01” variable to 2D convolution transpose that get data “x_10”, number of filters equal to 16, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_01” variable to concatenate between “Up_01” and “x_00”

Initialize “x_01” variable to conv_block that get data “x_01” and number of filters equal to 32

Initialize “Up_22” variable to 2D convolution transpose that get data “x_31”, number of filters equal to 64, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_22” variable to concatenate between “Up_22”, “x_21”, and “x_20”

Initialize “x_22” variable to conv_block that get data “x_22” and number of filters equal to 64

Initialize “Up_12” variable to 2D convolution transpose that get data “x_21”, number of filters equal to 32, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_12” variable to concatenate between “Up_12”, “x_11”, and “x_10”

Initialize “x_12” variable to conv_block that get data “x_12” and number of filters equal to 32

Initialize “Up_02” variable to 2D convolution transpose that get data “x_11”, number of filters equal to 16, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_02” variable to concatenate between “Up_02”, “x_01”, and “x_00”

Initialize “x_02” variable to conv_block that get data “x_02” and number of filters equal to 16

Initialize “Up_13” variable to 2D convolution transpose that get data “x_22”, number of filters equal to 32, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_13” variable to concatenate between “Up_13”, “x_12”, “x_11”, and “x_10”

Initialize “x_13” variable to conv_block that get data “x_13” and number of filters equal to 32

Initialize “Up_03” variable to 2D convolution transpose that get data “x_12”, number of filters equal to 16, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_03” variable to concatenate between “Up_03”, “x_02”, “x_01”, and “x_00”

Initialize “x_03” variable to conv_block that get data “x_03” and number of filters equal to 16

Initialize “Up_04” variable to 2D convolution transpose that get data “x_13”, number of filters equal to 16, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “x_04” variable to concatenate between “Up_04”, “x_03”, “x_02”, “x_01”, and “x_00”

Initialize “x_04” variable to conv_block that get data “x_04” and number of filters equal to 16

outputs = 2D convolution that get filters equal to 1, kernel size equal to 1, padding is same, activation function is sigmoid, and “x_04”

COMPUTE model that gets “inputs” and “outputs”

Return model

END build_unet_plus_plus

BEGIN conv_block

Initialize “x” variable to 2D convolution that get data “inputs”, number of filters, 3 × 3 kernel size, and padding is same.

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to ReLU activation that get data “x”

Initialize “x” variable to 2D convolution that get data “x”, number of filters, 3 × 3 kernel size, and padding is same.

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to ReLU activation that get data “x”

Return “x”

END conv_block

BEGIN dice_coef

Initialize “smooth” variable to 1.0

COMPUTE reference data into 1D using flatten function

COMPUTE prediction data into 1D using flatten function

intersection = computes the sum of elements across dimensions of reference data and prediction data

Return (2 times intersection plus smooth) divided by (computes the sum of elements across dimensions of reference data plus computes the sum of elements across dimensions of prediction data plus smooth)

END dice_coef

BEGIN dice_loss

Return 1 minus dice_coef

END dice_loss

BEGIN Test

Initialize “seed” variable to 42 both Numpy and Tenserflow libraries.

Initialize “name_model” variable to U_net

Initialize “name_enhance” variable to enhance name

Initialize “name_noise” variable to noise levels

Initialize “batch_size” variable to 16

Initialize “epochs” variable to 100

GET model path, csv path, test_img and test_mask

Initialize “my_model” variable to load model from model path

Initialize “Score” variable to empty list

Initialize “counts” variable to 0

FOR i and j in zip that get test_img and test_mask

test_img = load test image from index i

test_mask = load test mask from index j

test_image = expand the shape of an array of test_img that setting axis equal to 0

pred = prediction of model using predict function that get test_image and setting index equal to 0

pred = using squeeze function to convert data to 1 dimension

COMPUST create loss, dsc, and IoU graph from loss_graph

COMPUST create graph for compare MRI images and mask predicted from create_graph

add counts to 1

test_mask = (more test_mask than 0.5) and use flatten to 1 dimension

pred = (more pred than 0.5) and use flatten to 1 dimension

Initialize “Dice_value” variable to calculate dsc score

Initialize “jac_value” variable to calculate jaccard score

Initialize “recall_value” variable to calculate recall score

Initialize “precision_value” variable to calculate precision score

Initialize “accuracy_value” variable to calculate accuracy score

IF precision_value not equal to 0

Append “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” into “Score” list

ENDIF

ENDFOR

score = list of s(1:) FOR data in Score list ENDFOR

COMPUTE Average of “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” in score lists

DISPLAY “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value”

Save Average of “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” to excel file

END Test

BEGIN loss_graph

loss = get the loss history from model

val_loss = get the validation loss history from model

epoch = 1 to number of loss plus 1

DETERMINE plot loss data and label line of graph is green color

DETERMINE plot validation loss data and label line of graph is blue color

DETERMINE graph name and use font size equal to 14

DETERMINE x axis name to Epochs and use font size equal to 14

DETERMINE y axis name to graph name and use font size equal to 14

Save graph

END loss_graph

BEGIN create_graph

Initialize “figsize” variable to 12 × 8

Plot the six graph and title determines consists of t1n Image, t1c Image, flair Image, t2w Image, reference mask, and prediction mask

Save graph

END create_graph

9. โค้ดสำหรับการฝึกฝนโมเดลการเรียนรู้เชิงลึกที่ใช้สถาปัตยกรรม Third Conv

BEGIN Train

Initialize “seed” variable to 42 both Numpy and Tenserflow libraries.

Initialize “name_model” variable to My_net

Initialize “name_enhance” variable to enhance name

Initialize “name_noise” variable to noise levels

Initialize “batch_size” variable to 16

Initialize “epochs” variable to 100

Initialize “learning rate” variable to 0.001

GET model path, train_img, val_img, , train_mask, and val_mask

COMPUTE get the list of train_img, and val_img in the specified directory and get the data to train_img_list, and val_img, respectively.

COMPUTE get the list of train_mask, and val_mask in the specified directory and get the data to train_mask_list, and val_mask_list, respectively.

GET image from ImageLoader that input train_img, train_img_list, train_mask, train_mask_list, and batch_size, then get into train_img_data

GET image from ImageLoader that input val_img, val_img_list, val_mask, val_mask_list, and batch_size, then get into val_img_data

epoch step train = number of train_img_list floor divided by batch_size

epoch step validation = number of val_img_list floor divided by batch_size

COMPUTE build_Mynet

COMPUTE compile the model that determines loss functions is dice loss, Optimizer is Adam that get the learning rate, and metrics is dice_coef.

Call functions “callbacks” for use ModelCheckpoint, ReduceLROnPlateau, EarlyStopping, and CSVLogger in Tensorflow library

COMPUTE fit of model that input values train_img_data, epoch step train, epoch, val_img_data, epoch step validation, and callbacks

Save the model

END Train

```
BEGIN ImageLoader
    Initialize “num” variable to number of image list
    WHILE True
        Initialize “batch_start” variable to 0
        Initialize “batch_end” equal to batch_size
        WHILE less batch_start than “num”
            limit = minimum of batch_end and “num”
            X = load_img that get img_dir, img_list(batch_start:limit)
            Y = load_img that get mask_dir, mask_list(batch_start:limit)
            YIELD X and Y
            add batch_start to batch_size
            add batch_end to batch_size
        ENDWHILE
    ENDWHILE
END ImageLoder

BEGIN load_img
    Initialize “images” variable to empty set
    FOR enumerate of index and image name in img_list
        IF File extension as npy
            image = load image that get img_dir plus image name
            Append image into “images”
        ENDIF
    ENDFOR
    images = convert “images” to array
    Return “images”
END load_img

BEGIN build_Mynet
    Initialize “inputs” variable to input shape using Input function
```

Initialize “e1” and “p1” variables to encoder_block that get data “inputs” and number of filters equal to 16

Initialize “e2” and “p2” variables to encoder_block that get data “p1” and number of filters equal to 32

Initialize “e3” and “p3” variables to encoder_block that get data “p2” and number of filters equal to 64

Initialize “e4” and “p4” variables to encoder_block that get data “p3” and number of filters equal to 64

Initialize “b1” variable to conv_block that get data “p4” and number of filters equal to 256

Initialize “sk1” variable to connect that get data “e1”, “e2”, and number of filters equal to 16

Initialize “sk2” variable to connect that get data “e2”, “e3”, and number of filters equal to 32

Initialize “sk3” variable to connect that get data “e3”, “e4”, and number of filters equal to 64

Initialize “sk4” variable to connect that get data “e4”, “b1”, and number of filters equal to 128

Initialize “d1” variable to decoder_block that get data “b1”, “sk4”, and number of filters equal to 128

Initialize “d2” variable to decoder_block that get data “d1”, “sk3”, and number of filters equal to 64

Initialize “d3” variable to decoder_block that get data “d2”, “sk2”, and number of filters equal to 32

Initialize “d4” variable to decoder_block that get data “d3”, “sk1”, and number of filters equal to 16

outputs = 2D convolution that get filters equal to 1, kernel size equal to 1, padding is same, activation function is sigmoid, and “d4”

COMPUTE model that gets “inputs” and “outputs”

Return model

END build_Mynet

BEGIN encorder_block

Initialize “x” variable to conv_block that get data “inputs”, number of filters and 7 × 7 kernel size

Initialize “y” variable to conv_block that get data “inputs”, number of filters and 5 × 5 kernel size

Initialize “z” variable to conv_block that get data “inputs”, number of filters and 3 × 3 kernel size

Initialize “total” variable to addition parameters “x”, “y”, and “z”

Initialize “p” variable to 2 × 2 Max pooling and get the “total” values to compute

Return “total” and “p”

END encorder_block

BEGIN connect

Initialize “x” variable to 2D convolution that gets “inputs”, number of filters, 3 × 3 kernel size, and padding is same

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to Leaky ReLU activation that get data “x”

Initialize “u” variable to 2D convolution transpose that get data “inputs”, number of filters, 2 × 2 kernel size, 2 × 2 strides, and padding is same.

Initialize “t” variable to addition parameters “x” and “u”

Return “t”

END connect

BEGIN decorder_block

Initialize “x” variable to 2D convolution transpose that get data “inputs”, number of filters, 2 × 2 kernel size, 2 × 2 strides, and padding is same

Initialize “x” variable to concatenate between “x” and “skip_connect”

Initialize “x” variable to conv_block that get data “x” and number of filters

Return “x”

END decorder_block

BEGIN conv_block

Initialize “x” variable to 2D convolution that get data “inputs”, number of filters, 3 × 3 kernel size, and padding is same.

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to Leaky ReLU activation that get data “x”

Initialize “x” variable to 2D convolution that get data “x”, number of filters, 3 × 3 kernel size, and padding is same.

Initialize “x” variable to batch normalization that get data “x”

Initialize “x” variable to Leaky ReLU activation that get data “x”

Return “x”

END conv_block

BEGIN dice_coef

Initialize “smooth” variable to 1.0

COMPUTE reference data into 1D using flatten function

COMPUTE prediction data into 1D using flatten function

intersection = computes the sum of elements across dimensions of reference data and prediction data

Return (2 times intersection plus smooth) divided by (computes the sum of elements across dimensions of reference data plus computes the sum of elements across dimensions of prediction data plus smooth)

END dice_coef

BEGIN dice_loss

Return 1 minus dice_coef

END dice_loss

BEGIN Test

Initialize “seed” variable to 42 both Numpy and Tenserflow libraries.

Initialize “name_model” variable to U_net

Initialize “name_enhance” variable to enhance name

Initialize “name_noise” variable to noise levels

Initialize “batch_size” variable to 16

Initialize “epochs” variable to 100

GET model path, csv path, test_img and test_mask

Initialize “my_model” variable to load model from model path

Initialize “Score” variable to empty list

Initialize “counts” variable to 0

FOR i and j in zip that get test_img and test_mask

test_img = load test image from index i

test_mask = load test mask from index j

test_image = expand the shape of an array of test_img that setting axis equal to 0

pred = prediction of model using predict function that get test_image and setting index equal to 0

pred = using squeeze function to convert data to 1 dimension

COMPUST create loss, dsc, and IoU graph from loss_graph

COMPUST create graph for compare MRI images and mask predicted from create_graph

add counts to 1

test_mask = (more test_mask than 0.5) and use flatten to 1 dimension

pred = (more pred than 0.5) and use flatten to 1 dimension

Initialize “Dice_value” variable to calculate dsc score

Initialize “jac_value” variable to calculate jaccard score

Initialize “recall_value” variable to calculate recall score

Initialize “precision_value” variable to calculate precision score

Initialize “accuracy_value” variable to calculate accuracy score

IF precision_value not equal to 0

Append “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” into “Score” list

ENDIF

ENDFOR

score = list of s(1:) FOR data in Score list ENDFOR

COMPUTE Average of “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” in score lists

DISPLAY “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value”

Save Average of “Dice_value”, “jac_value”, “recall_value”, “precision_value”, and “accuracy_value” to excel file

END Test

BEGIN loss_graph

loss = get the loss history from model

val_loss = get the validation loss history from model

epoch = 1 to number of loss plus 1

DETERMINE plot loss data and label line of graph is green color

DETERMINE plot validation loss data and label line of graph is blue color

DETERMINE graph name and use font size equal to 14

DETERMINE x axis name to Epochs and use font size equal to 14

DETERMINE y axis name to graph name and use font size equal to 14

Save graph

END loss_graph

BEGIN create_graph

Initialize “figsize” variable to 12 × 8

Plot the six graph and title determines consists of t1n Image, t1c Image, flair Image, t2w Image, reference mask, and prediction mask

Save graph

END create_graph

บรรณานุกรม

บรรณานุกรม

1. Worldwide cancer data [Internet]. World Cancer Research Fund International; 2022 [updated 2022 Mar 23; cited 2024 Mar 26]. Available from: https://www.wcrf.org/cancer-trends/worldwide-cancer-data/.

2. Brain Tumor: Statistics [Internet]. Cancer.Net; 2023 [updated 2023 Mar; cited 2024 Mar 26]. Available from: https://www.cancer.net/cancer-types/brain-tumor/statistics.

3. Global cancer data by country [Internet]. World Cancer Research Fund International; 2020 [updated 2022 Mar 23; cited 2024 Mar 26]. Available from: https://www.wcrf.org/cancer-trends/global-cancer-data-by-country/.

4. Baid U, Ghodasara S, Mohan S, Bilello M, Calabrese E, Colak E, et al. The rsna-asnr-miccai brats 2021 benchmark on brain tumor segmentation and radiogenomic classification. arXiv preprint arXiv:210702314. 2021.

5. Lin M, Momin S, Lei Y, Wang H, Curran WJ, Liu T, et al. Fully automated segmentation of brain tumor from multiparametric MRI using 3D context deep supervised U-Net. Medical Physics. 2021;48(8):4365-74.

6. Walsh KM, Ohgaki H, Wrensch MR. Chapter 1 - Epidemiology. In: Berger MS, Weller M, editors. Handbook of Clinical Neurology. 134: Elsevier; 2016. p. 3-18.

7. Sheard. R. Understaning Brain Tumours: Cancer Council Australia; 2022.

8. Vo T, Dave P, Bajpai G, Kashef R, Khan N, editors. Brain Tumor Segmentation in MRI Images Using A Modified U-Net Model. 2022 IEEE International Conference on Digital Health (ICDH); 2022: IEEE.

9. Jain R, Dixit S, Kumar V, Verma B, editors. SEMC-Net: A Shared-Encoder Multi-Class Learner. 2023 4th International Conference for Emerging Technology (INCET); 2023: IEEE.

10. Güneş AM, van Rooij W, Gulshad S, Slotman B, Dahele M, Verbakel W. Impact of imperfection in medical imaging data on deep learning-based segmentation performance: An experimental study using synthesized data. Medical physics. 2023;50(10):6421-32.

11. Menze BH, Jakab A, Bauer S, Kalpathy-Cramer J, Farahani K, Kirby J, et al. The multimodal brain tumor image segmentation benchmark (BRATS). IEEE transactions on medical imaging. 2014;34(10):1993-2024.

12. Bakas S, Akbari H, Sotiras A, Bilello M, Rozycki M, Kirby JS, et al. Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features. Scientific data. 2017;4(1):1-13.

13. Bakas S, Akbari H, Sotiras A, Bilello M, Rozycki M, Kirby J, et al. Segmentation labels and radiomic features for the pre-operative scans of the TCGA-LGG collection. The cancer imaging archive. 2017;286.

14. Bakas S, Akbari H, Sotiras A, Bilello M, Rozycki M, Kirby J, et al. Segmentation labels and radiomic features for the pre-operative scans of the TCGA-GBM collection. The cancer imaging archive. Nat Sci Data. 2017;4(170117):10.1038.

15. พวงทอง ไกรพิบูลย์. ไกลโอมา เนื้องอกไกลโอมา (Glioma) [Internet]. หาหมอ แนะแนวทุกโรค และนำทุกหมอ; 2565 [updated 2565 พ.ค. 15; cited 2567 มี.ค. 26]. Available from: https://haamor.com/ไกลโอมา.

16. ธนันต์ จิตรวัชรโกมล. มะเร็งเนื้อเยื่อสมอง [Internet]. คณะแพทยศาสตร์ศิริราชพยาบาล มหาวิทยาลัยมหิดล; [ม.ป.ป] [cited 2567 มี.ค. 26]. Available from: https://sirirajradonco.org/Knowledge-detail/25.

17. Cancer in Thailand [Internet]: Ministry of Public Health; 2021 [cited 2024 Mar 26]. Available from: https://www.nci.go.th/e_book/cit_x/index.html.

18. Board WCoTE. Central Nervous System Tumours: International Agency for Research on Cancer; 2022.

19. Horbinski C, Nabors LB, Portnow J, Baehring J, Bhatia A, Bloch O, et al. Central Nervous System Cancers, Version 2.2022 Featured Updates to the NCCN Guidelines. JOURNAL OF THE NATIONAL COMPREHENSIVE CANCER NETWORK. 2023;21(1):12-20.

20. Magnetic Resonance Imaging (MRI) of the Brain and Spine: Basics [Internet]. 2016 [updated 2016 Apr 07; cited 2024 Mar 26]. Available from: https://case.edu/med/neurology/NR/MRI%20Basics.htm.

21. Lawtomated. A.I. Technical: Machine vs Deep Learning [Internet]. 2020 [updated 2019 Apr 28; cited 2024 Mar 26]. Available from: https://lawtomated.com/a-i-technical-machine-vs-deep-learning/.

22. Davi C. Why Perceptron Neurons Need Bias Input? [Internet]. Towards AI; 2020 [updated 2020 Mar 07; cited 2024 Mar 27]. Available from: https://pub.towardsai.net/why-perceptron-neurons-need-bias-Input-2144633bcad4.

23. Naven. What is multilayer perceptron? [Internet]. 2022 [updated 2022 Apr 17; cited 2024 Mar 27]. Available from: https://www.nomidl.com/natural-language-processing/what-is-multilayer-perceptron/.

24. DhanushKumar. MAX POOLING [Internet]. medium; 2023 [updated 2023 Nov 29; cited 2024 Api 03]. Available from: https://medium.com/@danushidk507/max-pooling-ef545993b6e4.

25. Swapna KE. Convolution Neural Network (CNN) [Internet]. Developers Breach; [n.p] [cited 2024 Mar 27]. Available from: https://developersbreach.com/convolution-neural-network-deep-learning/.

26. Sewoong Oh. CSE 446: Machine Learing [Internet]. University of Washington; 2022 [cited 2024 Mar 27]. Available from: https://courses.cs.washington.edu/courses/cse446/22wi/sections/08/convolutional_networks.html.

27. Ronneberger O, Fischer P, Brox T, editors. U-net: Convolutional networks for biomedical image segmentation. Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18; 2015: Springer.

28. Zhou Z, Rahman Siddiquee MM, Tajbakhsh N, Liang J, editors. Unet++: A nested u-net architecture for medical image segmentation. Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4; 2018: Springer.

29. Digital Image Processing [Internet]. tutorialspoint; [n.p.] [cited 2024 Mar 27]. Available from: https://www.tutorialspoint.com/dip/Histogram_equalization.htm.

30. Histograms - 2: Histogram Equalization [Internet]. [n.p] [cited 2024 Mar 27]. Available from: https://docs.opencv.org/4.x/d5/daf/tutorial_py_Histogram_equalization.html.

31. Samuel Samsudin Ng. Introduction to Histogram Equalization for Digital Image Enhancement [Internet]. Medium: Level Up Coding; 2021 [updated 2021 Jan 11; cited 2024 Mar 27]. Available from: https://levelup.gitconnected.com/introduction-to-Histogram-equalization-for-digital-image-enhancement-420696db9e43.

32. Kashif E. What is Histogram equalization in Python? : educative; [n.p] [cited 2024 Mar 27]. Available from: https://www.educative.io/answers/what-is-Histogram-equalization-in-python.

33. Wen H, Qi W, Shuang L, editors. Medical X-ray image enhancement based on wavelet domain homomorphic filtering and CLAHE. 2016 International Conference on Robots & Intelligent System (ICRIS); 2016: IEEE.

34. Zuiderveld K. Contrast limited adaptive Histogram equalization. Graphics gems IV1994. p. 474-85.

35. Cao G, Huang L, Tian H, Huang X, Wang Y, Zhi R. Contrast enhancement of brightness-distorted images by improved adaptive Gamma correction. Computers & Electrical Engineering. 2018;66:569-82.

36. Huang Z, Zhang T, Li Q, Fang H. Adaptive Gamma correction based on cumulative Histogram for enhancing near-infrared images. Infrared Physics & Technology. 2016;79:205-15.

37. Shreyamsha Kumar B. Image denoising based on non-local means filter and its method noise thresholding. Signal, image and video processing. 2013;7:1211-27.

38. Buades A, Coll B, Morel J-M, editors. A non-local algorithm for image denoising. 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05); 2005: Ieee.

39. Chandrashekar L, Sreedevi A, editors. Assessment of non-linear filters for MRI images. 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT); 2017: IEEE.

40. Non-local means denoising for preserving textures [Internet]. scikit-image; [n.p.] [cited 2024 Mar 27]. Available from: https://scikit-image.org/docs/stable/auto_examples/filters/plot_nonlocal_means.html.

41. Min A, Kyu ZM, editors. MRI images enhancement and tumor segmentation for brain. 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT); 2017: IEEE.

42. Baraniuk. RG. Wiener Filtering [Internet]. 1997 [updated 1995 Nov 24; cited 2024 Mar 27]. Available from: https://www.owlnet.rice.edu/~elec539/Projects99/BACH/proj2/wiener.html.

43. Pranjal Datta. All about Structural Similarity Index (SSIM): Theory + Code in PyTorch [Internet]. Medium: SRM MIC; 2020 [updated 2020 Sep 03; cited 2024 Mar 27]. Available from: https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e.

44. PSNR [Internet]. mathworks; [n.p.] [cited 2024 Mar 27]. Available from: https://www.mathworks.com/help/vision/ref/psnr.html.

45. Peak Signal-to-Noise Ratio as an Image Quality Metric [Internet]. EMERSON; 2023 [updated 2023 Dec 07. Available from: https://www.ni.com/en/shop/data-acquisition-and-control/add-ons-for-data-acquisition-and-control/what-is-vision-development-module/peak-signal-to-noise-ratio-as-an-image-quality-metric.html.

46. Haque IRI, Neubert J. Deep learning approaches to biomedical image segmentation. Informatics in Medicine Unlocked. 2020;18:100297.

47. Pagon Gatchalee. Confusion Matrix เครื่องมือสำคัญในการประเมินผลลัพธ์ของการทำนาย ในMachine learning [Internet]. Meduim; 2019 [updated 2019 Oct 03; cited 2024 Mar 27]. Available from: https://medium.com/@pagongatchalee/confusion-matrix-เครื่องมือสำคัญในการประเมินผลลัพธ์ของการทำนาย-ในmachine-learning-fba6e3f9508c.

48. Aboudi F, Drissi C, Kraiem T, editors. Efficient U-Net CNN with data augmentation for MRI ischemic stroke brain segmentation. 2022 8th International Conference on Control, Decision and Information Technologies (CoDIT); 2022: IEEE.

49. Pasha SS, Babu PS, Vakil Z, editors. Enhancement of MRI Brain Images with Histogram Equalization Techniques. 2019 International Conference on Emerging Trends in Science and Engineering (ICESE); 2019: IEEE.

50. Sahnoun M, Kallel F, Dammak M, Mhiri C, Mahfoudh KB, Hamida AB, editors. A comparative study of MRI contrast enhancement techniques based on Traditional Gamma Correction and Adaptive Gamma Correction: Case of multiple sclerosis

pathology. 2018 4th international conference on advanced technologies for signal and image processing (ATSIP); 2018: IEEE.

51. Gu Z, Cheng J, Fu H, Zhou K, Hao H, Zhao Y, et al. Ce-net: Context encoder network for 2d medical image segmentation. IEEE transactions on medical imaging. 2019;38(10):2281-92.

52. Ghosh S, Santosh K, editors. Tumor segmentation in brain MRI: U-Nets versus feature pyramid network. 2021 IEEE 34th International Symposium on Computer-Based Medical Systems (CBMS); 2021: IEEE.

53. Al Nasim MA, Al Munem A, Islam M, Palash MAH, Haque MMA, Shah FM, editors. Brain tumor segmentation using enhanced u-net model with empirical analysis. 2022 25th International Conference on Computer and Information Technology (ICCIT); 2022: IEEE.

54. Manasa K, Krishnaveni V, editors. Brain Tumor Segmentation Using Zernike Moments in U-Net. 2022 International Conference on Intelligent Innovations in Engineering and Technology (ICIIET); 2022: IEEE.

55. Yan C, Ding J, Zhang H, Tong K, Hua B, Shi S. SEResU-Net for multimodal brain tumor segmentation. IEEE Access. 2022;10:117033-44.

ประวัติผู้วิจัย

ชื่อ-นามสกุล นายกศตพรรษ นาคแนม

ประวัติการศึกษา วิทยาศาสตรบัญฑิต สาขาฟิสิกส์ มหาวิทยาลัยพะเยา

ผลงานตีพิมพ์ A Comparative Study of Pre-Processing Methods to Improve Glioma Segmentation Performance in Brain MRI using Deep Learning