การพัฒนาอัลกอริธึมนิวโรอีโวลูชั่นสำหรับปัญหาเกม

โดย
นายไพรวัน พัชรบำรุง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ แผน ก แบบ ก 2 ระดับปริญญามหาบัณฑิต
ภาควิชาวิศวกรรมไฟฟ้า
มหาวิทยาลัยศิลปากร
ปีการศึกษา 2565

มหาวิทยาลัยศิลปากร

การพัฒนาอัลกอริธึมนิวโรอีโวลูชั่นสำหรับปัญหาเกม

โดย
นายไพรวัน พัชรบำรุง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ แผน ก แบบ ก 2 ระดับปริญญามหาบัณฑิต
ภาควิชาวิศวกรรมไฟฟ้า
มหาวิทยาลัยศิลปากร
ปีการศึกษา 2565

DEVELOPMENT OF NEUROEVOLUTION ALGORITHMS FOR GAME PROBLEMS

By

MR. Praiwan PATCHARABUMRUNG

A Thesis Submitted in Partial Fulfillment of the Requirements

for Master of Engineering (ELECTRICAL AND COMPUTER ENGINEERING)

Department of ELECTRICAL ENGINEERING

Silpakorn University

Academic Year 2022

หัวข้อ	การพัฒนาอัลกอริธึมนิวโรอีโวลูชั่นสำหรับปัญหาเกม
โดย	นายไพรวัน พัชรบำรุง
สาขาวิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์ แผน ก แบบ ก 2 ระดับปริญญามหาบัณฑิต
อาจารย์ที่ปรึกษาหลัก	ผู้ช่วยศาสตราจารย์ ดร. ยุทธนา เจวจินดา

คณะวิศวกรรมศาสตร์และเทคโนโลยีอุตสาหกรรม มหาวิทยาลัยศิลปากร ได้รับพิจารณาอนุมัติให้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์และ
(ผู้ช่วยศาสตราจารย์ ดร. อรุณศรี ลีจีรจำเนียร) เทคโนโลยีอุตสาหกรรม

พิจารณาเห็นชอบโดย

.....ประธานกรรมการ
(รองศาสตราจารย์ ดร. ชูเกียรติ สอดศรี)

.....อาจารย์ที่ปรึกษาหลัก
(ผู้ช่วยศาสตราจารย์ ดร. ยุทธนา เจวจินดา)

.....ผู้ทรงคุณวุฒิภายนอก
(รองศาสตราจารย์ ดร. ยรรยงค์ พันธ์สวัสดิ์)

61407206 : วิศวกรรมไฟฟ้าและคอมพิวเตอร์ แผน ก แบบ ก 2 ระดับปริญญามหาบัณฑิต

นาย ไพรวัน พัชรบำรุง: การพัฒนาอัลกอริธึมนิวโรอีโวลูชั่นสำหรับปัญหาเกม อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก : ผู้ช่วยศาสตราจารย์ ดร. ยุทธนา เจวจินดา

งานวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนาอัลกอริทึมนิวโรอีโวลูชั่น และประยุกต์ใช้งานกลุ่มของโครงข่ายประสาทเทียมแบบคอนโวลูชันที่เป็นผลลัพธ์เพื่อสร้างเอเย่นต์ที่ฉลาดเพื่อเล่นเกมส์คอมพิวเตอร์ ส่วนแรกของวิทยานิพนธ์นำเสนอการพัฒนาระเบียบวิธีเชิงพันธุกรรมแบบหลายวัตถุประสงค์เพื่อวิวัฒนาการโครงข่ายประสาทเทียมแบบคอนโวลูชัน ระเบียบวิธีเชิงพันธุกรรมแบบหลายวัตถุประสงค์พัฒนาต่อจาก แนวคิดการจัดเรียงแบบไม่ข่มของเอนเอสจีเอรุ่นที่สอง อย่างไรก็ตามเรานำเสนอแนวทางการแบ่งประชากรออกเป็นสองกลุ่มเพื่อเพิ่มความหลากหลายในกระบวนการค้นหา เราได้ศึกษาผลกระทบของโอเปอเรเตอร์เชิงพันธุกรรม ได้แก่การผสมพันธ์ และการกลายพันธ์ รวมทั้งนำเสนอการเข้ารหัสโครโมโซมของโครงข่ายประสาทเทียม หลังจากได้โครงข่ายประสาทเทียมแบบคอนโวลูชันที่เหมาะสมที่สุดจำนวนหนึ่ง ได้นำโครงข่ายประสาทเทียมเหล่านี้ไปในงานการเรียนแบบคิวเชิงลึก เพื่อทำการเรียนรู้แบบออนไลน์ในขณะเล่นเกมส์คอมพิวเตอร์แข่งรถ ผลการทดลองพบว่านิวโรอีโวลูชั่นที่ใช้งานระเบียบวิธีเชิงพันธุกรรมแบบหลายวัตถุประสงค์สามารถดัดแปลงโครงข่ายประสาทเทียมแบบคอนโวลูชันที่ดี แต่อย่างไรก็ตาม ผลลัพธ์การใช้งานกับการเล่นเกมส์แข่งรถยังคงต้องปรับปรุงให้ดีขึ้น

61407206 : Major (ELECTRICAL AND COMPUTER ENGINEERING)

MR. Praiwan PATCHARABUMRUNG : Development of Neuroevolution Algorithms for game problems Thesis advisor : Assistant Professor YUTANA JEWAJINDA, Ph.D.

This research aims to develop a new neuroevolution algorithm and apply the results, a group of convolutional neural networks to create intelligent agents to play computer games. The first part of the thesis presents a development of a multi-objective genetic algorithm to evolve convolutional neural networks. The multi-objective genetic algorithm is based on the non-dominated sorting of NSGA-II. However, we propose using a dual population approach. We studied the effects of genetic operators, namely crossover and mutation, on the quality of the evolving process. With our dual population approach, the diversity of the search process is increased. After deriving a group of optimized convolutional neural networks, those networks are used in the deep Q-learning to online learning to play a computer racing game. The experimental results show that the neuroevolution using multi-objective genetic algorithms delivered a competitive group of convolutional neural networks. However, the results of creating intelligent agents to play still need to be improved.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดีจากความกรุณาของผู้ช่วยศาสตราจารย์ ดร. ยุทธนา เจวจินดา ซึ่งเป็นอาจารย์ที่ปรึกษาวิทยานิพนธ์ที่คอยช่วยเหลือ ให้แนวคิด และคำปรึกษา ตลอดการทำวิจัยที่ผ่านมา รวมทั้งขอขอบคุณรองศาสตราจารย์ ดร. ชูเกียรติ สอดศรี ประธานกรรมการสอบวิทยานิพนธ์ และรองศาสตราจารย์ ดร. ยรรยงค์ พันธ์สวัสดิ์ ผู้ทรงคุณวุฒิภายนอก ที่กรุณาให้ข้อเสนอแนะแก่การทำวิทยานิพนธ์ฉบับนี้ให้มีความถูกต้องสมบูรณ์มากขึ้น

ผู้วิจัยขอกราบขอบพระคุณในความกรุณาของอาจารย์ทุกท่านเป็นอย่างสูง ผู้วิจัยขอกราบขอบพระคุณคณาจารย์ประจำภาควิชาวิศวกรรมไฟฟ้าทุกท่าน ที่ให้ความรู้, การสนับสนุนไม่ว่าจะเป็นด้านทุนสำหรับการศึกษาเล่าเรียน อุปกรณ์ต่างๆสำหรับงานวิจัย และประสบการณ์อันมีค่าแก่ผู้วิจัย

ท้ายที่สุดนี้ขอขอบคุณนายสถาวร พัชรบำรุง และนางรุ่งนภา พัชรบำรุง ผู้เป็นบิดามารดาของผู้วิจัย และตลอดจนครอบครัว และมิตรสหายที่คอยเป็นกำลังใจ ให้ความช่วยเหลือในด้านต่างๆ จนกระทั่งวิทยานิพนธ์ฉบับนี้เสร็จสมบูรณ์

นาย ไพรวัน พัชรบำรุง

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

แนวคิดการทำให้เครื่องคำนวณทำงานเลียนแบบสมองมนุษย์ถือกำเนิดมายาวนานตั้งแต่ยุคเริ่มต้นของปัญญาประดิษฐ์ ซึ่งถือว่าเป็นจุดเริ่มต้นของศาสตร์ด้านโครงข่ายประสาทเทียม (Artificial neural network) การพัฒนาในยุคแรกเริ่มจากเพอร์เซ็ปตรอน ที่เป็นแบบจำลองการคำนวณที่เลียนแบบเซลล์ประสาทสมองเซลล์เดียว จากนั้นนำแบบจำลองของเซลประสาทมาเชื่อมโยงกันในลักษณะเป็นชั้นๆ [1] ศาสตร์ด้านโครงข่ายประสาทเทียมได้พัฒนามาโดยตลอด จนกระทั่งพัฒนามาสู่การเรียนรู้เชิงลึก (Deep learning) ในปัจจุบัน [2] เนื่องจากประสิทธิภาพของโครงข่ายประสาทเทียมขึ้นกับโครงสร้าง การเชื่อมต่อ และค่าพารามิเตอร์ต่างๆ ภายใน ทำให้เกิดแนวคิดหนึ่งในการนำอัลกอริทึมเชิงวิวัฒนาการมาประยุกต์ใช้งานร่วมกับโครงข่ายประสาทเทียมเรียกว่า “นิวโรอีโวลูชั่น” (Neuroevolution)

นิวโรอีโวลูชั่น เป็นวิธีการหนึ่งในการนำอัลกอริทึมเชิงวิวัฒนาการมาใช้ในการปรับโครงสร้างและค่าพารามิเตอร์ของโครงข่ายประสาทเทียมเพื่อให้เรียนรู้งานที่เจาะจง [3] อัลกอริทึมเชิงวิวัฒนาการเป็นวิธีการคำนวณที่ใช้แนวคิดเชิงชีววิทยา และมีการประยุกต์ใช้งานที่หลากหลาย โดยมักใช้งานในการค้นหาคำตอบที่มีค่าเหมาะที่สุด (Optimization) [4] เราสามารถใช้งานนิวโรอีโวลูชั่นทำงานร่วมกับการเคลื่อนลงตามความชัน (Gradient Descent Algorithm) และการแพร่ย้อนกลับ (Backpropagation) ในการปรับค่าพารามิเตอร์ของโครงข่ายประสาทเทียมได้ โดยทำการค้นหาสถาปัตยกรรม และการเชื่อมต่อของโครงข่ายประสาทเทียมด้วยอัลกอริทึมเชิงวิวัฒนาการ และใช้การปรับค่าน้ำหนักด้วยวิธีแพร่ย้อนกลับ

นิวโรอีโวลูชั่น และอัลกอริทึมเชิงวิวัฒนาการเอง ต่างก็ถูกนำไปใช้ทำงานควบคุมการทำงานของเกมส์ เพื่อให้คอมพิวเตอร์สามารถเล่นเกมส์แทนที่มนุษย์ได้ โดยมักเริ่มจากเกมส์กระดานทั่วไป เช่น หมากฮอส หมากรุก หมากล้อม โอเทลโล่ ไปจนถึงวีดีโอเกมส์ ที่มักจะมีความซับซ้อนสูงกว่าเกมส์กระดานทั่วไป [5] ปัญหาในการประยุกต์ใช้งานนิวโรอีโวลูชั่นกับเกมส์ โดยที่โครงข่ายประสาทเทียมแบบคอนโวลูชันเหมาะสมสำหรับเกมส์ที่ผู้เล่นต้องใช้การเห็นในการเล่นเกมส์ เนื่องจากเน้นใช้งานการรู้จำภาพ จากนั้นพัฒนา หรือเลือกใช้อัลกอริทึมเชิงวิวัฒนาการในการค้นหา หรือปรับปรุงโครงข่ายประสาทเทียม เพื่อนำไปใช้งานในเกมส์ การนำนิวโรอีโวลูชั่นใช้งานเพื่อเล่นเกมส์เหล่านี้ มักเป็นเกมส์ที่แข่งกันระหว่างสองผู้เล่น หรือ ตัวเองแข่งกับหลายผู้เล่น การวิวัฒนาการโครงข่ายประสาทเทียมในอดีตมักมุ่งเน้นไปที่อัลกอริทึมเชิงวิวัฒนาการเป้าหมายเดียว ในการวิจัยนี้จึงมุ่งเน้นการพัฒนานิวโรอีโวลูชั่นที่มีพื้นฐานมาจากอัลกอริทึมเชิงวิวัฒนาการหลายเป้าหมายเพื่อทำให้ได้โครงข่ายประสาทเทียม

ที่มีความสมดุลทั้งในส่วนของขนาด และความสามารถในการจำแนกภาพ ทำให้เมื่อนำไปใช้งานจริงจะทำงานได้เร็วขึ้นเมื่อมีขนาดไม่ใหญ่เกินไป

1.2 วัตถุประสงค์ของงานวิจัย

1. พัฒนาอัลกอริทึมนิวโรอีโวลูชั่นแบบใหม่
2. นำอัลกอริทึมที่ได้พัฒนาขึ้นมาประยุกต์ใช้งานกับปัญหาเกมส์
3. ทำการเปรียบเทียบประสิทธิภาพของอัลกริธึมที่ได้กับวิธีการอื่นๆ

1.3 สมมติฐานของงานวิจัย

นิวโรอีโวลูชั่น เป็นวิธีที่ผสมหลักการของโครงข่ายประสาทเทียมร่วมกับอัลกอริทึมเชิงวิวัฒนาการซึ่งใช้หลักการจากทฤษฎีวิวัฒนาการ [6] [7] โดยที่การทำงานเริ่มจากการสุ่มคำตอบขึ้นมากลุ่มหนึ่ง ซึ่งสำหรับนิวโรอีโวลูชั่น แต่ละคำตอบคือโครงข่ายประสาทเทียม โดยทำการแทนคำตอบของปัญหาในรูปรหัส หรือเรียกว่าโครโมโซม แทนคำตอบโดยตรง โดยที่แต่ละคำตอบจะถูกตรวจสอบด้วยฟิตเนสฟังก์ชั่น หรือฟังก์ชันต้นทุน ซึ่งอาจจะเป็นสมการ หรือ การจำลองการทำงานให้ได้ค่าออกมาเป็นตัวเลข สำหรับโครงข่ายประสาทเทียมคือการทดสอบประสิทธิภาพการใช้งานกับปัญหาที่เจาะจง กลุ่มของคำตอบรวมกันเรียกว่าประชากร โดยที่สมาชิกของประชากรเหล่านี้ จะถูกเลือกเพื่อนำไปผสมพันธุ์ และ กลายพันธุ์ หรือ กลายพันธุ์อย่างเดียว เพื่อสร้างเป็นกลุ่มของคำตอบรุ่นใหม่หรือประชากรรุ่นใหม่ แล้วก็เลือกคำตอบที่ดีกว่า ไปเรื่อยๆ จนกว่าจะได้คำตอบที่ถูกต้องพอที่จะยอมรับได้ ข้อดีประการหนึ่งของนิวโรอีโวลูชั่นคือสามารถนำไปใช้กับปัญหาที่แตกต่างกันได้โดยง่ายและไม่จำเป็นต้องให้ ฟังก์ชันต้นทุนอยู่ในรูปแบบสมการคณิตศาสตร์ที่ตายตัวเสมอไป [3]

เกมส์ประเภทแข่งขันกัน เป็นเป้าทดสอบมาตรฐานสำหรับปัญญาประดิษฐ์มาอย่างยาวนาน [8, 9] และโครงข่ายประสาทเทียมที่ถูกสร้างโดยนิวโรอีโวลูชั่นก็เป็นเช่นเดียวกัน สมมุติฐานในการวิจัยนี้จึงเป็นการพัฒนาอัลกอริทึมนิวโรอีโวลูชั่นสำหรับเอเย่นต์เดียว โดยเน้นให้เอเย่นต์สามารถพัฒนาตัวเองได้ โดยมีการพึงพาความรู้ในการเล่นเกมส์นั้นๆ โดยที่ประเด็นสำคัญคือการพัฒนาอัลกอริทึมเชิงวิวัฒนาการแบบหลายวัตถุประสงค์ [4, 10] เพื่อดัดแปลงโครงข่ายประสาทเทียมเพื่อสร้างกลุ่มของโครงข่ายประสาทเทียมที่ตอบหลายวัตถุประสงค์ ทำให้ได้โครงข่ายประสาทเทียมที่มีความสมดุล ทั้งในส่วนของจำนวนพารามิเตอร์ และความสามารถในการทำงานให้สำเร็จ โดยที่จำนวนพารามิเตอร์ต้องไม่มากเกินไป หรือใช้เวลาในการคำนวนภายในขอบเขต จากนั้นนำโครงข่ายประสาทเทียมที่ได้ไปใช้เป็นส่วนหนึ่งของการเล่นเกมส์ร่วมกับวิธีการเรียนรู้แบบเสริมแรง (Reinforcement learning) เพื่อสร้างเอเยนต์ที่สามารถเล่มเกมส์คอมพิวเตอร์ได้ด้วยตัวเอง เพื่อตอบสนองต่อการเปลี่ยนแปลง และสถานการณ์สิ่งแวดล้อมที่เปลี่ยนไป โดยที่เน้นเกมส์คอมพิวเตอร์ที่มีความซับซ้อนขึ้น

และต้องการการมองเห็นภาพในขณะที่เล่นซึ่งเป็นลักษณะของคอมพิวเตอร์เกมส์สมัยใหม่ ผลลัพธ์ทำให้ได้เอเย่นต์ที่สามารถเล่นเกมส์ที่ต้องใช้การมองเห็น และตอบสนองกับภาพ หรือสถานการณ์ที่เปลี่ยนไป โดยใช้โครงข่ายประสาทเทียมที่ได้จากนิวโรอีโวลูชั่น

1.4 ขอบเขตของการวิจัย

1. ศึกษาพื้นฐานทฤษฎีนิวโรอีโวลูชั่น
2. ศึกษาแนวทางการประยุกต์ใช้งานนิวโรอีโวลูชั่นกับปัญหาเกมส์
3. พัฒนาและปรับปรุงอัลกอริทึมนิวโรอีโวลูชั่นแบบใหม่
4. นำอัลกอริทึมที่ได้พัฒนาขึ้นมาประยุกต์ใช้งานกับปัญหาเกมส์ที่เหมาะสม
5. วัดประสิทธิภาพการทำงานของอัลกอริทึมที่ได้พัฒนาขึ้นในการแก้ปัญหาเกมส์เมื่อเปรียบเทียบกับวิธีอื่นๆ

1.5 ความจำกัดของการวิจัย

ในงานวิจัยนี้จะใช้นิวโรอีโวลูชั่นสร้างโครงข่ายประสาทเทียม สำหรับใช้เล่นเกมส์ จากนั้นนำโครงข่ายประสาทเทียมเหล่านั้นไปเล่นเกมส์ ในการทำการทดลองจำเป็นต้องใช้เครื่องคอมพิวเตอร์ประสิทธิภาพสูงซึ่งผู้วิจัยไม่สามารถจัดหาได้ ทำให้ต้องใช้เครื่องคอมพิวเตอร์ที่มีความสามารถจำกัด ทำให้การทดลองใช้เวลานานมากเป็นเวลาหลายวัน ในแต่ละการทดลอง

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้รับความรู้เกี่ยวกับการพัฒนางานนิวโรอีโวลูชั่นกับปัญหาเกมส์ที่ ซึ่งสามารถนำไปต่อยอดเพื่อประยุกต์ใช้ในระบบควบคุมที่ต้องใช้เอเย่นต์หลายตัว
2. เผยแพร่ในวารสารวิชาการระดับนานาชาติ หรือการประชุมวิชาการระดับนานาชาติ

บทที่ 2
ทฤษฎี และวรรณกรรมที่เกี่ยวข้อง

ในบทนี้นำเสนอทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้อง ซึ่งประกอบด้วย พื้นฐานโครงข่ายประสาทเทียม โครงข่ายประสาทเทียมแบบคอนโวลูชัน วิธีเชิงพันธุกรรม วิธีเชิงคัดสรรผลเฉลยที่ไม่ถูกครอบงำเพื่อใช้กับการหาค่าเหมาะแบบหลายวัตถุประสงค์ วรรณกรรมที่เกี่ยวข้องกับการประยุกต์ใช้งานกับปัญหาเกมส์

2.1 พื้นฐานโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียมเป็นแนวคิดหนึ่งในปัญญาประดิษฐ์ที่เลียนแบบการทำงานของโครงข่ายประสาทของสิ่งมีชีวิต โดยเฉพาะเมื่อโครงข่ายประสาทรับกระแสประสาทแล้วปรับตัวเองตามกระแสประสาทที่ได้รับ จนกระทั่งปรับตัว และเรียนรู้จนกลายเป็นโครงข่ายประสาทที่เหมาะสม เซลล์ประสาทแท้และการโยงใยถูกเลียนแบบด้วยเมทริกซ์และกราฟ กระบวนการปรับตัวตามกระแสประสาทถูกเลียนแบบด้วยสูตรคณิตศาสตร์ [1]

โครงข่ายประสาทเทียมแบบต่างๆ เป็นแบบจำลองการคำนวณ ที่ถูกออกแบบตามหลักการทำงานของสมองโดยเฉพาะในข้อสำคัญสองประการ คือ

1. โครงข่ายประสาทเทียมเรียนรู้จากสิ่งแวดล้อม ด้วยกระบวนการการเรียนรู้
2. สิ่งที่โครงข่ายประสาทเทียมเรียนรู้ได้ ถูกเก็บอยู่ในรูปของน้ำหนักการเชื่อมต่อ

ข้อสำคัญทั้งสองข้อนี้ ทำให้โครงข่ายประสาทเทียมสามารถถูกปรับใช้กับงานและปัญหาต่างๆ ที่แตกต่างกัน โดยไม่จำเป็นต้องใช้ความรู้เฉพาะทางจำนวนมาก เพราะการแทนปัญหาและงานเหล่านั้นเป็นสิ่งแวดล้อม ต้องการเพียงความรู้ทางคณิตศาสตร์ในระดับหนึ่ง โดยไม่จำเป็นต้องรู้เบื้องลึกของงานและปัญหาเหล่านั้น แต่ใช้ข้อมูลจำนวนมากในการสอนให้เกิดการเรียนรู้ พื้นฐานเบื้องต้นของโครงข่ายประสาทเทียมประกอบไปด้วย

2.1.1 เพอร์เซ็ปตรอน (Perceptron)

หน่วยที่เล็กที่สุดของโครงข่ายประสาทเทียม คือเซลล์ประสาทซึ่งสามารถสร้างแบบจำลองด้วยเพอร์เซ็ปตรอน และนำมาเชื่อมต่อกันเพื่อให้เกิดการเรียนรู้ที่ซับซ้อนขึ้นได้ เพอร์เซ็ปตรอนมีความสำคัญในวงการของโครงข่ายประสาทเทียมเป็นพิเศษ เพราะโดยหลักการพื้นฐานแล้ว เพอร์เซ็ปตรอนในยุคแรกของปัญญาประดิษฐ์ จนถึงเพอร์เซ็ปตรอนในทุกวันนี้ ก็ยังคงเป็นโครงสร้างเดิมๆ ไม่ได้มีการเปลี่ยนแปลงมากนัก โครงสร้างดังกล่าวประกอบไปด้วย อินพุต น้ำหนักการเชื่อมต่อ โหนดบวกรวม ลิมิเตอร์ ฟังก์ชันกระตุ้น และเอาท์พุต โดยอินพุตแต่ละตัว จะถูกนำไปคูณกับน้ำหนักการเชื่อมต่อ แล้วก็ส่งไปรวมกันในโหนดบวกรวม จากนั้น ผลลัพธ์ที่ถูกรวมแล้ว ก็จะถูกนำไปผ่านแอคทิเวชั่นฟังก์ชั่น

หรือฟังก์ชันกระตุ้น ซึ่งหลากหลายรูปแบบในปัจจุบัน เพื่อสร้างผลลัพธ์ที่ได้ออกมาก็จะกลายเป็นเอาท์พุตของเซลประสาท

2.1.2 การแพร่ย้อนกลับ (Backpropagation)

การถือกำเนิดของการแพร่ย้อนกลับถือเป็นสิ่งสำคัญในวงการโครงข่ายประสาทเทียม ก่อนที่การแพร่ย้อนกลับจะถือกำเนิดเต็มตัว ไม่มีวิธีการใดๆ ที่สามารถสอนโครงข่ายประสาทเทียมได้อย่างมีประสิทธิภาพ [1] ซึ่งเป็นปัญหาใหญ่มากในยุคแรกๆ ของโครงข่ายประสาทเทียมก่อนช่วงปี ค.ศ. 1980 [11]

ขั้นตอนของการแพร่ย้อนกลับถูกแบ่งเป็นสองส่วนหลักๆ ส่วนแรก ปล่อยข้อมูลไปทางด้านหน้า (ในความหมายนี้ จากอินพุตไปเอาท์พุต) เพื่อเอาความแตกต่างระหว่างเอาท์พุตที่ควรจะเป็น กับเอาท์พุตที่ได้จากโครงข่ายประสาทเทียม ที่เรียกว่าสัญญาณความผิดพลาด ส่วนที่สอง คือปล่อยสัญญาณความผิดพลาด ที่ได้จากส่วนแรก กลับมาด้านหลัง (ในความหมายนี้ จากเอาท์พุตไปอินพุต) เพื่อใช้ส่วนต่างนั้นแก้ไขน้ำหนักการเชื่อมต่อต่างๆ ในโครงข่ายประสาทเทียม ขั้นตอนเหล่านี้จะถูกทำไปเรื่อยๆ จนกว่าสัญญาณความผิดพลาด จะลดลงจนเหลือน้อยพอที่จะยอมรับได้

ในส่วนที่สองของการแพร่ย้อนกลับที่ปล่อยสัญญาณความผิดพลาดกลับมาด้านหลัง สัญญาณความผิดพลาดจะถูกหาอนุพันธ์ เทียบกับค่าน้ำหนัก ตามลำดับการส่งสัญญาณกลับมาด้านหลัง (ในกรณีที่โครงข่ายประสาทเทียมไม่ได้ไปข้างหน้าอย่างเดียว ก็ไล่ทีละเพอร์เซ็ปตรอน ตั้งแต่เอาท์พุตมาจรดอินพุต) ดังนั้นลิมิเตอร์ของเพอร์เซ็ปตรอนตัวนั้นจำเป็นต้องหาอนุพันธ์ได้

ข้อจำกัดที่ชัดเจนที่สุดของการแพร่ย้อนกลับ คือความที่ตัวเอง โดยรากฐานแล้ว เป็นเทคนิคการเคลื่อนลงตามความชันตามเส้นทางกราฟการคำนวณ (Computational graph) โดยมองจากมิติของน้ำหนักการเชื่อมต่อ ดังนั้นปัญหาที่พบในเทคนิคการเคลื่อนลงตามความชัน เช่นการติดขัดในโลคอลมินิม่า โดยเฉพาะตอนที่เวกเตอร์ค้นหาใหญ่มากๆ ก็จะพบในการแพร่ย้อนกลับเช่นกัน นั่นทำให้เกิดงานวิจัยเพื่อแก้ปัญหานี้ขึ้น อีกทั้งข้อจำกัดของวิธีการแพร่ย้อนกลับในระยะแรก จะไม่สามารถหาค่าน้ำหนักได้ดีถ้าจำนวนชั้นของโครงข่ายประสาทเทียมมีความลึกขึ้น โดยเฉพาะมากกว่า 3 ชั้นไปหรือมากกว่านั้นมากๆ [2]

2.1.3 โครงข่ายประสาทเทียมเชิงลึก (Deep Neural Network)

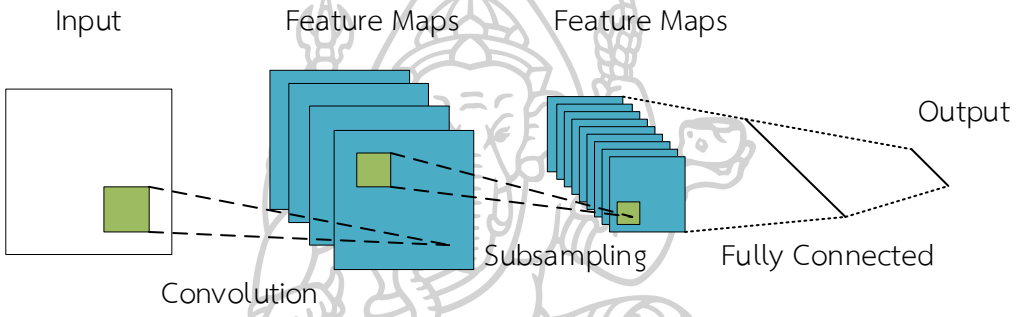
โดยทฤษฎี การแพร่ย้อนกลับสามารถใช้ฝึกสอนโครงข่ายประสาทเทียมขนาดใหญ่ที่มีหลายชั้น ซึ่งถูกเรียกรวมๆ ว่า โครงข่ายประสาทเทียมเชิงลึก แต่ในทางปฏิบัติ การแพร่ย้อนกลับเสียประสิทธิภาพเมื่อนำไปใช้สอนโครงข่ายประสาทเทียมที่มีขนาดใหญ่จำนวนชั้นมาก

โครงข่ายประสาทเทียมเชิงลึกโดยแท้จริง ถือกำเนิดขึ้นจาก การใช้วิธีการเรียนรู้แบบไม่มีผู้สอนสร้างตัวโครงข่ายประสาทเทียมขึ้นมาก่อน จากนั้นก็ใช้วิธีการเรียนแบบมีผู้สอน ทำการสอนตัวโครงข่ายประสาทเทียมที่ได้มาอีกครั้ง และสามารถแก้ปัญหาละเอียดอ่อนสูงต่างๆ ได้โดยตลอด[4]

ตั้งแต่นั้นเป็นต้นมา ระเบียบวิธีต่างๆ ทั้งสำหรับสร้างตัวโครงข่ายประสาทเทียม และสำหรับฝึกสอนตัวโครงข่ายประสาทเทียมที่ถูกสร้างขึ้น ก็ถูกพัฒนาต่อเนื่องมาเป็นอันมาก ภายใต้ชื่อ การเรียนรู้เชิงลึก [2]

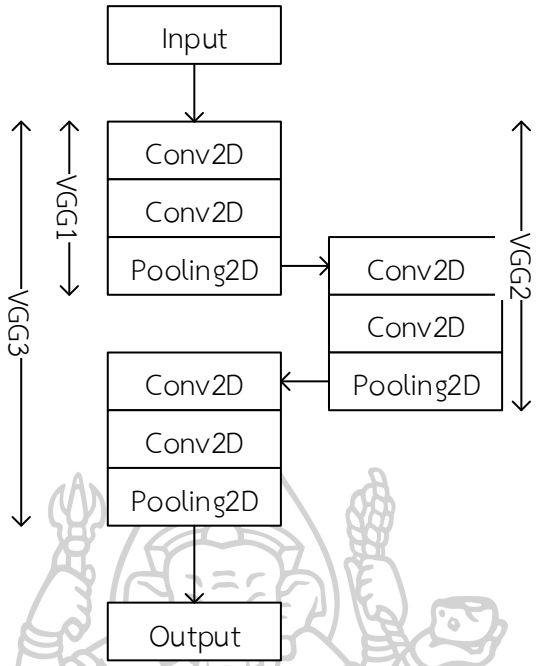
ในขณะที่โครงข่ายประสาทเทียม มีที่มาจากหลักการทำงานของสมอง โดยเฉพาะส่วนของเพอร์เซ็ปตรอน การเรียนรู้เชิงลึก และโครงข่ายประสาทเทียมเชิงลึก มีที่มาจากหลักการทำงานของสมองของมนุษย์ โดยเฉพาะส่วนของการแบ่งการทำงานเป็นลำดับชั้น แต่ละชั้นของโครงข่ายประสาทเทียมเชิงลึก โดยที่แต่ละชั้นจะเรียนรู้ลักษณะเด่นที่แตกต่างกัน ในแต่ละส่วนเป็นขั้นๆ โดยปัจจุบัน มักใช้การเรียนรู้จากข้อมูลตัวอย่างจำนวนมาก [4]

2.2 โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network)



ภาพที่ 1: รูป CNN

โครงข่ายประสาทเทียมแบบคอนโวลูชันเป็นโครงข่ายประสาทเทียมที่มีคุณสมบัติในการสกัดลักษณะเด่นในชุดข้อมูลประเภทรูปภาพ จุดเด่นของโครงข่ายประสาทเทียมแบบคอนโวลูชันที่ต่างจากโครงข่ายประสาทเทียมปกติ คือการมีชั้นพิเศษสำหรับทำหน้าที่สกัดฟีเจอร์ (ลักษณะเด่น) ชั้นเหล่านั้นจะมีลักษณะคล้ายกันอย่างหนึ่ง คือเป็นชั้นที่จะแบ่งรูปภาพเป็นส่วนๆ แล้วนำไปเข้ากระบวนการคอนโวลูชันกับฟิลเตอร์ ออกมาเป็นแผนผังฟีเจอร์ [12]



ภาพที่ 2: โครงสร้างเบื้องต้นของ VGG3 ที่เป็น CNN ชนิดหนึ่ง

ภาพที่ 2 เป็นตัวอย่างของโครงข่ายประสาทเทียมชนิดคอนโวลูชันชนิดหนึ่งที่เรียกว่า VGG ซึ่งมีโครงสร้างเรียบง่าย หนึ่งบล็อกประกอบด้วยสองชั้นคอนโวลูชันและหนึ่งชั้นพูลิ่ง ทุกชั้นและทุกบล็อกต่อแบบป้อนข้างหน้า (ไม่มีป้อนกลับ ไม่มีป้อนข้าม) และ VGG1, VGG2, VGG3 ก็หมายถึงการต่อบล็อกพวกนี้เข้าด้วยกัน 1, 2, และ 3 บล็อก ตามลำดับ

2.2.1 ชั้นคอนโวลูชัน

จุดประสงค์ของชั้นคอนโวลูชัน คือเพื่อสกัดฟีเจอร์ออกมา นำไปสร้างเป็นแผนผังฟีเจอร์

การสกัดฟีเจอร์ที่แตกต่าง ต้องใช้ฟิลเตอร์ที่แตกต่าง ด้วยเหตุนี้ ฟิลเตอร์ของชั้นคอนโวลูชัน (เคอร์เนล) จะประกอบด้วยเซลล์ประสาทเทียม ที่จะเรียนรู้จากส่วนรูปภาพ จนกว่าจะได้ฟิลเตอร์ที่เหมาะสมต่อการสกัดฟีเจอร์ที่ต้องการ นอกจากนั้นยังอาจจะสกัดหลายครั้งด้วยฟิลเตอร์หลายตัว (โดยทั่วไปแล้ว ฟิลเตอร์พวกนี้แต่ละตัวขนาดเท่ากัน) ได้เป็นแผนผังฟีเจอร์

ภาพที่ 1 มีส่วนที่แสดงการสกัดฟีเจอร์ (Convolution) ซึ่งสังเกตได้จากสิ่งที่สกัด (พื้นที่สีเขียว) ได้ถูกสกัดเป็นแผนผังฟีเจอร์ (พื้นที่สีฟ้า)

2.2.2 ชั้นพูลิ่ง

จุดประสงค์ของชั้นพูลิ่ง คือเพื่อลดขนาดข้อมูลโดยไม่เสียฟีเจอร์ ก่อนส่งต่อไปชั้นถัดไป

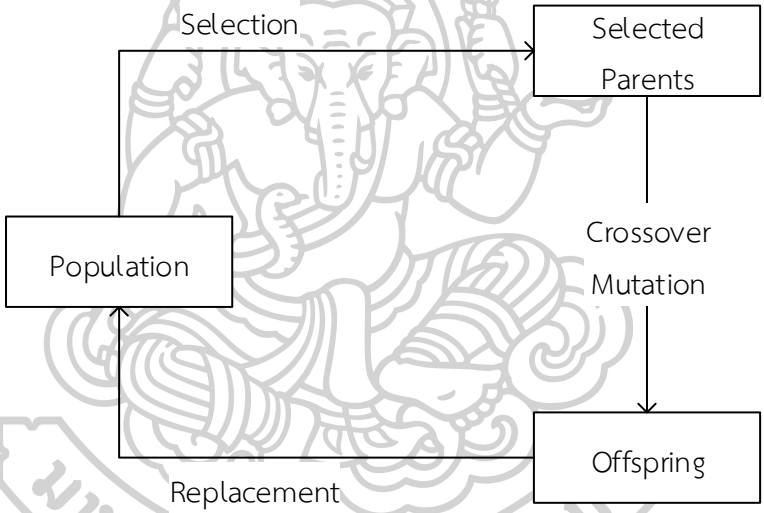
การลดขนาดข้อมูลที่ใช้ในชั้นพูลิ่ง มีฟิลเตอร์ที่นิยมอยู่สองแบบ คือฟิลเตอร์ที่เมื่อคอนโวลูชันกับส่วนรูปภาพแล้วจะเลือกจุดค่าสูงสุดออกมาจากส่วนรูปภาพนั้น กับฟิลเตอร์ที่เมื่อคอนโวลูชันกับส่วนรูปภาพแล้วจะได้ค่าเฉลี่ยจากทุกจุดในส่วนรูปภาพนั้น

ภาพที่ 1 มีส่วนที่แสดงการลดขนาดข้อมูล (Subsampling) ซึ่งสังเกตได้จากสิ่งที่สกัด (พื้นที่สีเขียว) ได้ถูกลดขนาดลงในชั้นถัดไป (พื้นที่สีเขียว)

2.3 การคำนวณเชิงวิวัฒนาการ (Evolutionary Computation)

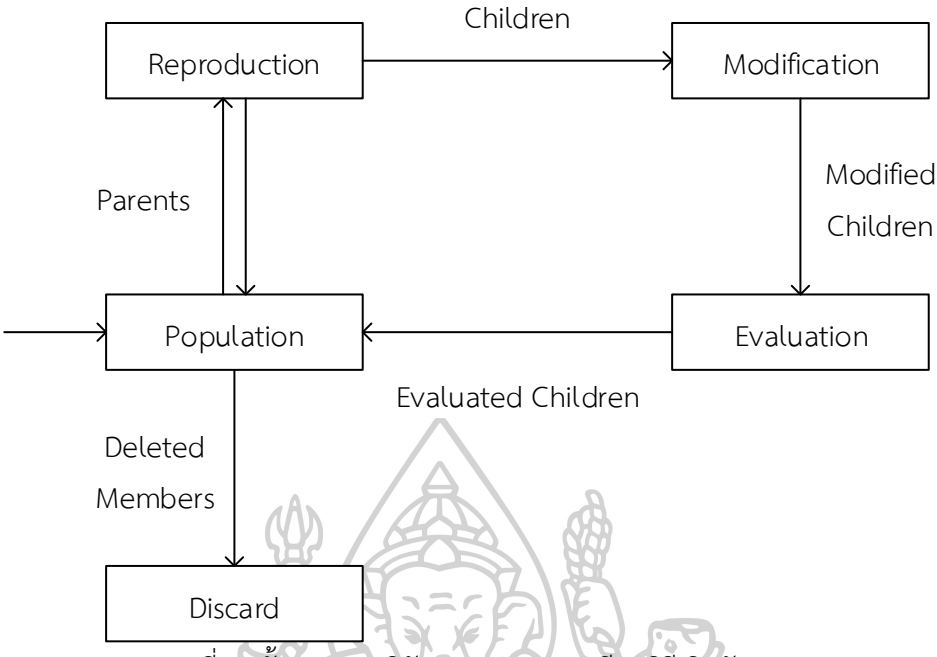
การคำนวณเชิงวิวัฒนาการ ได้รับแรงบันดาลใจจากการที่สิ่งมีชีวิตมีวิวัฒนาการจากรุ่นสู่รุ่นซึ่งโดยพื้นฐานอัลกอริทึมเชิงวิวัฒนาการประด้วยด้วย ระเบียบวิธีเชิงพันธุกรรม กลยุทธ์เชิงวิวัฒนาการ การโปรแกรมเชิงพันธุกรรม และความฉลาดแบบกลุ่ม [4]

2.3.1 ระเบียบวิธีเชิงพันธุกรรม (Genetic Algorithm)



ภาพที่ 3: แผนผังการทำงานของวิธีเชิงพันธุกรรมอย่างง่าย

ระเบียบวิธีเชิงพันธุกรรม มีหลักการมาจากการที่สิ่งมีชีวิตเข้ารหัสข้อมูลของตัวเองทั้งหมดเอาไว้ในรหัสพันธุกรรม หรือเรียกว่าโครโมโซม ซึ่งสามารถถูกถอดเพื่อสร้างเป็นสิ่งมีชีวิตตนนั้นอีกครั้ง โดยแนวคิดหลักคือ การทำงานเป็นรุ่นๆ โดยแทนคำตอบของปัญหาเป็นโครโมโซม และปรับปรุงพันธุ์ด้วยการสืบพันธุ์ และกลายพันธุ์เพื่อให้คำตอบที่ดีขึ้นในรุ่นถัดไป โดยที่ในการประเมินคุณภาพของคำตอบที่ถูกเข้ารหัสในรูปแบบโครโมโซมนั้น และสร้างประชากรรุ่นใหม่ในรูปแบบโครโมโซมซึ่งจำต้องถูกคัดเลือกให้ไปต่อในรุ่นถัดไปโดยที่จะต้องแทนเป็นคำตอบจริงเพื่อไปทำการประเมินประสิทธิภาพด้วยฟังก์ชันต้นทุน หรือฟิตเนสฟังก์ชัน (Fitness function) จากนั้นลือกประชากร หรือคำตอบที่ดีเพื่อส่งต่อเป็นประชากรในรุ่นถัดไป [4]



ภาพที่ 4: ขั้นตอนการวิวัฒนาการของระเบียบวิธีเชิงพันธุกรรม

ภาพที่ 4 แสดงหลักการของระเบียบวิธีเชิงพันธุกรรมซึ่งประกอบด้วย

1. กลุ่มคำตอบเริ่มต้นถูกสุ่มสร้างขึ้นใหม่ในรอบแรก และถอดรหัสเอาจากกลุ่มรหัสที่ได้จากข้อ 5. ในรอบก่อน ถ้าเป็นรอบสอง หรือรอบถัดๆมา
2. นำคำตอบในกลุ่มคำตอบทั้งหมดไปผ่านฟิตเนสฟังก์ชัน หรือฟังก์ชันต้นทุนเพื่อตรวจสอบว่าแต่ละคำตอบนั้น ใกล้เคียงกับความถูกต้อง แค่ไหน และหยุดที่ขั้นตอนนี้ เมื่อพบคำตอบที่ใกล้เคียงพอ (หรือถูกต้อง 100%)
3. เข้ารหัสคำตอบทั้งหมด ให้อยู่ในรูปที่คล้ายคลึงกับรหัสพันธุกรรม หรือโครโมโซม
4. นำคำตอบในรูปของรหัส ไปผ่านขั้นตอนที่คล้ายคลึงกับการผสมพันธุ์ และ/หรือ การกลายพันธุ์ ได้เป็นรหัสกลุ่มใหม่ จากนั้นทำการประเมิน และคัดเลือก เพื่อสร้างประชากรของคำตอบในรุ่น หรือรอบถัดไป
5. เริ่มขั้นตอนที่ 1. ใหม่

2.3.2 กลยุทธ์เชิงวิวัฒนาการ (Evolutionary Strategy)

กลยุทธ์เชิงวิวัฒนาการ ได้รับแรงบันดาลใจมาจากการวิวัฒนาการของสิ่งมีชีวิต และตัวมันเองก็มีรูปแบบที่หลากหลาย แต่โดยทั่วไปแล้ว กลยุทธ์เชิงวิวัฒนาการมีจุดร่วมกันอยู่สามอย่าง [4]

1. มีการเลือกคำตอบบางตัว (ตัวเดียวหรือหลายตัวก็ได้) ไปเพื่อไปผลิตเป็นคำตอบกลุ่มใหม่ด้วยการกลายพันธ์เป็นหลัก โดยเน้นสร้างลูกจากพ่อหรือ แม่ 1 ตัวเท่านั้น

2. คำตอบกลุ่มใหม่ที่ได้จะถูกผ่านขั้นตอนที่คล้ายคลึงกับการกลายพันธุ์
3. มีการเลือกคำตอบบางตัวออกจากกลุ่ม หรือ เพิ่มเข้าไปในกลุ่ม

ด้วยเหตุผลดังที่ได้อธิบายไปแล้วในระเบียบวิธีเชิงพันธุกรรม เนื้อหาบางอย่าง เช่น การ ผสมพันธุ์ การกลายพันธุ์ และการเข้ารหัสถอดรหัสพันธุกรรม ที่จะแยกแยะได้ว่ามาจากระเบียบวิธีเชิงพันธุกรรม หรือกลยุทธ์เชิงวิวัฒนาการ

เนื้อสำคัญของกลยุทธ์เชิงวิวัฒนาการอยู่ในส่วนของการกลายพันธุ์ หลักการใหญ่ๆ ของกลยุทธ์เชิงวิวัฒนาการ เน้นไปที่การปรับตัวแปรในการสุ่มต่างๆ ซึ่งมีผลต่อการกลายพันธุ์โดยตรง ให้ค่าที่ได้จากการกลายพันธุ์เข้าใกล้คำตอบที่ถูกต้องได้เร็วขึ้น

2.3.3 ความฉลาดแบบกลุ่ม (Swarm Intelligence)

หลักการของความฉลาดแบบกลุ่ม มาจากพฤติกรรมของฝูงสัตว์ เช่น มด ผึ้ง นก ฯลฯ โดยเฉพาะในส่วนที่สัตว์หลายตัวรวมกันสามารถทำงานที่ซับซ้อนขึ้นได้ [4]

ข้อสำคัญของความฉลาดแบบกลุ่ม คือการจัดระเบียบกลุ่มด้วยตัวเอง โดยที่ไม่มีศูนย์ควบคุมกลาง การจัดระเบียบดังกล่าว เกิดขึ้นจากปฏิกิริยาพื้นฐานง่ายๆ ระหว่างแต่ละตัว จนเกิดเป็นระเบียบ และ ในที่สุด ความฉลาด

การทำให้เหมาะสมแบบกลุ่มอนุภาค (Particle Swarm Optimization) เป็นตัวอย่างของอัลกอริทึมประเภทความฉลาดแบบกลุ่ม ที่ได้รับแนวคิดมาจากพฤติกรรมของฝูงนกฝูงปลา โดยในการค้นหาคำตอบของปัญหา ก็ให้กลุ่มอนุภาคกลุ่มหนึ่ง เคลื่อนที่ไปในพื้นที่ที่จะค้นหาคำตอบ โดยแต่ละอนุภาคจะติดต่อกันในกลุ่ม ว่าอนุภาคไหนกำลังอยู่ที่ไหน ด้วยความเร็วเท่าไหร่ เข้าใกล้คำตอบที่ต้องการหรือไม่ เพื่อใช้การติดต่อดังกล่าว ชักนำอนุภาคที่อยู่ไกลจากคำตอบที่ต้องการ หรือกำลังหลงทาง ให้เดินทางมายังคำตอบที่ถูกต้อง

2.4 การวิวัฒนาการโครงสร้างของโครงข่ายประสาทเทียมเชิงลึกด้วยอัลกอริทึมเชิงวิวัฒนาการ (Evolving Deep Neural Architecture with Evolutionary Algorithm)

อัลกอริทึมเชิงวิวัฒนาการนานาชนิดได้ถูกนำมาใช้ค้นหาโครงสร้างโครงข่ายประสาทเทียมแทนมือมนุษย์ [13] [14] โครงข่ายประสาทเทียมที่เป็นที่นิยมถูกค้นหาโดยอัลกอริทึมเชิงวิวัฒนาการมากที่สุดในระยะแรกก็คือ โครงข่ายประสาทเทียมแบบคอนโวลูชัน หรือ CNN ซึ่งมักถูกใช้ทำการเปรียบเทียบระหว่างงานวิจัยที่เกี่ยวข้องกับการค้นหาโครงข่ายประสาทเทียมแบบอัตโนมัติจำนวนมาก ในส่วนของชุดข้อมูลมาตรฐานนั้น MNIST CIFAR10 และ CIFAR100 ซึ่งข้อมูลภาพ เป็นชุดข้อมูลที่

นิยมใช้ทดสอบโครงข่ายประสาทเทียมที่สุด เป้าหมายคือการค้นหาโครงสร้างโครงข่ายประสาทเทียมที่ได้เพื่อไปทำการปรับหาค่าน้ำหนักต่อไป และนำไปประยุกต์ใช้กับปัญหาอื่นๆต่อไป [15]

2.5 การหาค่าเหมาะที่สุดแบบหลายวัตถุประสงค์ (Multi Objective Optimization)

การหาค่าเหมาะที่สุดแบบหลายวัตถุประสงค์ได้ถูกนำมาใช้งานอย่างกว้างขวาง โดยเฉพาะการหาค่าค่าเหมาะที่สุดแบบหลายวัตถุประสงค์เชิงวิวัฒนาการ (Evolutionary multi-objective optimization) โดยแนวคิดคือการสร้างกลุ่มคำตอบที่คำนึงหลายวัตถุประสงค์ในระหว่างการค้นหา โดยทำการสร้างกลุ่มคำตอบที่เหมาะสมแบบพาเรโต้ (Pareto optimality) คือการสร้างกลุ่มคำตอบแบ่งแบบระนาบที่เรียกว่าว่าพาเรโตฟร้อน (Pareto front) โดยแนวทางที่ได้รับความนิยมมักพัฒนาต่อยอดจากแนวคิดการเรียงโดยคำนึงถึงการถูกครอบงำ สำหรับกลุ่มคำตอบที่คำนึงถึงหลายวัตถุประสงค์สำหรับปัญหาเดียวกัน ถ้าคำตอบหนึ่งไม่ด้อยกว่าอีกคำตอบหนึ่งในทุกวัตถุประสงค์ของปัญหา และคำตอบหนึ่งเหนือกว่าอีกคำตอบหนึ่งในอย่างน้อยหนึ่งวัตถุประสงค์ของปัญหา ถือว่าคำตอบหนึ่งครอบงำคำตอบสอง [10] พาเรโตฟร้อนเป็นกลุ่มคำตอบแบบระนาบชนิดหนึ่ง โดยในพาเรโตฟร้อนเดียวกันจะไม่มีคำตอบใดครอบงำคำตอบอื่น

2.5.1 วิธีเชิงคัดสรรผลเฉลยที่ไม่ถูกครอบงำ (Non-domination Sorting)

วิธีเชิงคัดสรรผลเฉลยที่ไม่ถูกครอบงำ เป็นกลยุทธ์หาค่าเหมาะสมหลายวัตถุประสงค์ชนิดหนึ่ง ในจุดประสงค์นี้ การครอบงำ หมายถึง การที่ผลเฉลยหนึ่ง มีคะแนนในทุกวัตถุประสงค์ เป็นที่พึงประสงค์กว่า ผลเฉลยอีกผลเฉลยหนึ่ง ก็จะยอมรับว่า ผลเฉลยแรก ครอบงำผลเฉลยหลัง และเมื่อใช้วิธีเชิงคัดสรรผลเฉลยที่ไม่ถูกครอบงำในการจัดเรียงผลเฉลย ผลเฉลยจะถูกเรียงเป็นขั้น โดยในแต่ละขั้นจะไม่มีผลเฉลยใดที่ครอบงำผลเฉลยอื่นในขั้นเดียวกัน ขั้นแรกสุด คือขั้นที่ไม่ถูกครอบงำใดๆ เลย ขั้นรองลงมา ก็จะถูกครอบงำโดยขั้นก่อนๆ หน้า ไปจนถึงขั้นสุดท้าย ที่ไม่สามารถครอบงำขั้นอื่นใดๆ เลย [16]

2.6 นิวโรอีโวลูชั่น (Neuroevolution)

นิวโรอีโวลูชั่น คือการใช้อัลกอริทึมเชิงวิวัฒนาการทำการปรับปรุง หรือดัดแปลงโครงข่ายประสาทเทียมเพื่อให้ได้โครงข่ายประสาทเทียมที่เหมาะสมที่สุดนำไปใช้งาน [17, 18] ในช่วงแรกก่อนการพัฒนาสู่โครงข่ายประสาทเทียมเชิงลึก นิวโรอีโวลูชั่น ใช้งานกับโครงข่ายประสาทเทียมแบบเพอร์เซ็ปตรอนหลายชั้น ซึ่งถือว่าเป็นโครงข่ายประสาทเทียมแบบตื้น [3, 6, 18] เมื่อโครงข่ายประสาทเทียมพัฒนาสู่การเรียนรู้เชิงลึก จึงเริ่มมีการใช้งานนำนิวโรอีโวลูชั่นมาปรับปรุงและใช้งานอีกครั้ง [7,

14] ประเด็นที่สำคัญของนิวโรอีโวลูชั่นคือ การใช้งานอัลกอริทึมเชิงวิวัฒนาการที่มีประสิทธิภาพเหมาะสมกับปัญหาเนื่องจากมีอัลกอริทึมที่หลากหลายมากซึ่งได้กล่าวถึงในหัวข้อก่อนหน้า และวิธีการเข้ารหัสโครงข่ายประสาทเทียมเป็นเรื่องสำคัญมาก [19] วิธีการเข้ารหัสโครงข่ายประสาทเทียมสามารถแบ่งเป็นสามกลุ่มใหญ่ๆ ดังต่อไปนี้

2.6.1 การแทนตัวโดยตรง (Direct Representation)

การเข้ารหัสแบบที่เรียบง่ายที่สุด คือการเปลี่ยนข้อมูลทั้งหมดของโครงข่ายประสาทเทียมเป็นบิตโดยตรง สำหรับการคำนวณหาโครงข่ายประสาทเทียมโดยกำหนดโครงสร้างของโครงข่ายประสาทเทียมเอาไว้ตายตัว โครงข่ายประสาทเทียมสามารถถูกเขียนในรูปเวกเตอร์ของตัวเลข [18] ซึ่งนอกจากจะทำให้ขั้นตอนการเข้ารหัสถอดรหัสไม่ซับซ้อน เพราะสามารถใช้รหัสชุดเดียวแทนโครงข่ายประสาทเทียมทั้งชุด แล้วยังทำให้สามารถนำอัลกอริทึมทางคณิตศาสตร์หลายอย่างมาประยุกต์ใช้ด้วยได้ [9]

ในโครงข่ายประสาทเทียมที่มีโครงสร้างไม่ตายตัว อย่างน้อยต้องมีรหัสสองชุดเพื่อแทนโครงข่ายประสาทเทียมชุดเดียว รหัสชุดแรกจะใช้แทนค่าน้ำหนักการเชื่อมต่อของเพอร์เซ็ปตรอนแต่ละตัว รหัสชุดที่สองจะใช้แทนโครงสร้างของโครงข่ายประสาทเทียมโดยรวม [18] ขั้นตอนเข้ารหัสถอดรหัส หรือแม้แต่การคำนวณเชิงวิวัฒนาการ ก็จะยุ่งยากซับซ้อนขึ้น แต่ข้อดีใหญ่ของวิธีนี้ คือโครงข่ายประสาทเทียมสุดท้ายที่ได้มาจะถูกลดขนาดส่วนเกินออกตามการวิวัฒนาการ ทำให้ได้โครงข่ายประสาทเทียมที่เล็ก [6] ซึ่งก็จะหมายถึงทำงานเร็วขึ้น และประหยัดทรัพยากรเครื่องมากขึ้นเช่นกัน

2.6.2 การแทนตัวเชิงพัฒนาการ (Developmental Representation)

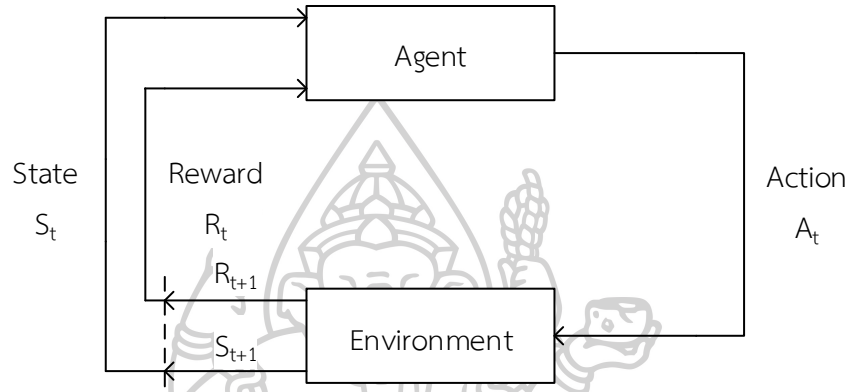
แทนที่จะเปลี่ยนโครงข่ายประสาทเทียมเป็นบิตโดยตรง ก็เปลี่ยนข้อมูลโครงข่ายประสาทเทียมเป็นสัญลักษณ์ แล้วค่อยเปลี่ยนสัญลักษณ์เป็นบิต วิธีนี้จำเป็นต้องตั้งกฎการเปลี่ยนสัญลักษณ์เป็นโครงข่ายประสาทเทียม [18] และมีขั้นตอนการคำนวณมากขึ้น แต่มีข้อดีคือสามารถแทนที่โครงข่ายประสาทเทียมขนาดใหญ่ได้ด้วยรหัสขนาดเล็ก [9] ซึ่งโครงข่ายประสาทเทียมขนาดใหญ่ย่อมหมายถึงขีดความสามารถในการเรียนรู้ที่ซับซ้อนขึ้น

2.6.3 การแทนตัวโดยอ้อม (Indirect Representation)

หลักการของการแทนตัวโดยอ้อมคือการใช้สัญลักษณ์แบ่งส่วนในรหัส เพื่อระบุว่าส่วนไหนของรหัสคือส่วนที่จะถูกถอดเป็นโครงข่ายประสาทเทียม ส่วนไหนของรหัสเป็นส่วนที่ถูกปล่อยทิ้งไม่ถูกนำมาใช้ตอนถอดรหัส วิธีการเช่นนี้มีข้อดีตรงที่ทำให้ไม่ต้องมีกฎพิเศษเพื่อปกป้องรหัสขณะทำการคำนวณเชิงวิวัฒนาการในส่วนการผสมพันธุ์กับการกลายพันธุ์ เพราะส่วนที่ถูกปล่อยทิ้งจะทำหน้าที่เป็นกันชนไม่ให้ส่วนของรหัสที่ถูกถอดเป็นโครงข่ายประสาทเทียมเสียหายในขั้นตอนเหล่านั้น [18] และเนื่องจากนี่เป็นวิธีธรรมชาติที่สิ่งมีชีวิตจริงๆ ทำ จึงรับประกันความสำเร็จในระดับหนึ่ง

2.7 การเรียนรู้แบบเสริมกำลัง (Reinforcement Learning)

การเรียนรู้แบบเสริมกำลัง เป็นส่วนการเรียนรู้ของเครื่อง [2, 12, 20] ชนิดหนึ่ง ที่ให้เอเย่นต์เรียนรู้ในสภาพแวดล้อมเพื่อให้ได้รับรางวัลเป็นคะแนน ซึ่งการประยุกต์ใช้ที่สำคัญคือเกมส์เนื่องจากระบบหรือเอเย่นต์ต้องมีการปฏิสัมพันธ์กับสิ่งแวดล้อมตลอดเวลาในระหว่างการเล่นเกมส์ การเรียนรู้แบบเสริมกำลังประกอบด้วย สิ่งแวดล้อม เรียนรู้สภาวะ การกระทำ และค่ารางวัล จากสิ่งแวดล้อมวนไปเป็นวัฏจักร แสดงดังภาพที่ 5



ภาพที่ 5: แผนผังการเรียนรู้แบบเสริมกำลังอย่างง่าย

เป้าหมายของเอเย่นต์ปัญญาประดิษฐ์ที่อยู่ในการเรียนรู้แบบเสริมกำลังคือเรียนรู้หลักการที่จะทำให้เมื่อนำไปปฏิบัติในสิ่งแวดล้อมแล้วจะสามารถได้รับคะแนนสูงสุดในความเป็นจริง การเรียนรู้แบบเสริมกำลัง ถือเป็นวิชาที่มีเนื้อหาที่ลึกซึ้งซึ่งนำเสนอใน [20] สิ่งที่เรานำมาใช้มีเพียงส่วนเล็กๆ ส่วนเดียวเท่านั้น นั่นคือ Deep Q Learning [17]

2.8 การใช้งานนิวโรอีโวลูชั่นกับเกมส์

เกมส์เป็นเป้าทดสอบของระบบปัญญาประดิษฐ์นานาชนิดมาช้านาน [5] มีการประยุกต์ใช้นิวโรอีโวลูชั่นกับปัญหาเกมส์มาอย่างต่อเนื่อง [9] โครงข่ายประสาทเทียมได้ถูกประยุกต์ใช้กับเกมส์เนื่องจากสามารถเรียนรู้ และเก็บความรู้ไว้ได้ในขณะที่มีการเล่นเกมส์ทำให้เป็นการเรียนรู้แบบออนไลน์ สิ่งนี้มีความสำคัญเป็นพิเศษ เพราะโครงข่ายประสาทเทียมที่ทำงานหนึ่งได้ สามารถถูกดัดแปลงให้ทำงานอื่นที่คล้ายๆ กันได้โดยง่าย [9] ซึ่งมีประเด็นที่สำคัญต่อไปนี้

1) ระดับความความซับซ้อนของเกมส์ เพื่อรับประกันว่า โครงข่ายประสาทเทียมที่สามารถเล่นเกมส์นี้ได้ดี ย่อมสามารถนำไปแก้ปัญหาจริงๆ ที่มีความซับซ้อนสูงได้เฉกเช่นกัน ในจุดประสงค์ของส่วนนี้ เราจะใช้สเตทสเปซ (จำนวนความเป็นไปได้ทั้งหมดของสภาวะเกมส์ตั้งแต่ต้นจนจบ) เป็นเกณฑ์วัดความซับซ้อนของเกมส์ที่สามารถวัดสเตทสเปซได้

(หมากรุกมีสเตทสเปซ 10^{43} [5, 21] และหมากล้อมแบบ 19x19 มีสเตทสเปซ 10^{170} [21]) และให้เหตุผลว่าทำไมถึงวัดสเตทสเปซไม่ได้ ถ้าเกมส์นั้นวัดสเตทสเปซไม่ได้

2) การประเมินประสิทธภาพของเอเย่นต์ที่ถูกพัฒนาขึ้น รวมถึงระดับความยากง่ายในการเขียนโปรแกรมควบคุมลงไป เช่น เป็นเกมส์ดั้งเดิมที่ใช้ทดสอบเอเย่นต์ปัญญาประดิษฐ์ มีผู้ทำวิจัยการใช้ปัญญาประดิษฐ์ในเกมส์นั้นสำเร็จแล้วมาก่อน เพื่อรับประกันว่า จะมีต้นแบบสำหรับเปรียบเทียบประสิทธิภาพ (จริงๆ แล้ว เราตั้งเงื่อนไขนี้เอง เพื่อจำกัดเวลาในการทำการวิจัยโดยรวมลง)

ในบทนี้จะทำการสำรวจเกมส์ที่เกี่ยวข้อง โดยเกมส์ดังต่อไปนี้เป็นตัวอย่างเกมส์ที่ถูกออกแบบมาให้มนุษย์เล่นและเคยมีผู้นำไปทดลองใช้กับปัญญาประดิษฐ์จนเป็นที่ประจักษ์มาแล้ว

2.8.1 เกมส์ Civilization II

Civilization II เป็นเกมส์วางแผนการรบประเภท 4X ผู้เล่นจะเริ่มจากนำผู้อพยพไปตั้งรกรากถิ่นฐาน จากนั้นก็สำรวจ ขยับขยาย และในที่สุด เอาชนะผู้เล่นอื่น มีสเตทสเปซขนาด 10188 สเตท ในการทดสอบ ให้มีผู้เล่นสู้กันตัวต่อตัว ถ้าให้เพิ่มจำนวนคนหรือมีการเล่นเป็นทีม จะใหญ่กว่านี้ได้อีก [22] ซึ่งใหญ่กว่าหมากล้อมเป็นล้านล้านล้านเท่า เป็นเกมส์ที่สามารถเล่นได้ด้วย Monte-Carlo Search Tree อยู่ ด้วยผลสำเร็จอันเป็นที่น่าพอใจ [22] ข้อที่น่าสนใจตรงนี้ อยู่ที่มีการดัดแปลง Monte-Carlo Search Tree ให้ใช้ข้อมูลจากคู่มือเกมส์ Civilization II ประกอบการตัดสินใจ

2.8.2 เกมส์ Ms. Pacman

Pacman เป็นเกมส์ที่ผู้เล่นจะเล่นเป็นแป๊กแมน แล้วไล่กินเม็ดยาในด่าน ไปพร้อมๆ กับคอยหลบผีซึ่งมีหลายตัวในด่าน ถ้าโดนผีกินจะแพ้ และจะมีเม็ดยาพิเศษที่ทำให้แป๊กแมนกินผีได้ชั่วคราว [23]

Pacman เป็นเกมส์ที่เล่นในเวลาจริง หรือเรียลไทม์ จึงแทบจะเป็นไปไม่ได้ที่จะคำนวณสเตทสเปซ โดยจุดประสงค์ของการวิจัย Ms. Pacman ต่างจาก Pacman ตรงที่มีส่วนที่ถูกสุ่ม ไม่ได้ขยับตามกฎ 100% ดังนั้นในการวัดสเตทสเปซ สามารถใช้กฎเดียวกัน มีคนตั้งการแข่งขันสร้างบอทควบคุมฝั่งแป๊กแมนและฝั่งผี และมีผู้ทดลองใช้การคำนวณเชิงวิวัฒนาการในการสร้างบอทสำหรับทั้งฝั่งแป๊กแมนและฝั่งผี เขาใช้อัลกอริทึมที่ชื่อ Competitive Coevolution ที่ให้บอททั้งสองฝ่ายสู้กันและปรับตัวขึ้นเรื่อยๆ ผลลัพธ์เป็นที่น่าพอใจสำหรับฝั่งแป๊กแมน แต่ไม่เป็นที่น่าพอใจนักสำหรับฝั่งผี [23]

2.8.3 เกมส์ Hero Academia

Hero Academia เป็นเกมส์กระดานขนาด 5x9 ผู้เล่นจะเริ่มเกมส์มาโดยมีคริสตัลคนละอันในกระดาน และเล่นเกมส์โดยวางยูนิตลงในกระดาน ให้ยูนิตในกระดานเคลื่อนไหว หรือใช้เวทจากบนมือ

(คำสั่งของผู้เล่นใช้ Action Point และตอนเริ่มตาตัวเอง ผู้เล่นจะมี Action Point 5) เป้าหมายของเกมส์คือกำจัดยูนิตของฝ่ายตรงข้ามทุกตัว หรือกำจัดคริสตัลของฝ่ายตรงข้าม[5]

Hero Academia มีสเตทสเปซขนาด 10199 สเตท [5] (ตัดข้อมูลของฝ่ายตรงข้ามที่ผู้เล่นไม่รู้ เช่น เวทที่ฝ่ายตรงข้ามถือครอง แล้ว) ใหญ่กว่าหมากล้อมแสนล้านล้านล้านล้านเท่า

จากข้อมูลบทความ มีการทดลองเล่น Hero Academia ด้วย Monte-Carlo Search Tree และ Online Evolution Planning (ใช้การคำนวณเชิงวิวัฒนาการ) ด้วยผลสำเร็จที่ใกล้เคียงกัน [5] ข้อที่น่าสนใจตรงนี้อยู่ที่ ในขณะที่การใช้ Monte-Carlo Search Tree จำเป็นต้องดัดแปลงก่อน Online Evolution Planning สามารถนำมาใช้ได้โดยไม่ต้องดัดแปลงมากนัก

2.8.4 เกมส์ Starcraft

Starcraft เป็นเกมส์วางแผนการรบประเภท RTS ที่ผู้เล่นจะเริ่มเกมส์มาด้วยฐานทัพ และต้องรวบรวมทรัพยากร สร้างฐานเพิ่ม สร้างกองทัพ และทำลายฐานทัพของผู้เล่นอื่นให้หมด ผู้เล่นในเกมส์ Starcraft จะควบคุมเกมส์พร้อมกันในเวลาจริง จึงแทบจะเป็นไปไม่ได้ที่จะคำนวณสเตทสเปซ จากข้อมูลบทความ Starcraft เคยถูกใช้เป็นเป้าทดลองการวางแผนวิวัฒนาการแบบออนไลน์ (Online Evolution Planning) [24, 25] แต่ผู้ทดลองไม่สามารถใช้ การวางแผนวิวัฒนาการแบบออนไลน์ตรงๆ ได้ ด้วยเหตุผลทางเทคนิคบางอย่าง เลยต้องมีการดัดแปลงก่อน

2.8.5 เกมส์ Robocode

Robocode เป็นเกมส์เขียนบอทคุมรถถังต่อสู้กัน ที่ใช้คะแนนที่ได้จากการโจมตีรถถังศัตรูเป็นเกณฑ์วัดผลแพ้ชนะ สามารถมีกรณีที่คนที่รถถังพังก่อน แต่โจมตีได้มากกว่า จนได้คะแนนมากกว่า และเป็นฝ่ายชนะ และกรณีที่คนที่รถถังไม่พัง แต่โจมตีได้น้อยกว่า ได้คะแนนน้อยกว่า จนเป็นฝ่ายแพ้ไปในที่สุด [8, 26]

Robocode เป็นเกมส์ควบคุมในเวลาจริง แม้ผู้เล่นจะไม่ได้คุมเอง เพราะต้องเขียนบอทคุม ดังนั้นการคำนวณสเตทสเปซจึงแทบจะเป็นไปไม่ได้ ข้อสำคัญของ Robocode คือ บอทที่ควบคุมรถถัง จะรับรู้ข้อมูลโลกภายนอกได้แค่ทางเซ็นเซอร์ ไม่ได้รับรู้ทั้งสนาม หรือเป็นวงกว้างจากบทความมีการใช้การคำนวณเชิงวิวัฒนาการ [26] และโครงข่ายประสาทเทียมโดยเฉพาะนิวโรอีโวลูชั่น [8] ในการเล่น Robocode ได้สำเร็จมาแล้ว

2.9 วรรณกรรมที่เกี่ยวข้อง

1 A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II

งานวิจัยนี้ได้อธิบายถึง NSGA-II [16] ซึ่งเป็น วิธีเชิงพันธุกรรม ที่ใช้วิธีคัดสรรผลเฉลยที่ไม่ถูกครอบงำ เป็นตัวปฏิบัติทางพันธุกรรมคัดผู้เหมาะสม โดยรายละเอียดแล้ว NGSA-II ปรับปรุงวิธีคัดสรรผลเฉลยที่ไม่ถูกครอบงำเดิม โดยวิธีเดิมจะใช้การคัดผลเฉลยที่ไม่ถูกครอบงำออกทีละขั้น ตั้งแต่ขั้นแรก

สุด ลงไปจนกว่าจะถึงขั้นสุดท้าย แต่วิธีของพวกเขาจะมีการจดคะแนนการไม่ถูกครอบงำของแต่ละผลเฉลยตั้งแต่การคัดรอบแรกสุดเก็บไว้ ซึ่งคะแนนนี้จะถูกนำไปใช้ในการคัดผลเฉลยชั้นถัดๆ ไปได้ทันที ทำให้เสียเวลาคัดสรรน้อยลง ในตัวอย่างที่พวกเขาให้มา NGSA-II ถูกใช้กับการหาค่าเหมาะสมแบบสองวัตถุประสงค์ แต่ในความเป็นจริงแล้ว NGSA-II สามารถใช้ได้กับมากกว่าสองวัตถุประสงค์

2 NSGA-Net: Neural Architecture Search using Multi-Objective Genetic Algorithm

ในงานวิจัยนี้ได้กล่าวถึง NSGA-Net [27] ซึ่งคือการใช้ NSGA (ณ จุดนี้ มี NGSA หลายชนิด) ค้นหาผลเฉลยซึ่งก็คือโครงข่ายประสาทเทียมในตัวอย่างที่พวกเขาให้มา พวกเขาเลือกรหัสพันธุกรรมที่มีผลเฉลยเป็นโครงข่ายประสาทเทียมแบบคอนโวลูชันที่มีขนาดจำกัด แต่สามารถมีโครงสร้างที่ไม่ถูกจำกัดแค่จากป้อนจากชั้นหน้าไปชั้นหลัง โดยรหัสพันธุกรรมจะเป็นกราฟแสดงทิศทางการเชื่อมต่อระหว่างแต่ละชั้นของโครงข่ายประสาทเทียม ที่ถูกเข้ารหัสเป็นบิท 0-1 ซึ่งรหัสพันธุกรรมทุกตัวจะมีความยาวเท่ากัน เพราะโครงข่ายประสาทเทียมมีขนาดจำกัด

3 Neuroevolution in Deep Neural Networks: Current Trends and Future Challenges

จุดสำคัญที่เราใช้ก็คือ ส่วนที่ 3 ในเนื้อหางานวิจัย การวิวัฒนาการโครงสร้างของโครงข่ายประสาทเทียมเชิงลึกด้วยอัลกอริทึมเชิงวิวัฒนาการ แต่ความจริงแล้ว งานวิจัยนี้ [28] ครอบคลุมการใช้วิธีเชิงวิวัฒนาการและการเรียนรู้เชิงลึกโดยกว้าง โดยมีเป้าหมายหลักไปที่คนที่มีพื้นฐานความเข้าใจอยู่ก่อน

4 Effects of Genetic Operators on Neural Architecture Search Using Multi-Objective Genetic Algorithm

คืองานวิจัยที่เราทำการทดลองสร้างยีนพูลให้ค้นหาโครงข่ายประสาทเทียมสำหรับ CIFAR10 [29] ผลการค้นหาโครงข่ายประสาทเทียมเบื้องต้นก่อนที่เราจะนำไปใช้กับเกมส์ ประสิทธิภาพของเครือข่ายประสาทเทียมขึ้นอยู่กับโครงสร้างทางสถาปัตยกรรมและพารามิเตอร์ของแบบจำลอง การเติบโตอย่างมหาศาลของการเรียนรู้เชิงลึกทำให้เกิดสถาปัตยกรรมโครงข่ายประสาทเทียมเชิงลึก (DNN) ที่หลากหลายซึ่งออกแบบมาสำหรับการใช้งานเฉพาะประเภท เช่น โครงข่ายประสาทเทียมแบบคอนโวลูชัน (CNN) สำหรับการจำแนกประเภทภาพ จนกระทั่งเมื่อไม่นานมานี้ สถาปัตยกรรมดังกล่าวจำนวนมากของ CNN ได้รับการออกแบบด้วยมือมนุษย์ อย่างไรก็ตาม วงการการเรียนรู้เชิงลึกมีความเห็นเป็นเอกฉันท์มากขึ้นว่าการออกแบบสถาปัตยกรรม DNN แบบอัตโนมัตินั้นเป็นสิ่งจำเป็น สิ่งนี้นำไปสู่ Neural Architecture Search (NAS) ซึ่งเป็นกระบวนการอัตโนมัติในการค้นหาสถาปัตยกรรม DNN และไฮเปอร์พารามิเตอร์ อัลกอริทึมเชิงวิวัฒนาการ (EAs) เป็นตัวแก้ปัญหาที่มีแนวโน้มสำหรับ NAS ด้วยการเข้ารหัสโครงสร้างของ DNNs ในโครโมโซมและดำเนินการกับตัวดำเนินการทางพันธุกรรมเพื่อให้ได้การแสดงที่ดีที่สุด NAS (ENAS) ที่ใช้วิวัฒนาการสามารถ

ค้นพบคำตอบคุณภาพสูงได้ วิธีการ NAS สามารถถูกมองว่าเป็นกระบวนการเพิ่มประสิทธิภาพ เพื่อปรับใช้โมเดล DNN

สำหรับการใช้งานจริง มีเกณฑ์การออกแบบที่ต้องปรับให้เหมาะสม เช่น การลดเวลาแฝงและเพิ่มความแม่นยำในการทำนายในเวลาเดียวกัน ดังนั้น งาน NAS ในปัจจุบันสามารถถูกเขียนเป็นปัญหาการหาค่าเหมาะสมหลายวัตถุประสงค์ (MOPs)

อัลกอริทึมการหาค่าเหมาะสมหลายวัตถุประสงค์เชิงวิวัฒนาการ (EMOA) ได้รับการศึกษาอย่างกว้างขวางและนำไปใช้กับ MOP ได้สำเร็จ ดังนั้นจึงเป็นที่ต้องการสำหรับการแก้ปัญหา NAS ที่กำหนดเป็น MOP สำหรับปัญหาต่างๆ ของ NAS นั้น ความแม่นยำและความซับซ้อนของโมเดลเป็นสองวัตถุประสงค์ในการปรับให้เหมาะสม MNasNet คำนึงถึงความเร็วและความแม่นยำในการทำนาย อย่างไรก็ตาม อัลกอริทึมการปรับให้เหมาะสมคือการเรียนรู้แบบเสริมแรงแทน EMOA สำหรับ EMOA สำหรับ NAS, NSGA-Net และ NSGANetV1 ใช้อัลกอริทึมการคัดสรรผลเฉลยที่ไม่ถูกครอบงำด้วยวิธีเชิงพันธุกรรม (NSGA-II) เพื่อเพิ่มประสิทธิภาพปัญหา NAS ในแง่ของข้อผิดพลาดในการคาดคะเนและการดำเนินการแบบทศนิยม เนื่องจากเป็น EMOA หลักในแนวทางที่ไม่ถูกครอบงำ NSGA-II จึงถูกนำไปใช้ใน DeepMaker และให้ผลลัพธ์ที่น่าพึงพอใจบน NAS เมื่อเปรียบเทียบกับแนวทาง EMOA อื่นๆ อย่างไรก็ตาม MODE-CNN ซึ่งเป็น EMOA ที่ใช้การคัดผลเฉลยไม่ถูกครอบงำเช่นกัน ก็ได้ประสิทธิผลเทียบเคียงได้กับ NAS การใช้แนวทางจุดอ้างอิง RNSGA-Net รวม NSGA- II แบบไม่ครอบงำอย่างรวดเร็วเข้ากับจุดอ้างอิงเพื่อเลือก พื้นที่การตั้งค่า ใกล้กับจุดอ้างอิง ของพาเรโตฟร้อน สำหรับ NAS ที่ใช้ EMOAs ครอสโอเวอร์และการกลายพันธุ์เป็นตัวดำเนินการทางพันธุกรรมเชิงวิวัฒนาการที่ใช้มากที่สุด สำหรับ EAs วิธีการต่างๆ จะสนับสนุนเฉพาะตัวดำเนินการกลายพันธุ์เท่านั้น อย่างไรก็ตาม สำหรับ EMOA ส่วนใหญ่แล้ว ตัวดำเนินการทางพันธุกรรมคือการรวมกันของการผสมข้ามและการกลายพันธุ์ มีการแลกเปลี่ยนระหว่างการผสมข้ามและการกลายพันธุ์บน ENAS และผลกระทบที่มีต่อโซลูชัน ดังนั้นในบทความนี้ เราจึงนำเสนอ วิธี NAS แบบหลายวัตถุประสงค์ด้วยประชากรคู่ที่อิงตาม NSGA-II ที่ดัดแปลงแล้ว และการศึกษาผลกระทบของตัวดำเนินการทางพันธุกรรม ได้แก่ ครอสโอเวอร์และการกลายพันธุ์

บทที่ 3
วิธีดำเนินการวิจัย

ขั้นตอนวิธีดำเนินการวิจัยนี้สามารถถูกแบ่งได้เป็น 6 ขั้นตอนดังนี้ 1) รวบรวมข้อมูลและกำหนดขอบเขตงาน 2) ศึกษาทฤษฎีที่เกี่ยวข้อง 3) การออกแบบอัลกอริทึม 4) การออกแบบการทดลอง 5) ทำการทดลอง เก็บรวบรวมข้อมูล และวิเคราะห์ผล และ 6) สรุปผลการวิจัยและจัดทำรายงานวิทยานิพนธ์

3.1 รวบรวมข้อมูลและกำหนดขอบเขตของงาน

ขั้นตอนนี้เป็นการศึกษาเกี่ยวกับโครงข่ายประสาทเทียมชนิดต่างๆ และการสร้างโครงข่ายประสาทเทียม ทั้งด้วยตัวเอง ทั้งด้วยชุดโปรแกรมสำเร็จรูป

3.1.1 กำหนดโปรแกรมที่จะใช้ในการสร้างการเรียนรู้เชิงลึกสำหรับการวิจัย

ในการศึกษาหาข้อมูลรอบแรก ชุดโปรแกรมสำเร็จรูปสำหรับสร้างชุดข้อมูลที่จะใช้เริ่มจากบนภาษา Java แต่ความเป็นจริงที่ว่าการเขียนโปรแกรมข้ามภาษานั้นไม่ง่าย บวกกับทางเราไม่เจอชุดโปรแกรมสำเร็จรูปสำหรับสร้างโครงข่ายประสาทเทียมที่มีคุณสมบัติเป็นที่ต้องการบนภาษา Java ทำให้เมื่อเวลาผ่านไป เราจึงหาโปรแกรมที่จะใช้จากในภาษา Python และในที่สุด ก็เลือกจากตัวเลือกระหว่าง PyTorch กับ TensorFlow-Keras

3.1.2 กำหนดฐานข้อมูลของรูปภาพที่ใช้ในการทดสอบและประเมินความสามารถของระบบ

ณ ในปัจจุบันนี้ เนื่องจากโปรแกรมถูกทำใน TensorFlow-Keras เพื่อความง่าย (บวกกับหลายงานวิจัยใช้) ชุดข้อมูลก็เลยกลายเป็น CIFAR10 ที่ TensorFlow-Keras มีให้มากับตัวด้วย

ชุดข้อมูล CIFAR10 เป็นชุดข้อมูลที่ประกอบด้วยภาพขนาด 32h×32w×3c ที่แต่ละภาพจะคู่กับป้ายบอกประเภทภาพ 1 ใน 10 ประเภท ประเภทละ 6,000 ชุด (รวมเป็นทั้งสิ้น 60,000 ชุด)

3.1.3 กำหนดเกมส์ที่ใช้ในการทดสอบและประเมินความสามารถของระบบ

หลังจากได้โครงข่ายประสาทเทียมที่พึงประสงค์แล้ว ก็จะนำโครงข่ายประสาทเทียมดังกล่าวมาใช้ทดลองสวมเข้าไปในระบบที่ใช้โครงข่ายประสาทเทียมเล่นเกมส์ ซึ่งในกรณีนี้คือเกมส์ CarRacing ที่ทำโดยทีมงาน gym ซึ่งตอนนี้ย้ายไปอยู่ Farama แล้ว [30]

เกมส์ CarRacing เดิมเป็นเกมส์รถแข่งหนึ่งผู้เล่นมุมสูงที่เขียนโดยค่าย Box2D แต่ด้วยความเป็นโอเพ่นซอร์ส ทำให้ค่ายอื่นที่จัดหาระบบสิ่งแวดล้อมสำหรับฝึกปัญญาประดิษฐ์ ได้นำเกมส์ CarRacing ไปประกอบเนื้อหาของตัวเอง เวอร์ชั่นที่เราใช้ นำมาจากค่าย Gymnasium[31] (gym เดิม ที่ย้ายไปสังกัดมูลนิธิ Farama จึงเปลี่ยนชื่อ)

3.2 ศึกษาทฤษฎี และงานวิจัยที่เกี่ยวข้อง

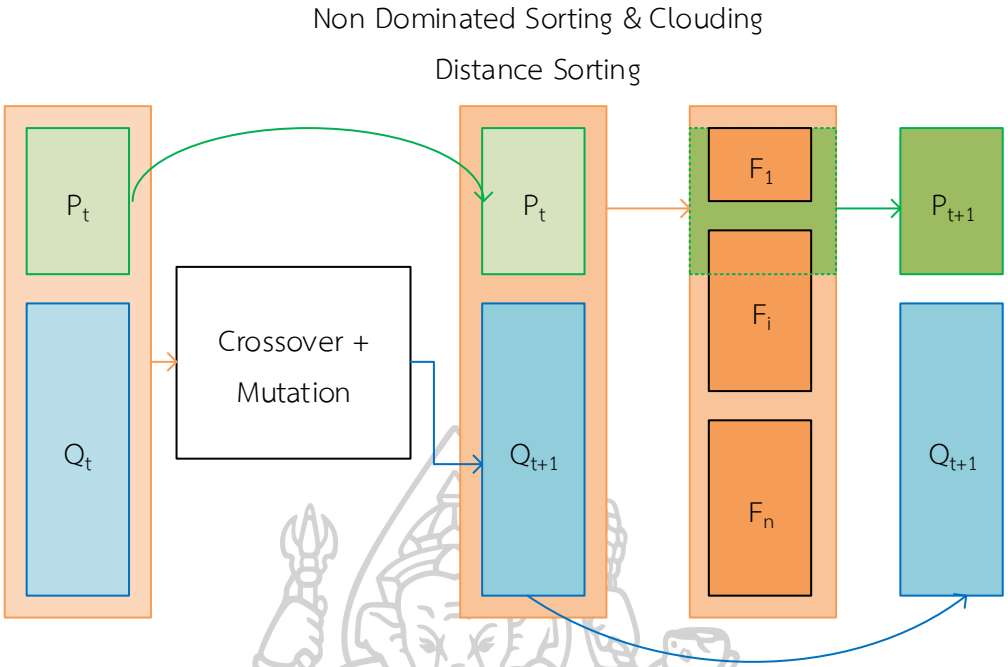
ในขั้นตอนนี้จะเริ่มจากศึกษาโครงข่ายประสาทเทียม ไปจนถึงเจาะจงที่โครงข่ายประสาทเทียมแบบคอนโวลูชัน เพื่อใช้เป็นแนวทางในการออกแบบและพัฒนาระบบ

3.3 การออกแบบอัลกอริทึม

ขั้นตอนนี้เริ่มจากการทดลองทำความเข้าใจกับโครงข่ายประสาทเทียมทั่วไปตั้งแต่จากภาษา C มาจนถึงภาษา Python โดยทดลองใช้กรอบงานสำเร็จรูปที่เหมาะสมในการพัฒนา โดยมีการใช้งานกรอบงาน TensorFlow-Keras ในการออกแบบอัลกอริทึมนิวโรอีโวลูชั่นนั้น งานวิจัยนี้ให้ความสำคัญกับการออกแบบ และใช้งานอัลกอริทึมการหาค่าเหมาะที่สุดหลายวัตถุประสงค์เชิงวิวัฒนาการซึ่งจะกล่าวถึงโดยละเอียดในหัวข้อต่อไปนี้

3.3.1 การหาค่าเหมาะสมหลายวัตถุประสงค์

วิธีการค่าเหมาะสมแบบหลายวัตถุประสงค์เชิงวิวัฒนาการที่ได้รับการยอมรับคือ NSGA-II [16] แนวคิดคือการค้นหาพาเรโตฟร้อน โดยใช้การคัดสรรผลเฉลยที่ไม่ถูกครอบงำของ NSGA-II โดยที่เราทำการดัดแปลง NSGA-II โดยนำเสนอวิธีการสร้าง ประชากรสองกลุ่ม และทำการสร้างประชากรใหม่ด้วยครอสโอเวอร์ และการกลายพันธุ์ดังแสดงใน ภาพที่ 7 อัลกอริทึม 1 แสดงอัลกอริทึมหลักที่เสนอ โดยเก็บประชากรไว้สองกลุ่ม: กลุ่มประชากรคัดสรรที่เกิดจากการคัดสรรผลเฉลยที่ไม่ถูกครอบงำ และกลุ่มประชากรที่ไม่ได้คัดสรรเพื่อเพิ่มความหลากหลายในการค้นหา ประชากรที่ไม่ได้คัดสรรมีจำนวนเป็นสองเท่าของประชากรคัดสรร อัลกอริทึมทำงานดังนี้ ขั้นแรก จะสร้างประชากรใหม่ที่มีการผสมข้าม และการกลายพันธุ์ หรือเฉพาะการกลายพันธุ์จากการรวมประชากรทั้งสองเข้าด้วยกัน จากนั้นรวมประชากรที่สร้างขึ้นใหม่นี้เข้ากับประชากรชั้นยอดที่จัดไว้เพื่อสร้างกลุ่มใหม่ ต่อมา ดำเนินการคัดสรรผลเฉลยที่ไม่ถูกครอบงำกับกลุ่มใหม่นี้ ทำให้เกิดชุดอันดับ (F) สุดท้าย สร้างกลุ่มประชากรคัดสรรค์ที่เรียงลำดับใหม่จากชุดอันดับ *F* ตามใน NSGA-II ฟังก์ชัน *Fast-non-domination-sort* (R_t) และ *Crowding-distance-assignment* (F_i) จะเหมือนกับใน NSGA-II



ภาพที่ 6: ขั้นตอน NSGA_II ที่ถูกดัดแปลง

Algorithm 1: The proposed algorithm

1 $R_{temp} = P_t \cup Q_t$
2 Q_t+1 = offspring-generating(R_{temp})
3 $R_t = P_t \cup Q_{t+1}$
4 F = fast-non-domination-sort(R_t)
5 $P_t+1 = \emptyset$ and i = 1
6 **while** $|P_t+1| + |F_i| \leq N$ **do**
7 crowding-distance-assignment(F_i)
8 $P_t+1 = P_t+1 \cup F_i$
9 i = i+1
10 **end**
11 Sort(F_i, n)
12 $P_t+1 = P_t+1 \cup F_i[1:N - |P_t+1|]$
13 t = t+1

ภาพที่ 7: อัลกอริทึม 1 การค้นหาพาเรโตฟร้อนที่เราใช้

Algorithm 2: Offspring Generating

Input: The two populations P_t and Q_t, the probability of crossover operation p_c, the probability of mutation p_m, crossover control CR, and mutation control MU.

Output: The offspring population Q_{t+1}

```
1   R = P_t ∪ Q_t
2   Q_{t+1} = ∅
3   while | Q_{t+1} | < | Q_t | do
4   p_1 = Randomly select one individual from R
5   p_2 = Randomly select one individual from R
6   while p_1 == p_2 do
7       Repeat Line 5
8       end
9       If CR == true then
10          o_1, o_2 = crossover (p_1, p_2, p_c)
11      else
12          o_1 = p_1
13          o_2 = p_2
14      end
15      if MU == true then
16          if MU_CTRL = true then
17              o_1 = mutate (o_1, p_m) with the control gene
18              o_2 = mutate (o_2, p_m) with the control gene
19          else
20              o_1 = mutate (o_1, p_m) freely
21              o_2 = mutate (o_1, p_m) freely
22          end
23      end
24      Q_{t+1} = Q_{t+1} ∪ o_1 ∪ o_2
25  end
26  return Q_{t+1}
```

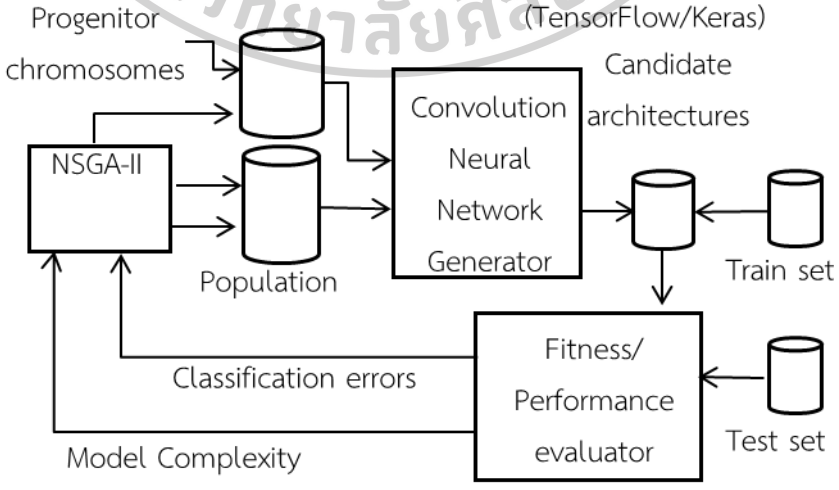
ภาพที่ 8: อัลกอริทึม 2 การผลิตกลุ่มประชากรรุ่นถัดไป

อัลกอริทึม 2 แสดง pseudo-code ของ ฟังก์ชัน *offspring- generating( )* (บรรทัดที่ 2) ใน อัลกอริทึม 1 ในอัลกอริทึม 2 มีชุดค่าผสมสามตัวของตัวดำเนินการทางพันธุกรรมที่ควบคุมโดยค่าคงที่ CR และ MU คงที่สำหรับครอสโอเวอร์และการกลายพันธุ์ตามลำดับ การครอสโอเวอร์ (บรรทัดที่ 9-14) เป็นวิธีการขยายแบบเสมือนจริงที่อธิบายไว้ในส่วนก่อนหน้า อัลกอริทึม 2 ดำเนินการกลายพันธุ์ โดยมีหรือไม่มียีนควบคุม (15-23)

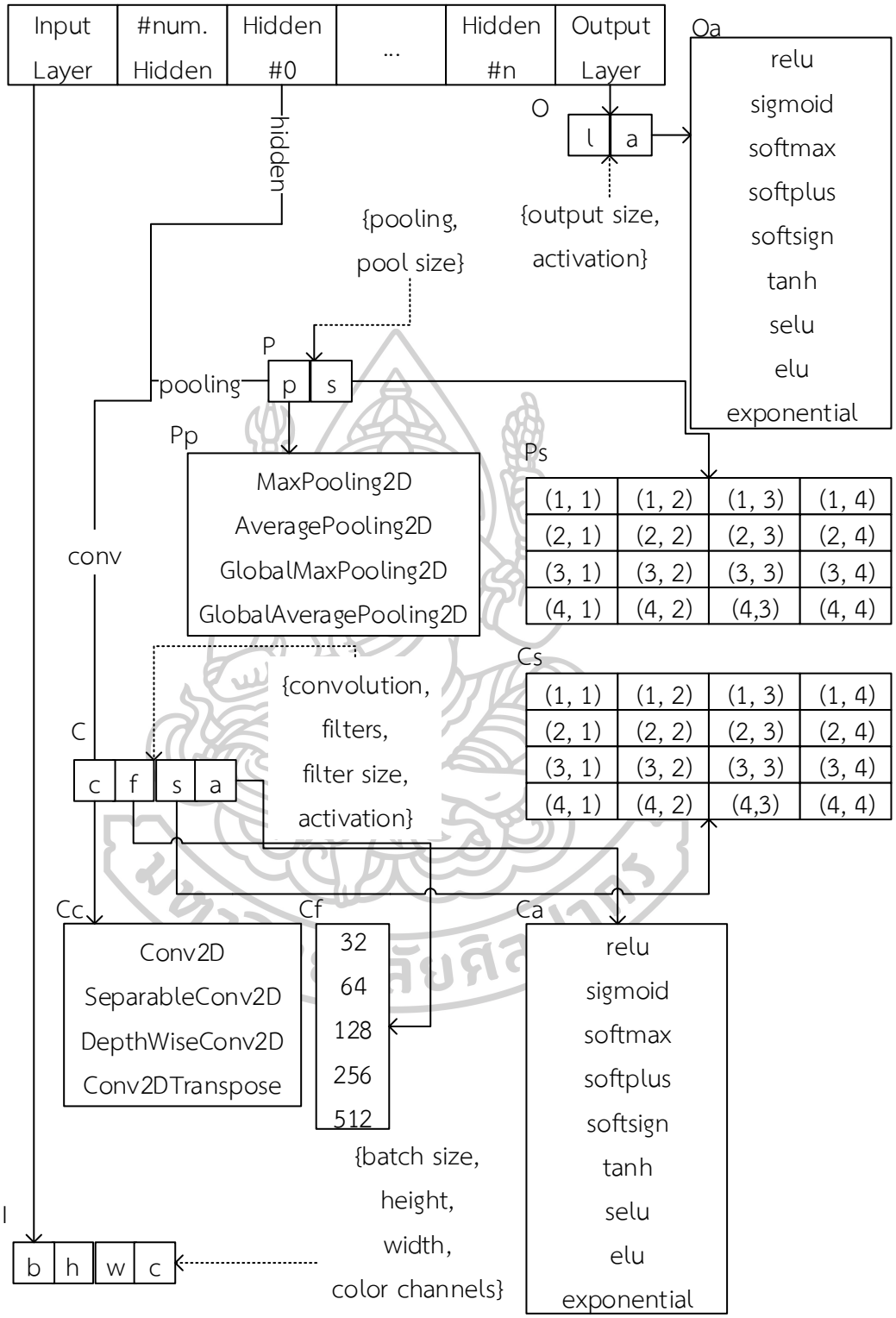
3.3.2 การวิวัฒนาการโครงข่ายประสาทเทียม

งานวิจัยฉบับนี้ใช้โครงข่ายประสาทเทียมที่มีต้นแบบมาจาก VGG ซึ่งเป็นโครงข่ายประสาทเทียมชนิดคอนโวลูชันที่มีโครงสร้างแบบป้อนไปข้างหน้า ไม่มีวนกลับ ไม่มีแตกแขนง ทำให้การเพิ่มลดชั้นเป็นไปได้โดยง่าย โครงข่ายประสาทเทียมจะขึ้นกับพันธุกรรมที่โดยเทคนิคเป็นดาต้าคลาสที่เขียนเป็นกึ่งโปรแกรมเชิงวัตถุ ไม่ได้นำไปเข้ารหัสเป็นบิท 0-1 จุดที่สำคัญของพันธุกรรมนี้คือขนาดจำนวนชั้นของโครงข่ายประสาทเทียมที่ได้ สามารถเพิ่มลดได้ และขนาดจำนวนการเชื่อมต่อ ของโครงข่ายประสาทเทียมนี้ ก็จะเป็นวัตถุประสงค์หนึ่งในการค้นหาด้วย และอีกวัตถุประสงค์หนึ่งคือความแม่นยำในการจำแนกภาพของโครงข่าย

ภาพที่ 9 แสดงภาพรวมของ วิธีการที่นำเสนอ กระบวนการทำงานดังต่อไปนี้ อันดับแรก เริ่มต้นด้วยการเริ่มต้นของ ประชากร คู่ ซึ่งเป็นโครโมโซมที่เป็นตัวแทนของ แบบจำลอง CNN ประการที่สอง โครโมโซมถูก สร้างขึ้น ไปยัง Tensorflow / Keras CNN โมเดล ประการที่สาม สำหรับการประเมินความเหมาะสม การฝึก และการทดสอบ จะดำเนินการโดยใช้ชุดข้อมูล CIFAR10 ผลลัพธ์ที่ได้ มี อัตราความผิดพลาดในการจำแนกประเภท และความซับซ้อนของแบบโครงข่ายประสาทเทียม ที่แสดงเป็นจำนวนพารามิเตอร์ ประการที่สี่ NSGA-II ปรับให้เหมาะสมและสร้างประชากรคู่เป็นโครโมโซม สุดท้าย ทำซ้ำขั้นตอนนี้จนกว่าจะถึงเกณฑ์การหยุด



ภาพที่ 9: กระบวนการโดยรวมของวิธีการที่เสนอ (TensorFlow/Keras)



ภาพที่ 10: โครโมโซมของโมเดล CNN

3.3.3 การแทนโครงข่ายประสาทเทียมด้วยโครโมโซม

ภาพที่ 10 แสดงการเข้ารหัสของโครงสร้างของโครงข่ายประสาทเทียมแบบคอนโวลูชันในรูปแบบโครโมโซม โดยที่โครโมโซมมีความยาวผันแปรได้ การเข้ารหัสเป็นแบบเลเยอร์ หรือการเข้ารหัสแบบบล็อกเชนซึ่งแบ่งเป็นชั้นๆ เชื่อมต่อกันไปในทิศทางเดียว สำหรับระดับบนสุดใน ภาพที่ 10 บล็อกแรกคือชั้นอินพุต ตามด้วยบล็อกที่สองที่ระบุจำนวนเลเยอร์ที่ซ่อนอยู่ ระหว่างบล็อกที่สองและบล็อกสุดท้ายที่เป็นชั้นเอาท์พุท มีชั้นซ่อน (Hidden layer) อยู่เป็นจำนวน n ซึ่งแต่ละชั้นสามารถเป็นชั้นคอนโวลูชันหรือชั้นพูลิ่งก็ได้ โดยไม่มีการข้ามหรือการเชื่อมต่อแบบขนาน การเข้ารหัสจะแสดงถึงโมเดลที่เหมือน VGG แต่รองรับประเภทและขนาดต่างๆ ของคอนโวลูชันและพูลิ่ง

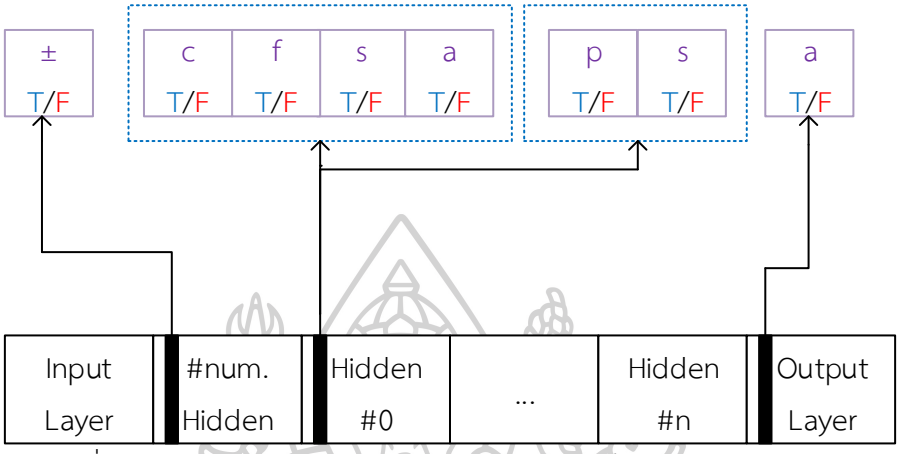
จากภาพที่ 10 ชั้นอินพุตประกอบด้วยพารามิเตอร์สี่ตัวที่กำหนดให้กับชุดข้อมูล: ขนาดแบทช์ ความสูง ความกว้าง และช่องสัญญาณ บล็อกที่สองคือจำนวนชั้นซ่อน ชั้นเอาต์พุตประกอบด้วยขนาดเอาต์พุตและประเภทของแอคทิเวชั่นฟังก์ชั่น สำหรับชั้นฮิดเด้นที่อยู่ตรงกลาง แต่ละชั้นฮิดเด้นสามารถเป็นคอนโวลูชันหรือพูลิ่ง โดยระบุด้วยตัวอักษร C และ H ตามลำดับ ชั้นคอนโวลูชันประกอบด้วย ประเภทคอนโวลูชัน จำนวนฟิลเตอร์ ขนาดฟิลเตอร์ และแอคทิเวชั่นฟังก์ชั่น ตารางทั้งสี่ที่มีป้ายกำกับด้วยตัวอักษร (Cc, Cf, Cs และ Ca) จะแสดงค่าทั้งสี่นี้โดยละเอียดดังแสดงใน ภาพที่ 10

สำหรับชั้นคอนโวลูชัน อักษรทั้งสี่ตัวจะอธิบายตารางทั้งสี่ดังนี้ ประเภทของคอนโวลูชัน (Cc) คือ Conv2D, SeparableConv2D, DepthWiseConv2D และ Conv2DTranspose ค่าที่เป็นไปได้สำหรับขนาดฟิลเตอร์ (Cs) คือ (1, 1) (1, 2) (1, 3) (1, 4) (2, 1) (2, 2) (2, 3) (2 , 4) (3, 1) (3, 2) (3, 3) (3, 4) (4, 1) (4, 2) (4, 3) (4, 4). ค่าที่เป็นไปได้สำหรับจำนวนฟิลเตอร์ (Cf) คือ 32, 64, 128, 256 และ 512 แอคทิเวชั่นฟังก์ชั่น (Ca) สามารถเป็น ReLU , Sigmoid, Softmax , Softplus , Softsign , Tanh, SELU และ ELU Exponential

ชั้นพูลิ่งประกอบด้วยประเภทชั้นพูลิ่งและขนาดพูล ซึ่งแสดงด้วยตัวอักษร Pp และ Ps ตามลำดับ ชั้นพูลิ่งที่เป็นไปได้ที่แสดงใน Pp คือ MaxPooling2D, AveragePooling2D, GlobalMaxPooling2D และ GlobalAveragePooling2D ขนาดพูลิ่งที่เป็นไปได้ซึ่งแสดงใน Ps คือ (1, 1) (1, 2) (1, 3) (1, 4) (2, 1) (2, 2) (2, 3) (2 , 4) (3, 1) (3, 2) (3, 3) (3, 4) (4, 1) (4, 2) (4, 3) (4, 4).

เราเสนอโครโมโซมควบคุมเพื่อควบคุมการกลายพันธุ์ ทำให้สามารถส่งต่อบล็อกที่ต้องการไปยังรุ่นต่อไปได้ อีกทางหนึ่ง เราจำกัดการกลายพันธุ์หากบางชั้นของโมเดลมีคุณสมบัติเหมาะสมอยู่แล้ว ภายในแต่ละชั้นฮิดเด้นภายในโครโมโซมใน ภาพที่ 10 โครโมโซมควบคุมจะถูกแทรกลงในแต่ละส่วนเพื่อควบคุมการกลายพันธุ์ ดังแสดงใน ภาพที่ 11 โครโมโซมควบคุมจะสั่งการการกลายพันธุ์ในแต่ละบล็อกย่อย โดยที่แต่ละค่าในโครโมโซมควบคุม คือ บูลีน (จริงหรือเท็จ) โครโมโซมควบคุมมีสองโหมด: เปิดหรือปิดโครโมโซมควบคุมทั้งหมดสำหรับทุกเลเยอร์ โหมดแรก เปิดใช้งาน การกลายพันธุ์

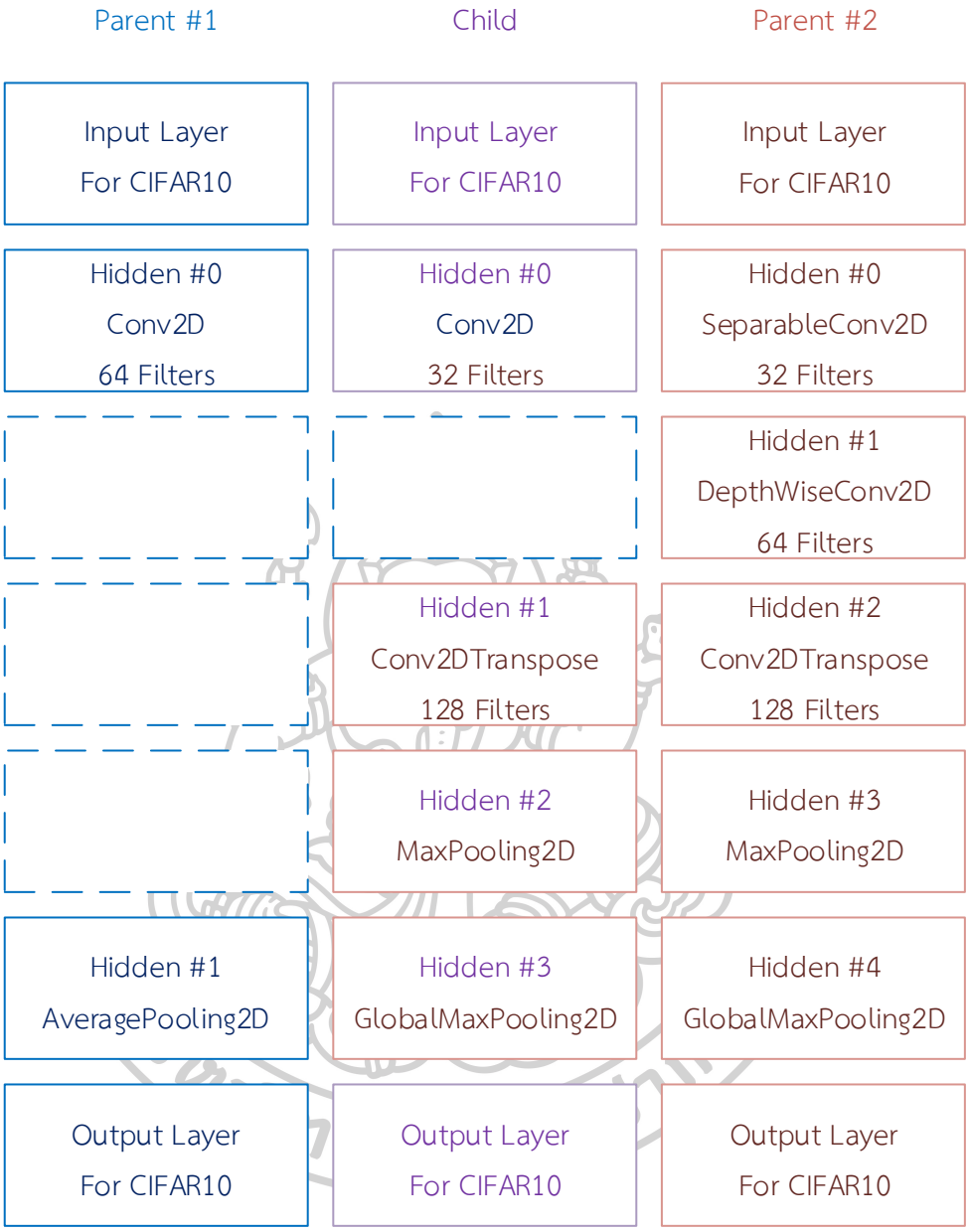
ของแต่ละบล็อกย่อยขึ้นอยู่กับค่าของโครโมโซมควบคุมในบล็อกนั้น ดังนั้นจึงช่วยให้บล็อกย่อยที่ดีบางส่วนสามารถอยู่รอดได้โดยไม่มีการเปลี่ยนแปลง โหมดที่สอง โครโมโซมควบคุมทั้งหมดถูกปิดใช้งาน ส่งผลให้การกลายพันธุ์ของบล็อกย่อยไม่ขึ้นกับโครโมโซมควบคุมในบล็อก



ภาพที่ 11: โครโมโซมควบคุมการกลายพันธุ์ภายในบล็อกย่อยของรูปผลเฉลย

3.3.4 การผสมพันธุ์

สำหรับการเข้ารหัสความยาวผันแปร ครอสโอเวอร์ต้องรองรับประชากรที่มีความยาวต่างกันสองตัวเพื่อสร้างลูกหลาน ครอสโอเวอร์แบบหนึ่งตำแหน่งที่ใช้กันทั่วไปใช้ได้กับประชากรที่มีความยาวเท่ากันสองตัว ดังนั้นเราจึงเสนอครอสโอเวอร์ใหม่เพื่อจัดการกับประชากรที่มีความยาวผันแปรได้สองคน ไม่เหมือนซันและคณะซึ่งใช้ความน่าจะเป็นในการส่งมอบตัวดำเนินการครอสโอเวอร์ที่มีประสิทธิภาพสำหรับบุคคลที่มีความยาวต่างกัน



ภาพที่ 12: ครอสโอเวอร์แบบขยายเสมือน

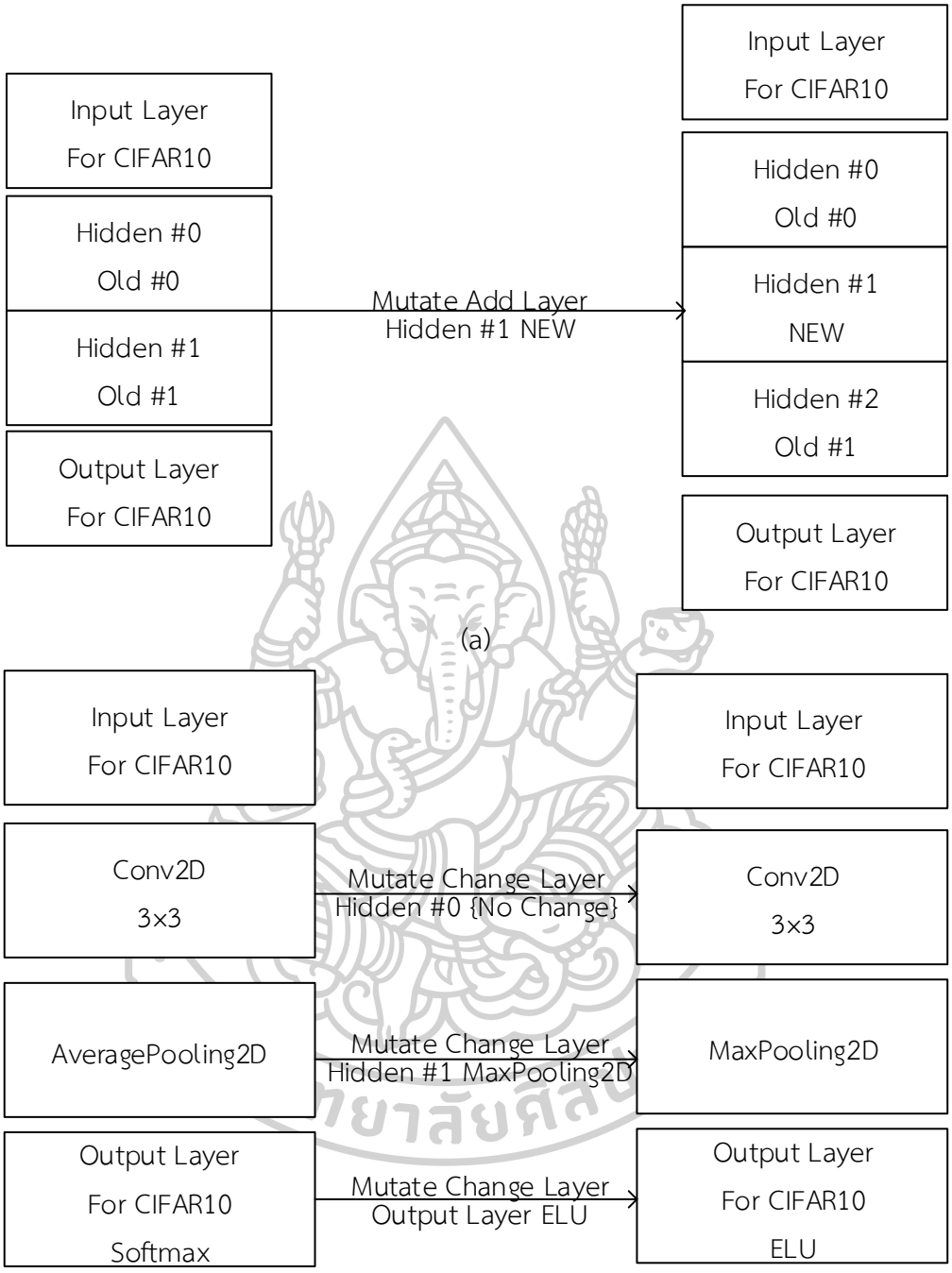
เราเพิ่มขั้นตอนพิเศษก่อนที่จะทำการครอสโอเวอร์แบบหนึ่งตำแหน่งเนื่องจากใช้งานได้กับบุคคลที่มีความยาวเท่ากันสองคนเท่านั้น วิธีการครอสโอเวอร์ที่เสนอจะสุ่มเลือกพ่อแม่สองคน เหมือนกับการเลือกทัวร์นาเมนต์ไบนารี่ เพื่อจัดการกับความยาวที่แตกต่างกันของพ่อแม่สองคน พ่อพันธุ์แม่พันธุ์ที่สั้นกว่าจะถูกขยายตามขนาดบล็อกเพื่อให้ตรงกับขนาดของพ่อแม่ที่ยาวกว่า ดังแสดงในภาพที่ 12

ครอสโอเวอร์ที่เสนอเริ่มต้นด้วยการสุ่มเลือกประชากรสองตัวจากกลุ่มประชากรปัจจุบัน จากนั้น ขยายบล็อกย่อยของประชากรตัวที่สั้นกว่าให้ตรงกับความยาวของประชากรตัวที่ยาวกว่า หลังจากนั้นให้สุ่มสร้างจุดตัดสำหรับพ่อแม่ที่ตอนนี้มีความยาวเท่ากันแล้ว ดังแสดงใน ภาพที่ 12 ครอสโอเวอร์เกิดขึ้นระหว่างบล็อกย่อยของพ่อแม่ที่มีดัชนีบล็อกเดียวกัน *i* ดังนั้นความยาวของลูกหลานจึงอยู่ระหว่างความยาวของพ่อแม่สองคน สมมติเช่นตัวอย่างใน ภาพที่ 12 ที่การผสมข้ามระหว่างตัวที่สั้นกว่าที่มีชั้นฮิดเด้น 2 และตัวที่ยาวกว่าที่มีชั้นฮิดเด้น 5 ได้ลูกหลานที่มีชั้นฮิดเด้น 3 ชั้น

3.3.5 การกลายพันธุ์

ดังที่แสดงในรูปที่ 3 เราเสนอให้ใช้ยีนควบคุมเพื่อรักษาบล็อคย่อยบางส่วนไม่ให้กลายพันธุ์ อย่างไรก็ตาม การกลายพันธุ์ของยีนควบคุมขึ้นอยู่กับอัตราการกลายพันธุ์ของยีนควบคุม Wistuba และคณะแสดงให้เห็นว่าการใช้เฉพาะการกลายพันธุ์แบบอนุรักษ์เท่านั้นเป็นตัวดำเนินทางพันธุกรรมก็สามารถได้ผลเฉลยที่พึงประสงค์ได้ แม้จะเริ่มต้นด้วยโครงข่ายประสาทเทียมต้นแบบที่มีชั้นคอนโวลูชันสองชั้น หลังจากการฝึกฝนเป็นระยะเวลานานขึ้น ระยะเวลาการฝึกที่สั้นลงได้จากการกลายพันธุ์ประชากรที่เป็นที่พึงประสงค์ที่สุด วิธีการที่เรานำเสนอแตกต่างจาก ของ Wistuba เนื่องจากเราใช้ยีนควบคุมเพื่อรักษาบล็อคการสร้างที่ดีโดยไม่อนุญาตให้บล็อคย่อยเหล่านั้นกลายพันธุ์อย่างอิสระและใช้ประชากรจำนวนมากขึ้น

การกลายพันธุ์มีสามประเภท: การเพิ่ม การลบ และการเปลี่ยนแปลงเลเยอร์ที่ซ่อนอยู่ การกลายพันธุ์นี้คล้ายกับของ Sun และ Xue การเพิ่มชั้นฮิดเด้นจะเลือกชั้นฮิดเด้นเป้าหมาย จากนั้นสร้างชั้นฮิดเด้นใหม่แบบสุ่ม ซึ่งอาจเป็นได้ทั้งชั้นคอนโวลูชันหรือชั้นพูลิ่ง จากนั้นใส่ไว้ข้างหน้าชั้นเป้าหมายที่จะถูกผลักกลับลงไปเป็นลำดับ การลบชั้นฮิดเด้น จะสุ่มเลือกชั้นฮิดเด้นแล้วลบออก การแก้ไขชั้นฮิดเด้นจะสุ่มค่าใหม่ที่เข้าได้กับชั้นฮิดเด้นเดิมและแทนที่ ดังแสดงใน ภาพที่ 13

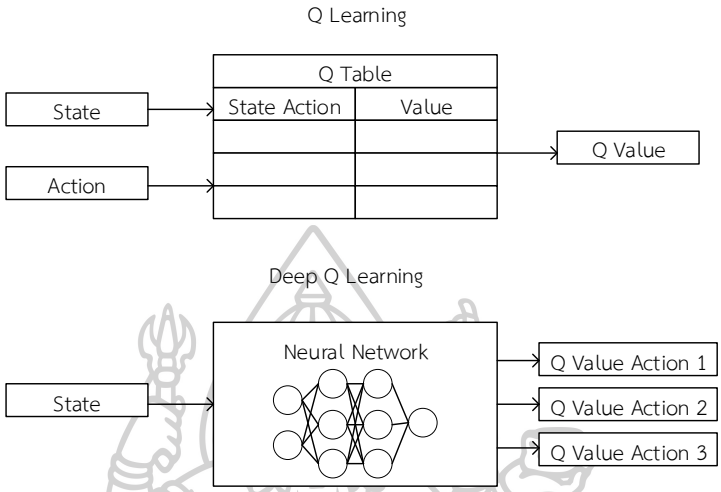


ภาพที่ 13: ตัวอย่างของการกลายพันธุ์โดย (a) การเพิ่ม (b) การเปลี่ยนแปลง

3.4 การออกแบบการทดลอง

ทำการออกแบบการทดลองทั้งในส่วนการใช้งานอัลกอริทึมในการสร้างโครงข่ายประสาทเทียม และการใช้งานโครงข่ายประสาทเทียมที่ได้การวิวัฒนาการกับปัญหาเกมส์ ซึ่งการออกแบบโครงข่ายประสาทเทียมด้วยนิวโรอีโวลูชั่นนั้นจำเป็นต้องระบุลักษณะของโครงข่ายประสาทเทียม ซึ่งระบุได้โดย

ทำการออกแบบย้อยหลังจากการใช้งานกับปัญหาเกมส์ ซึ่งปัญหาการใช้งานโครงข่ายประสาทเทียมกับปัญหาเกมส์จำเป็นระบุเกมส์ที่เราใช้ในการทดลอง และวิธีการที่ใช้ในเกมส์ในการสร้างเอเย่นต์ด้วย Deep Q Learning ซึ่งอธิบายดังหัวข้อต่อไปนี้

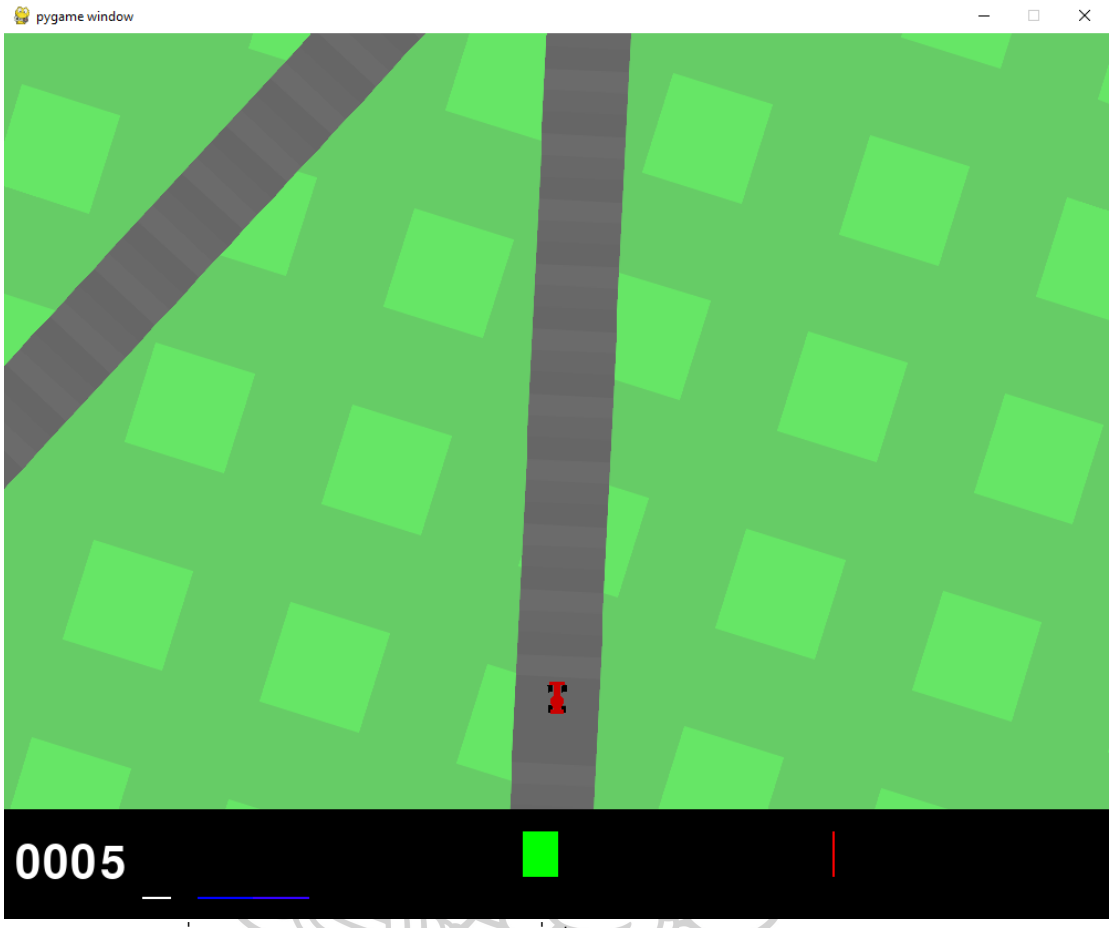


ภาพที่ 14: แสดงความแตกต่างของ Q Learning (บน) และ Deep Q Learning (ล่าง)

3.4.1 Deep Q Learning

เป็นวิธีการเรียนรู้แบบเสริมแรงประภทหนึ่ง [20] ที่พัฒนาต่อยอดจาก Q Learning แต่มีการใช้งานโครงข่ายประสาทเทียมเชิงลึกร่วมด้วย แสดงดังภาพที่ 14 (บน) เอเย่นต์จะเรียนรู้สภาพสิ่งแวดล้อมแล้วให้น้ำหนักเป็นตารางคะแนน โดยตารางคะแนนนี้ ที่ถูกเรียกว่า Q Table จะถูกใช้ประกอบการตัดสินใจในการปฏิบัติในสิ่งแวดล้อม คือจะให้น้ำหนักสภาพสิ่งแวดล้อมที่ให้คะแนนสูงระยะยาว แล้วพยายามปฏิบัติไปจนถึงสภาพเหล่านั้นให้ได้ ใน Deep Q Learning ภาพที่ 14 (ล่าง) เอเย่นต์จะใช้โครงข่ายประสาทเทียมเป็นฟังก์ชั่นประมาณการของตารางคะแนน โดยให้โครงข่ายประสาทเทียมเรียนรู้สภาพสิ่งแวดล้อม แล้วให้เอเย่นต์ใช้โครงข่ายประสาทเทียมแทนตารางคะแนนเวลาประกอบการตัดสินใจ ทฤษฎีของ Deep Q Learning มีความซับซ้อน และไม่ได้อธิบายโดยละเอียดในหัวข้อนี้ ซึ่งสามารถหาข้อมูลได้ใน [2, 20]

3.4.2 การทำงานของเกมส์แข่งรถ CarRacing



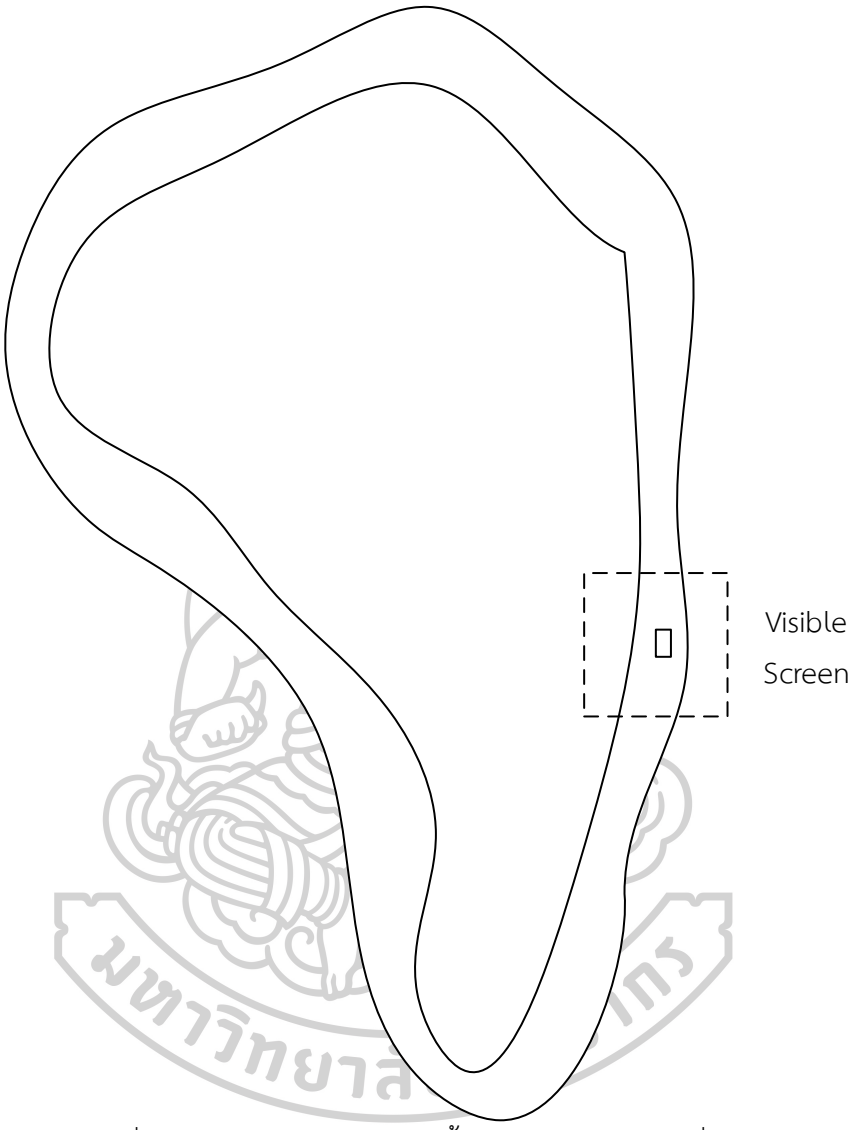
ภาพที่ 15: หน้าตาเกมส์ CarRacing ที่เป็นเกมส์รถแข่ง 1 ผู้เล่นง่ายๆ จาก gym

ภาพที่ 15 เกมส์ CarRacing เป็นเกมส์ในภาษา Python ที่มีผู้เล่นคนเดียว และจับเวลาในการวิ่งจนครบรอบสนาม โดยมีสูตรการคำนวณคะแนนรวมคือ

คะแนนรวม = + (1000/จำนวนช่องพื้นถนนที่รถต้องเหยียบเพื่อชนะ) – (0.1×จำนวนเฟรมที่ต้องใช้เพื่อชนะ)

แต่ในเวลาคำนวณคะแนนจริง สังเกต จากในภาพที่ 15 คะแนนเริ่มต้นจะไม่เท่ากับศูนย์ คือเกินศูนย์มาเล็กน้อย หรือเพียงไม่กี่คะแนน และไม่ใช่เริ่มจาก 1000 คะแนน ซึ่งมาจากจำนวนช่องล่างซ้ายของเกมส์ที่รถเหยียบอยู่ตั้งแต่ตอนต้นเกมส์ และเมื่อเริ่มเกมส์ คะแนนจะลด 0.1 คะแนนทุกๆ เฟรมที่ผู้เล่นเกมส์เห็น (ถ้ามนุษย์เล่น 1 วินาที คือ 50 เฟรม เท่ากับเสียคะแนนวินาทีละ 5 คะแนน) และเพิ่ม 1000 คะแนนต่อจำนวนช่องที่ต้องเหยียบเพื่อชนะ ทุกๆ ช่องที่เหยียบ ถ้าใช้สนามดั้งเดิม สนามจะมีความยาวเท่ากับเส้นรอบวงของวงกลมรัศมี 900 พิกเซล คิดเป็นเส้นรอบวงประมาณ 5655 พิกเซล และมีความกว้าง 40 พิกเซล ซึ่งแปลว่าช่องที่ต้องเหยียบเพื่อให้ได้ 1 คะแนน จะมีขนาด

ประมาณ 6×40 พิกเซล (บิดพลิ้วได้บ้างตอนทางโค้ง) เรียงกันเป็นจำนวนทั้งสิ้น 943 ช่องถนน ซึ่งเมื่อใช้สูตร 1000 คะแนนต่อช่องฯ ก็จะได้ค่าเป็นช่องละประมาณ 1.06 คะแนน



ภาพที่ 16 ภาพเกมส์ CarRacing ทั้งสนาม เทียบกับส่วนที่ผู้เล่นเห็น

ภาพที่ 16 แสดงถึงมุมมองของผู้เล่นที่มองเห็นตัวเกมส์ ตัวเกมส์ CarRacing จะตัดสนามรอบตัวรถ ขนาด 600×400 พิกเซล มาเป็นภาพวีดีโอ แล้วสเกลเป็นขนาดใหม่ (1000×800 สำหรับผู้เล่นมนุษย์ 96×96 สำหรับผู้เล่นที่เป็นปัญญาประดิษฐ์ รวมทั้งโครงข่ายประสาทเทียมด้วย)

ผู้เล่นที่เป็นมนุษย์จะควบคุมรถโดยใช้ปุ่มลูกศร (KEY_UP = คันเร่ง KEY_DOWN = เบรก KEY_LEFT = เลี้ยวซ้าย KEY_RIGHT = เลี้ยวขวา) ส่วนคอมพิวเตอร์ (โดยเฉพาะโครงข่ายประสาทเทียม) จะมีรูปแบบการควบคุมได้สองแบบ แบบแยกส่วน (0 = ไม่ทำอะไรเลย 1 = เลี้ยวซ้าย 2 = เลี้ยวขวา 3 = คันเร่ง 4 = เบรก) ซึ่งมีอัตราการควบคุมคล้ายปุ่มลูกศรของมนุษย์ และแบบต่อเนื่อง (0

= พวงมาลัย (-1 = เลี้ยวซ้าย +1 = เลี้ยวขวา) 1 = คันเร่ง 2 = เบรก) ซึ่งมีความสามารถในการควบคุมได้ละเอียดอ่อนกว่ามนุษย์ (โดยเฉพาะในส่วนพวงมาลัย)

ภาพที่ 17: ต่อให้หันหัวกลับ 180 องศา แบบนี้ แล้วเข้าเส้นชัยจากทิศตรงข้าม ก็ชนะเช่นกัน

ภาพที่ 17 แสดงให้เห็นถึงสิ่งที่จะเกิดถ้าผู้เล่นตัดสินใจเข้าเส้นชัยในทิศตรงกันข้าม แม้การตีวงจนถึงขั้นหลุดลู่วิ่งลงไปบนพื้นหญ้าเช่นนี้จะทำให้เสียคะแนน (เพราะเสียเวลาไปเหยียบช่องที่ไม่ใช่พื้นถนน) แต่ความจริงมีอยู่ว่า ทิศทางในการวิ่งครบรอบไม่มีผล รถจะวิ่งครบรอบด้วยการเดินหรือหน้าถอยหลังผู้เล่น หรือเข้าเส้นชัยจากทางด้านหน้าหรือด้านหลัง ก็ได้ผลเท่ากัน เพราะเกมส์จะเช็คแค่ว่าเราเหยียบพื้นถนนในลู่วิ่งไปถึงเกณฑ์ที่กำหนดเป็นเกณฑ์ชนะ โดยไม่เช็คทิศทางการวิ่ง และไม่เช็คคะแนนที่เสียไปจากการเหยียบพื้นหญ้าด้วย

ภาพที่ 18: ต่อให้มีคะแนนเยอะแค่ไหนแต่ถ้าชนขอบด้านอกสนามแบบนี้คือรถพังแพ้ทันที

ภาพที่ 18 แสดงถึงเงื่อนไขการแพ้ที่สำคัญอย่างหนึ่ง คือการวิ่งออกนอกตัวสนามที่เกมส์ทำขึ้น (สังเกตเขตสีดำสนิท) ซึ่งจะทำรถให้รถพังและถูกปรับแพ้ทันที และถ้าผู้เล่นถูกปรับแพ้ด้วยวิธีนี้ คะแนนตอนจบเกมจะถูกหักออก 100 คะแนน สำหรับเกมส์ CarRacing ต้นฉบับของค่าย Box2D นี่เป็นกรณีเดียวที่จะนับว่าผู้เล่นแพ้โดยแท้จริง (ในโค๊ดเกมส์ระบุว่า เกมส์จบด้วยรถออกนอกสนาม ตัวยกเกมส์จะถูก terminate ผิดกับเกมส์จบด้วยกรณีอื่น (เช่น รถวิ่งครบรอบเข้าเส้นชัย) ที่จะถูก truncate)

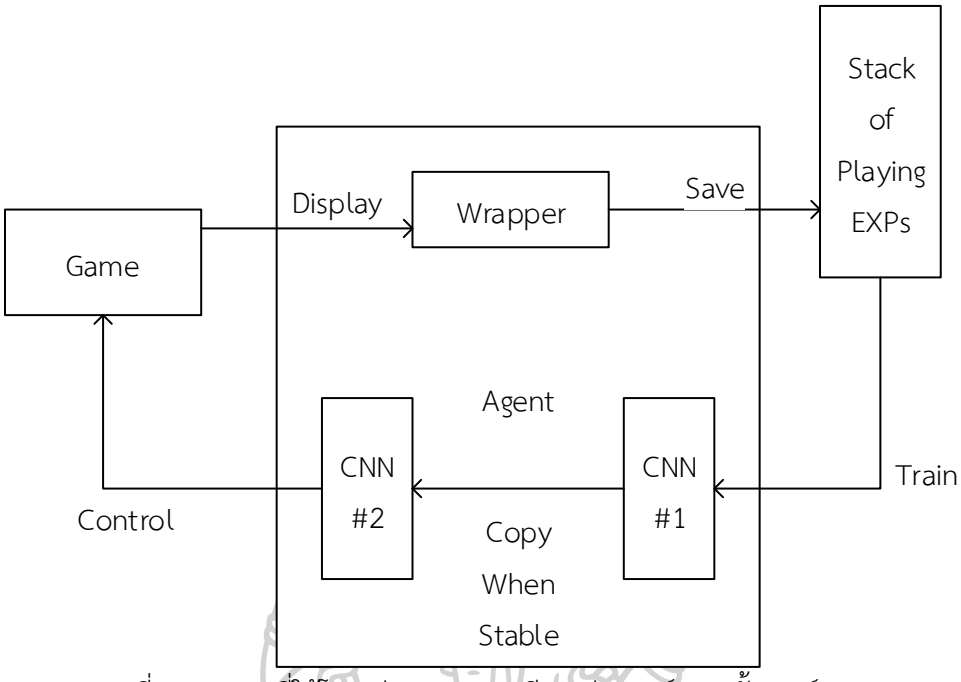
ข้อสังเกตที่สำคัญของจุดนี้ คือทางค่าย Box2D ได้กำหนดมาตรฐานเอาไว้ว่า ถ้าค่ายอื่นที่นำโค๊ดของเกมส์ CarRacing ไปดัดแปลง ตัดสินใจใส่ให้มีการแพ้เพราะหมดเวลา ให้ถือการแพ้เพราะหมดเวลานี้ เป็นกรณีเดียวกับการชนะด้วยการเข้าเส้นชัย (คือให้ truncate ยกเกมส์ที่จบเพราะหมดเวลา และไม่ปรับคะแนน ซึ่งเป็นการปฏิบัติแบบเดียวกับเมื่อยกเกมส์จบเพราะรถวิ่งครบรอบเข้าเส้นชัย แทนที่จะให้ terminate เหมือนกรณีรถวิ่งออกนอกสนามนี้)

ภาพที่ 19: ถ้าอยู่เฉยๆ ไม่ทำอะไร เวลาผ่านไปคะแนนก็จะติดลบไปเรื่อยๆ จนในที่สุดก็แพ้อยู่ดี

ภาพที่ 19 แสดงให้เห็นว่าจะเกิดอะไรขึ้นกับรถที่จอดอยู่เฉยๆ ในความเป็นจริงแล้ว เกมส์ CarRacing ต้นฉบับ ไม่มีเงื่อนไขแพ้ชนะที่คะแนน (หรืออย่างน้อย ก็เท่าที่เราพยายามอ่านโค้ดเกมส์ดูแล้ว) แต่ในทางปฏิบัติ ถ้ารถไม่ทำอะไรเลยจนคะแนนติดลบถึงขนาด (จากที่เราทดสอบดู ประมาณ -100 คะแนน) เกมส์ก็จะตัดให้แพ้ทันที (ซึ่งนี่อาจเป็นส่วนที่ค่าย gym หรือ Gymnasium ทำเพิ่ม ไม่เกี่ยวกับเกมส์ CarRacing เดิม ที่เป็นเกมส์ของค่าย Box2D)

การแพ้ด้วยวิธีนี้ถือเป็นการแพ้เพราะหมดเวลาชนิดหนึ่ง (ซึ่งก็คือตัวยกเกมส์จะถูก truncate แทนที่จะถูก terminate) ซึ่งโดยทฤษฎีแล้วตัวเกมส์ไม่ได้นับว่าผู้เล่นแพ้ (คือยกเกมส์ไม่ได้จบเพราะถูก terminate) แต่ในทางปฏิบัติ เนื่องจากเราใช้ Deep Q Learning ที่ค่าคะแนนมีความสำคัญต่อการฝึกสอนโครงข่ายประสาทเทียมในตัวเอเย่นต์ บางครั้งการแพ้เพราะหมดเวลาเช่นนี้ จะถูกเอเย่นต์นับเป็นประสบการณ์ไม่พึงประสงค์ ยิ่งกว่ากรณีแพ้เพราะรถออกนอกสนาม (ซึ่งเป็นไปได้ว่าจะได้คะแนนสูงถ้าคะแนนเดิมก่อนรถออกนอกสนามสูง)

3.4.3 การใช้งานโครงข่ายประสาทเทียมแบบคอนโวลูชันกับเกมส์แข่งรถ



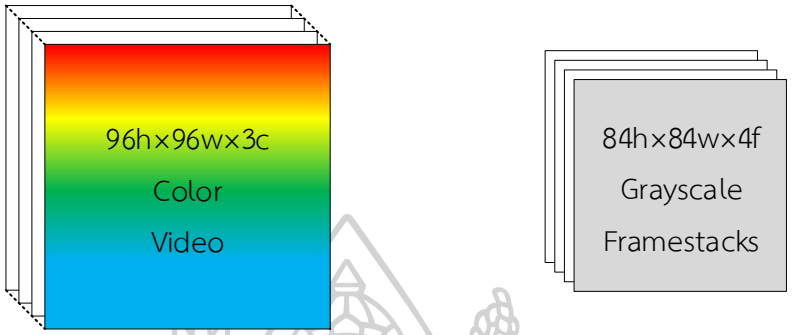
ภาพที่ 20: ระบบที่ใช้โครงข่ายประสาทเทียมเล่นเกมส์ รวมทั้งเกมส์ CarRacing

ในปัจจุบันมีการใช้โครงข่ายประสาทเทียมแบบ CNN เล่นเกมส์นี้ โดยให้รับอินพุทเป็นรูปตอนเกมส์กำลังเล่น และเอาท์พุทไปควบคุมรถในเกมส์ที่กำลังเล่น ซึ่งเป็นเหตุผลที่มีคนใช้ CNN กับเกมส์นี้ เนื่องจาก CNN ถูกออกแบบมาให้สกัดฟีเจอร์จากรูปภาพ ทำให้เข้าภาพสเมือนมนุษย์กำลังเล่นเกมส์

ภาพที่ 20 เป็นภาพที่แสดงให้เห็นลำดับขั้นตอน เริ่มจากตัวผู้เล่นปัญญาประดิษฐ์บันทึกประสบการเกมส์เก็บไว้เป็นชุดของภาพต่อเนื่อง หรือเฟรมสแตค (Frame stack) ปริมาณมาก (รายละเอียดดัง ภาพที่ 21) แล้วสุ่มภาพหรือเฟรมจำนวนหนึ่ง เรียกว่าเฟรมสแตค ออกมาฝึกโครงข่ายประสาทเทียมที่ได้ออกแบบโครงสร้างจากนิวโรอีโวลูชั่นในขั้นก่อนหน้า ซึ่งโครงข่ายประสาทเทียมตรงนี้จะมีสองตัว ตัวแรกเรียนจากประสบการณ์ที่บันทึกไว้ ตัวที่สองเล่นเกมส์ จนตัวแรกมีเสถียรภาพ จึงค่อยให้คัดลอกมาให้ตัวที่สองจากนั้นก็ให้โครงข่ายประสาทเทียมในตัวไปเล่นเกมส์ ได้เป็นประสบการณ์เกมส์ออกมา วนไปเรื่อยๆ เป็นวัฏจักร

เทคนิคนี้เป็นแขนงย่อยหนึ่งของ Q Learning ที่เรียกว่า Deep Q Learning เป็นการใช้โครงข่ายประสาทเทียม (CNN #2 ใน ภาพที่ 20) แทนที่ตารางค่าประสบการณ์ (Stack of Playing EXPs ใน ภาพที่ 20) ในการตัดสินใจเล่นเกมส์ นั่นเพราะตารางค่าประสบการณ์ที่มาจากวีดีโอเกมส์เวลาจริงเช่นเกมส์ CarRacing จะมีขนาดใหญ่มาก แต่โครงข่ายประสาทเทียมที่เรียนจากตารางค่าประสบการณ์จนสามารถคาดเดาค่าในตารางค่าประสบการณ์ สามารถทำหน้าที่แทนตารางค่า

ประสบการณ์ได้ และมีขนาดเล็กกว่าตารางค่าประสบการณ์มาก และการใช้โครงข่ายประสาทเทียมสองตัว (CNN #1 และ CNN #2 ใน ภาพที่ 20) ก็เป็นเทคนิคในการรักษาเสถียรภาพอย่างหนึ่ง เพื่อให้โครงข่ายประสาทเทียมไม่ถูกเปลี่ยนไปเปลี่ยนมาตามสิ่งแวดล้อมเกมส์บ่อยเกินไป จนเสียโอกาสสำรวจสิ่งแวดล้อมเกมส์



ภาพที่ 21: (ซ้าย) วีดีโอเดิมที่เกมส์ CarRacing ให้ผู้เล่นปัญญาประดิษฐ์เห็น (ขวา) เฟรมสแตคที่ผู้เล่นปัญญาประดิษฐ์แปลงจากวีดีโอเดิมก่อนบันทึกไว้เพื่อให้โครงข่ายประสาทเทียมในตัวเองใช้ฝึก

ภาพที่ 21 เป็นการเปลี่ยนจากวีดีโอเดิมของวีดีโอเกมส์ (ในกรณีนี้คือของ CarRacing แต่ความจริง สามารถใช้กับวีดีโอเกมส์โดยทั่วไป) เป็นเฟรมสแตค เพื่อใช้ฝึกโครงข่ายประสาทเทียม โดยเฟรมสแตคจะเป็นภาพขาวดำที่ถูกนำมาซ้อนกันหลายเฟรมโดยใช้ช่องสีเดิมเป็นช่องเฟรม สมมติเช่นในภาพที่ 21 (ขวา) เป็นเฟรมสแตคขนาด 84h×84w×4f

3.5 ทำการทดลอง เก็บรวบรวมข้อมูล และวิเคราะห์ผล

การทดลองแบ่งเป็น 2 ส่วนคือ

1. การใช้งานนิวโรอีโวลูชั่นเพื่อสร้างโครงข่ายประสาทเทียมแบบคอนโวลูชัน
2. ใช้งานโครงข่ายประสาทเทียมที่ได้จากข้อที่ 1 นำมาเล่นเกมส์ โดยจะเรียนรู้แบบออนไลน์ในขณะเล่นเกมส์

ในขั้นที่หนึ่งจะเริ่มจากการทดลองสร้างยีนพูลเพื่อให้ไปค้นหาโครงข่ายประสาทเทียมที่เหมาะสมโดยใช้ชุดข้อมูล CIFAR10 เป็นตัวตั้งต้น หลังจากได้โครงสร้างโครงข่ายประสาทเทียมที่เหมาะสมแล้ว ก็จะนำมาทดสอบกับเกมส์ในขั้นที่สอง โดยเกมส์เป้าหมายในปัจจุบันเป็นเกมส์ CarRacing ที่เป็นเกมส์รถแข่งสำเร็จรูปจากชุด Gymnasium ซึ่งมีผู้รู้ใช้ Deep Q Learning ในการเล่นเกมส์นี้จนสำเร็จมาแล้ว (และมีโปรแกรมสำเร็จรูปบน GitHub) สิ่งที่เราจะทำ คือใช้โครงข่ายประสาทเทียมที่ค้นหามาซึ่งอาจเป็น 1 ตัว หรืออย่างมาก 2 – 3 ตัว เพื่อแทนที่โครงข่ายประสาทเทียมเดิม

3.6 สรุปผลการดำเนินงานวิจัยและจัดทำรายงานวิทยานิพนธ์

การสรุปผลของการดำเนินงานวิจัยและจัดทำรายงานวิทยานิพนธ์ถือเป็นขั้นตอนสุดท้ายของการวิจัยนี้ โดยจะนำผลลัพธ์ที่ได้จากการทดลองทั้ง 2 ขั้นตอน มาวิเคราะห์ผลลัพธ์ และเขียนรายงานวิทยานิพนธ์

บทที่ 4
ผลการดำเนินงาน

ในบทนี้จะประกอบไปด้วยผลการทดสอบประสิทธิภาพอัลกอริทึมที่ออกแบบและสร้างขึ้น (กับ CIFAR10) และผลการทดลองใช้กับปัญหาเกมส์ (CarRacing)

4.1 ผลการวิวัฒนาการโครงข่ายประสาทเทียมแบบคอนโวลูชัน

ตารางที่ 1: คอนโทรลยีนมีผลต่อการกลายพันธุ์เท่านั้น การผสมพันธุ์ล้วนๆ คอนโทรลยีนไม่มีผล

	ไม่มีคอนโทรลยีน	**มีคอนโทรลยีน**
ผสมพันธุ์	Test 2	
ผสมพันธุ์ + กลายพันธุ์	Test 4	Test 0
กลายพันธุ์	Test 5	Test 1

ตารางที่ 1 เป็นส่วนที่อธิบายว่าการทดสอบไหนมีความหมายอะไรโดยคร่าวๆ ซึ่งผู้อ่านจะสังเกตว่าไม่มีการทดสอบที่ 3 เพราะว่าคอนโทรลยีนที่เราใช้ควบคุมแต่การกลายพันธุ์ ทำให้การทดสอบที่ 2 และ 3 ซึ่งเป็นการผสมพันธุ์โดยไม่มีการกลายพันธุ์ ไม่ได้รับผลใดๆ จากคอนโทรลยีน

ตารางที่ 2: ความหมายของผลลัพธ์การทดลองทั้ง 6

การทดสอบ	**รหัสย่อ**	**ความหมาย**
Test 0	CR+MU+Ctrl	การใช้คอนโทรลยีนเพื่อควบคุมการผสมพันธุ์และกลายพันธุ์
Test 1	MU+Ctrl	การใช้คอนโทรลยีนเพื่อควบคุมการกลายพันธุ์
Test 2	CR	การใช้การผสมพันธุ์
Test 3		
Test 4	CR+MU	การไม่ใช้คอนโทรลยีนควบคุมการผสมพันธุ์และกลายพันธุ์
Test 5	MU	การไม่ใช้คอนโทรลยีนควบคุมการกลายพันธุ์

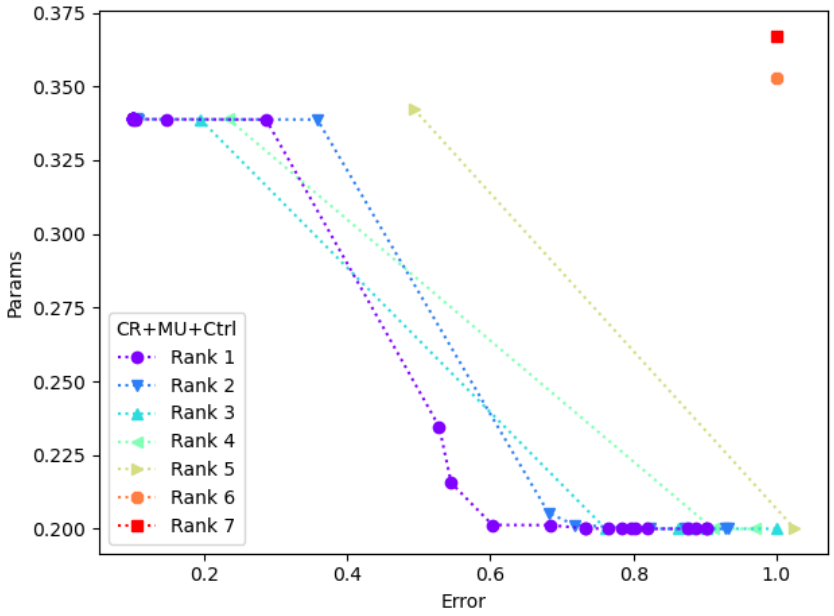
ตารางที่ 2 ได้อธิบายในสิ่งเดียวกับที่ ตารางที่ 1 อธิบาย นั่นคือความหมายการทดสอบ ตั้งแต่ที่ 0 ถึงที่ 5 แต่ในกรณีนี้เราใส่การทดสอบที่ 3 ลงไป เพื่อประกอบการอธิบายว่าการทดสอบที่ 2 และ 3 เป็นการทดสอบที่ผลลัพธ์ไม่ต่างกันใดๆ

กราฟข้างล่างต่อไปนี้ แกนตั้งเป็นคะแนนขนาดโครงข่ายประสาทเทียม (ยิ่งโครงข่ายเล็กยิ่งเป็นที่พึงประสงค์) แกนนอนเป็นคะแนนความแม่นยำ (ยิ่งแม่นยำมากยิ่งเป็นที่พึงประสงค์) โดยคะแนนพึงประสงค์ดีที่สุดคือ 0 และคะแนนตัวเลขมากขึ้นหมายถึงเป็นที่พึงประสงค์น้อยลง

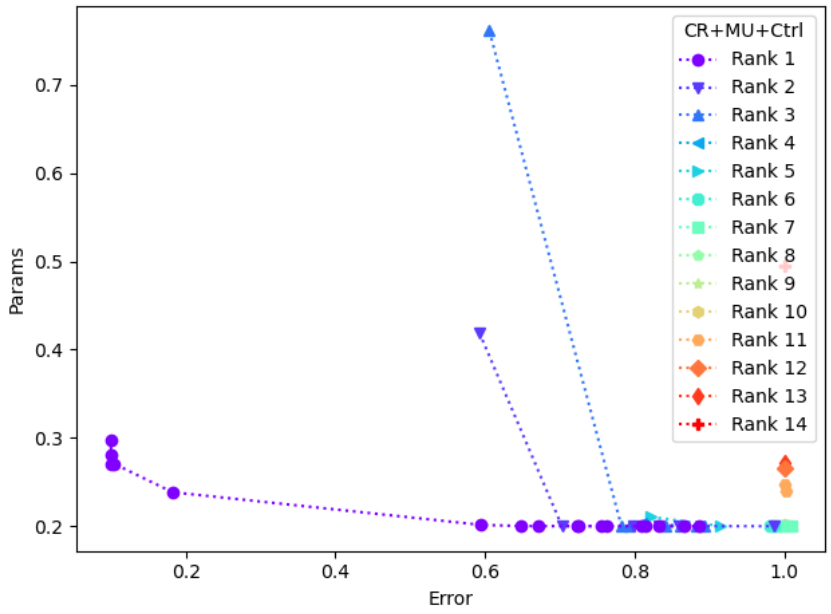
ในแกนตั้ง (แกน Y, Params) ค่าคะแนนขนาดโครงข่ายประสาทเทียมมาจากการนำขนาดของโครงข่ายประสาทเทียมแต่ละตัว (ซึ่งคือจำนวนพารามิเตอร์ในโครงข่ายประสาทเทียมนั้น) ตั้ง แล้วหารด้วยขนาดโครงข่ายประสาทเทียมที่ใหญ่ที่สุดของกลุ่มประชากร (ดังนั้นค่าจะมีระยะในช่วง 0 – 1) ดังนั้น คะแนนขนาดโครงข่ายประสาทเทียมที่เข้าใกล้ 0 จึงหมายถึงขนาดของโครงข่ายประสาทเทียมที่เล็ก (ซึ่งเป็นที่พึงประสงค์ ขนาดของโครงข่ายประสาทเทียมที่เล็กย่อมใช้ทรัพยากรคอมพิวเตอร์น้อยลง)

ในแกนนอน (แกน X, Error) ค่าคะแนนความแม่นยำนี้ ความจริงคืออัตราการทำนายผิดพลาดเมื่อทำนายชุดข้อมูล CIFAR10 หลังฝึกโครงข่ายประสาทเทียมกับ CIFAR10 แล้ว ซึ่งเดิมเป็นร้อยละ (ดังนั้นค่าจะมีระยะในช่วง 0 – 1) ดังนั้น คะแนนความแม่นที่เข้าใกล้ 0 จึงหมายถึงความแม่นยำของโครงข่ายประสาทเทียมหลังฝึกที่สูงขึ้น (ซึ่งเป็นที่พึงประสงค์ โครงข่ายประสาทเทียมที่ทำนายแม่นยำย่อมแสดงถึงความสามารถในการเรียนที่สูง)

คำว่า คะแนนรวม คะแนนเฉลี่ย คะแนนกลุ่ม คะแนนดิบ (หรือเทอมที่คล้ายกันอื่นๆ) ของผลเฉลยใดๆ หมายถึงค่าระยะแมนฮัตตันระหว่างตำแหน่งของผลเฉลยนั้นๆ ในกราฟ ไปจนถึงจุดกำเนิดของกราฟนั้นๆ ซึ่ง เนื่องจากธรรมชาติของแกนตั้งและแกนนอนของกราฟเหล่านี้ (ดังที่ได้อธิบายไปในสองย่อหน้าก่อน) ผลเฉลยที่มีระยะแมนฮัตตันระหว่างตัวเองกับจุดกำเนิดกราฟเข้าใกล้ 0 จึงหมายถึงผลเฉลยที่เป็นที่พึงประสงค์ (เพราะทั้งแกนตั้งแกนนอน เข้าใกล้ 0 หมายถึงเป็นที่พึงประสงค์ ทั้งสองแกน)

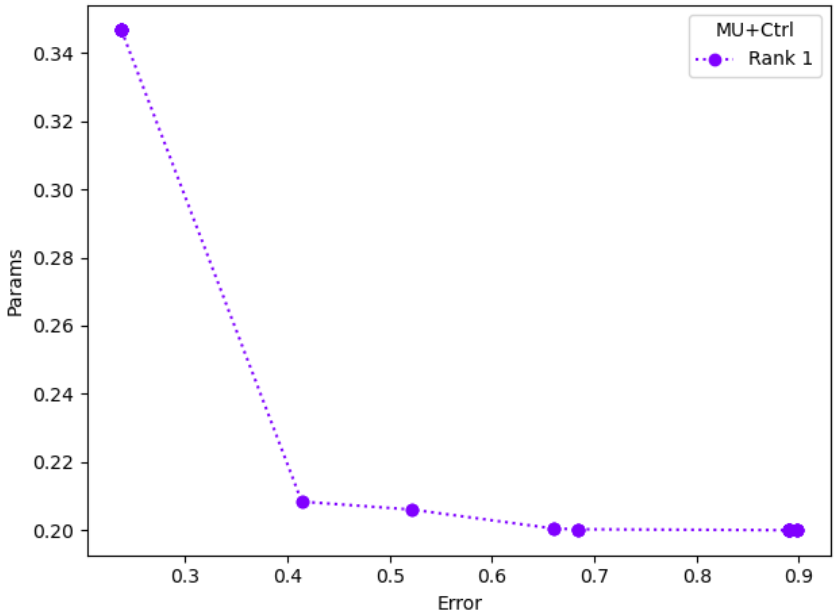


ภาพที่ 22: ผลลัพธ์ของการใช้คอนโทรลยีนเพื่อควบคุมการผสมพันธุ์และกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายต่ำ

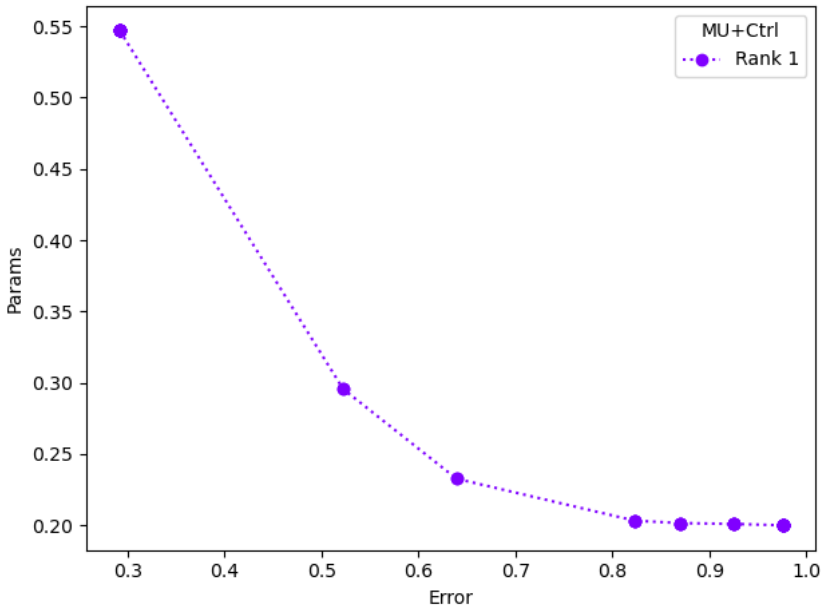


ภาพที่ 23: ผลลัพธ์ของการใช้คอนโทรลยีนเพื่อควบคุมการผสมพันธุ์และกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายสูง

ในภาพที่ 22 และภาพที่ 23 เป็นการทดสอบที่ 0 (CR+MU+Ctrl) ซึ่งจะสามารถเห็นได้ชัดเจนว่า จำนวนประชากรตั้งต้นที่เพิ่มขึ้น ทำให้ผลลัพธ์คะแนนรวมของกลุ่มผลเฉลยขั้น 1 ดีขึ้นอย่างมีนัยยะสำคัญ (สังเกตจากเส้นสีม่วงที่ใกล้จุดศูนย์มากขึ้น)



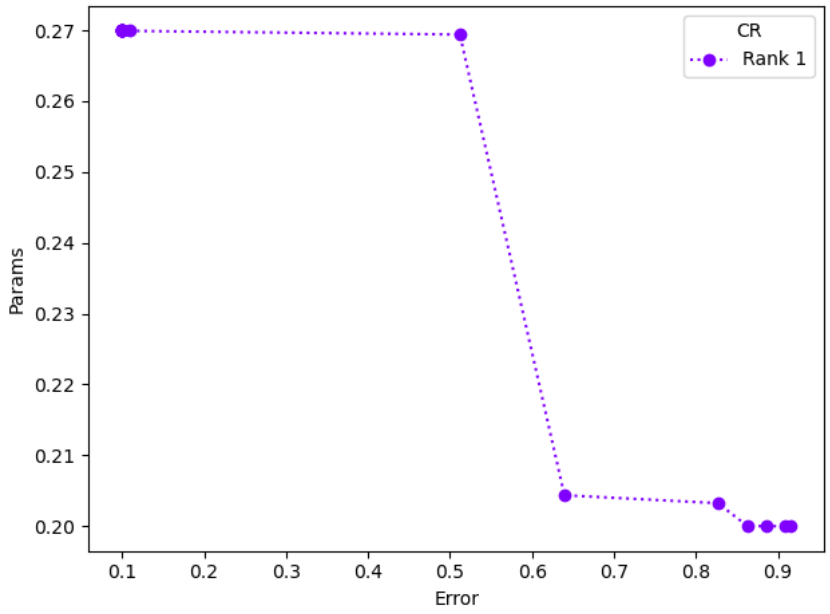
ภาพที่ 24: ผลลัพธ์ของการใช้คอนโทรลยีนเพื่อควบคุมการกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายต่ำ



ภาพที่ 25: ผลลัพธ์ของการใช้คอนโทรลยีนเพื่อควบคุมการกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายสูง

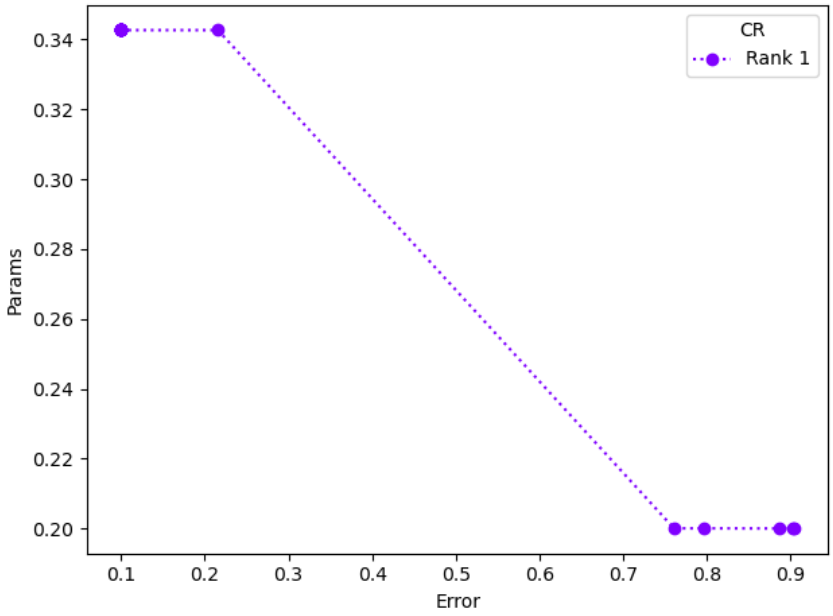
ในภาพที่ 24 และภาพที่ 25 เป็นการทดสอบที่ 1 (MU+Ctrl) สังเกตการขาดความหลากหลายทางชีวภาพของกลุ่มผลเฉลย และผลลัพธ์คะแนนรวมที่ดีกว่าในส่วนที่เริ่มจากจำนวนประชากรตั้งต้นต่ำ

ในการทดสอบที่ 0 กลุ่มผลเฉลยถูกแบ่งเป็นหลายชั้นอย่างชัดเจน ในขณะที่กลุ่มผลเฉลยในการทดสอบที่ 1 อยู่รวมกันทั้งหมดในชั้นเดียว



ภาพที่ 26: ผลลัพธ์ของการใช้การผสมพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายต่ำ

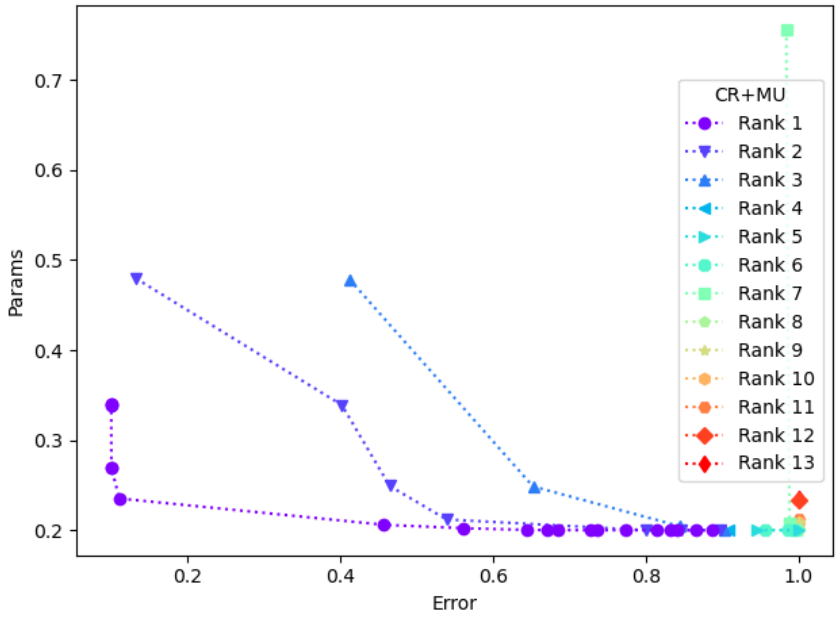
กลุ่มผลเฉลยที่ถูกแบ่งเป็นกลุ่มย่อยชัดเจนเช่นนี้ ในแง่หนึ่ง แสดงถึงความสามารถในการรักษาความหลากหลายของกลุ่มผลเฉลย อันสังเกตได้จากการที่ผลเฉลยแบ่งตัวเป็นหลายกลุ่มย่อย ที่แต่ละกลุ่มย่อยมุ่งเน้นไปที่เป้าหมายที่ต่างกัน แต่ในอีกแง่หนึ่ง ก็แสดงถึงการขาดความสามารถในการหาผลเฉลยที่สมดุลย์กันในระหว่างสองเป้าหมาย



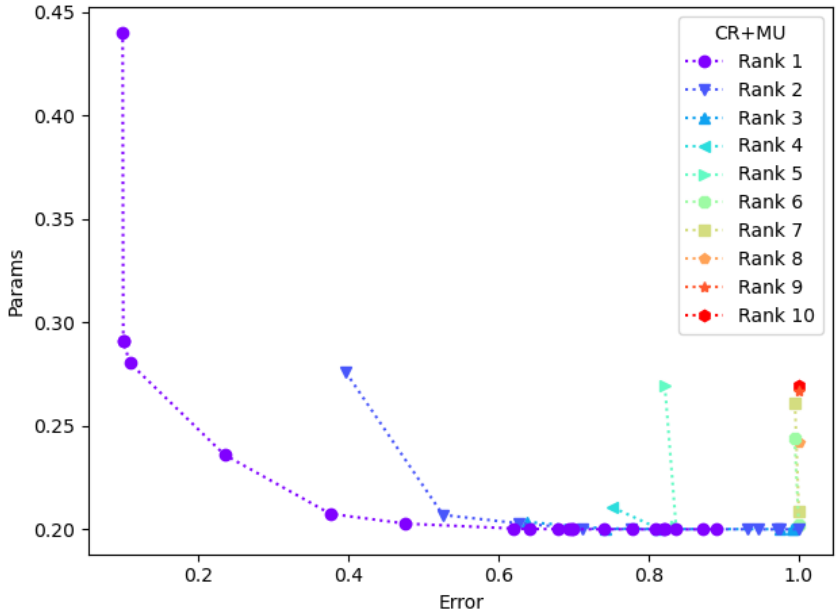
ภาพที่ 27: ผลลัพธ์ของการใช้การผสมพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายสูง

ในภาพที่ 26 และภาพที่ 27 เป็นการทดสอบที่ 2 และ 3 สังเกตว่าเมื่อคอนโทรลยีนไม่มีผลใดๆ ต่อการผสมพันธุ์ การทดสอบที่ 2 (CR) และ 3 (CR+Ctrl) จึงไม่มีความแตกต่างใดๆ ดังนั้นเราจึงแสดงเพียงการทดสอบที่ 2 เท่านั้น อีกจุดที่สำคัญตรงนี้ คือ ผลลัพธ์คะแนนรวมที่แบ่งเป็นสองกลุ่ม (กลุ่มได้โครงข่ายประสาทเทียมขนาดเล็ก กับกลุ่มได้โครงข่ายประสาทเทียมความแม่นยำสูง) โดยเฉพาะในยีนพูลประชากรตั้งต้นความหลากหลายสูงยิ่งเห็นสภาพนี้ได้อย่างชัดเจน

ทิศทางการแบ่งกลุ่มคำตอบของการกลายพันธุ์อย่างเดียว (ซึ่งกลุ่มคำตอบกระจายตัวทั่วทั้งชั้น และมีหลายคำตอบอยู่ในส่วนตรงกลาง) และการผสมพันธุ์อย่างเดียว (ซึ่งกลุ่มคำตอบแบ่งตัวเป็นหลายกลุ่ม และแต่ละกลุ่มมุ่งเน้นไปที่แต่ละเป้าหมาย) ถือว่าตัดกันอย่างชัดเจน เราจึงสามารถคาดการณ์ว่า การกลายพันธุ์ และการผสมพันธุ์ มีข้อดีข้อเสียที่ต่างกันไป และการนำมารวมกัน ก็อาจจะสืบทอดข้อดีของทั้งสองวิธี หรือข้อเสียของทั้งสองวิธี ก็เป็นได้



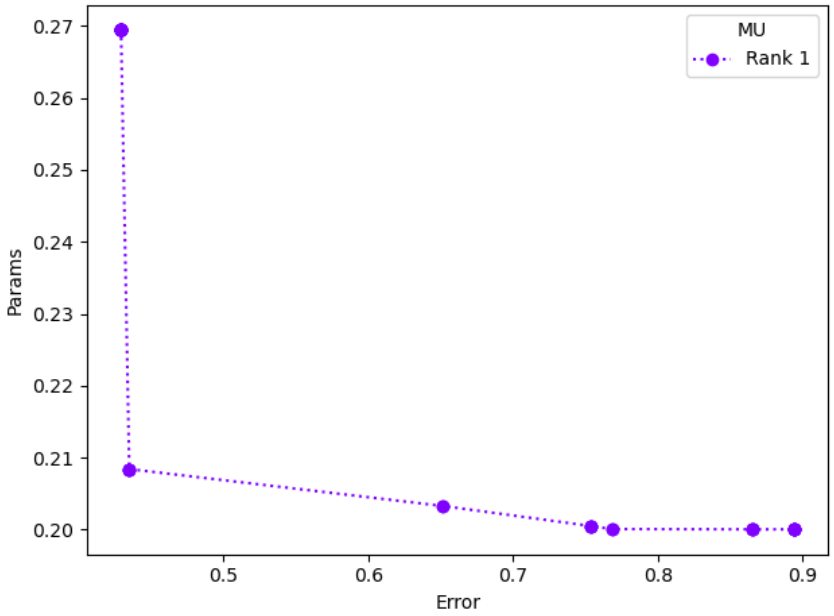
ภาพที่ 28: ผลลัพธ์ของการไม่ใช้คอนโทรลยีนควบคุมการผสมพันธุ์และกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายต่ำ



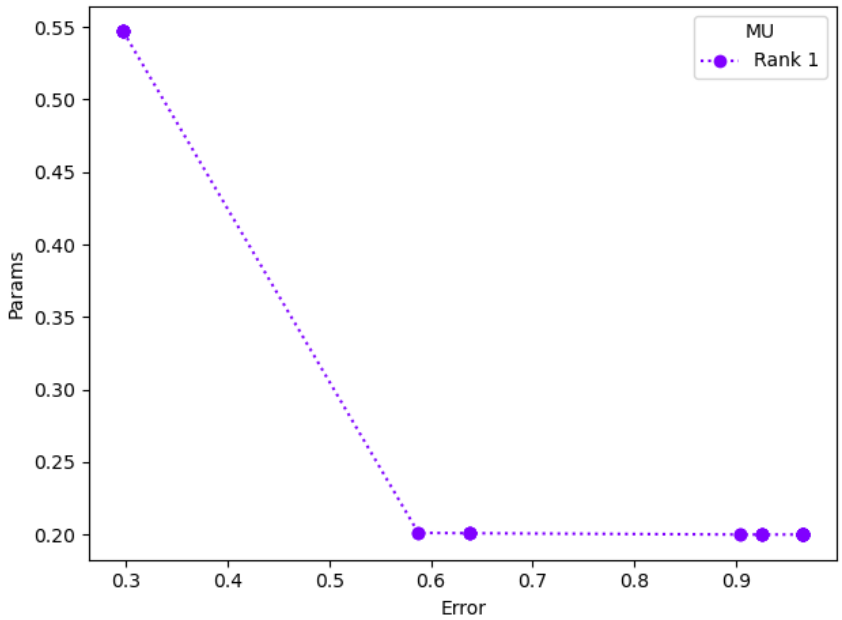
ภาพที่ 29: ผลลัพธ์ของการไม่ใช้คอนโทรลยีนควบคุมการผสมพันธุ์และกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายสูง

ในภาพที่ 28 และภาพที่ 29 เป็นการทดสอบที่ 4 (CR+MU) ซึ่งมีความหลากหลายทางชีวภาพคล้ายกับการทดสอบที่ 0 (CR+MU+Ctrl) ซึ่งสังเกตได้จากทั้งการทดสอบที่ 0 และการทดสอบ

ที่ 4 ต่างก็มีกลุ่มคำตอบที่แบ่งเป็นหลายระนาบ สังเกตข้อสำคัญตรงที่ผลของมันตรงข้ามกับการทดสอบที่ 0 เนื่องจากครั้งนี้กลุ่มประชากรเริ่มต้นความหลากหลายต่ำให้ผลลัพธ์คะแนนรวมที่ดีกว่า



ภาพที่ 30: ผลลัพธ์ของการไม่ใช้คอนโทรลยีนควบคุมการกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายต่ำ



ภาพที่ 31: ผลลัพธ์ของการไม่ใช้คอนโทรลยีนควบคุมการกลายพันธุ์ โดยใช้กลุ่มประชากรเริ่มต้นความหลากหลายสูง

ในภาพที่ 30 และภาพที่ 31 เป็นการทดสอบที่ 5 (MU) เช่นเดียวกับการทดสอบที่ 1 ผลการทดสอบที่ 5 ขาดความหลากหลายทางชีวภาพของกลุ่มผลเฉลย (การทดสอบที่ 1 การทดสอบที่ 2 และการทดสอบที่ 5 ต่างก็มีกลุ่มคำตอบเรียงกันบนระนาบเดียว) และประชากรตั้งต้นที่เพิ่มขึ้น ทำให้ผลลัพธ์คะแนนรวมของการทดสอบด้อยลง

ในการกลายพันธุ์ ทั้งในแบบใช้คอนโทรลยีน (MU+Ctrl) และไม่ใช้คอนโทรลยีน (MU) จะสังเกตได้ว่า กลุ่มผลเฉลยกระจายตัวเฉลี่ยกว่าการผสมพันธุ์ (ที่ผลลัพธ์แบ่งเป็นสองกลุ่มค่อนข้างชัดกว่า)

เมื่อเราได้เห็นทั้ง 6 การทดสอบแล้ว เราจึงสามารถสรุปได้

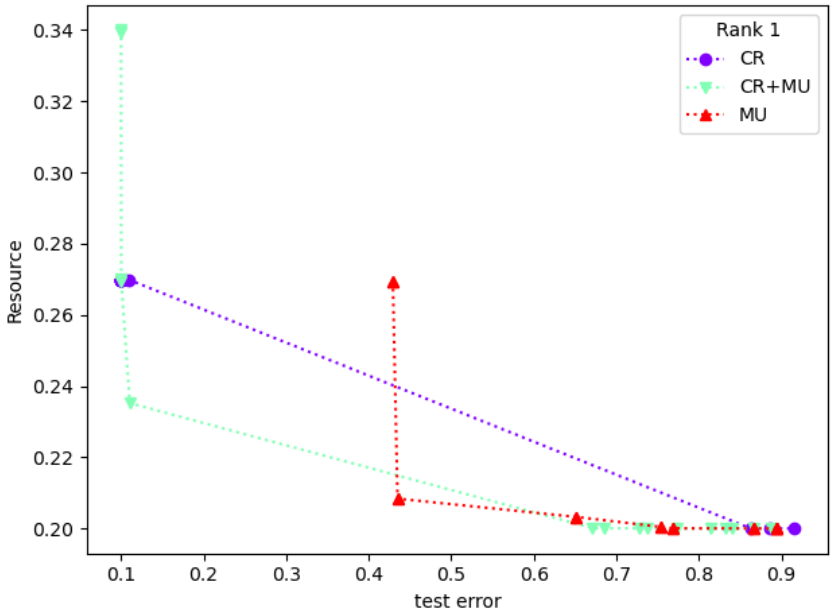
1) ในการทดสอบที่ใช้การกลายพันธุ์และการผสมพันธุ์ ไม่ว่าจะคอนโทรลยีนหรือไม่ กลุ่มผลเฉลยก็จะแบ่งเป็นหลายชั้น แสดงถึงความหลากหลายของผลเฉลย
2) ในการทดสอบที่ใช้การกลายพันธุ์อย่างเดียว และ/หรือ ผสมพันธุ์อย่างเดียว ไม่ว่าจะคอนโทรลยีนหรือไม่ กลุ่มผลเฉลยจะรวมอยู่ในชั้นเดียว
3) ในการทดสอบที่ใช้การกลายพันธุ์อย่างเดียว ผลเฉลยจะแบ่งเป็นสองกลุ่ม กลุ่มโครงข่ายประสาทเทียมความแม่นยำสูง และกลุ่มโครงข่ายประสาทเทียมขนาดเล็ก
4) ในการทดสอบที่ใช้การผสมพันธุ์อย่างเดียว ผลเฉลยจะค่อนข้างกระจายตัวตลอดทั้งเส้นชั้น

กราฟข้างล่างต่อไปนี้ แกนตั้งเป็นคะแนนขนาดโครงข่ายประสาทเทียม (ยิ่งโครงข่ายเล็กยิ่งเป็นที่พึงประสงค์) แกนนอนเป็นคะแนนความแม่นยำ (ยิ่งแม่นยำมากยิ่งเป็นที่พึงประสงค์) โดยคะแนนพึงประสงค์ดีที่สุดคือ 0 และคะแนนตัวเลขมากขึ้นหมายถึงเป็นที่พึงประสงค์น้อยลง

ในแกนตั้ง (แกน Y, Resource) ค่าคะแนนขนาดโครงข่ายประสาทเทียม มีที่มาเดียวกับแกนตั้ง (แกน Y, Params) ในการทดสอบที่ 0 – 5 โดยไม่มีผิดเพี้ยน

ในแกนนอน (แกน X, test error) ค่าคะแนนอัตราการผิดพลาดนี้ มีที่มาเดียวกับแกนนอน (แกน X, Error) ในการทดสอบที่ 0 – 5 โดยไม่มีผิดเพี้ยน

คำว่า คะแนนรวม คะแนนเฉลี่ย คะแนนกลุ่ม คะแนนดิบ (หรือเทอมที่คล้ายกันอื่นๆ) ของผลเฉลยใดๆ จึงยังคงมีความหมายเดิมเหมือนในการทดสอบที่ 0 – 5 โดยไม่มีผิดเพี้ยนเช่นกัน

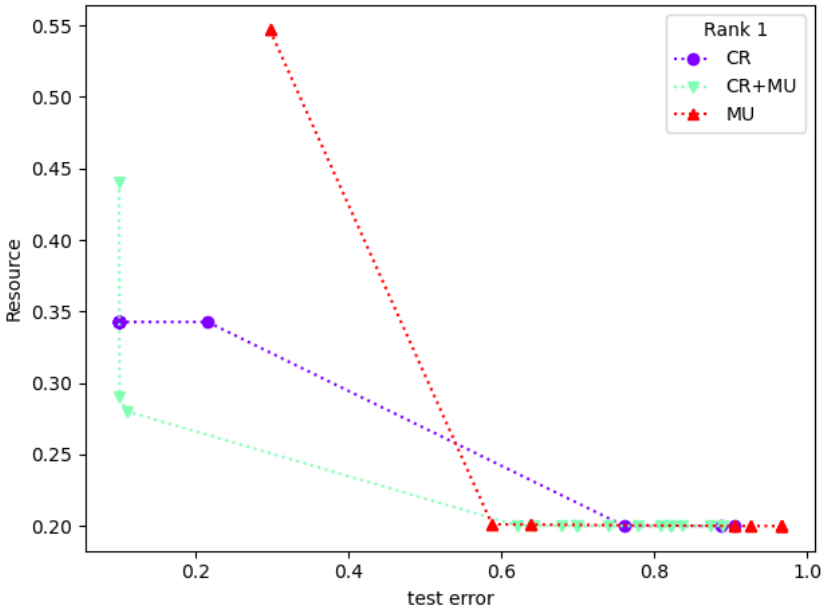


ภาพที่ 32: ในขณะความหลากหลายประชากรเริ่มต้นยังไม่มาก กลายพันธุ์ล้วนๆ สามารถทำคะแนนกลุ่มได้ดีกว่า

ภาพที่ 32 เป็นการนำระนาบขั้นหนึ่ง ของทั้งกลายพันธุ์บวกผสมพันธุ์ กลายพันธุ์อย่างเดียว และผสมพันธุ์อย่างเดียว โดยไม่มีคอนโทรลยีน และเริ่มจากยีนพูลความหลากหลายต่ำ มาเทียบคะแนนดิบกันให้เห็นผลโดยชัดเจน

ผลปรากฏว่า ในสภาพยีนพูลหลากหลายต่ำ และไม่มีคอนโทรลยีน การกลายพันธุ์ล้วนๆ ให้คะแนนรวมที่ดีที่สุด อันสามารถสังเกตได้จากระนาบขั้นหนึ่งของการกลายพันธุ์ล้วนๆ มีส่วนที่เข้าใกล้จุดกำเนิดของกราฟที่สุด แต่ในขณะเดียวกัน หากไม่ใช่เรื่องคะแนนรวม ผสมพันธุ์ล้วนๆ และผสมพันธุ์บวกกลายพันธุ์ จะได้คะแนนดีกว่าในแง่ขนาดโครงข่ายประสาทเทียมอย่างชัดเจน

การที่การกลายพันธุ์ล้วนๆ ได้คะแนนรวมดีที่สุด สามารถถูกมองได้ว่า เพราะการกลายพันธุ์ล้วนๆ ที่มีคะแนนกระจายตัวตลอดทั้งชั้น มีคำตอบบางตัวหลุดเข้าไปในกลุ่มตรงกลาง สังเกตได้จากคะแนนรวมดีที่สุดของการกลายพันธุ์ล้วนๆ มาจากคำตอบส่วนที่อยู่ค่อนไปทางกลางชั้น เพราะหากสังเกตการกระจายตัวของกลุ่มคะแนน จะเห็นได้ว่ากลุ่มคำตอบของการกลายพันธุ์ล้วนๆ จะยื่นค่อนไปทางเป้าหมายเดียว ผิดกับการผสมพันธุ์ล้วนๆ และการผสมพันธุ์และกลายพันธุ์ ที่กลุ่มคำตอบจะยื่นไปหาทั้งสองเป้าหมาย

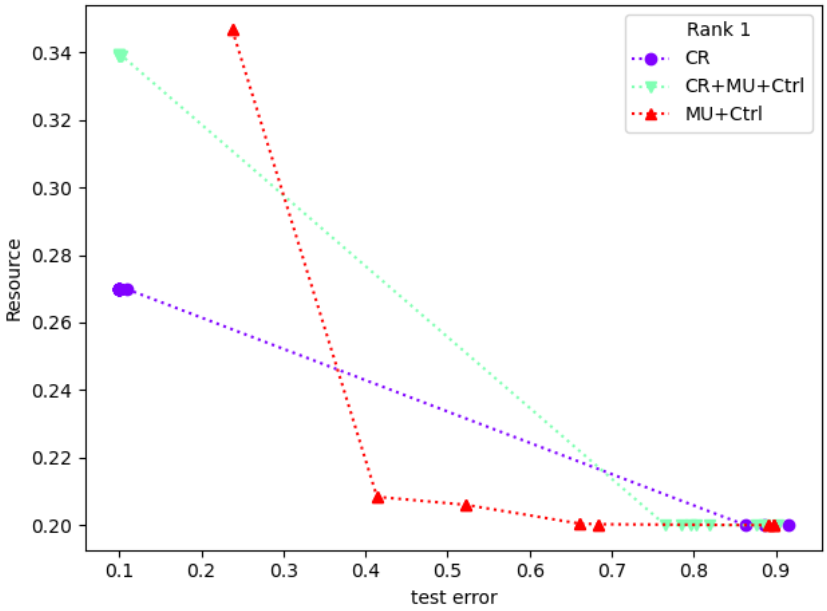


ภาพที่ 33: เมื่อประชากรเกิดความต่างชั้นของสมาชิกขึ้นสูง กลายพันธุ์บวกผสมพันธุ์ก็จะทำคะแนนดีขึ้นจนแซงกลายพันธุ์ล้วนๆ ขึ้นมา

ภาพที่ 33 เป็นการนำระนาบขั้นหนึ่ง ของทั้งกลายพันธุ์บวกผสมพันธุ์ กลายพันธุ์อย่างเดียว และผสมพันธุ์อย่างเดียว โดยไม่มีคอนโทรลยีน และเริ่มจากยีนพูลความหลากหลายสูง มาเทียบคะแนนดิบกันให้เห็นผลโดยชัดเจน

ในสภาวะไม่มีคอนโทรลยีน และเพิ่มความหลากหลายทางชีวภาพขึ้น คะแนนรวมของผสมพันธุ์บวกกลายพันธุ์จะเพิ่มขึ้นจนทัดเทียมกลายพันธุ์ล้วนๆ โดยยังรักษาคะแนนนำในแง่ขนาดโครงข่ายประสาทเทียมเอาไว้ได้

การที่ผสมพันธุ์และกลายพันธุ์สามารถทำคะแนนได้ดีขึ้น จะสังเกตได้ว่าคะแนนรวมส่วนที่ดีขึ้นมาจากจำนวนคำตอบที่อยู่ในส่วนกลางชั้นมากขึ้น (เทียบกับภาพที่ 32 ที่ผสมพันธุ์บวกกลายพันธุ์ไม่มีคำตอบที่อยู่ในส่วนกลางชั้นมากนัก)

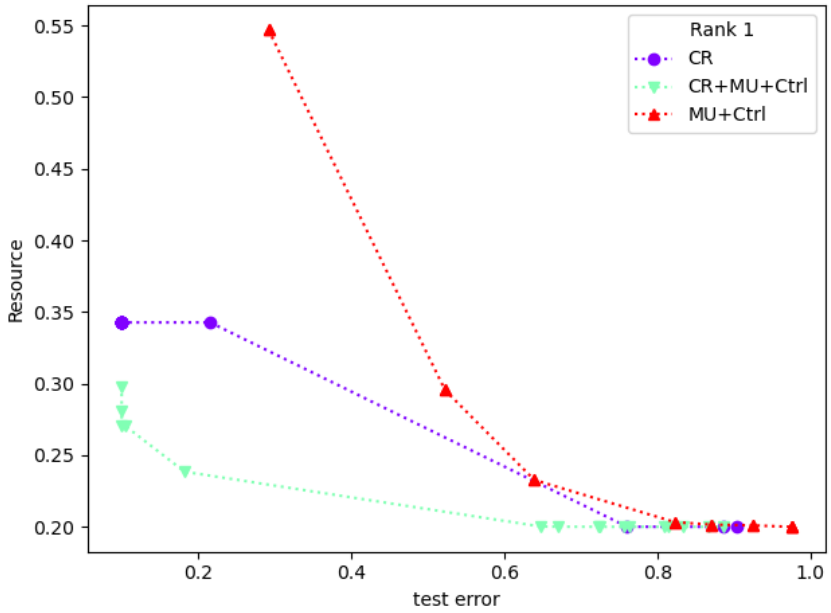


ภาพที่ 34: คอนโทรลยีนที่กลุ่มประชากรความต่างต่ำ ทำให้ผลลัพธ์ของกลายพันธุ์ล้วนๆ ดีขึ้น แต่กลายพันธุ์บวกผสมพันธุ์แย่ลง

ภาพที่ 34 เป็นการนำระนาบขั้นหนึ่ง ของทั้งกลายพันธุ์บวกผสมพันธุ์ กลายพันธุ์อย่างเดียว และผสมพันธุ์อย่างเดียว และมีคอนโทรลยีน โดยเริ่มจากยีนพูลความหลากหลายต่ำ มาเทียบคะแนนดิบกันให้เห็นผลโดยชัดเจน

เมื่อเพิ่มคอนโทรลยีนเข้าไป และในสภาวะยีนพูลเริ่มต้นหลากหลายต่ำ การกลายพันธุ์ล้วนๆ ยังคงรักษาที่หนึ่งในด้านคะแนนรวม และสามารถทำคะแนนขนาดโครงข่ายประสาทเทียมได้ดีขึ้นอย่างเห็นได้ชัด

จุดหนึ่งที่ควรสังเกตคือคอนโทรลยีนส่งผลเสียต่อความหลากหลายของกลุ่มคำตอบอย่างชัดเจน สังเกตได้จากรูปร่างของกลุ่มคำตอบที่มาจากการผสมพันธุ์บวกกลายพันธุ์ ที่กลุ่มคำตอบกระจุกรวมตัวกันเป็นกลุ่มที่เป้าหมายใดเป้าหมายหนึ่งมากขึ้นกว่าเดิม (เมื่อเทียบกับจากภาพที่ 32 ที่กลุ่มคำตอบจากการผสมพันธุ์บวกกลายพันธุ์กระจายตัวไปทั้งชั้น)



ภาพที่ 35: คอนโทรลยีนที่กลุ่มความต่างประชากรสูง ทำให้กลายพันธุ์ล้วนๆ คะแนนแย่ลง แต่กลายพันธุ์บวกผสมพันธุ์ดีขึ้น

ภาพที่ 35 เป็นการนำระนาบขั้นหนึ่ง ของทั้งกลายพันธุ์บวกผสมพันธุ์ กลายพันธุ์อย่างเดียว และผสมพันธุ์อย่างเดียว และมีคอนโทรลยีน โดยเริ่มจากยีนพูลความหลากหลายสูง มาเทียบคะแนนดิบกันให้เห็นผลโดยชัดเจน

ในสภาวะสุดท้ายที่มีคอนโทรลยีนในยีนพูลเริ่มต้นความหลากหลายสูง คะแนนของการผสมพันธุ์บวกกลายพันธุ์กลับเพิ่มขึ้นมาก จนชนะการกลายพันธุ์ล้วนๆ อย่างชัดเจนในทุกด้าน และการกลายพันธุ์ล้วนๆ ก็คะแนนตกต่ำลงมาก จนแพ้แม้แต่การผสมพันธุ์ล้วนๆ ที่ตัวเองเคยเอาชนะได้ในสมัยความหลากหลายต่ำ และ/หรือ ไม่มีคอนโทรลยีน

แนวโน้มที่เหมือนกันในทั้งสี่รูปคือ

1) ผสมพันธุ์ล้วนๆ (Test 2) ให้ผลลัพธ์กลางๆ เมื่อเทียบกับวิธีการที่เหลือ (ไม่กลายพันธุ์ คอนโทรลยีนจึงไม่มีผล)
2) กลายพันธุ์ล้วนๆ (Test 1 และ Test 5) ให้ผลลัพธ์ด้อยลงเมื่อความต่างชั้นประชากรเริ่มต้นสูงขึ้น (เปรียบเทียบก่อนหน้า Test 1 (มีคอนโทรลยีน) ดีกว่า Test 5 (ไม่มีคอนโทรลยีน) ที่ความต่างชั้นประชากรต่ำ)

3) ผสมพันธุ์บวกกลายพันธุ์ (Test 0 และ Test 4) ให้ผลลัพธ์ดีขึ้นเมื่อความต่างชั้นประชากรเริ่มต้นสูงขึ้น (เปรียบเทียบก่อนหน้า Test 0 (มีคอนโทรลยีน) ดีกว่า Test 4 (ไม่มีคอนโทรลยีน) ที่ความต่างชั้นประชากรสูง)

ตารางที่ 3: ตารางสรุปผลการทดลองโดยสังเขป

Genetic Operators	**ไม่มี Control Gene**	
	0 – 4	0 – 36
Crossover (CR)	~	~
Mutation (MU)	✓	↓
Crossover and Mutation (CR + MU)	↓	✓

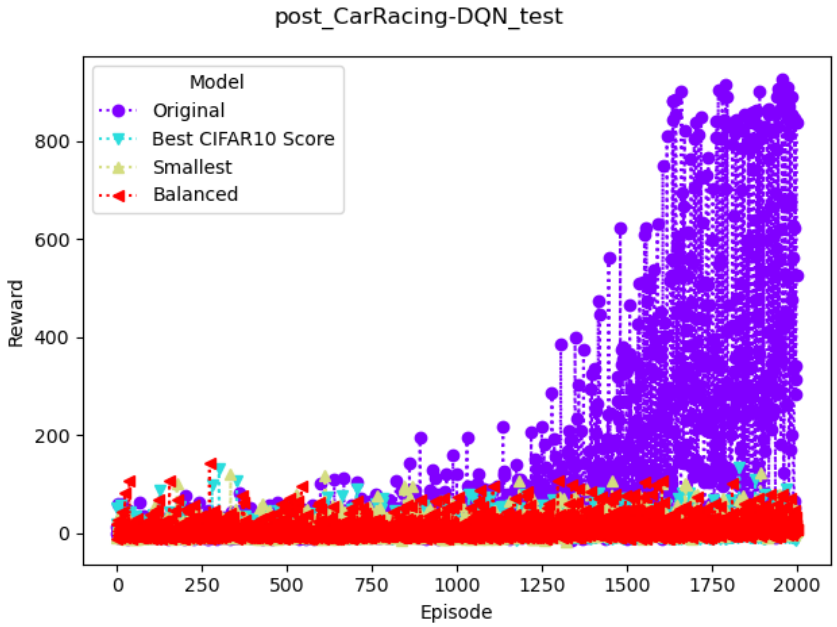
4.2 ผลการทดลองในการใช้งานกับปัญหาเกมส์

ในการทดลองเราได้ให้โครงข่ายประสาทเทียมของเรา 3 ตัว ทดลองเล่นเกมส์ CarRacing เป็นจำนวน 2000 ยก แล้วเทียบผลกับการให้โครงข่ายประสาทเทียมดั้งเดิม (ที่รู้ว่าเริ่มเรียนสำเร็จในประมาณยกที่ 1600) แน่แล้วเป็นคนเล่น แล้วเก็บผลการทดลองเป็นกราฟดังในภาพที่ 36

ในแกนตั้ง (แกน Y, Reward) เป็นคะแนนที่ตัวเกมส์ CarRacing ให้ตัวผู้เล่นรับรู้ (ดังที่ได้อธิบายแล้วในหัวข้อที่ 3.4.2) ซึ่งในกรณีนี้ เราเก็บมาเฉพาะคะแนนตอนจบยก (เช่น ตอนที่ชนะเพราะรถเข้าเส้นชัย หรือตอนที่แพ้ด้วยเหตุผลต่างๆ)

ในแกนนอน (แกน X, Episode) เป็นจำนวนยกที่ตัวเอเย่นต์เล่นเกมส์ผ่านไป ซึ่งตัวเอเย่นต์เกมส์จะฝึกโครงข่ายประสาทเทียมไปเรื่อยๆ ด้วยประสบการณ์การเล่น (ซึ่งจะได้อธิบายต่อไปในหัวข้อที่ 4.2.1)

4.2.1 ผลการทดลอง



ภาพที่ 36: กราฟผลการทดลอง เริ่มจากโครงข่ายประสาทเทียม ดั้งเดิม (ม่วง) คะแนนดี (ฟ้า) ขนาดเล็ก (เขียว) สมดุล (แดง)

เมื่อเอเย่นต์ทำการเล่นเกมส์ ตัวเอเย่นต์จะแปลงภาพวีดีโอ ครั้งละ 4f×96h×96w×3c (4 เฟรม สูง 96 กว้าง 96 3 สี) กลายเป็นเฟรมสแตค 84h×84w×4f (ขาวดำ สูง 84 กว้าง 84 4 เฟรม) บวกกับผลคะแนน (และอื่นๆ ที่ต้องใช้สำหรับสูตรคำนวณสำหรับการเรียนรู้แบบเสริมกำลัง) แล้วก็บันทึกเก็บไว้ (อย่างน้อย 10,000 ชุด (ถึงจะเริ่มใช้สอน) อย่างมาก 40,000 ชุด (ถ้าเกินจะเริ่มลบทิ้งเริ่มจากเก่าสุด)) เพื่อใช้เป็นชุดสอนสำหรับโครงข่ายประสาทเทียม ที่ทำหน้าที่เป็นฟังก์ชั่นประมาณค่าแทน Q Table โครงข่ายประสาทเทียมตัวนี้เอง ที่เราจะนำโครงข่ายประสาทเทียมที่เราได้จากยีนพูล CIFAR10 เข้ามาใช้แทนที่โครงข่ายประสาทเทียมดั้งเดิม

โครงข่ายประสาทเทียมที่ถูกใช้ทดลองประกอบด้วย โครงข่ายประสาทเทียมดั้งเดิมที่ระบบให้มา โครงข่ายประสาทเทียมที่คะแนนแม่นยำ (ตอน CIFAR10) สูงสุด โครงข่ายประสาทเทียมที่ขนาดเล็กที่สุด และโครงข่ายประสาทเทียมที่สมดุลย์ในทั้งสองด้าน

ภาพที่ 36 เป็นกราฟ แนวนอนแทนจำนวนยกที่เล่น แนวตั้งแทนคะแนนที่ทำได้ ซึ่งจะสังเกตได้ว่าโครงข่ายประสาทเทียมของเราทำคะแนนได้ในช่วงไม่เกิน 200 คะแนน ในขณะที่โครงข่ายประสาทเทียมเดิมได้ผลเกินกว่า 800 คะแนน

4.2.2 อภิปรายผลการทดลอง กับการใช้งานกับปัญหาเกมส์แข่งรถ

ผลการทดลองเบื้องต้นออกมาถือว่าไม่ดี โครงข่ายประสาทเทียมดั้งเดิมได้คะแนนดีกว่าโครงข่ายประสาทเทียมของเราทั้งสามตัวอย่างชัดเจน และโครงข่ายประสาทเทียมของเราก็ไม่มีเค้าเรียนรู้สำเร็จเป็นที่น่าพอใจ (แม้จะพ้นยกที่ 2000 ไปแล้วก็ตาม) แต่อย่างไรก็ตาม เราได้คาดเหตุผลของความล้มเหลวครั้งนี้ไว้แล้ว

1) โอเวอร์ฟิต: โครงข่ายประสาทเทียมพวกนี้ถูกฝึกกับ CIFAR10 ซึ่งเป็นข้อมูลที่มีจำนวนฟีเจอร์ภายในต่างกับของ CarRacing อย่างเห็นได้ชัด
2) เวอร์ชั่น: โค๊ด DQN ที่ใช้กับ CarRacing เป็น Tensorflow 1 ตัวโครงข่ายประสาทเทียมที่เราวิวัฒนาการขึ้นมาเป็น Tensorflow 2
3) ไม่มีชั้นเดนส์: ชั้นเชื่อมต่อถึงกันหมด (ชั้น Dense) ที่ในโครงข่ายประสาทเทียมเดิมที่สำเร็จใน CarRacing มี แต่ของเราไม่มี *ไม่น่าใช่ เราทดลองเอาชั้นเดนส์ใส่ตามเขาดูแล้ว ก็ไม่ดีขึ้น*
4) มีชั้นพูลิ่งเยอะเกินไป: ชั้นพูลิ่งจำนวนมากสามารถลดขนาดของโครงข่ายประสาทเทียมได้ แต่ก็ลดความสามารถในการสกัดฟีเจอร์ลงเช่นกัน

บทที่ 5
สรุป อภิปรายผล และข้อเสนอแนะ

5.1 บทสรุปของงานวิจัย

วิทยานิพนธ์นี้ได้นำเสนอการพัฒนาอัลกอริทึมนิวโรอีโวลูชั่น และการใช้งานกับปัญหาเกมส์ โดยขั้นตอนการวิจัยเริ่มจากการสืบค้นงานวิจัยและข้อมูลต่างๆ เกี่ยวกับโครงข่ายประสาทเทียม เกมส์ และระบบที่ใช้โครงข่ายประสาทเทียมในการเล่นเกมส ได้ออกแบบการทดลองโดยการใช้ยีนพูลที่ดัดแปลงมาจาก NSGA_II ในการค้นหาโครงข่ายประสาทเทียมที่เหมาะสม เราได้ทำระบบของเราโดยใช้ภาษา Python และกรอบงาน Tensorflow-Keras โดยมีเป้าหมายตั้งต้นเป็นชุดข้อมูล CIFAR10 และนำโครงข่ายประสาทเทียมที่ได้ไปใช้ในระบบการเรียนรู้แบบเสริมกำลังที่เล่นเกมส์ CarRacing

ผลการทดลองในส่วนที่ทำกับชุดข้อมูล CIFAR10 การผสมพันธุ์และกลายพันธุ์ได้ผลลัพธ์เป็นกลุ่มผลเฉลยที่แบ่งเป็นหลายขั้น การผสมพันธุ์อย่างเดียว และการกลายพันธุ์อย่างเดียว ได้ผลลัพธ์รวมกันอยู่ในขั้นเดียว แต่การผสมพันธุ์อย่างเดียวได้ผลลัพธ์ที่แบ่งตัวออกเป็นสองกลุ่มที่สุดปลายขั้น กลุ่มที่เป็นโครงข่ายประสาทเทียมความแม่นยำสูง และกลุ่มที่เป็นโครงข่ายประสาทเทียมขนาดเล็ก ในขณะที่การกลายพันธุ์อย่างเดียวได้ผลลัพธ์ที่เฉลี่ยกระจายทั้งขั้นนมากกว่า และสำหรับคอนโทรลยีน คอนโทรลยีนไม่มีผลต่อการผสมพันธุ์อย่างเดียว แต่ในกรณีที่เหลือ คอนโทรลยีนสามารถช่วยในกรณีกลายพันธุ์อย่างเดียวที่ประชากรเริ่มต้นความหนาแน่นต่ำ และผสมพันธุ์บวกกลายพันธุ์ที่ประชากรเริ่มต้นความหนาแน่นสูง แต่ได้ผลลัพธ์แย่ลงในกรณีกลายพันธุ์อย่างเดียวที่ประชากรเริ่มต้นความหนาแน่นสูง และผสมพันธุ์บวกกลายพันธุ์ที่ประชากรเริ่มต้นความหนาแน่นต่ำ

การทดลองในส่วนที่ทำกับเกมส์ CarRacing ที่เกิดขึ้น ยีนพูลสามารถค้นหาโครงข่ายประสาทเทียมที่เหมาะสมกับ CIFAR10 แต่โครงข่ายประสาทเทียมที่เหมาะสมกับ CIFAR10 กลับไม่เหมาะสมกับการใช้เล่นเกมส์ CarRacing คือไม่สามารถสกัดฟีเจอร์ที่จำเป็นต่อการให้เอเย่นต์เล่นเกมส์ CarRacing ได้จนถึงระดับที่เทียบเคียงกับโครงข่ายประสาทเทียมต้นฉบับ

5.2 ข้อเสนอแนะ

การที่ยีนพูลไม่สามารถค้นหาโครงข่ายประสาทเทียมที่เหมาะกับเกมส์ CarRacing ได้ นั้นมาจากการที่ชุดข้อมูล CIFAR10 มีฟีเจอร์ซับซ้อนน้อยกว่าชุดเฟรมสแตคของเกมส์ CarRacing โครงข่ายประสาทเทียมที่ถูกปรับให้พอเหมาะกับ CIFAR10 จึงไม่เพียงพอที่จะใช้กับ CarRacing ดังนั้นหากยีนพูลใช้ชุดเฟรมสแตคของเกมส์ CarRacing ที่มีการทำสำเร็จมาก่อน โครงข่ายประสาทเทียมที่ได้ย่อมสามารถสกัดฟีเจอร์ของเกมส์ CarRacing ได้ดีขึ้น

รายการอ้างอิง

1. Haykin, S.S., *Neural networks and learning machines*. 2009, New York: Prentice Hall.
2. Goodfellow, I., Y. Bengio, and A. Courville, *Deep Learning*. 2016: MIT Press.
3. Floreano, D., P. Dürr, and C. Mattiussi, *Neuroevolution: from architectures to learning.* Evolutionary Intelligence, 2008. **1**(1): p. 47-62.
4. Kacprzyk, J. and W. Pedrycz, *Springer handbook of computational intelligence*. 2015: Springer.
5. Justesen, N., et al., *Playing Multiaction Adversarial Games: Online Evolutionary Planning Versus Tree Search.* IEEE Transactions on Games, 2018. **10**(3): p. 281-291.
6. Stanley, K.O. and R. Miikkulainen, *Evolving neural networks through augmenting topologies.* Evolutionary computation, 2002. **10**(2): p. 99-127.
7. Stanley, K.O., et al., *Designing neural networks through neuroevolution.* Nature Machine Intelligence, 2019. **1**(1): p. 24-35.
8. Gade, M., et al., *Applying Machine Learning to Robocode*, in *Aalborg University, Aalborg, Denmark*. 2003.
9. Risi, S. and J. Togelius, *Neuroevolution in games: State of the art and open challenges.* IEEE Transactions on Computational Intelligence and AI in Games, 2017. **9**(1): p. 25-41.
10. Deb, K., *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley-Interscience Series in Systems and Optimization. 2001, England: John Wiley & Son.
11. Rumelhart, D.E., G.E. Hinton, and R.J. Williams, *Learning representations by back-propagating errors.* Cognitive modeling, 1988. **5**(3): p. 1.
12. Ekman, M., *Learning Deep Learning: Theory and Practice of Neural Networks, Computer Vision, Natural Language Processing, and Transformers Using TensorFlow*. 2021: Addison-Wesley Professional.
13. Miikkulainen, R., et al., *Evolving deep neural networks*, in *Artificial Intelligence in*

the Age of Neural Networks and Brain Computing. 2019, Elsevier. p. 293-312.

14. Iba, H., *Evolutionary Approach to Machine Learning and Deep Neural Networks: Neuro-Evolution and Gene Regulatory Networks 1st ed. 2018 Edition*. 1st ed. 2018: Springer.
15. Zhan, Z.-H., J.-Y. Li, and J. Zhang, *Evolutionary deep learning: A survey.* Neurocomputing, 2022. **483**: p. 42-58.
16. Deb, K., et al., *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II.* IEEE Transactions on Evolutionary Compuation, 2002. **6**(2): p. 182-197.
17. Omelianenko, I., *Hands-On Neuroevolution with Python: Build high-performing artificial neural network architectures using neuroevolution-based algorithms*. 1st ed. 2019: Packt Publishing.
18. Hoekstra, V., *An overview of neuroevolution techniques*, in *Business Mathematics and Informatics course program at the VU university in Amsterdam*. 2011.
19. Gustavo-Adolfo, V.-H., M.-M. Efrén, and A.-M. Héctor-Gabriel, *A Review on Convolutional Neural Network Encodings for Neuroevolution.* IEEE Transactions on Evolutionary Compuation, 2022. **26**(1): p. 12-27.
20. Sutton, R.S., *Reinforcement Learning, second edition: An Introduction*. Adaptive Computation and Machine Learning series. 2018: Bradford Books.
21. 70.249.220.170. *Game complexity - Wikipedia*. 2006 [cited 2019 17 May 2019]; Available from: https://en.wikipedia.org/wiki/Game_complexity.
22. Branavan, S., D. Silver, and R. Barzilay. *Non-linear monte-carlo search in civilization ii*. in *Twenty-Second International Joint Conference on Artificial Intelligence*. 2011. Barcelona, Catalonia, Spain.
23. Cardona, A.B., J. Togelius, and M.J. Nelson. *Competitive coevolution in ms. pac-man*. in *2013 IEEE Congress on Evolutionary Computation*. 2013. Cancun, Mexico: IEEE.
24. Wang, C., et al. *Portfolio online evolution in StarCraft*. in *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*. 2016. Burlingame, California, USA.
25. Justesen, N. and S. Risi. *Continual online evolutionary planning for in-game*

build order adaptation in starcraft. in *Proceedings of the Genetic and Evolutionary Computation Conference*. 2017. Berlin, Germany: ACM.

26. Eisenstein, J., *Evolving robocode tank fighters*, in *Massachusetts Institute of Technology Computer Science and Artificial Intelligence Laboratory*. 2003.
27. Lu, Z., et al., *NSGA-Net: Neural Architecture Search using Multi-Objective Genetic Algorithm*, in *Proceedings of the genetic and evolutionary computation conference*. 2019: Prague, Czech Republic. p. 419–427.
28. Galván, E. and P. Mooney, *Neuroevolution in Deep Neural Networks: Current Trends and Future Challenges.* IEEE Transactions on Artificial Intelligence, 2021. **2**(6): p. 476-493.
29. Patcharabumrung, P., Y. Jewajinda, and K. Praditwong. *Effects of Genetic Operators on Neural Architecture Search Using Multi-Objective Genetic Algorithm*. in *20th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. 2023. Phitsanulok.
30. Klimov, O. *Car Racing - Gymnasium Documentation*. 2022 [cited 2023 28 May 2023]; Available from: https://gymnasium.farama.org/environments/box2d/car_racing/.
31. PIERRÉ, A. *gym/car_racing.py at master · openai/gym · GitHub*. 2022 4 October 2022 [cited 2023 28 May 2023]; Available from: https://github.com/openai/gym/blob/master/gym/envs/box2d/car_racing.py.

มหาวิทยาลัยศิลปากร

ประวัติผู้เขียน

ชื่อ-สกุล	ไพรวัน พัชรบำรุง
วัน เดือน ปี เกิด	14 มิถุนายน 2535
สถานที่เกิด	จังหวัดสมุทรสาคร
วุฒิการศึกษา	วิศวกรรมศาสตรบัณฑิต (วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยมหิดล
ที่อยู่ปัจจุบัน	71/1 หมู่ 8 ตำบลหนองสองห้อง อำเภอบ้านแพ้ว จังหวัดสมุทรสาคร