การปรับปรุงอัลกอริทึมการเรียนรู้เชิงลึกด้วยอัตราการเรียนรู้ที่ปรับตัวเองได้

IMPROVING DEEP LEARNING ALGRITHM BY SELF-ADAPTIVE LEARNING RATE

สุทิศ องอาจ
SUTIT ONGART

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าคุณทหารลาดกระบัง
พ.ศ. 2563
KMITL-2020-EN-M-070-022

การปรับปรุงอัลกอริทึมการเรียนรู้เชิงลึกด้วยอัตราการเรียนรู้ที่ปรับตัวเองได้

IMPROVING DEEP LEARNING ALGRITHM BY SELF-ADAPTIVE LEARNING RATE

สุทิศ องอาจ

SUTIT ONGART

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2563

KMITL-2020-EN-M-070-022

IMPROVING DEEP LEARNING ALGRITHM BY SELF-ADAPTIVE LEARNING RATE

SUTIT ONGART

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN COMPUTER ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2020

KMITL-2020-EN-M-070-022

หัวข้อวิทยานิพนธ์	การปรับปรุงอัลกอริทึมการเรียนรู้เชิงลึกด้วยอัตราการเรียนรู้ที่ปรับตัวเองได้
นักศึกษา	นายสุทิศ องอาจ
รหัสประจำตัว	60601038
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
พ.ศ.	2563
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ.ดร.เกียรติกูล เจียรนัยธนะกิจ

บทคัดย่อ

บทความนี้จะนำเสนอการปรับปรุงอัลกอริทึมการเรียนรู้สำหรับโครงข่ายประสาทเทียมแบบสังวัตนาการโดยการใช้อัตราการเรียนรู้ที่ปรับเองได้ ซึ่งจะทำให้การลู่เข้าสู่คอนเวอร์เจนซ์เร็วขึ้นกว่าเดิมด้วยวิธีการใช้อัตราการเรียนรู้ที่ปรับเองได้ เนื่องจากวิธีแบบดั้งเดิมนั้นการกำหนดอัตราการเรียนรู้จะขึ้นอยู่กับประสบการณ์และการทดลอง แต่ในงานวิจัยนี้จะนำเสนอให้ว่าอัตราการเรียนรู้ปรับตัวเองได้สามารถใช้กฎพื้นฐานของรูปแบบสมการเทย์เลอร์ ที่มีความสัมพันธ์ระหว่างฟังก์ชันการเปลี่ยนแปลงความผิดพลาดกำลังสองกับน้ำหนักการเชื่อมต่อและไบอัสการเปลี่ยนแปลงที่ได้รับ ซึ่งสมการสำหรับหาอัตราการเรียนรู้ด้วยตนเองจะมีการปรับค่าอัตราเรียนรู้แต่ละรอบขึ้นอยู่กับความผิดพลาดกำลังสองเฉลี่ยและแนวโน้มความผิดพลาดของเกรเดียนต์ และผลจากทดลองการจำลองของโปรแกรมแสดงให้เห็นว่าจะใช้เวลาเข้าสู่คอนเวอร์เจนซ์น้อยกว่าวิธีแบบดั้งเดิมที่กำหนดอัตราการเรียนรู้คงที่

Thesis	Improving Deep Learning Algorithm by Self-Adaptive Learning Rate
Student	Mr.Sutit Ongart
Student ID.	60601038
Degree	Master of Engineering
Program	Computer Engineering
Year	2020
Thesis Advisor	Assoc.Prof.Dr. Kietikul Jearanaitanakij

ABSTRACT

This thesis improves the training algorithm for Convolutional Neural Network by using self-adaptive learning rate. Tuning for appropriate the value of learning rate usually requires human intervention. In this work, the learning rate can be adaptive based on Taylor's formula. This formula has the relationship among the root mean square errors changed, connection template, weights and biases changes. The proposed self-adaptive learning rate is calculated from the root mean square error and the error curve surface gradient. From the experimental results, this proposed system can reach the convergence with smaller number of training iterations than the traditional algorithm with a constant learning rate.

กิตติกรรมประกาศ

วิจัยฉบับนี้สำเร็จได้ด้วยได้รับความช่วยเหลือจากอาจารย์ รศ.ดร.เกียรติกูล เจียรนัยธนะกิจ ที่ได้ให้โอกาสและให้คำแนะนำในการทำวิจัยรวมถึงตรวจทานแก้ไขเนื้อหาในวิจัยนี้ ซึ่งผู้วิจัยได้รับแนวทางในการศึกษาค้นคว้าหาความรู้เพิ่มเติมและประสบการณ์อย่างมากในการทำวิจัยครั้งนี้จึงขอขอบพระคุณเป็นสูงมา ณ โอกาสนี้

สุทิศ องอาจ

สารบัญ

หน้า

บทคัดย่อ I
ABSTRACT II
กิตติกรรมประกาศ III
สารบัญ IV
สารบัญรูป VI
สารบัญตาราง VIII
บทที่ 1 บทนำ 1
1.1 ความเป็นมาของปัญหา 1
1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา 1
1.3 สมมุติฐานของการศึกษา 1
1.4 ทฤษฎีหรือแนวคิดที่ใช้ในการวิจัย 2
1.5 ขอบเขตการวิจัย 2
1.6 ขั้นตอนการศึกษา 2
1.7 เครื่องมือและอุปกรณ์ที่ใช้ในงานวิจัย 2
1.8 โครงสร้างของวิทยานิพนธ์ 2
บทที่ 2 ทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้อง 4
2.1 โครงข่ายประสาทเทียมเบื้องต้น 4
2.2 Convolutional Neural Network 5
2.2.1 การทำงานของ Convolutional Neural Network 5
2.2.2 Convolution 6
2.2.3 Pooling layer 6
2.2.4 Fully connected 7
2.2.5 ความผิดพลาดของการเรียนรู้ 8
2.2.6 กระบวนการความผิดพลาดของการแพร่กลับ 9
2.2.7 การประมาณค่าฟังก์ชันด้วยวิธีของอนุกรมเทย์เลอร์ (Taylor series Method) 10
2.2.8 การประมาณค่าสมการ 1 ตัวแปร 11
2.2.9 การประมาณค่าสมการหลายตัวแปร 11
บทที่ 3 ปัญหาของการปรับค่าอัตราการเรียนรู้ 14
3.1 การทดลองข้อมูลชุด MNIST 15
3.1.1 อัตราการเรียนรู้ที่มีค่าเข้าใกล้ 0 16
3.1.2 อัตราการเรียนรู้ที่มีค่าระหว่าง 0.1 -1 16
3.1.3 อัตราการเรียนรู้ที่มีค่ามากกว่า 2-10 20
3.1.4 สรุปความผิดพลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ระหว่าง 0.01 – 10 23

สารบัญ(ต่อ)

3.2 การทดลองข้อมูลชุด CIFAR-10 25
3.2.1 ทดลองโครงสร้างในการทดลอง 25
3.2.2 สรุปความผิดพลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ระหว่าง 0.01 - 10 28
3.3 สรุปปัญหาที่เกิดจากการปรับค่าอัตราการเรียนรู้ 30
บทที่ 4 อัตราการเรียนรู้ที่สามารถปรับตัวเองได้ 31
4.1 อัตราการเรียนรู้ที่ปรับตัวเองได้ โดยอาศัยทฤษฎีอนุกรมเทย์เลอร์ 31
4.2 Procedure ขั้นตอนการทำงาน 33
บทที่ 5 ผลการทดลอง 35
5.1 ทดลองโดยใช้ข้อมูล MNIST database 35
5.1.1 ค่าพารามิเตอร์ที่ใช้ในการทดลอง 35
5.1.2 ผลการทดลอง 36
5.2 ทดลองโดยใช้ข้อมูล CIFAR-10 38
5.2.1 ค่าพารามิเตอร์ที่ใช้ในการทดลอง 38
5.2.2 ผลการทดลอง 39
5.3 สรุปความผิดพลาดในการทดสอบ 42
บทที่ 6 สรุปและข้อเสนอแนะ 44
6.1 ข้อสรุป 44
6.2 ข้อเสนอแนะ 44
เอกสารอ้างอิง 45
ภาคผนวก 46
ภาคผนวก ก 47
ภาคผนวก ข 50
ประวัติผู้เขียน 57

สารบัญรูป

หน้า

รูปที่ 2.1 โครงสร้างทั่วไปของโครงข่ายประสาทเทียม 4
รูปที่ 2.2 โครงสร้างของ อัลกอริทึม Deep learning 5
รูปที่ 2.3 การทำ Max pooling ขนาด (2X2) 7
รูปที่ 2.4 การทำAverage pooling ขนาด (2X2) 7
รูปที่ 2.5 โครงสร้างFully connected 7
รูปที่ 2.6 กระบวนการแพร่กลับ 9
รูปที่ 2.7 กราฟตัวอย่างของเทร์เลอร์ 11
รูปที่ 3.1 ตัวอย่างชุดข้อมูลMNIST 14
รูปที่ 3.2 ตัวอย่างชุดข้อมูลCIFAR-10 15
รูปที่ 3.3 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.1 16
รูปที่ 3.4 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.2 17
รูปที่ 3.5 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.3 17
รูปที่ 3.6 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.4 18
รูปที่ 3.7 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.5 18
รูปที่ 3.8 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.6 18
รูปที่ 3.9 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.7 19
รูปที่ 3.10 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.75 19
รูปที่ 3.11 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.8 19
รูปที่ 3.12 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.9 20
รูปที่ 3.13 แสดงความผิดพลาด RMS เมื่อ Learning Rate =1 20
รูปที่ 3.14 แสดงความผิดพลาด RMS เมื่อ Learning Rate =2 21
รูปที่ 3.15 แสดงความผิดพลาด RMS เมื่อ Learning Rate =3 21
รูปที่ 3.16 แสดงความผิดพลาด RMS เมื่อ Learning Rate =4 21
รูปที่ 3.17 แสดงความผิดพลาด RMS เมื่อ Learning Rate =5 22
รูปที่ 3.18 แสดงความผิดพลาด RMS เมื่อ Learning Rate =10 22
รูปที่ 3.19 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.01-10 23
รูปที่ 3.20 กราฟเปรียบเทียบค่าความผิดพลาด ในการเรียนรู้และทดสอบ 24
รูปที่ 3.21 กราฟเปรียบเทียบค่าความผิดพลาด ในการเรียนรู้และทดสอบ 29
รูปที่ 3.22 กราฟเปรียบเทียบค่าความผิดพลาด ในการเรียนรู้และทดสอบ 30
รูปที่ 5.1 เวลาที่ใช้ในการเรียนรู้แต่ละอัตราการเรียนรู้ 36
รูปที่ 5.2 ความผิดพลาดทั้ง 3 แบบ 37
รูปที่ 5.3 เวลาที่ใช้ในการเรียนรู้แต่ละอัตราการเรียนรู้ 40
รูปที่ 5.4 ความผิดพลาดทั้ง 3 แบบ 40
รูปที่ 5.5 อัตราการเรียนรู้ที่ปรับตัวเองได้ 41

สารบัญรูป(ต่อ)

รูปที่ 5.6 ความผิดพลาด ในกรณีที่อัตราการเรียนรู้ปรับตัวเองได้ 41
รูปที่ 5.7 ความผิดพลาด เทียบกับ อัตราการเรียนรู้ปรับตัวเองได้ 42
รูปที่ ก.1 ตัวอย่างชุดข้อมูล MNIST 47
รูปที่ ก.2 แปลงข้อจาก 1x784 เป็น 28x28 47
รูปที่ ก.3 ตัวอย่างชุดข้อมูล CIFAR-10 dataset 48
รูปที่ ก.4 การจัดรูปแบบ input เป็นแบบarray2 มิติ 49

สารบัญตาราง

หน้า

ตารางที่ 3.1 โครงสร้างในการทดลอง 15
ตารางที่ 3.2 เปรียบเทียบความผิดผลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ โดยใช้ข้อมูลMNIST 24
ตารางที่ 3.3 ตารางสรุปการผลทดลองเมื่อเปลี่ยนโครงสร้างในการทดลองในค่าต่าง ๆ 26
ตารางที่ 3.4 โครงสร้างในการทดลอง 28
ตารางที่ 3.5 เปรียบเทียบความผิดพลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ โดยใช้ข้อมูลชุดCIFAR-10 29
ตารางที่ 5.1 ค่าพารามิเตอร์ที่ใช้ในการทดลอง 35
ตารางที่ 5.2 ผลการทดลอง 36
ตารางที่ 5.3 ค่าพารามิเตอร์ที่ใช้ในการทดลอง 39
ตารางที่ 5.4 ผลการทดลอง 39
ตารางที่ 5.5 สรุปผลความผิดพลาดในการทดสอบ 43
ตารางที่ ก.1 แสดงความสัมพันธ์ระหว่าง note กับชื่อกลุ่ม 48
ตารางที่ ก.2 แสดงความสัมพันธ์ระหว่าง note กับชื่อกลุ่ม 49

บทที่ 1
บทนำ

1.1 ความเป็นมาของปัญหา

โครงข่ายประสาทเทียมแบบแพร่กลับเป็นที่นิยมกันมาก แต่ก็ยังมีจุดด้อยหลายข้อหลักๆ ได้แก่จำนวนรอบในการเรียนรู้ที่ต้องใช้จำนวนหลายรอบ ปัญหาความสามารถในการเรียนรู้แบบท้องถิ่นที่ยังไม่มีประสิทธิภาพ และปัญหาจำนวนน้ำหนักที่มีมากตามจำนวนอินพุตของข้อมูล โดยเฉพาะเมื่อนำไปใช้กับข้อมูลที่มีการแบ่งกลุ่มที่เกี่ยวกับรูปภาพแล้วจะพบว่าจำนวนข้อมูลอินพุตมีจำนวนมหาศาลเนื่องจากข้อมูลรูปภาพประกอบด้วยจุดสีและแม่สีจำนวน 3 แม่สี ซึ่งสามารถคำนวณเป็นจำนวนโหนดอินพุตคือ ความกว้าง x ความยาว x 3 ทำให้ชุดจำนวนโหนดอินพุตมีจำนวนมากมาย ในเวลาต่อมามีการนำเสนออัลกอริทึม Deep learning ที่มีโครงสร้างที่สามารถแก้ปัญหาจำนวนโหนดของอินพุตที่มีจำนวนมากได้ สำหรับการทำงานจะมีความคล้ายกับโครงข่ายประสาทเทียมแบบแพร่กลับในแง่ของการปรับค่าน้ำหนักของเส้นที่เชื่อมต่อของแต่ละโหนด การทำงานของอัลกอริทึม Deep learning มีการทำงานอยู่ 2 ส่วน คือ forward ซึ่งจะเป็นการนำข้อมูลกระทำกับค่าน้ำหนัก ในส่วนนี้แบ่งแยกย่อยไปอีกได้ 3 ส่วน ได้แก่ การคอนโวลูชัน,Max-Pooling และ Fully connected ส่วนที่สองคือ กับปรับค่าน้ำหนักหรือ Backpropagationโดยส่วนนี้จะเป็นกระบวนการย้อนกลับซึ่งเป็นการหาความไวของความผิดพลาดเมื่อเทียบกับค่าน้ำหนักนั้น

หลักการของอัลกอริทึม Deep learning ในส่วน Fully connected จะมีการกำหนดค่าคงที่ของการเรียนรู้ (Learning rate) เป็นค่าใด ซึ่งส่วนใหญ่จะกำหนดตามความเชี่ยวชาญของผู้ทำวิจัย หากค่า Learning rate มีค่ามาก ผลของความผิดพลาดช่วงแรกๆจะดีมาก เพราะมีการปรับน้ำหนักค่ามากๆ แต่จะมีข้อเสียเมื่อทำงานลู่เข้าสู่คำตอบ อาจจะมีการกระโดดข้ามคำตอบ และมีการแกว่งไปมาไม่สามารถลู่เข้าสู่คำตอบที่ดีที่สุดได้ หากปรับ Learning Rate ค่าน้อยเกินไป ผลการเรียนรู้ก็จะขยับช้าและส่งผลให้การเรียนรู้ใช้เวลานานเกินไป งานวิจัย [1] ได้เสนอวิธีแก้ปัญหาการปรับ Learning Rate ให้สามารปรับตัวเองได้ โดยอาศัยหลักการ นำค่าของ loss function มาเทียบกับความไวความความผิดพลาด ซึ่งวิธีนี้ได้นำมาใช้กับการเรียนรู้ในโครงข่ายประสาทเทียมแบบแพร่กลับและได้นำมาประยุกต์ใช้ในอัลกอริทึม Deep learning เพื่อให้ Learning Rate สามารถปรับตัวเองได้ จากผลการทดลองจะเห็นได้ว่า สามารถลู่เข้าหาคำตอบได้เร็วกว่าวิธีแบบดั้งเดิม

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

ความมุ่งหมายเพื่อนำเสนอวิธีการปรับอัตราการเรียนรู้ที่ปรับตัวเองได้ เพื่อให้สามารถหาคำตอบได้เร็วขึ้นโดยไม่ต้องเปลี่ยนค่าอัตราการเรียนรู้

1.3 สมมุติฐานของการศึกษา

อัตราการเรียนรู้ ถือเป็นค่าคงที่ที่สิ่งสำคัญในการลู่เข้าหาคำตอบ ดังนั้นถ้าอัตราการเรียนรู้สามารถปรับค่าขึ้นลงได้ด้วยตัวเองก็น่าจะส่งผลให้การเรียนรู้ใช้เวลาเร็วขึ้นในการลู่เข้าหาคำตอบ

1.4 ทฤษฎีหรือแนวคิดที่ใช้ในการวิจัย

อนุกรมเทย์เลอร์นั้นสามารถประมาณค่าฟังก์ชันรอบจุดใดๆ ได้ด้วยการทำอนุพันธ์ย่อย ดังนั้นอนุกรมเทย์เลอร์ก็สามารถประมาณค่าฟังก์ชันความผิดพลาดของการเรียนรู้โครงข่ายประสาทเทียมด้วยการทำอนุพันธ์ย่อยกับตัวแปรที่มีผลต่อความผิดพลาด และสามารถจัดรูปแบบสมการใหม่เพื่อหาอัตราการเรียนรู้ที่ปรับตัวเองได้

1.5 ขอบเขตการวิจัย

วิทยานิพนธ์ฉบับนี้จะทำการศึกษาเมื่อมีการปรับค่าคงที่ของอัตราการเรียนรู้ในค่าต่างๆ และอัตราการเรียนรู้ที่สามารถปรับตัวเองได้

วิธีการควบคุมจำนวนประชากรให้อยู่ในค่าที่เหมาะสมโดยกำหนดขอบเขตดังนี้

1. ต้องสามารถลดเวลาในการประมวลผล ในเมื่อเทียบกับการกำหนดค่าอัตราการเรียนรู้ที่คงที่
2. ความผิดพลาดต้องมีค่าน้อยกว่าหรือเท่ากับวิธีที่กำหนดค่าอัตราการเรียนรู้ที่คงที่

1.6 ขั้นตอนการศึกษา

1.6.1 ศึกษาทฤษฎีและความรู้พื้นฐานที่เกี่ยวข้องกับอัลกอริทึมการเรียนรู้เชิงลึก

1.6.2 ศึกษาทฤษฎีและความรู้พื้นฐานที่เกี่ยวข้องกับการคำนวณความผิดพลาดในการปรับอัตราการเรียนรู้

1.6.3 ศึกษาวิธีการปรับค่าอัตราการเรียนรู้

1.6.4 ทดลองวิธีการปรับค่าอัตราการเรียนรู้เพื่อศึกษาข้อดีและข้อเสีย

1.6.5 ปรับปรุงวิธีการปรับค่าอัตราการเรียนรู้

1.6.6 สรุปผลการทดลองและวิเคราะห์ผลลัพธ์ที่ได้

1.6.7 จัดทำเอกสารประกอบวิทยานิพนธ์

1.7 เครื่องมือและอุปกรณ์ที่ใช้ในงานวิจัย

1.7.1 เครื่องคอมพิวเตอร์ส่วนบุคคล

- หน่วยประมวลผลกลางยี่ห้อ Intel รุ่น Core TM i5-2450M ความเร็ว 2.5GHz
- หน่วยความจำหลักขนาด 8 GB

1.7.2 ระบบปฏิบัติการ Windows Seven Professional

1.7.3 โปรแกรม MATLAB

1.8 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้แบ่งเนื้อหาออกเป็น 6 บทดังนี้

บทที่ 1 กล่าวถึงที่มาของงานวิจัยความมุ่งหมายวัตถุประสงค์สมมุติฐานทฤษฎีและแนวคิดที่ใช้ รวมถึงของเขตของการวิจัยและขั้นตอนของการศึกษา

บทที่ 2 กล่าวถึงทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้อง

บทที่ 3 กล่าวถึงปัญหาของการปรับค่าอัตราการเรียนรู้

บทที่ 4 กล่าวถึงแนวคิดที่ใช้แก้ปัญหาการปรับค่าเรียนรู้ที่คงที่เป็นสามารถปรับค่าตัวเองได้และงานวิจัยที่นำเสนอ

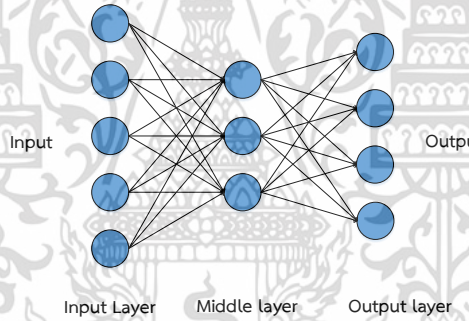
บทที่ 5 กล่าวถึงผลการทดลองเปรียบเทียบการปรับค่าอัตราการเรียนรู้ใช้ค่าคงที่หลายๆค่าเทียบกับการปรับด้วยงานวิจัยที่นำเสนอ

บทที่ 6 กล่าวถึงบทสรุปและข้อเสนอแนะของงานวิจัย

บทที่ 2
ทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้อง

2.1 โครงข่ายประสาทเทียมเบื้องต้น

โครงข่ายประสาทเทียมมีโครงสร้างการทำงานที่เลียนแบบการทำงานของระบบประสาทในสิ่งมีชีวิต ซึ่งสมองประกอบด้วยเซลล์ประสาทที่เรียกกันว่า นิวรอน(neuron) ประมาณ หนึ่งหมื่นล้านเซลล์ และจุดเชื่อมต่อประสาท(synapses) ประมาณ 60 ล้านล้านจุด เทียบได้กับค่าน้ำหนักและโหนดในโครงสร้างของโครงข่ายประสาทเทียม ตามลำดับ กระบวนการคำนวณ จะปรับค่าน้ำหนักให้กับอินพุตตามโครงสร้างการเรียนรู้นั้นๆ โครงข่ายประสาทเทียมมีการนำไปประยุกต์ใช้งานมากมาย เช่น การจดจำรูปแบบ (Pattern recognition), การจับกลุ่ม (Clustering /categorization), การประมาณค่าฟังก์ชัน (Function approximation) ,การทำนาย (Prediction / forecasting), การหาค่าเหมาะที่สุด (Optimization) และ ระบบควบคุม (Control system) เป็นต้น



รูปที่ 2.1 โครงสร้างทั่วไปของโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียมและสมองมนุษย์ (Biological and artificial neural network) ประกอบไปด้วย

1. นิวรอน (Neurons)เซลล์หน่วย
2. นิวเคลียส (Nucleus)ส่วนประกอบผลกลางของนิวรอน
3. เดรนไดน์ Dendriteส่วนของ biological neuron ที่รับอินพุตเข้าสู่เซลล์
4. แอ็กซอน (Axon)จุดต่อด้านออก
5. ไซแนพส์ (Synapse)การเชื่อมต่อ ที่มีการให้น้ำหนัก (Weights) ระหว่างส่วนประมวลผลต่างๆ

การเรียนรู้ของโครงข่ายประสาทเทียมนั้นมีหลายโครงข่าย ยกตัวอย่างที่นิยมกันได้แก่

1. การเรียนรู้โครงข่ายเพอร์เซ็พตรอน (Perceptron learning) ถือเป็นโครงข่ายต้นกำเนิดนิวรอน เนื่องจากมีความซับซ้อนไม่มากทำให้เหมาะกับผู้รู้ศึกษา แต่ลักษณะงานที่แก้ได้จะเป็นปัญหาที่เป็นเชิงเส้นเท่านั้น

2. การเรียนรู้แบบเฮ็บเบียน (Hebbian learning) เมื่อเทียบกับการเรียนรู้โครงข่ายเพอร์เซ็พตรอนก็จะมีความต่างที่มีความสามารถในการแก้สมการเป็นเชิงเส้นได้และยังเป็นจุดเริ่มต้นในการพัฒนาเป็นการเรียนรู้แบบแพร่กลับในเวลาต่อมา

3. การเรียนรู้แบบวิโดร์-ฮอฟฟ์ (Windrow Hoff learning) กระบวนการเรียนรู้เมื่อได้คำตอบแต่ละรอบ จะทำการเปรียบเทียบเอาต์พุตเทียบกับค่าจริง โดยอาศัยหลักการหาความผิดพลาดกำลังสองเฉลี่ยที่น้อยที่สุดในการปรับเส้นโดยอาศัยโครงข่ายADALINE and MADALINE ในการปรับเส้นด้วย
การเรียนรู้แบบแพร่กลับ (Back-propagation learning)ต้องยอมรับว่าเป็นที่รู้จักและนำไปประยุกต์ใช้งานมากที่สุด การเรียนรู้แบบแพร่กลับมาสามารถใช้ฝึกสอนโครงข่ายแบบหลายชั้นได้

4. การเรียนรู้แบบแข่งขัน (Competitive learning) เป็นการเรียนรู้แบบไม่ต้องมีผู้ฝึกสอน จะมีการจัดกลุ่มของข้อมูล
การเรียนรู้ของโครงข่ายทฤษฎีเรโซแนนซ์แบบปรับตัว (Learning by adaptive resonance theory network)

5. การเรียนรู้ของโครงข่ายฟังก์ชันฐานรัศมี (Learning of radial basis function network)

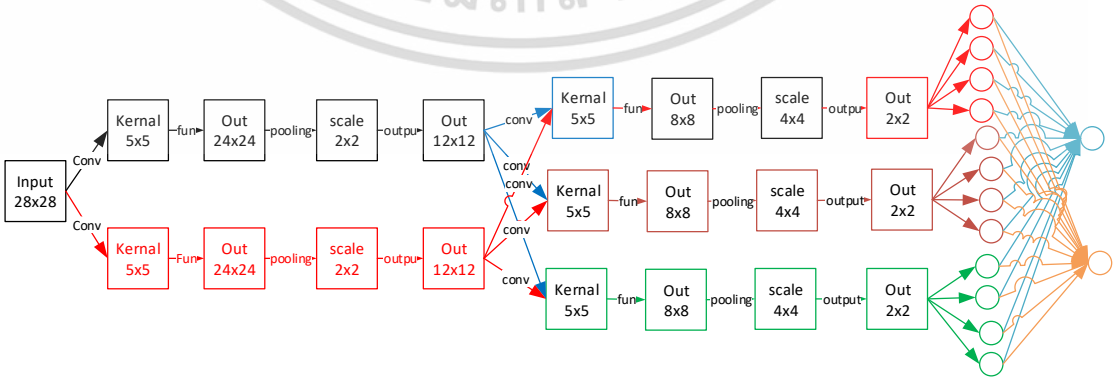
การประยุกต์ใช้งาน

การนำทฤษฎีการเรียนรู้โครงข่ายประสาทเทียมไปประยุกต์ใช้งานสามารถปรับใช้ได้หลากหลาย เช่นการจดจำรูปแบบ Pattern recognition,การจับกลุ่ม (Clustering /categorization),การประมาณค่าฟังก์ชัน Function approximation ,การทำนาย Prediction / forecasting ,การหาค่าเหมาะที่สุด Optimization,หน่วยความจำอ้างอิงด้วยเนื้อหา Content-addressable memory,ระบบควบคุม Control systemเป็นต้น

2.2 Convolutional Neural Network

2.2.1 การทำงานของ Convolutional Neural Network

Convolutional Neural Network มีโครงสร้างดังรูปที่ 2.1 ซึ่งประกอบไปด้วย convolution,Max pooling และ Fully connected ซึ่งการทำงานจะแบ่งออกเป็นสองส่วนคือ Forward และ Backward ซึ่งจะอธิบายตามลำดับ



รูปที่ 2.2 โครงสร้างของ อัลกอริทึม Deep learning

2.2.2 Convolution

Convolution ในส่วนนี้จะแบบได้ออกเป็น 2 ส่วน คือ จำนวนเทมเพลตและขนาดเทมเพลตส่วนที่ 1 เทมเพลต (template) ทำหน้าที่แปลงข้อมูลภาพโดยการกำหนดจำนวนเทมเพลตซึ่งจำนวนเทมเพลตมีผลให้มีการแยกภาพจาก 1 รูปที่เป็นอินพุตและเป็นต้นฉบับให้มีรูปภาพใหม่ตามค่าของเทมเพลตและมีจำนวนเอาต์พุตตามจำนวนเทมเพลตนั้น จำนวนเทมเพลตถือเป็นอีกเรื่องที่สำคัญเช่นกันหากกำหนดจำนวนเทมเพลตน้อยเกินไปอาจทำให้ไม่สามารถแยกแยะความแตกต่างของภาพได้และหากกำหนดจำนวนมากเกินไปการคำนวณก็ใช้เวลานานขึ้นรวมถึงการนำไปใช้งานหลังจากเรียนรู้ก็ไม่เหมาะสมเพราะต้องใช้เวลาคำนวณนานเกินไป และการกำหนดขนาดของเทมเพลตเล็กก็มีความสำคัญไม่ต่างจากการกำหนดจำนวนเทมเพลต เช่นถ้ากำหนดขนาดเล็กเกินไป การเรียนรู้ก็จะใช้เวลานานแต่ผลการเรียนรู้จะออกมาดี แต่ถ้ากำหนดขนาดใหญ่เกินไปจะทำให้การลู่เข้าสู่คำตอบนั้นอาจไม่ได้ผลเพราะอาจจะทำให้มีการแกว่งของ Gradient ได้

วิธีการกรองข้อมูลภาพที่กล่าวมาส่วนใหญ่อาศัยหลักของการหาค่าเฉลี่ยโดยอาจเป็นการหาค่าเฉลี่ยของจุดเดียวกันจากภาพหลายๆ ภาพ หรืออาจเป็นการหาค่าเฉลี่ยจากจุดต่างๆ ที่อยู่รอบๆ จุดที่สนใจ เนื่องจากการหาค่าเฉลี่ยเป็นการลดการเปลี่ยนแปลงของข้อมูลวิธีการนี้จึงใช้ได้ดีกับการกำจัดสัญญาณรบกวนที่เป็นสัญญาณความถี่สูง การกรองสัญญาณมีวัตถุประสงค์เพื่อเน้นคุณสมบัติบางอย่างที่ต้องการในภาพให้เด่นชัดขึ้น ในขณะที่ลดทอนคุณสมบัติที่ไม่ต้องการลง หากเราต้องการเน้นการเปลี่ยนแปลงของระดับความเข้มของจุดต่างๆ ภายในภาพให้เด่นชัดขึ้น ในที่นี่จะเสมือนกับการกรองสัญญาณความถี่สูงผ่านจะไม่สามารถใช้วิธีการหาค่าเฉลี่ยได้วิธีที่สามารถนำมาใช้ได้คือการคอนโวลูชัน (convolution)

การคอนโวลูชันคือการประมวลผลระหว่างภาพ (image) กับ เทมเพลต (template) กำหนดให้เทมเพลตคือเมตริกซ์ขนาด n x m ของชุดตัวเลขที่จะนำไปซ้อนทับภาพที่ตำแหน่งต่างๆ เพื่อหาผลลัพธ์ และ K(x,y) เป็นเทมเพลตขนาด n x m และภาพ I(X,Y) มีขนาด N x M การคอนโวลูชันระหว่างเทมเพลตกับภาพสามารถแสดงได้ดังสมการต่อไปนี้

$$I'(X,Y) = K(I)\ \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} K(i,j)\cdot I(X-i,Y-j) \tag{2.1}$$

$$O(X,Y) = f\big(I'(X,Y)\big) \tag{2.2}$$

$$f(x) = \max(0,x) \tag{2.3}$$

โดย $I'(X,Y)$ คือภาพผลลัพธ์จากการคอนโวลูชัน

$f(x)$ คือฟังก์ชัน Relu

$O(X,Y)$ คือค่าเอาต์พุตในส่วนของการทำคอนโวลูชัน

m,n คือ ขนาดของเทมเพลต

2.2.3 Pooling layer

Max pooling layer ในส่วนนี้จะเป็นการกำหนดขนาด Matrix ของ Max pooling ซึ่งหน้าที่ของ Max pooling จะเป็นการปรับค่าที่ได้จากการคำนวณในส่วน Convolution ซึ่งค่าที่มากที่สุดจะเป็นเอาต์พุตของเล

เอาท์นี้ซึ่งขนาดของ Max pooling ถ้ามีขนาดมากไปการลู่เข้าคำตอบก็จะมีการแกว่ง แต่ถ้าน้อยไปการหลู่เข้าคำตอบก็จะช้า

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

6	8
3	4

รูปที่ 2.3 การทำ Max pooling ขนาด (2X2)

Average pooling layer ก็มีการทำงานคล้ายกับ Max pooling layer แต่ในส่วนนี้จะเป็นการหาค่าเฉลี่ย

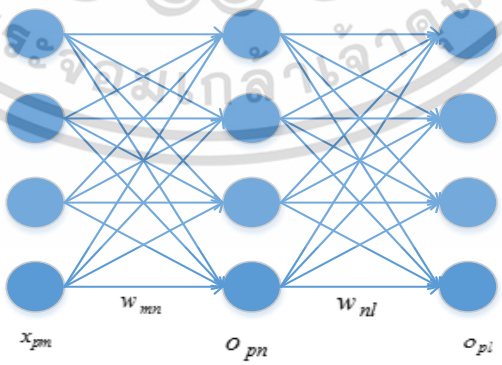
1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

3.25	5.25
2	2

รูปที่ 2.4 การทำAverage pooling ขนาด (2X2)

2.2.4 Fully connected

Fully connected ในส่วนนี้เปรียบเสมือน Forward ในการเรียนในโครงสร้าง neural network ซึ่งต้องกำหนดขนาดของ Hidden layer ว่ามีกี่โหนด และกำหนด output layer ว่ามีกี่โหนดเช่นกันเพื่อให้แยกข้อมูลเป็นกลุ่ม ตามจำนวนเอาต์พุตได้



รูปที่ 2.5 โครงสร้างFully connected

$$o_j = f\left(\sum_{i-1}^{M} w_{ji}\, x_i + b_j\right) \quad (2.4)$$

$$o_{pk} = f\left(\sum_{j-1}^{N} w_{kj}\, x_i + b_k\right) \quad (2.5)$$

โดยที่

o_j, o_{pk} คือ ค่าเอาต์พุตของโหนดซ้อนเร้นและโหนดเอาต์พุตตามลำดับ

x_i, x_j คือ อินพุตของชั้นซ้อนเร้นและโหนดเอาต์พุตตามลำดับ

b_j, b_k คือ ค่าไบอัสของโหนดซ้อนเร้นและโหนดเอาต์พุตตามลำดับ

w_{ji}, w_{kj} คือ ค่าน้ำหนักระหว่างโหนดอินพุตกับโหนดซ้อนเร้นและโหนดซ้อนเร้นกับเอาต์พุตตามลำดับ

o_j, o_{pk} คือ ขนาดของโหนดอินพุต ซ้อนเร้น และเอาต์พุตตามลำดับ

2.2.5 ความผิดพลาดของการเรียนรู้

การวัดประสิทธิภาพของอัลกอริทึมจะวัดจากความผิดพลาดระหว่างเอาร์พุตที่ได้เทียบกับค่าเป้าหมายยิ่งค่าเข้าใกล้ศูนย์เท่าไรก็ยิ่งมีประสิทธิภาพ

การหาค่าของความผิดพลาดที่มีเอาต์พุต1 โหนด สามารถหาได้จาก

$$E = t - y \quad (2.6)$$

ในกรณีที่เอาต์พุตมีแค่ 1 โหนดไม่ค่อยพบบ่อยหนัก มักจะพบกรณีที่เอาต์พุตหลายโหนด ซึ่งจะทำการรวมความผิดพลาดทุกโหนดเข้าด้วยกันในสมการที่2.7 แต่วิธีนี้มีความเหมาะสมกับกรณีที่มีความผิดพลาดมีค่าเป็นบวกทุกค่า หรือเป็นค่าลบทุกค่า อย่างใดอย่างหนึ่งเท่านั้น แต่ถ้าว่ามีความผิดพลาดทั้งค่าบวกและลบผสมกันไม่สามารถใช้วิธีการนี้ในการวัดความผิดพลาดได้

$$E_s = \sum_{L=1}^{L} (t_1 - y_1) \quad (2.7)$$

ความผิดพลาดกำลังสองของข้อมูลจำนวนเอาต์พุต L โหนด เหมาะสมกับข้อมูลที่มีความผิดพลาดบวกลบผสมกันสามารถกำหนดได้ดังนี้

$$E_p = \frac{1}{2}\sum_{L=1}^{L} (t_1 - y_1)^2 \quad (2.8)$$

ความผิดพลาดกำลังสองของชุดข้อมูลทั้งหมด P ชุดข้อมูล สำหรับข้อมูลตัวอย่างทั้งหมด ความผิดพลาดกำลังสองสามารถกำหนดได้ดังนี้

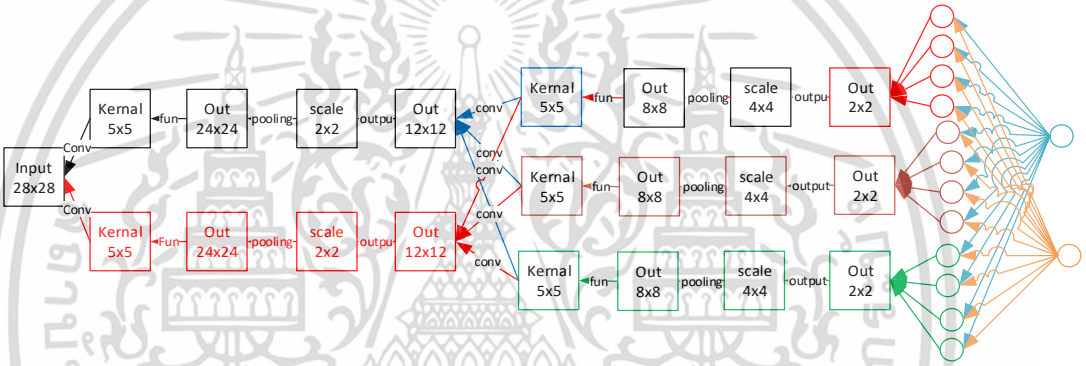
$$E = \sum_{P=1}^{P} \mathrm{E_p} = \frac{1}{2\mathrm{P}} \sum_{\mathrm{P=1}}^{\mathrm{P}} \sum_{\mathrm{L=1}}^{\mathrm{L}} (\mathrm{t_{pl}} - \mathrm{y_{pl}})^2 \quad (2.9)$$

ความผิดพลาดกำลังสองเฉลี่ย สามารถกำหนดได้ดังนี้

$$E_m = \frac{1}{2} \sum_{P=1}^{P} \mathrm{E_p} = \frac{1}{2\mathrm{P}} \sum_{\mathrm{P=1}}^{\mathrm{P}} \sum_{\mathrm{L=1}}^{\mathrm{L}} (\mathrm{t_{pl}} - \mathrm{y_{pl}})^2 \quad (2.10)$$

เมื่อ $\mathrm{E_p}$, E และ E_{m} คือ ความผิดพลาดจำนวนข้อมูล 1 ชุด ความผิดพลาดทุกข้อมูลและ ความผิดพลาดเฉลี่ยตามลำดับ $\mathrm{t_{p1}}$ และ $\mathrm{y_{p1}}$ คือ ค่าเอาต์พุตที่ต้องการและค่าเอาต์พุตจากการคำนวณของ 1_{th} โหนดในชั้นเอาต์พุตที่ตรงกับข้อมูลตัวอย่างที่ $\mathrm{p_{th}}$ ตามลำดับ

2.2.6 กระบวนการความผิดพลาดของการแพร่กลับ



รูปที่ 2.6 กระบวนการแพร่กลับ

อัลกอริทึมการเรียนรู้แพร่กลับแบบแบ็ท batch ซึ่งอัลกอริทึมนี้ได้มาจากสูตรของอัลกอริทึมเรียนรู้แพร่กลับแบบมาตรฐานแต่แตกต่างกันตรงที่ในการอัพเดทค่าน้ำหนัก ในการอัพเดทของอัลกอริทึมแพร่กลับมาตรฐานคือจะดำเนินการหลังจากคำนวณข้อมูลแต่ละชุด แต่สำหรับ batch จะอัพเดทน้ำหนักหลังจากทำเนินกระบวนการเสร็จครบทุกข้อมูล

1. สำหรับค่าน้ำหนัก $\mathrm{w_{li}}$ และไบอัส $\mathrm{b_l}$ การเปลี่ยนแปลงหาได้ตามนี้

$$\Delta \mathrm{w_{li}} = -\eta \frac{\partial \mathrm{E}}{\partial \mathrm{w_{li}}} = \sum_{\mathrm{p=1}}^{\mathrm{P}} (-\eta \frac{\partial \mathrm{E}}{\partial \mathrm{w_{li}}}) = \sum_{\mathrm{p=1}}^{\mathrm{P}} \Delta_{\mathrm{p}} \mathrm{w_{li}} = \eta \sum_{\mathrm{p=1}}^{\mathrm{P}} \delta_{\mathrm{pl}} \mathrm{o_{pl}} \quad (2.11)$$

$$\Delta \mathrm{b_l} = -\eta \frac{\partial \mathrm{E}}{\partial \mathrm{b_l}} = \sum_{\mathrm{p=1}}^{\mathrm{P}} (-\eta \frac{\partial \mathrm{E}}{\partial \mathrm{b_l}}) = \sum_{\mathrm{p=1}}^{\mathrm{P}} \Delta_{\mathrm{p}} \mathrm{b_l} = \eta \sum_{\mathrm{p=1}}^{\mathrm{P}} \delta_{\mathrm{pl}} \quad (2.12)$$

$$\delta_{\mathrm{pl}} = (\mathrm{t_{pl}} - \mathrm{y_{pl}}) \mathrm{y_{pl}} (1 - \mathrm{y_{pl}}) \quad (2.13)$$

เมื่อ w_{li} คือ ค่าน้ำหนักเชื่อมต่อระหว่าง โหนด 1_{th} ในชั้นเอาต์พุตและ โหนด i_{th} ในชั้นซ่อนเร้น b_l คือ ไบอัสของโหนด l_{th} ในชั้นเอาต์พุตηคือ อัตราการเรียนรู้ i คือจำนวนอนุกรมของโหนดในชั้น ซ่อนเร้น $i = 1,2,\ldots,M, M$ คือ ผลรวมของโหนดชั้นซ่อนเร้น Δpคือการเปลี่ยนแปลงข้อมูลที่ p_{th}, δ_{pl} คือสัญญาณความผิดพลาดของโหนดในชั้นเอาต์พุต O_{pi} คือค่าเอาต์พุตของโนด i_{th} ในชั้นซ่อนเร้น ที่ตรงกับข้อมูล p_{th}

2. สำหรับน้ำหนัก w_{li} และไบอัส b_l การเปลี่ยนแปลงค่าได้ตามนี้

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = \sum_{p=1}^{P}(-\eta \frac{\partial E}{\partial w_{ij}}) = \sum_{p=1}^{P} \Delta_p w_{ij} = \eta \sum_{p=1}^{P} \delta_{pi} o_{pj} \quad (2.14)$$

$$\Delta b_i = -\eta \frac{\partial E}{\partial b_i} = \sum_{p=1}^{P}(-\eta \frac{\partial E}{\partial b_i}) = \sum_{p=1}^{P} \Delta_p b_i = \eta \sum_{p=1}^{P} \delta_{pi} \quad (2.15)$$

$$\delta_{pi} = O_{pi}(1 - O_{pi}) \sum_{l=1}^{L} w_{li} \delta_{pl} \quad (2.16)$$

เมื่อ w_{li} คือ น้ำหนักเชื่อมต่อระหว่างโหนด i_{th} ในชั้นซ่อนเร้นและโหนด j_{th} ในชั้นอินพุต b_l คือไบอัสของโหนด i_{th} ในชั้นซ่อนเร้น O_{pi} คือ ค่าเอาต์พุตของโหนดชั้นอินพุตที่ตรงกับข้อมูล p_{th}, δ_{pl} คือสัญญาณความผิดพลาดในโหนดชั้นซ่อนเร้น

3. สำหรับน้ำหนัก k_{li} และไบอัส b_l การเปลี่ยนแปลงค่าที่อยู่ในชั้น Convolution จะมีการคำนวณดังนี้

$$\Delta k_{ij} = -\eta \frac{\partial E}{\partial k_{ij}} = \sum_{p=1}^{P}(-\eta \frac{\partial E}{\partial k_{ij}}) = \sum_{p=1}^{P} \Delta_p k_{ij} = \eta \sum_{p=1}^{P} \mathrm{conv}(\delta_{pl} o_{pl}) \quad (2.17)$$

$$\Delta b_i = -\eta \frac{\partial E}{\partial b_i} = \sum_{p=1}^{P}(-\eta \frac{\partial E}{\partial b_i}) = \sum_{p=1}^{P} \Delta_p b_i = \eta \sum_{p=1}^{P} \delta_{pi} \quad (2.18)$$

$$\delta_{pi} = O_{pi}(1 - O_{pi}) \sum_{l=1}^{L} \mathrm{conv}(w_{li} \delta_{pl}) \quad (2.19)$$

อัลกอริทึม Batch ไม่ได้หาความผิดพลาดกำลังสองกับข้อมูลเพียงชุดเดียว แต่เหมาะกับการหาความผิดพลาดกำลังสองของข้อมูลทั้งหมด

2.2.7 การประมาณค่าฟังก์ชันด้วยวิธีของอนุกรมเทย์เลอร์ (Taylor series Method)

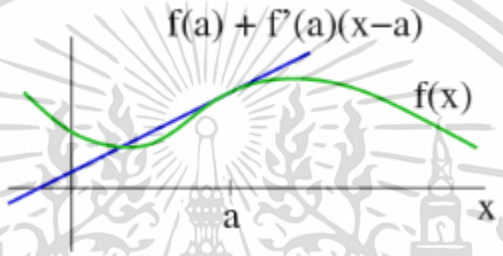
พหุนามเทย์เลอร์ เป็นพหุนามที่ใช้ประมาณค่าในบริเวณใกล้ๆกับจุดใดจุดหนึ่งที่ทราบค่าแน่นอน ยิ่งถ้าค่าที่ต้องประมาณนั้นใกล้กับจุดที่ทราบค่าจะทำให้ค่าที่ประมาณแม่นยำขึ้นแต่ถ้าค่าที่ต้องการประมาณอยู่ห่างจากจุดดังกล่าว ก็จะทำให้ความคลาดเคลื่อนของการประมาณเกินขึ้นได้มาก

2.2.8 การประมาณค่าสมการ 1 ตัวแปร

ที่อยู่ในรูปฟังก์ชันในรูป$f(x)$ซึ่งเราสามารถประมาณหาค่าที่ใกล้เคียง หรือรอบๆ สำหรับตัวอย่าง, การประมาณค่าฟังก์ชันด้วยทฤษฎีเทย์เลอร์ 1 ตัวแปร

$$f(x) \approx f(a) + f'(a)(x-a) \tag{2.20}$$

สมการเส้นตรงที่แสดงด้วยกราฟสีน้ำเงิน ที่เกิดจากการประมาณค่า $f(x)$ ที่แสดงด้วยกราฟสีเขียว ที่จุด $x = a$ ซึ่งตรงกับจัดความชัดของฟังก์ชันที่จุด a



รูปที่ 2.7 กราฟตัวอย่างของเทร์เลอร์

การประมาณค่าจากข้างต้นจะพบได้ว่า มีค่าแค่จุดเดียวที่มีค่าตรงกัน ซึ่งเราสามารถประมาณค่าฟังก์ชันให้เข้าใกล้ได้โดยการเพิ่มลำดับ ในการประมาณค่ามากขั้น เช่นการประมาณค่าในรูป ลำดับที่ 2 ดังสมการ

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 \tag{2.21}$$

หรือการประมาณค่าในฟังก์ชันเทอมสูงๆดังรูปสมการ

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \frac{1}{6}f'''(a)(x-a)^3 + \cdots \tag{2.22}$$

2.2.9 การประมาณค่าสมการหลายตัวแปร

รูปแบบสมการทั่วไปของ เทย์เลอร์ที่มีหลายทั่วแปรสามารถเขียนได้ดังสมการ

$$\boldsymbol{f(x) = f(x_1, x_2, \cdots, x_n)} \tag{2.23}$$

จากสมการ 1 ตัวแปรเมื่อนำมาใช้กับหลายตัวแปร มีการจัดรูปแบบการคำนวณเป็นแบบเมทริก ซึ่งเขียนในสมการใหม่

$$f(x) \approx f(a) + Df(a)(x-a) + \frac{1}{2}(x-x)^{T}Hf(a)(x-a) \tag{2.24}$$

ตัวอย่างการประมาณค่าฟังก์ชันด้วยทฤษฎีเทย์เลอร์ ในอันดับ 2 ในตำแหน่ง (0,0) และ ตำแหน่ง (1,2)

$$f(x,y) = e^{-(x^2+y^2)} \tag{2.9}$$

ประมาณค่าสมการที่อยู่ใกล้จุด (a , b)

$$f(x,y) \approx f(a,b) + Df(a,b)\begin{bmatrix} x-a \\ y-b \end{bmatrix} + \frac{1}{2}\begin{bmatrix} x-a \\ y-b \end{bmatrix}^T Hf(a,b)\begin{bmatrix} x-a \\ y-b \end{bmatrix} \tag{2.26}$$

คำนวณในรูปอนุพันธ์ย่อย

$$\frac{\partial f}{\partial x}(x,y) = -2xe^{-(x^2+y^2)} \tag{2.27}$$

$$\frac{\partial f}{\partial y}(x,y) = -2ye^{-(x^2+y^2)} \tag{2.28}$$

$$\frac{\partial^2 f}{\partial x^2}(x,y) = (-2+4x^2)e^{-(x^2+y^2)} \tag{2.29}$$

$$\frac{\partial^2 f}{\partial y^2}(x,y) = (-2+4y^2)e^{-(x^2+y^2)} \tag{2.30}$$

$$\frac{\partial^2 f}{\partial x\,\partial y}(x,y) = \frac{\partial^2 f}{\partial y\,\partial x}(x,y) = 4xye^{-(x^2+y^2)} \tag{2.31}$$

การประมาณค่าในตำแหน่ง 0,0

$$f(0,0) = e^{-(0^2+0^2)} = 1 \tag{2.32}$$

$$\frac{\partial f}{\partial x}(0,0) = -2(0)e^{-(0^2+0^2)} = 0 \tag{2.33}$$

$$\frac{\partial f}{\partial y}(0,0) = -2(0)e^{-(0^2+0^2)} = 0 \tag{2.34}$$

$$\frac{\partial^2 f}{\partial x^2}(0,0) = (-2+40^2)e^{-(0^2+0^2)} = -2 \tag{2.35}$$

$$\frac{\partial^2 f}{\partial y^2}(0,0) = (-2+40^2)e^{-(0^2+0^2)} = -2 \tag{2.36}$$

$$\frac{\partial^2 f}{\partial x\,\partial y}(0,0) = \frac{\partial^2 f}{\partial y\,\partial x}(0,0) = 0 \tag{2.37}$$

$$f(x,y) \approx f(0,0) + Df(0,0)\begin{bmatrix} x-0 \\ y-0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} x-0 \\ y-0 \end{bmatrix}^T Hf(0,0)\begin{bmatrix} x-0 \\ y-0 \end{bmatrix} \approx 1 + [0\ 0]\begin{bmatrix} x \\ y \end{bmatrix} + \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^T \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \quad (2.38)$$

$$\approx 1 - x^2 - y^2 \quad (2.39)$$

การประมาณค่าในตำแหน่ง 1,2

$$f(1,2) = e^{-(1^2+2^2)} = e^{-5} \quad (2.40)$$

$$\frac{\partial f}{\partial x}(1,2) = -2(1)e^{-(1^2+2^2)} = -2e^{-5} \quad (2.41)$$

$$\frac{\partial f}{\partial y}(1,2) = -2(2)e^{-(1^2+2^2)} = -4e^{-5} \quad (2.42)$$

$$\frac{\partial^2 f}{\partial x^2}(1,2) = (-2 + 4(1^2))e^{-(1^2+1^2)} = 2e^{-5} \quad (2.43)$$

$$\frac{\partial^2 f}{\partial y^2}(1,2) = (-2 + 4(2^2))e^{-(2^2+2^2)} = 14e^{-5} \quad (2.44)$$

$$\frac{\partial^2 f}{\partial x\,\partial y}(1,2) = \frac{\partial^2 f}{\partial y\,\partial x}(1,2) = 8e^{-5} \quad (2.45)$$

$$f(x,y) \approx e^{-5} + [-2e^{-5} - 4e^{-5}]\begin{bmatrix} x-1 \\ y-2 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} x-1 \\ y-2 \end{bmatrix}^T \begin{bmatrix} 2e^{-5} & 8e^{-5} \\ 8e^{-5} & 14e^{-5} \end{bmatrix}\begin{bmatrix} x-1 \\ y-1 \end{bmatrix} \quad (2.46)$$

$$\approx e^{-5}(1 - 2(x-1) - 4(y-2) + (x-1)^2 + 8(x-1)(y-2) + 7(y-2)^2 \quad (2.47)$$

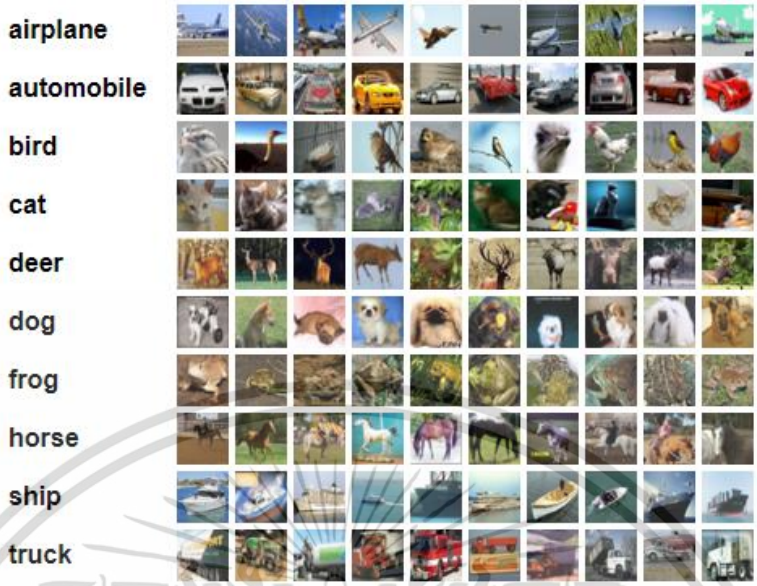
บทที่ 3
ปัญหาของการปรับค่าอัตราการเรียนรู้

ในบทนี้จะนำเสนอผลการทดลองที่มีการปรับค่าอัตราการเรียนที่มีค่าต่างๆ เพื่อสังเกตผลการทดลองว่าอัตราการเรียนรู้ที่ส่งผลต่อความคิดผิดพลาดอย่างไร

Dataset ในการทดลองจะประกอบไปด้วยชุดข้อมูลMNIST ดังรูปที่ 3.1 เป็นข้อมูลรูปภาพที่มีการเขียนด้วยหลายมือตั้งแต่ เลข 0 – 9 จำนวน 70,000 ข้อมูล ขนาด 28x28 เป็นรูปแบบโทนสีเทา โดยแบ่งรูปออกเป็น60,000 รูปใช้สำหรับเรียนรู้ และ 10,000 รูปใช้สำหรับทดสอบ

รูปที่ 3.1 ตัวอย่างชุดข้อมูลMNIST

ข้อมูลชุด CIFAR-10 ดังรูปที่ 3.2 ประกอบไปด้วย รูปภาพสี 60,000 รูป มีขนาด 32x32 แบ่งเป็น 10 กลุ่ม ได้แก่ เครื่องบิน, รถยนต์, นก, แมว, กว้าง, หมา, กบ, ม้า, เรือ, รถบรรทุก ซึ่งมีรูปภาพจำนวน 60,000รูปภาพสามารถแบ่งเป็น 50,000 รูปภาพสำหรับการเรียนรู้ และ 10,000 รูปภาพสำหรับทดสอบ.



รูปที่ 3.2 ตัวอย่างชุดข้อมูลCIFAR-10

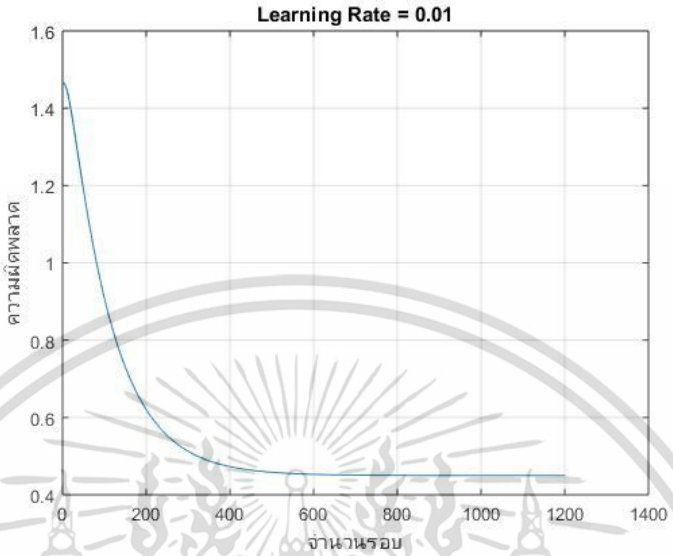
3.1 การทดลองข้อมูลชุด MNIST

ในการทดลองจะกำหนดอัตราการเรียนรู้อยู่ที่ 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1,2,3,4,5,10,20 โดยมีโครงสร้างในการทดลอง ในตารางที่ 3.1

ตารางที่ 3.1 โครงสร้างในการทดลอง

Parameters	Values
Data training	60,000 images
Data testing	10,000 images
Output label	10 labels
Convolution layer 1 Output maps	6 maps
Convolution layer 1 kernel size	5 x 5
Convolution layer 2	12 maps
Convolution layer 2 kernel size	5 x 5
batch size	500 images

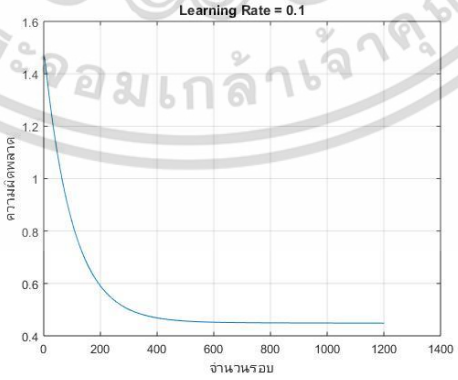
3.1.1 อัตราการเรียนรู้ที่มีค่าเข้าใกล้ 0



รูปที่ 3.1 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.01

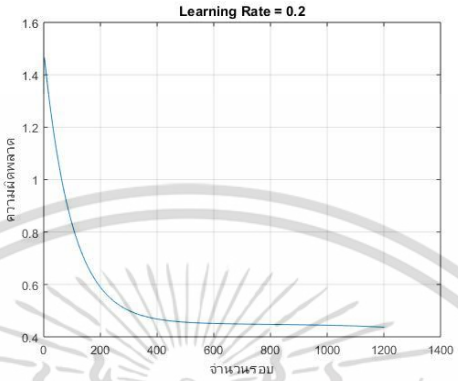
จากกราฟการทดลองโดยกำหนดให้แนวตั้งแสดงความผิดพลาดจากกเรียนรู้ และแนวนอนแสดงจำนวนรอบ และการทำลองจะกำหนดค่าอัตราการเรียนรู้เท่ากับ 0.1 นั้นจะสังเกตผลการทดลองจะได้ว่ามีค่าความผิดพลาดจากการเรียนรู้ลดลงมาที่อยู่ที่ค่าคงที่ค่าหนึ่งและไม่สามารถลดค่าความผิดพลาดลดลงไปกว่านี้ถึงจะเพิ่มจำนวนรอบมาขึ้นเรื่อย ๆ ก็ตามจากรูปจะเห็นว่าจำนวนรอบที่ 600 -1200 ความผิดพลาดที่ได้จากเรียนรู้จะคงที่อยู่ค่าใดค่าหนึ่งและไม่สามารถลดความผิดพลาดลงได้อีก

3.1.2 อัตราการเรียนรู้ที่มีค่าระหว่าง 0.1 -1



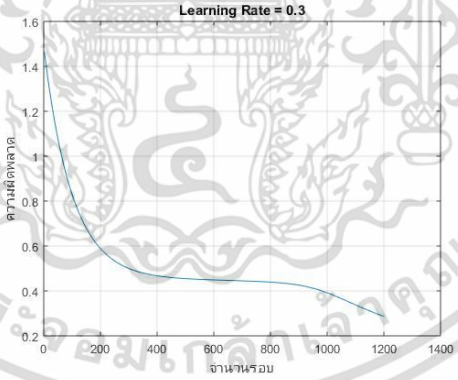
รูปที่ 3.3 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.1

จากกราฟการทดลองเมื่อกำหนดอัตราการเรียนรู้อยู่ที่ 0.1 จะเห็นได้ว่าความผิดพลาดมีการลดลงอย่างช้า ๆ และลงมาที่ค่าคงที่ค่าหนึ่ง เมื่อทำการเปรียบเทียบกับการทดลองที่กำหนดอัตราการเรียนรู้อยู่ที่ 0.01 นั้น จะเห็นได้ว่าแนวโน้นของกราฟทั้งสองมีผลการทดลองคล้ายกัน



รูปที่ 3.4 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.2

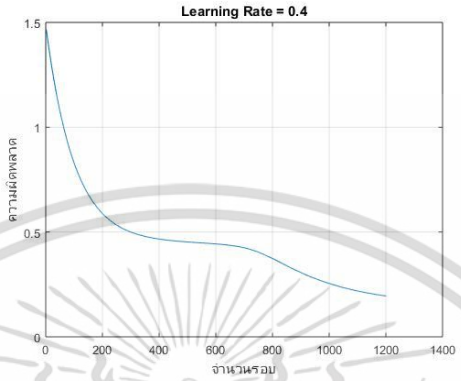
จากกราฟการทดลองมีการเพิ่มอัตราการเรียนรู้จาก 0.1 เป็น 0.2 เพื่อสังเกตความผิดพลาดจากการเรียนรู้ว่าอัตราการเรียนรู้ที่น้อยที่สุดที่จะส่งผลให้การเรียนรู้มีการปรับค่าความผิดพลาดลดลงได้ จากผลการทดลองจะเห็นได้ว่าความผิดพลาดมีการลดลงมาที่ค่าคงที่ค่าหนึ่งซึ่งสรุปได้ว่า ค่าอัตราการเรียนรู้ 0.2 ยังไม่เป็นค่าที่น้อยที่สุดที่สามารถลดค่าความผิดพลาดลงได้



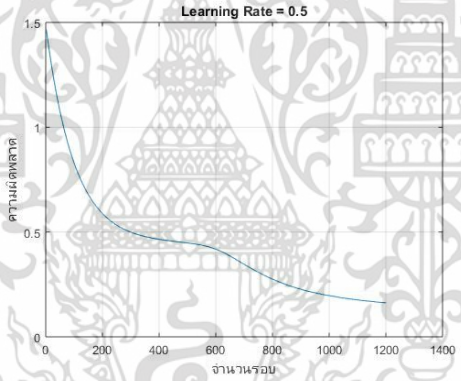
รูปที่ 3.5 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.3

จากกราฟการทดลองจะกำหนดค่าอัตราการเรียนรู้อยู่ที่ 0.3 ซึ่งเพิ่มมาจากเดิมอีก 0.1 เพื่อสังเกตความผิดพลาดว่าจะมีแนวโน้นไปทิศทางใด จากซังเกตจะได้ว่าได้ลักษณะกราฟจะมีการลดลง ซึ่งต่างจากการทดลองก่อนหน้าที่ค่าความผิดพลาดจะค่าที่ค่าใดค่าหนึ่ง เพราะฉะนั้นสามารถสรุปได้ว่า ค่าอัตราการเรียนรู้ที่ 0.3 คือค่าอัตราการเรียนรู้ที่น้อยที่สุดที่จะทำให้ค่าความผิดพลาดมีการลดลงได้เมื่อเพิ่มจำนวนรอบการเรียนรู้มากขึ้น

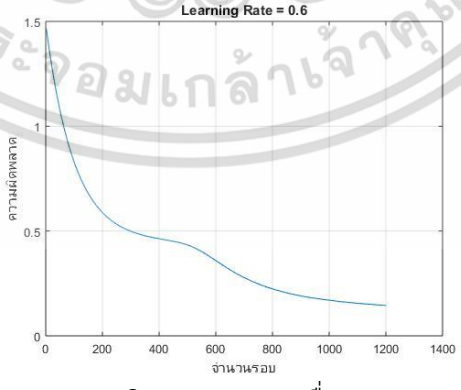
การทดลองต่อไปนี้จะทำการเพิ่มค่าอัตราการเรียนรู้ขึ้นทีละ 0.1 จนถึง 1 ซึ่งอยู่ในช่วงที่นักวิจัยส่วนใหญ่เลือกใช้ เพื่อสังการค่าความผิดพลาดว่ามีแนวโน้นไปทิศทางใดและเพื่อหาว่าค่าอัตราการเรียนรู้ใดที่ส่งผลให้ความผิดพลาดมีค่าน้อยที่สุด



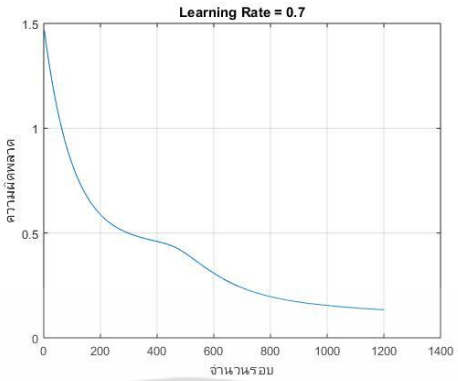
รูปที่ 3.6 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.4



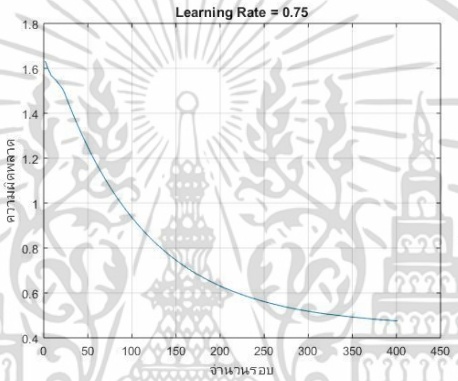
รูปที่ 3.7 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.5



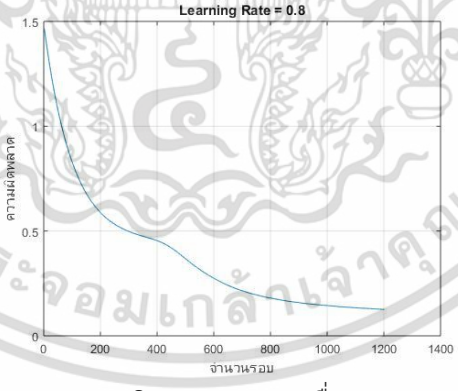
รูปที่ 3.8 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.6



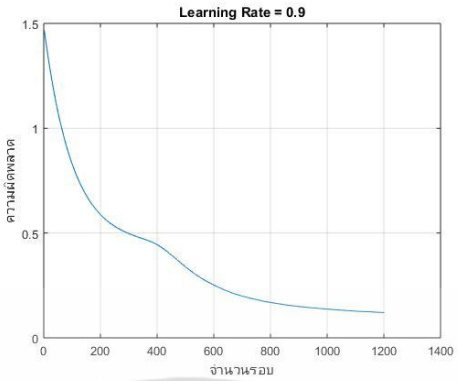
รูปที่ 3.9 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.7



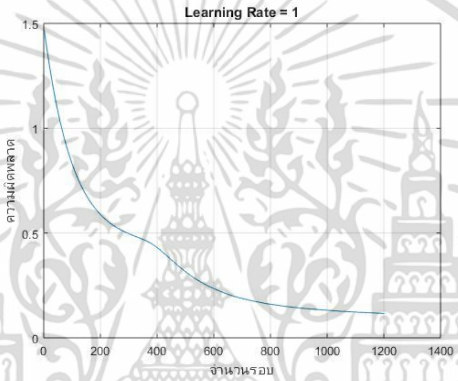
รูปที่ 3.10 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.75



รูปที่ 3.11 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.8



รูปที่ 3.12 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.9

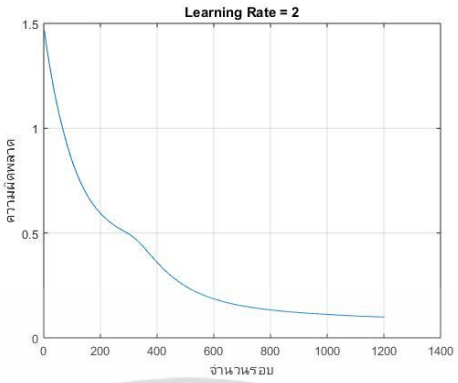


รูปที่ 3.13 แสดงความผิดพลาด RMS เมื่อ Learning Rate =1

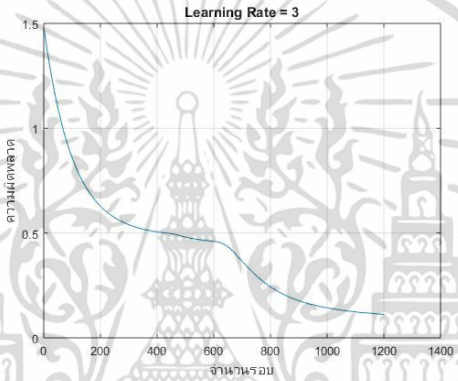
จากรูปการทดลองที่กำหนดอัตราการเรียนรู้ตั้งแต่ 0.1-1นั้นและมีการเพิ่มระดับการเรียนรู้อยู่ที่ 0.1 เพื่อสังเกตว่าอัตราการเรียนรู้แต่ละค่ามีผลต่อความผิดพลาดอย่างไร จากการสังเกตผลการทดลองจะเห็นได้ว่าความผิดพลาดจะแปรผกผันตามอัตราการเรียนรู้ อธิบายได้ว่าเมื่อทำการเพิ่มค่าอัตราการเรียนรู้มากขึ้งค่าความผิดผลาดก็จะลดลงน้อยตาม

3.1.3 อัตราการเรียนรู้ที่มีค่ามากกว่า 2-10

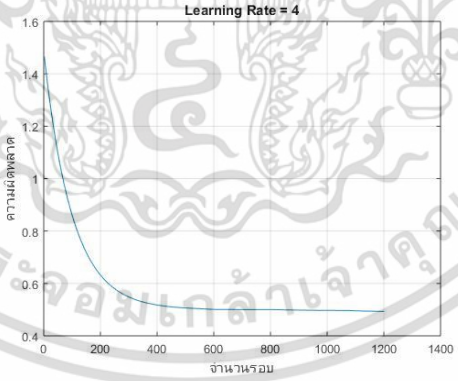
จากการทดลองก่อนหน้านี้จะเห็นได้ว่าเมื่อเพิ่มอัตราการเรียนรู้ก็จะส่งผลให้ความผิดพลาดลดน้อยลง การทดลองต่อไปนี้จะทำการเพิ่มอัตราการเรียนรู้เพิ่มขึ้นทีละ 1 ค่า เพื่อจะสังเกตว่าอัตราการเรียนรู้ค่าใดที่ส่งผลให้ความผิดพลาดไม่สามารถลดลงได้ โดยทำการทดลองตั้งแต่ 2– 10



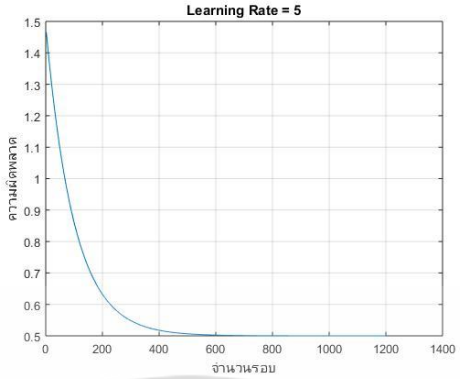
รูปที่ 3.14 แสดงความผิดพลาด RMS เมื่อ Learning Rate =2



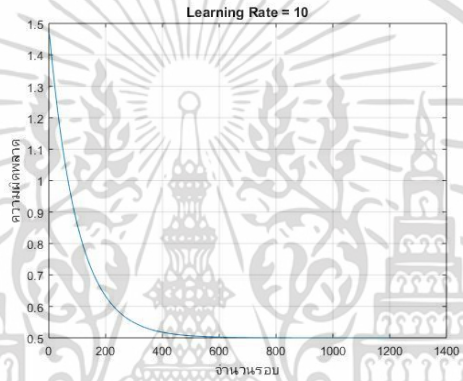
รูปที่ 3.15 แสดงความผิดพลาด RMS เมื่อ Learning Rate =3



รูปที่ 3.16 แสดงความผิดพลาด RMS เมื่อ Learning Rate =4



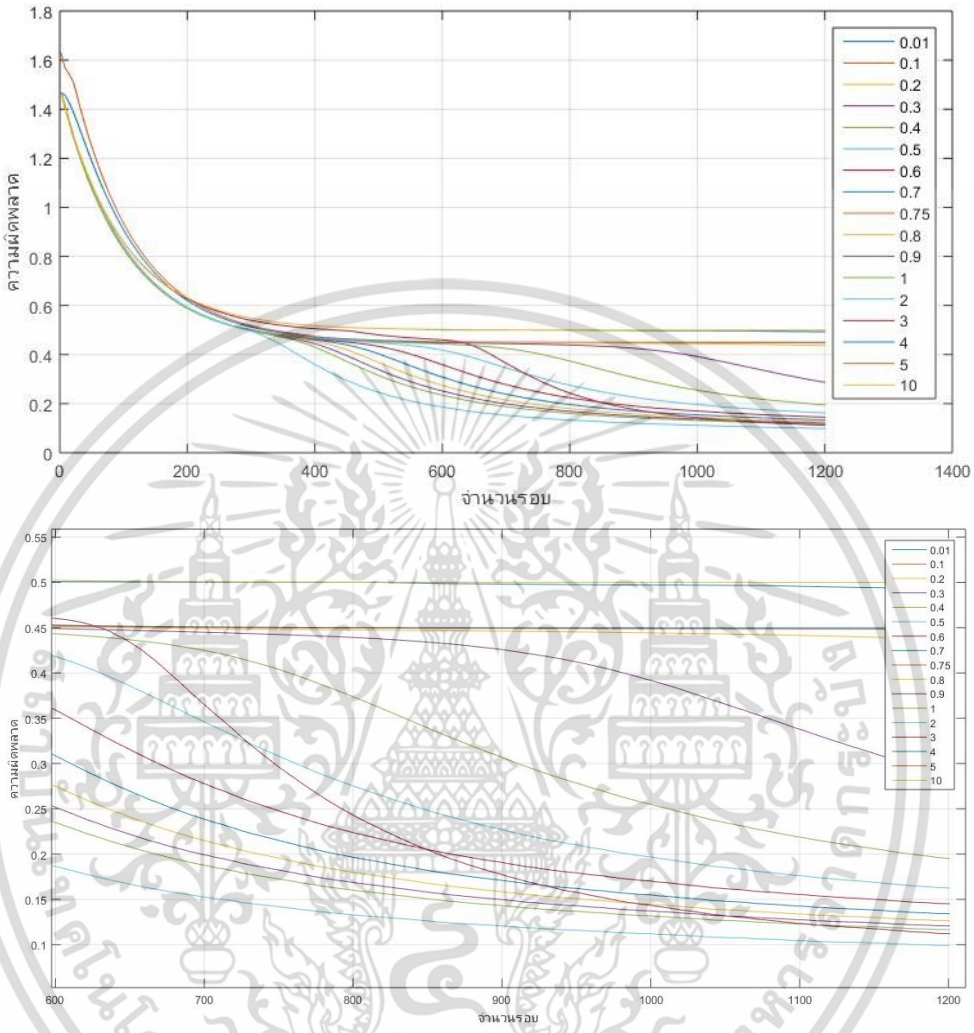
รูปที่ 3.17 แสดงความผิดพลาด RMS เมื่อ Learning Rate =5



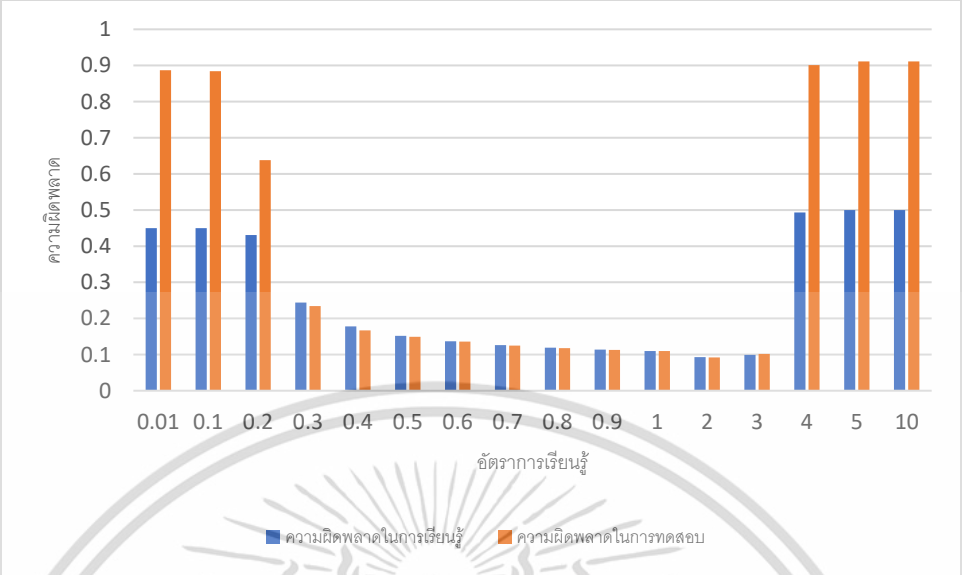
รูปที่ 3.18 แสดงความผิดพลาด RMS เมื่อ Learning Rate =10

จากการทดลองเมื่อเพิ่มค่าอัตราการเรียนรู้เพิ่มขึ้นจากเดิมทีละ 1 ค่านั้นจากผลการทดลองสังเกตได้ว่า ในช่วงอัตราการเรียนรู้ที่ 2 และ 3 สามารถลดค่าความผิดพลาดลงได้ ตั้งเมื่อเพิ่มอัตราการเรียนรู้เรียนมากขึ้นจะเห็นได้ชัดว่าอัตราการเรียนรู้ที่มีค่าตั้งแต่4จนถึง 10 นั้นไม่สามารถลดค่าความผิดพลาดลงได้ และยังคงค่าความผิดพลาดอยู่ที่ค่าใดค่าหนึ่งถึงแม้จะมีจำนวนรอบมาขึ้นมาก็ตามจะสรุปได้ว่าอัตราการเรียนรู้ที่มากที่สุดที่ยังส่งผลให้ความผิดพลาดจากการเรียนรู้ลดลงได้นั้นคืออัตราการเรียนรู้เท่ากับ 3 และเมื่อเพิ่มค่าอัตราการเรียนรู้มากกว่า 3 แล้วจะส่งผลให้การเรียนรู้ไม่สามารถลดค่าความผิดพลาดลงได้

3.1.4 สรุปความผิดพลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ระหว่าง 0.01 – 10



รูปที่ 3.19 แสดงความผิดพลาด RMS เมื่อ Learning Rate =0.01-10



รูปที่ 3.20 กราฟเปรียบเทียบค่าความผิดพลาด ในการเรียนรู้และทดสอบ

ตารางที่ 3.2 เปรียบเทียบความผิดผลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ โดยใช้ข้อมูลMNIST

อัตราการเรียนรู้	ความผิดพลาดในการเรียนรู้	เวลาที่ใช้ (นาที)	ความผิดพลาดในการทดสอบ
0.01	0.450	28.828	0.887
0.1	0.450	27.78	0.884
0.2	0.431	26.3	0.638
0.3	0.244	26.1	0.234
0.4	0.178	27.79	0.167
0.5	0.152	26.11	0.149
0.6	0.137	29.43	0.136
0.7	0.126	27.58	0.125
0.8	0.119	27.53	0.118
0.9	0.114	30.58	0.113
1	0.110	28.15	0.110
2	0.093	28	0.092
3	0.099	29.17	0.102
4	0.493	27.18	0.901
5	0.5	27.18	0.911
10	0.5	27.18	0.911

จากรูปที่ 3.21 และ ตารางที่ 3.2เมื่ออัตราการเรียนรู้มีค่าน้อยมากๆ ในการทดลองได้กำหนดเป็น 0.01 ซึ่งจะเห็นได้ว่า ผลความผิดพลาดจะคงที่อยู่ที่ 0.450 และความผิดพลาดในการทดสอบก็สูงมาก อยู่ที่ 0.0887 ในกรณีนี้สามารถสรุปได้ว่า อัตราการเรียนรู้ที่น้อยมากๆ ทำให้ผลของการเรียนรู้จะคงที่อยู่ค่าหนึ่งถึงจะให้มีการเรียนรู้ต่อไปนานแค่ไหนก็ตาม และเมื่อเมื่อปรับค่าอัตราการเรียนรู้ จาก 0.1 เพิ่มทีละ 0.1 จนถึง 1ซึ่งเป็นค่าช่วงที่นักวิจัยนิยมเลือกใช้ จะเห็นว่าอัตราการเรียนรู้ที่เพิ่มขึ้นทำให้ความผิดพลาดในการเรียนรู้ลดลงตามลำดับ และในการทดลองได้เพิ่มอัตราการเรียนรู้เพิ่มขึ้นจาก 1 ถึง 10 โดยเพิ่มทีละ 1 จะเห็นว่าความผิดพลาดที่น้อยที่สุดคืออัตราการเรียนรู้ที่มีค่าเท่ากับ 2 แต่ถ้าอัตราการเรียนรู้ที่มีค่ามากขึ้นค่าความผิดพลาดก็ไม่สามารถลดลงได้ และมีค่าคงที่อยู่ค่าใดค่าหนึ่งเท่านั้น

จากการทดลองสามารถสรุปได้ 2 ประเด็นคือ

1. ค่าอัตราการเรียนรู้ที่ส่งผลให้ความผิดพลาดน้อยที่สุดคือ 2 ซึ่งในทางทำงานจริง นักวิจัยไม่สามารถรู้ได้เลยว่าอัตราการเรียนรู้ค่าใดที่ส่งผลให้ความผิดพลาดน้อยที่สุด ซึ่งในประเด็นทางผู้วิจัยจึงได้พยายามหาวิธีที่จะประมาณค่าค่าอัตราการเรียนรู้ให้สามารถเลือกค่าอัตราการเรียนรู้ที่ดีที่สุด ซึ่งจะอธิบายในหัวข้อต่อไป
2. ค่าอัตราการเรียนรู้ที่ส่งผลให้ความผิดพลาดลดลงได้ซึ่งอยู่ในช่วงตั้งแต่ 0.3 จนถึง 3 เท่านั้น

3.2 การทดลองข้อมูลชุด CIFAR-10

ในการทดลองจะกำหนดค่าอัตราการเรียนรู้ ซึ่งโครงสร้างในการทดลองนั้นจะทำการทดลองโครงในรูปแบบต่าง ๆ เพื่อค้นหาว่าโครงสร้างใดที่เหมาะสมกับโครงสร้างข้อมูล CIFAR-10

3.2.1 ทดลองโครงสร้างในการทดลอง

โครงสร้างในการทดลองจะประกอบไปด้วยหลายๆ layer และแต่ละ layer นั้นจะประกอบไปด้วย จำนวนเอาต์พุต, kernel size และขนาดของ Pooling ซึ่งโครงสร้างเป็นเรื่องแรกที่ต้องคำนึงถึงว่าค่าใดจะเหมาะสมใน dataset แต่ละชนิด นอกจากโครงสร้างที่พูดถึงแล้วยังมีขนาดของ Batch size และจำนวนรอบ (Epochs) ที่ส่งผลต่อค่าความผิดพลาดในการเรียนรู้และทดสอบ ดังนั้นผู้วิจัยได้ทำการทดสอบโดยการเปลี่ยนค่าต่างๆที่ว่ามาข้างต้นเพื่อสังเกตว่าโครงสร้างชุดใดที่ให้ค่าความผิดพลาดน้อยที่สุดก็จะเลือกโครงสร้างนั้นเป็นโครงสร้างในการทดลองต่อไปสำหรับข้อมูลชุด CIFAR-10

ตารางที่ 3.3 ตารางสรุปการผลทดลองเมื่อเปลี่ยนโครงสร้างในการทดลองในค่าต่าง ๆ

Layer 1			Layer 2			Layer 3			Batch size	Epochs	Error learning	Error Test	Time (hr)	คอมพิวเตอร์
Output	Kernel	pooling	Output	Kernel	pooling	Output	Kernel	pooling						
8	5	2	24	5	2				20	100	0.293	0.448	28.98	B
12	5	2	36	5	2				50	200	0.283	0.43	30.66	A
12	5	2	20	5	2				20	100	0.3	0.464	14.57	A
12	5	2	24	5	2						0.315	0.481	18.57	B
18	5	2	36	5	2				50	100	0.327	0.509	22.56	A
18	5	2	36	5	2				200	60	0.375	0.62	12.32	A
9	5	2	10	5	2	36	2	2	100	100	0.315	0.484	10.27	A
8	5	2	24	5	2	36	2	2	20	100	0.314	0.484	73.27	B
9	5	2	18	5	2	36	2	2	200	100	0.336	0.527	10.08	A
8	5	2	24	5	2	36	2	2	20	100	0.314	0.484	73.27	B
8	5	2	16	5	2	24	2	2	100	60	0.351	0.559	4.67	A
12	5	2	24	5	2	45	2	2	20	100	0.291	0.448	40.63	A
18	5	2	36	5	2	45	2	2	50	60	0.335	0.528	22.53	A
15	5	2	30	5	2	48	2	2	50	60	0.346	0.62	17.74	A
12	5	2	24	5	2	36	2	2	200	30	0.33	0.616	27.49	B
12	5	2	24	5	2	48	2	2	20	20	0.4	0.635	9.19	A

เครื่องคอมพิวเตอร์ส่วนบุคคล A

- หน่วยประมาลผลกลางยี่ห้อ Intel รุ่น Core TM i5-2450M ความเร็ว 2.5GHz
- หน่วยความจำหลักขนาด 8 GB
- ระบบปฏิบัติการ windows 10

เครื่องคอมพิวเตอร์เซิร์ฟเวอร์ B

- หน่วยประมาลผลกลางยี่ห้อ Intel (r) รุ่น Xeon(r) CPU X3430 ความเร็ว 2.40 GH
- หน่วยความจำหลักขนาด 8 GB
- ระบบปฏิบัติการ windows Server 2016

จากตารางผลการทดลองสามารถสรุปแยกเป็นประเด็นต่าง ๆ ได้ดังนี้คือ

1. จำนวน layer ก็มีผลต่อการทดลอง จะเห็นว่าจำนวน layer มากขึ้นจะส่งผลให้ความผิดพลาดมีความผิดพลาดลดลงตามจำนวน layer แต่ในข้อดีก็ยังมีข้อเสียคือเวลาใช้ในการประมวลผลก็ใช้เวลามากขึ้นตามจำนวน layer เช่นเดียวกัน

2. จำนวน output แต่ละ layer ก็มีผลต่อการเรียนรู้เช่นกัน จำนวน output มากขึ้นการที่แยกเป็นกลุ่มของข้อมูลก็ได้เยอะขึ้นเช่นกัน แต่ข้อเสียก็จะใช้เวลาในการเรียนรู้มากเกินไปเพราะฉะนั้นการเลือกจำนวน output ไม่ใช้แค่ทำให้ได้ค่าความผิดพลาดที่น้อยลง แต่ต้องพิจารณาถึงเวลาที่ใช้ไปด้วยเช่นกัน

3. pooling ถ้าเรากำหนดขนาดใหญ่เกินไปจะทำการวิเคราะห์ของภาพจะเป็นแบบหยาบหรือไม่ละเอียดนั้นเองซึ่งทำให้การแบ่งกลุ่มก็ทำได้ยากยิ่งขึ้นนั้นหมายความว่าความผิดพลาดก็จะมากขึ้นมาได้ และอาจทำให้การเรียนรู้ไม่คอนเวอร์เจนซ์

4. batch size:ก็มีผลกับการทำงานของโปรแกรมที่สำคัญเพราะส่งผลไปยังอุปกรณ์ที่ใช้ทำงานว่าทำงานหนักหรือน้อยด้วย ถ้ากำหนดค่ามากเกินไปจะส่งผลกระทบกับหน่วยความจำ RAM เนื่องจากการคำนวณต้องกมีการเก็บข้อมูลที่มหาศาล และความผิดพลาดของการเรียนรู้จะมีค่ามากกว่าเมือเทียบกับจำนวน

5. batch size ที่น้อยกว่า แต่ถ้ากำหนดจำนวน batch size น้อยเกินไปความผิดพลาดจากการทำงานของโปรแกรมก็จะมีการแกว่งและอาจส่งผลให้การเรียนรู้ไม่คอนเวอร์เจนซ์ได้
อุปกรณ์คอมพิวเตอร์ที่ใช้ในการรันโปรแกรมจะไม่มีผลกับการทำงานแต่จะมีผลในเรื่องของเวลาในการทำงานเท่านั้น

จากการทดลองในตารางที่ ทางทีมวิจัยจึงได้เลือกโครงสร้างดังต่อไปนี้เพื่อใช้ในการทดสอบหาค่าความผิดพลาดในกรณีที่มีการกำหนดค่าอัตราการเรียนรู้แบบคงที่เพื่อหาค่าอัตราการเรียนรู้ที่ส่งผลให้ความผิดพลาดลดลงน้อยที่สุด

ตารางที่ 3.4 โครงสร้างในการทดลอง

Parameters	Values
Training Data	60,000 images
Testing Data	10,000 images
Output label	10 labels
Convolution layer 1 Output maps	12 maps
Convolution layer 1 kernel size	5 x 5
Convolution layer 2	18 maps
Convolution layer 2 kernel size	5 x 5
Convolution layer 3	36 maps
Convolution layer 3 kernel size	2 x 2
batch size	500 images

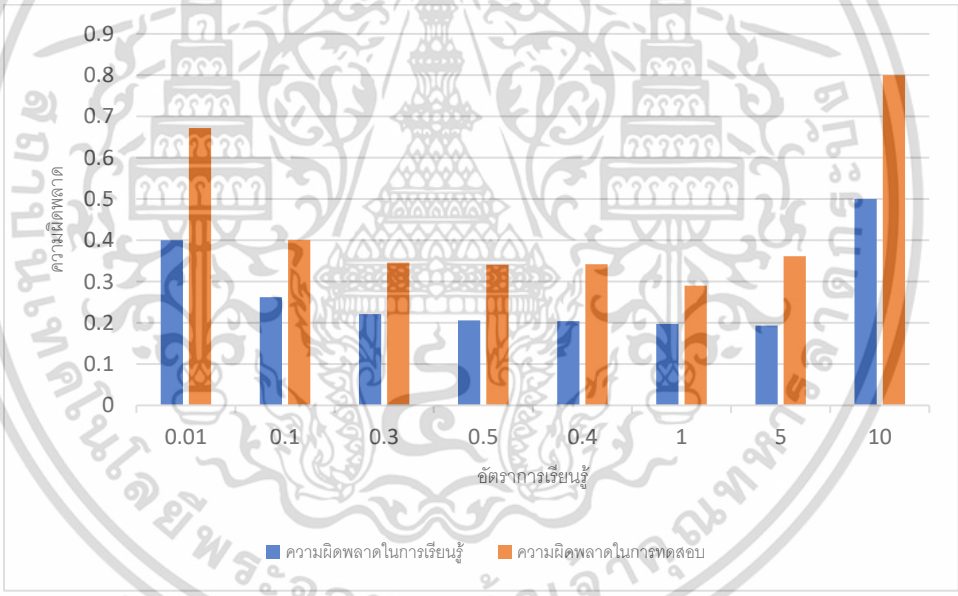
3.2.2 สรุปความผิดพลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ระหว่าง 0.01 - 10

การทดลองจะทำการปรับค่าอัตราการเรียนรู้เพื่อทำการเปรียบเทียบ ความผิดพลาดในการเรียนรู้ เวลาที่ใช้ในการทำงานและความผิดพลาดในการทดสอบ ซึ่งผลกรทดลองสรุปได้ในตารางที่ 3.4 และนำไปข้อมูลจากตารางไปวาดกราฟแผนภูมิได้ในรูปที่ 3.22 เพื่อให้เห็นความแต่งต่างของการทดลองได้ชัดเจน จะเห็นได้ว่าอัตราการเรียนรู้ที่ 0.01 และ 10 จะไม่สามารถเป็นค่าที่อัตราการเรียนรู้ไม่สามารถลดลงได้ และ อัตราการเรียนรู้ที่ 0.3, 0.5, 0.7, 1 และ 5 ให้ผลความผิดพลาด ไม่ต่างกันมากนัก แต่ค่าอัตราการเรียนรู้ที่ 5 ให้จะให้ค่าความผิดพลาดได้น้อยที่สุด แต่เมื่อดูความผิดพลาดในการทดสอบจะเห็นว่า อัตราการเรียนรู้ที่ 1 จะมีผลการทดลองความผิดพลาดในการทดสอบน้อยที่สุด

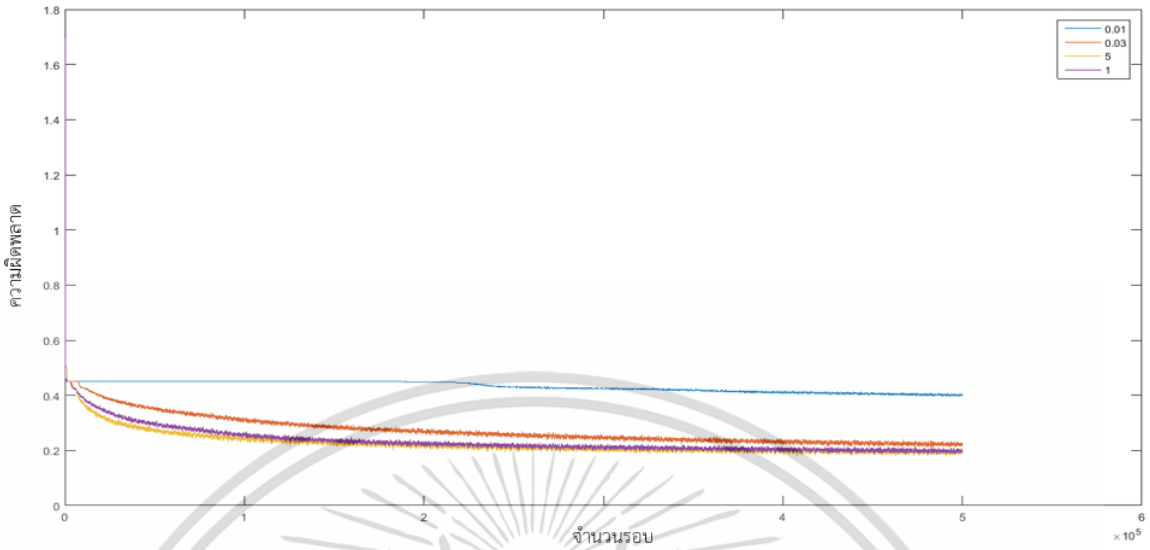
จากตางรางการทดลอง จะเห็นว่าความผิดพลาดมีค่าใกล้เคียงกันอาจจะเป็นผลมาจากการรันโปรแกรมที่จำนวนรอบมากเกินไป ซึ่งอาจทำให้ค่าอัตราการเรียนรู้ ที่อยู่ในช่วง 0.3 - 5 สามารถลดลงเข้าสู่คำตอบได้ สังเกตจากรูปที่ 3.23 จะเห็นว่าหลังช่วง $1x10^5$ จะเห็นว่าความผิดพลาดจะไม่ค่อยลดค่าลง

ตารางที่ 3.5 เปรียบเทียบความผิดพลาดเมื่อเปลี่ยนค่าอัตราการเรียนรู้ โดยใช้ข้อมูลชุดCIFAR-10

อัตราการเรียนรู้	ความผิดพลาดในการเรียนรู้	เวลาที่ใช้ (นาที)	ความผิดพลาดในการทดสอบ
0.01	0.400	55.2	0.6718
0.1	0.262	55	0.4009
0.3	0.221	55.3	0.346
0.5	0.206	55.43	0.341
0.7	0.20393	55.56	0.342
1	0.1973	55.7	0.29
5	0.193	55.73	0.3613
10	0.5	55.6	0.8



รูปที่ 3.21 กราฟเปรียบเทียบค่าความผิดพลาด ในการเรียนรู้และทดสอบ



รูปที่ 3.22 กราฟเปรียบเทียบค่าความผิดพลาด ในการเรียนรู้และทดสอบ

3.3 สรุปปัญหาที่เกิดจากการปรับค่าอัตราการเรียนรู้

จากการทดลองกับข้อมูลชุด MNIST และ CIFAR-10และมีการปรับค่าอัตราการเรียนรู้ในค่าต่าง ๆ ซึ่งจะเห็นอัตราการเรียนรู้ที่มีค่าน้อยไป หรือค่ามากไป ก็จะส่งผลให้ความผิดพลาดไม่สามารถลดลง หรือผลการเรียนรู้ไม่คอนเวอร์เจนซ์นั้นเอง และหากค่าอัตราการเรียนรู้ที่ส่งผลให้ความผิดพลาดลดลงน้อยที่สุดของแต่ละโครงสร้างและชุดข้อมูลไม่สามารถนำมาใช้ร่วมกันได้ ด้วยเหตุนี้ที่ว่าอัตราการเรียนรู้ที่ส่งผลให้ความผิดพลาดน้อยที่สุดขึ้นอยู่กับการทดลองและโครงสร้างนั้น ๆ จะเป็นการดีและมีประโยชน์กับนักวิจัย ถ้าสามารถกำหนดให้อัตราการเรียนรู้สามารถปรับตัวเองได้ตามโครงสร้างและชุดข้อมูล ซึ่งทางนักวิจัยได้นำเสนอวิธีการปรับค่าอัตราการเรียนรู้เองโดยอาศัยหลักการณ์ของทฤษฎีอนุกรมเทย์เลอร์มาช่วยซึ่งจะอธิบายต่อไปในบทที่ 4 และผลการทดลองในบทที่ 5

บทที่ 4
อัตราการเรียนรู้ที่สามารถปรับตัวเองได้

4.1 อัตราการเรียนรู้ที่ปรับตัวเองได้ โดยอาศัยทฤษฎีอนุกรมเทย์เลอร์

จากผลการทดลองในบทที่ 3 จะเห็นได้ว่าเป็นเรื่องยากมากที่จะกำหนดค่าของอัตราการเรียนรู้ให้เหมาะสม โดยปกติอัตราการเรียนรู้จะเลือกจากการทดลอง ซึ่ง η มีค่ามาก ความเร็วในการเรียนรู้จะเร็ว แต่ความผิดพลาดกำลังสองเฉลี่ยจะเกิดการแกว่ง โดยตรงกันข้าม ถ้า η มีค่าน้อย ความเร็วการเรียนรู้หรือการลู่เข้าก็จะช้าตาม ด้วยเหตุนี้ถ้าสามารถกำหนด η ให้สามารถปรับตัวเองได้ก็จะเป็นผลดีต่อนักวิจัยที่ไม่ต้องกังวลในการกำหนดค่า η.

ฟังก์ชันความผิดผลาดสามารถเขียนอยู่ในรูปฟังก์ชันอนุกรมเทย์เลอร์ดังสมการที่ 4.1

$$E = f(w_{nl}, b_{ll}, K_{nl}, b_l) \tag{4.1}$$

เมื่อ w_{nl} คือค่าน้ำหนักในชั้น fully connected

b_{ll} คือค่าไบอัสในชั้น full connected

K_{nl} คือค่าfilter ที่ทำกับรูปภาพในชั้น Convolution

b_l คือค่า ไบอัสในชั้น convolution

จากสมการที่ 4.1 อธิบายได้ว่า กระบวนการของเครือข่ายประสาทเทียมแบบแพร่กลับในการปรับน้ำหนัก เทมเพลต(template) และไบอัส จะวนปรับค่าเพื่อที่จะทำให้ค่าความผิดพลาด E_m ลดลง จนกระทั้งความผิดพลาดตรงตามต้องการซึ่งความผิดพลาดกำลังสองทั้งหมดนั้นสามารถประมาณค่าจากทฤษฎีอนุกรมเทย์เลอร์และจัดสมการอยู่ในรูปของอนุพันธ์ย่อย จากสมการที่ 4.1 เป็นดังสมการ 4.2

$$E(i+1) = E(i) + \frac{\partial E(o)}{\partial w_{nl}}\Delta w_{nl} + \frac{\partial E(i)}{\partial b_l}\Delta b_l + \frac{\partial E(i)}{\partial K_{nl}}\Delta K_{nl} + \frac{\partial E(i)}{\partial b_l}\Delta b_l$$

$$E(i+1) - E(i) = \frac{\partial E(o)}{\partial w_{nl}}\Delta w_{nl} + \frac{\partial E(i)}{\partial b_l}\Delta b_l + \frac{\partial E(i)}{\partial K_{nl}}\Delta K_{nl} + \frac{\partial E(i)}{\partial b_l}\Delta b_l$$

$$\Delta E = E(i+1) - E(i) \tag{4.2}$$

จากสมการ 2.11, 2.12, 2.14, 2.15, 2.17 และ 2.18 นำมาแทนในสมการ 4.2 และจัดรูปแบบสมการใหม่

$$\Delta E = -\eta \left[(\frac{\partial E(i)}{\partial w_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 + (\frac{\partial E(i)}{\partial k_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 \right] \leq 0 \qquad (4.3)$$

ย้ายสมการเพื่อต้องการหาค่า η เรียนรู้

$$\eta = \frac{-\Delta E}{\left[(\frac{\partial E(i)}{\partial w_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 + (\frac{\partial E(i)}{\partial k_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 \right]} \qquad (4.4)$$

$$\eta = \frac{-\Delta E}{gradient} \qquad (4.5)$$

เมื่อ

$$gradient = gradient1 + gradient2 \qquad (4.6)$$

$$gradient1 = \sum_{i=1}^{M} \sum_{j=1}^{N} \left(\frac{\partial E}{\partial w_{ij}} \right)^2 + \sum_{i=1}^{M} \left(\frac{\partial E}{\partial b_i} \right)^2 \qquad (4.7)$$

$$gradient2 = \sum_{l=1}^{L} \sum_{i=1}^{M} \left(\frac{\partial E}{\partial K_{nl}} \right)^2 + \sum_{l=1}^{L} \left(\frac{\partial E}{\partial b_1} \right)^2 \qquad (4.8)$$

แทนสมการ (2.9) ลงในในสมการ (4.2) จะได้

$$-\Delta E = \frac{1}{2} \sum_{p=1}^{P} \sum_{l=1}^{L} \left\{ \left[t_{pl} - y_{pl}(k) \right]^2 - \left[t_{pl} - y_{pl}(k+1) \right]^2 \right\} \qquad (4.9)$$

จัดรูปสมการ 4.9 ใหม่ ตามสมการเดิม

$$-\Delta E = A + B \qquad (4.10)$$

เมื่อ

$$A = \frac{\frac{1}{2} \sum_{p=1}^{P} \sum_{l=1}^{L} \left[t_{pl} - y_{pl} \right]^2}{p} \qquad (4.11)$$

$$B = \frac{1}{2} \sum_{p=1}^{P} \sum_{l=1}^{L} \left\{ \left[t_{pl} - y_{pl}(k) \right]^2 \frac{(p-1)}{p} - \left[t_{pl} - y_{pl}(k+1) \right]^2 \right\} \qquad (4.12)$$

เมื่อผลรวมของตัวอย่างการฝึกอบรมมีจำนวนมากเราประมาณได้ว่า

$$\frac{(p-1)}{p} \approx 1 \tag{4.13}$$

สำหรับกระบวนการอบรมที่เวลารอบสูงขึ้นจะได้สมการตามนี้

$$\left[t_{pl} - y_{pl}\right]^2 \approx \left[t_{pl} - y_{pl}(k+1)\right]^2 \tag{4.14}$$

แทนสมการที่ 4.131 4.14 ลงในสมการที่ 4.12 ซึ่งทำให้เทอมของ B = 0 ซึ่งจะตรงกับสมการที่ 2.10 ดังนั้นจะได้ว่า

$$-\Delta E = \frac{1}{2P}\sum_{p=1}^{P}\sum_{l=1}^{L}\left[t_{pl} - y_{pl}(k)\right]^2 = E_m \tag{4.15}$$

จากสมการที่ 4.15 เขียนใหม่ได้ว่า

$$\eta = \frac{E_m}{gradient} \tag{4.16}$$

อัตราการเรียนรู้จากสมการที่ จะเห็นได้ว่าขึ้นอยู่อัตราส่วน ค่าเฉลี่ยความผิดพลาด E_m ต่อความผิดพลาดเกรเดียนต์กำลังสอง ซึ่งในสมการนี้จะช่วยให้ระอัลกอริทึมสามารถเลือกอัตราการเรียนรู้แต่ละรอบได้อยากเหมาะสม

4.2 Procedure ขั้นตอนการทำงาน

กระบวนการหาค่าอัตราการเรียนรู้เองอัตโนมัติ สามารถทำตามขั้นตอนได้ทีละขั้นตอนดังนี้

ขั้นตอนที่ 1: กำหนดค่าเริ่มต้น

ขั้นตอนที่ 2: Forward computation

- 2.1 Convolution
- 2.2 pooling and
- 2.3 fully-connection

ขั้นตอนที่ 3: ความผิดพลาดและการอัพเดทค่าน้ำหนักและเทมเพลต

- 3.1 การคำนวณหาค่าความผิดพลาด $(\mathrm{E_{rms}})$
- 3.2 คำนวณหาความความไวของความผิดพลาด δ
- 3.3 คำนวณหาอัตราการเรียนรู้ $\eta = \frac{\mathrm{E_{rms}}}{\delta}$
- 3.4 อัพเดทค่าเทมเพลตในชั้น Convolution และค่าน้ำหนักในชั้น fully connected

ขั้นตอนที่ 4: ใช้ข้อมูลรอบถัดไปในการคำนวณและกลับไปเริ่มในขั้นตอนที่ 2 และทำซ้ำเรื่อยๆ จนกว่าความผิดพลาดจะลดลงตามเป้าหมายที่ต้องการ

บทที่ 5
ผลการทดลอง

ในการทดลองในบทที่5 จะใช้พารามิเตอร์เช่นเดียวกับการทดลองในบทที่ 3 แต่จะเพิ่มการทดสอบเพิ่มเติมจากบทที่3 ก็คือจะเพิ่มการทดลองในเรื่องของอัตราการเรียนรู้ที่สามารถปรับตัวเองได้ซึ่งได้นำเสนอไว้ในบที่ที่4 และอัลกอริทึมที่ปรับตัวเองได้ที่นำเสนอโดย Zhiwen Shao [11]เรื่อง“FACE ALIGNMENT BY DEEP CONVOLUTIONAL NETWORK WITH ADAPTIVE LEARNING RATE” เป็นการปรับอัตราการเรียนรู้ได้เองโดยการทำงานจะนำค่าอัตราการเรียนรู้คูณด้วย 0.1 ในการอับเดทค่าน้ำหนักแต่ละรอบ

ในการทดลองโดยกำหนดให้ อัลกริทึมที่นำเสนอโดย Zhiwen Shao เป็น AUTO1 และอัลกริทึมเรียนรู้ที่ผู้วิจัยได้นำเสนอ ตาม Procedure ในข้อหัวข้อที่ 4.2 เป็น AUTO2 เพื่อใช้ในการเปรียบเทียบในการทดลองในหัวข้อต่อไป

5.1 ทดลองโดยใช้ข้อมูล MNIST database

ในหัวข้อนี้ จะทำการทดสอบอัลกอริทึมที่นำเสนอในบทที่4 โดยใช้ข้อมูลตัวอย่างที่นำมาใช้ในการทดลองคือ MNIST database เป็นข้อมูลรูปภาพที่ให้เขียนเลข 0 – 9 จำนวน 70,000 ข้อมูล โดยแบ่ง 60,000 ใช้สำหรับเรียนรู้ และ 10,000 ข้อมูลใช้สำหรับทดสอบ

5.1.1 ค่าพารามิเตอร์ที่ใช้ในการทดลอง

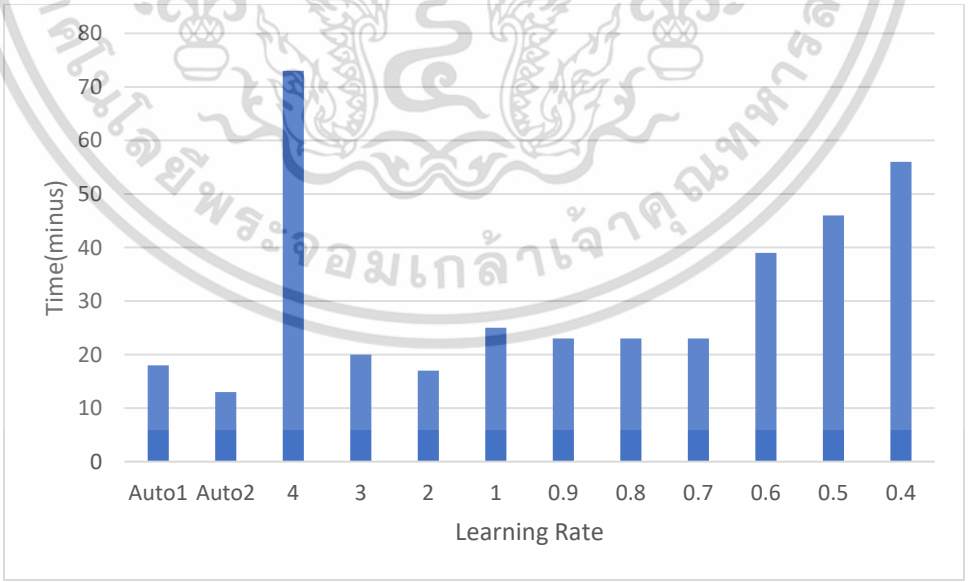
ตารางที่ 5.1 ค่าพารามิเตอร์ที่ใช้ในการทดลอง

Parameters	Values
Training data	60,000 images
Testing data	10,000 images
Output label	10 labels
Convolution layer 1 Output maps	6 maps
Convolution layer 1 kernel size	5 x 5
Convolution layer 2	12 maps
Convolution layer 2 kernel size	5 x 5
batch size	500 images
Target Error	0.09

5.1.2 ผลการทดลอง

ตารางที่ 5.2 ผลการทดลอง

อัตราการเรียนรู้	ความผิดพลาดที่ตั้งไว้	ในการเรียนรู้ 60,000 ข้อมูล			ในการทดสอบ 10,000 ข้อมูล		เวลาที่ใช้ (นาที)
		ความผิดพลาดเฉลี่ย	ความผิดพลาดจาก 60,000	ความผิดพลาด	ความผิดผลาดจากการทดสอบ	ความผิดพลาดจากการทดสอบ	
Auto1	0.09	0.0882	5317	0.0885	809	0.0809	18
Auto2	0.09	0.087	5,399	0.09	824	0.0824	13
4	0.09	0.08917	8,011	0.1335	1,279	0.1279	73
3	0.09	0.0882	5,317	0.0886	823	0.0823	20
2	0.09	0.087	5,312	0.0885	809	0.0809	17
1	0.09	0.0894	5,547	0.0925	847	0.0847	25
0.9	0.09	0.0894	5,547	0.0925	847	0.0847	23
0.8	0.09	0.089497	5,547	0.0925	847	0.0847	23
0.7	0.09	0.089497	5,547	0.0925	847	0.0847	23
0.6	0.09	0.089	5,655	0.0943	862	0.0862	39
0.5	0.09	0.0891	5,610	0.0935	853	0.0853	46
0.4	0.09	0.0895	5,639	0.094	853	0.0853	56

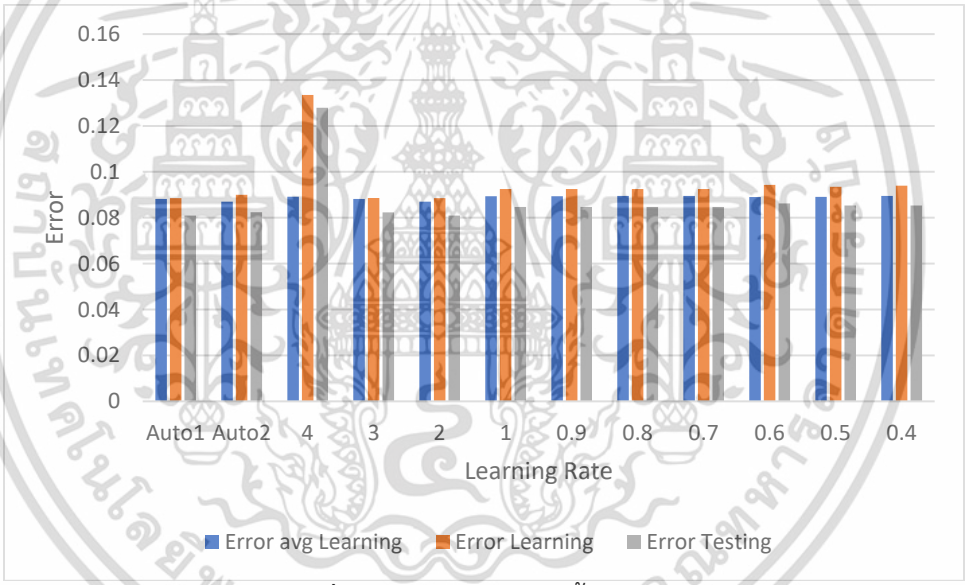


รูปที่ 5.1 เวลาที่ใช้ในการเรียนรู้แต่ละอัตราการเรียนรู้

จากรูปที่ 5.1 เมื่อทำการทดลองโดยกำหนดความความผิดพลาดไว้ที่ 0.09 และเปลี่ยนอัตราการเรียนรู้ในค่าต่างๆ และเทียบกับอัตราการเรียนรู้ที่ปรับตัวเองได้ จากรูปเมื่อเปลียนอัตราการเรียนรู้ระหว่าง 0.4 – 4 จะเห็นว่า อัตราการเรียนรู้เท่ากับ 2 จะใช้เวลาน้อยที่สุด และเมื่อเพิ่มการเรียนรู้มากขึ้นนั้นไม่ได้หมายความว่าอัตราการเรียนรู้จะใช้เวลาน้อยลง จะเห็นได้ว่าเมื่อเพิ่มอัตรการเรียนรู้เป็น 3 จะใช้เวลามากขึ้น และเพิ่มอัตราการเรียนรู้เป็น 4 จะใช้เวลามากถึง 73 นาที และข้อสังเกตจะเห็นว่าอัตราการเรียนรู้ที่ 1- 0.7 จะใช้เวลาใกล้เคียงกัน เมื่อลดอัตราการเรียนรู้ต่ำกว่า 0.7 จะใช้เวลามากขึ้นในการหลู่เข้าสู่คำตอบ

และได้ทำการทดลองกับวิธี Auto1 ก็จะเห็นว่าวิธี Auto1 ใช้เวลาในการเรียนรู้ 18 นาที เมื่อเทียบกับวิธีที่กำหนดค่าอัตราการเรียนรู้แบบคงที่ก็แสดงเห็นว่ามีผลดีกว่า

และเมื่อทำการทดลองให้อัตรการเรียนรู้ปรับตัวเองได้ Auto2 จะเห็นว่าใช้เวลาน้อยที่สุดเมื่อเทียบกับการกำหนดอัตราการเรียนรู้แบบคงที่ซึ่งใช้เวลาเพียง 13 นาที



รูปที่ 5.2 ความผิดพลาดทั้ง 3 แบบ

ความผิดพลาดจะมีการวัดความผิดพลาดอยู่ด้วยกัน 3 แบบ คือ ความผิดพลาดเฉลี่ยจากการเรียนรู้ , ความผิดพลาดจากการเรียนรู้ และ ความผิดพลาดจากการทดสอบ ซึ่งจะอธิบายต่อไปนี้

1.ความผิดพลาดเฉลี่ยจากเรียนรู้ คือ ความผิดพลาดที่เกิดขึ้นใน 1 รอบการเรียนรู้ ซึ่งแต่ละ 1 รอบการเรียนรู้จะมีการแบ่งข้อมูลเป็นชุด batch size ซึ่งทำให้ 1 รอบมีเรียนหลายครั้ง ยกตัวอย่างเช่นถ้ามีข้อมูลในการทดสอบ 60,000 ข้อมูล และจัดเป็นชุดหละ 500 ข้อมูล ก็หมายความว่า 1 รอบ จะมีการเรียนรู้ 120 ครั้ง ซึ่งแต่ละครั้งจะมีการอับเดทค่าน้ำหนักทุกครั้ง และความผิดพลาดก็จะมีค่าที่แตกต่างกัน ดังนั้นจึงค่าหาเฉลี่ยค่าความผิดพลาดเพื่อเป็นค่ากลางในการเรียนรู้ในหนึ่งรอบ

2.ความผิดพลาดทั้งหมดจากการเรียนรู้ คือ ความผิดพลาดทั้งหมดนี้ จะหาหลังจากความผิดผลาดเฉลี่ยน้อยกว่าหรือเท่ากับค่าความผิดพลาดที่ตั้งไว้ ซึ่งจะส่งข้อมูลทั้งทั้งหมดไปทดสอบในค่าน้ำหนัก

ตัวเดียวกัน จะไม่มีการแบ่งกลุ่มเป็น batch size การวัดผิพลาดจะเป็นการนับจำนวนที่ไม่ถูกต้องแล้วคำนวณกลับเป็นความผิดผลาด ซึ่งต่างจากข้อ 1 ที่นำเอาท์พุตจากกาเรียนรู้มาลบกับค่าจริงของข้อมูลชุดนั้น ยกตัวอย่างเช่น มีข้อมูลในการทดสอบ 60,000 ข้อมูล เมื่อนำไปเรียนรู้ มีข้อมูลที่ไม่ถูกต้องอยู่ 6,000 ข้อมูล เพราะฉะนั้นความผิดพลาดมีค่าเท่ากับ 6,000/60,000 = 0.1

3.ความผิดพลาดจากการทดสอบ คือ ความผิดผลาดที่นำข้อมูล ที่ไม่เคยนำไปเรียนรู้เลย ซึ่งได้แบ่งไว้เพื่อใช้ทดสอบ ซึ่งในการทดสอบจะใช้หลักการเดียวกับ ข้อ 2 ยกตัวอย่าง ว่ามีข้อมูลที่ใช้ในการทดสอบ 10,000 ข้อมูล เมื่อนำไปทดสอบ มีข้อมูลที่ไม่ถูกต้องจำนวน 1,000 ข้อมูล เพราะฉะนั้นความผิดพลาดในการทดสอบมีค่าเท่ากับ 1,000/10,000 = 0.1

ในการทดลอง จะมีการกำหนดความผิดพลาดไว้ที่ 0.09 เพื่อเป็นวัดว่าอัตราการเรียนรู้ที่คงที่ และอัตราการเรียนรู้ที่ปรับตัวเองได้โดยใช้หลักการณ์ในบทที่ 4 ว่าอัตราการเรียนรู้แบบไหนใช้เวลาในการเรียนรู้น้อยที่สุด

จากการทดลองจะเห็นได้ว่า อัตราการเรียนรู้แบบคงที่ ที่ใช้เวลาน้อยที่สุด คืออัตรการเรียนรู้เท่ากับ 2 และสังเกตได้ว่าถ้าเราเพิ่มเป็น 3 จะใช้เวลามากขึ้นมากๆ และเมื่อเทียบกับอัตราการเรียนรู้ที่ปรับตัวเองได้ กลับใช้เวลาน้อยกว่าที่เป็นแบบคงที่ค่าไว้ เป็นเพราะว่าอัตรการเรียนรู้ที่ปรับตัวเองได้จะมีการปรับตามความไวของความผิดพลาด ถ้าค่าความไวของความผิดพลาดมีค่ามากอัตราการเรียนรู้จะมีค่าน้อย และถ้าค่าความไว้มีค่าน้อยอัตรการเรียนรู้ก็จะมีค่ามาก ซึ่งค่าความไวของความผิดพลาดและค่าอัตราการเรียนรู้จะมีความสัมพันธ์แบบผกผันแบบอินเวอร์ส

จากการทดลองเมื่อเปรียบเทียบความผิดพลาดแบบเฉลี่ยและแบบทั้งหมด จะสังเกตได้ว่าความผิดพลาดจากแบบเฉลี่ยจะมีค่าน้อยกว่า เพราะว่าแบบเฉลี่ยจะเป็นผลต่างระหว่างเอาท์พุตจากการเรียนรู้เทียบกับค่าจริง

จากการทดลองบทที่ 3 จะเห็นได้ว่าถ้าเลือกอัตราการเรียนรู้น้อยหรือมากไป ก็อาจทำให้ไม่สามารถลดค่าความผิดพลาดจากเรียนรู้ได้ แต่จากการทดลองบทที่ 5 พิสูจน์แล้วว่า อัตราการเรียนรู้ที่ปรับตัวเองได้สามารถลงค่าความผิดพลาดได้และเมื่อเปรียบเทียบกับการกำหนดค่าอัตราการเรียนรู้คงที่ ที่ลดค่าความผิดพลาดน้อยที่สุดแล้วผลความผิดพลาดของอัตราการเรียนรู้ที่ปรับตัวเองได้ก็มีผลดีกว่ามากในช่วงแรกและในช่วงคงที่ก็มีค่าไม่ต่างกันมาก

5.2 ทดลองโดยใช้ข้อมูล CIFAR-10

ในหัวข้อนี้ จะทำการทดสอบอัลกอริทึมที่นำเสนอในบทที่4 โดยใช้ข้อมูลตัวอย่างที่นำมาใช้ในการทดลอง คือ CIFAR-10 database ซึ่งรูปภาพประกอบไปด้วย รูปภาพสี 60,000 รูป มีขนาด 32x32 ซึ่งมีรูปภาพจำนวน 60,000รูปภาพ สามารถแบ่งเป็น 50,000 รูปภาพ สำหรับการเรียนรู้ และ 10,000 รูปภาพสำหรับทดสอบ โดยจัดกลุ่มได้ 10 กลุ่มได้แก่ ได้แก่ เครื่องบิน,รถยนต์,นก,แมว,กว้าง,หมา,กบ,ม้า,เรือ,รถบรรทุก

5.2.1 ค่าพารามิเตอร์ที่ใช้ในการทดลอง

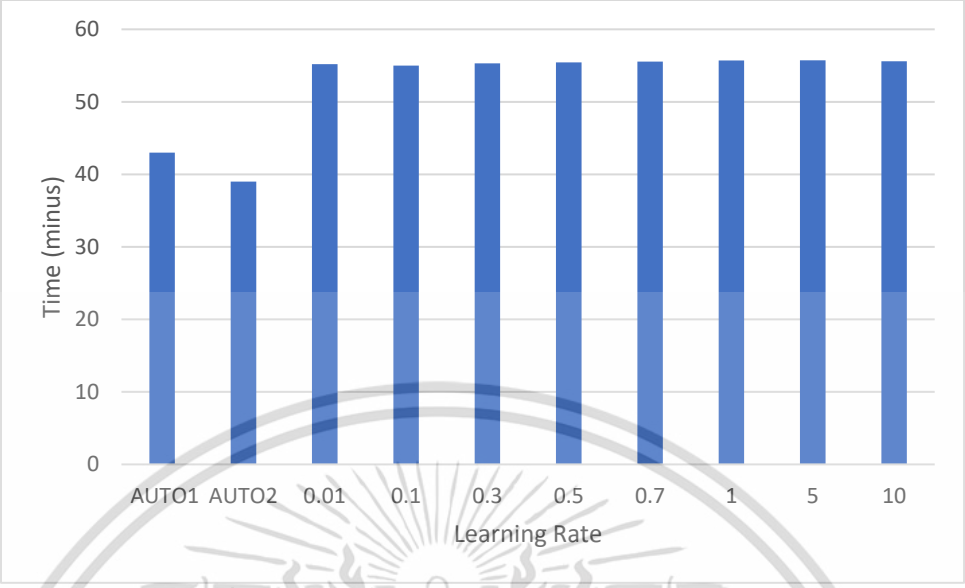
ตารางที่ 5.3 ค่าพารามิเตอร์ที่ใช้ในการทดลอง

Parameters	Values
Training data	60,000 images
Testing data	10,000 images
Output label	10 labels
Convolution layer 1 Output maps	12 maps
Convolution layer 1 kernel size	5 x 5
Convolution layer 2	18 maps
Convolution layer 2 kernel size	5 x 5
Convolution layer 3	36 maps
Convolution layer 3 kernel size	2 x 2
batch size	500 images

5.2.2 ผลการทดลอง

ตารางที่ 5.4 ผลการทดลอง

อัตราการเรียนรู้	ความผิดพลาดที่ตั้งไว้	ในการเรียนรู้ 50,000 ข้อมูล			ในการทดสอบ 10,000 ข้อมูล		เวลาที่ใช้ (นาที)
		ความผิดพลาดเฉลี่ย	ความผิดพลาดจาก 50,000	ความผิดพลาด	ความผิดผลาดจากการทดสอบ	ความผิดพลาดจากการทดสอบ	
AUTO1	0.2	0.195	8,450	0.197	0.22	895.00	43
AUTO2	0.2	0.197	8,435	0.199	0.213	890.00	39
0.01	0.2	0.4	3,850	0.4	0.9	8,405.00	55.2
0.1	0.2	0.194	8,512	0.198	0.22	872.00	55
0.3	0.2	0.198	8,325	0.199	0.21	882.00	55.3
0.5	0.2	0.196	8,442	0.197	0.233	897.00	55.43
0.7	0.2	0.194	8,501	0.193	0.202	881.00	55.56
1	0.2	0.198	8,494	0.198	0.223	894.00	55.7
5	0.2	0.193	8,502	0.195	0.21	887.00	55.73
10	0.2	0.5	3,981	0.5	0.89	8,259.00	55.6

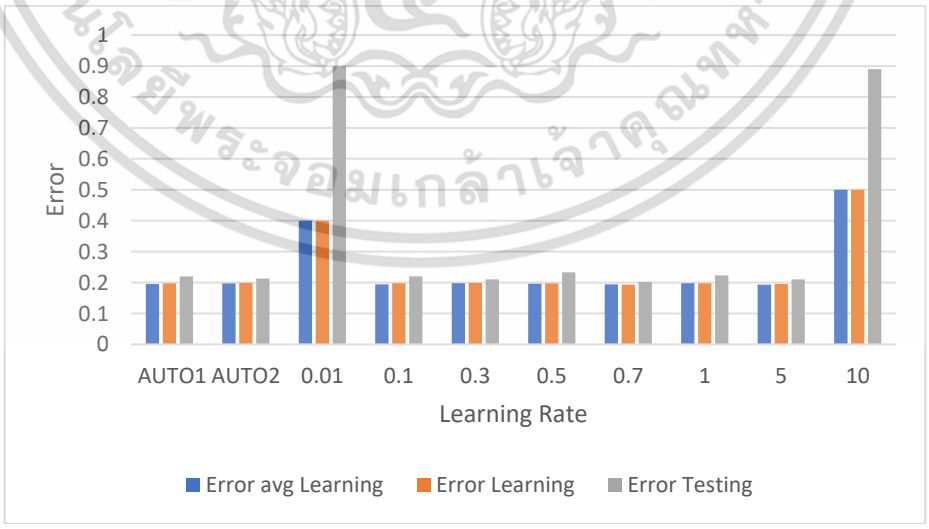


รูปที่ 5.3 เวลาที่ใช้ในการเรียนรู้แต่ละอัตราการเรียนรู้

จากรูปที่ 5.3 เมื่อทำการทดลองโดยกำหนดความความผิดพลาดไว้ที่ 0.2 และเปลี่ยนอัตราการเรียนรู้ในค่าต่างๆ และเทียบกับอัตราการเรียนรู้ที่ปรับตัวเองได้ จากรูปเมื่อเปลี่ยนอัตราการเรียนรู้ระหว่าง 0.01 – 10 จะเห็นว่าจะใช้เวลาใกล้เคียงกัน แต่จะเห็นว่า อัตราการเรียนรู้ที่ 0.01 และ 10 ไม่สามารถลู่เข้าสู่คำตอบได้ เนื่องจากอัตราการเรียนรู้น้อยเกินไป และมากเกินไปตามลำดับ

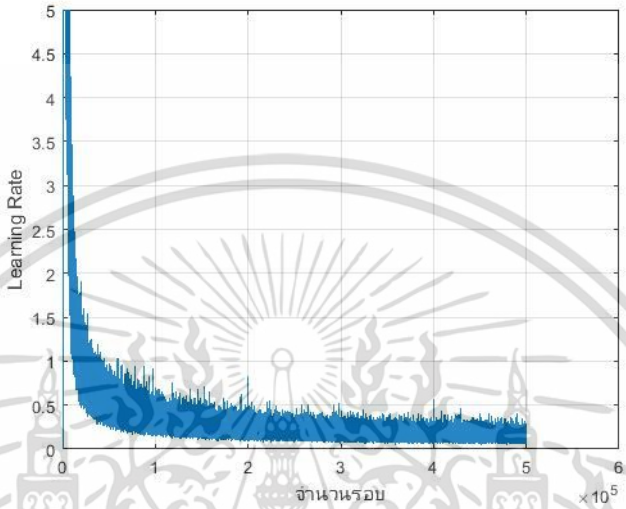
การทดลองด้วยวิธี Auto1 ก็จะเห็นว่าวิธีนี้ใช้เวลาในการเรียนรู้ 43 นาที เมื่อเทียบกับวิธีที่กำหนดค่าอัตราการเรียนรู้แบบคงที่ก็แสดงเห็นว่ามีผลความผิดพลาดดีกว่า

และเมื่อทำการทดลองให้อัตรการเรียนรู้ปรับตัวเองได้ Auto2 ซึ่งเป็นวิธีที่ผู้วิจัยได้นำเสนอ จะเห็นว่าใช้เวลาน้อยที่สุดเมื่อเทียบกับการกำหนดอัตราการเรียนรู้แบบคงที่ซึ่งใช้เวลาเพียง 39 นาที



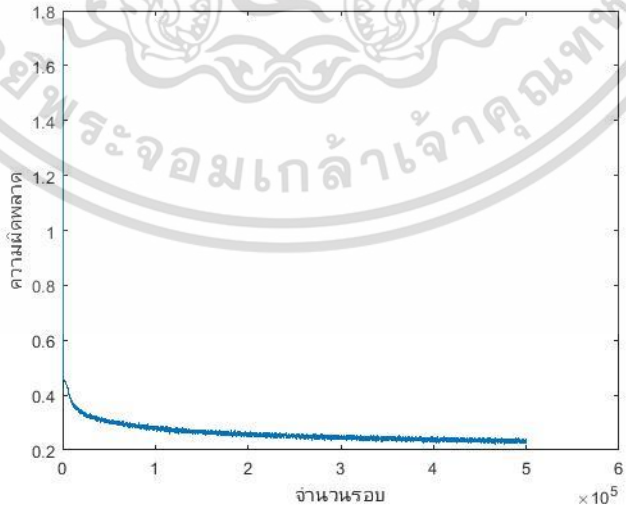
รูปที่ 5.4 ความผิดพลาดทั้ง 3 แบบ

จากการทดลองเมื่อเปรียบเทียบความผิดพลาดแบบเฉลี่ยและแบบทั้งหมด จะสังเกตได้ว่าความผิดพลาดจากแบบเฉลี่ยจะมีค่าน้อยกว่าเพราะว่าแบบเฉลี่ยจะเป็นผลต่างระหว่างเอาท์พุตจากการเรียนรู้เทียบกับค่าจริง



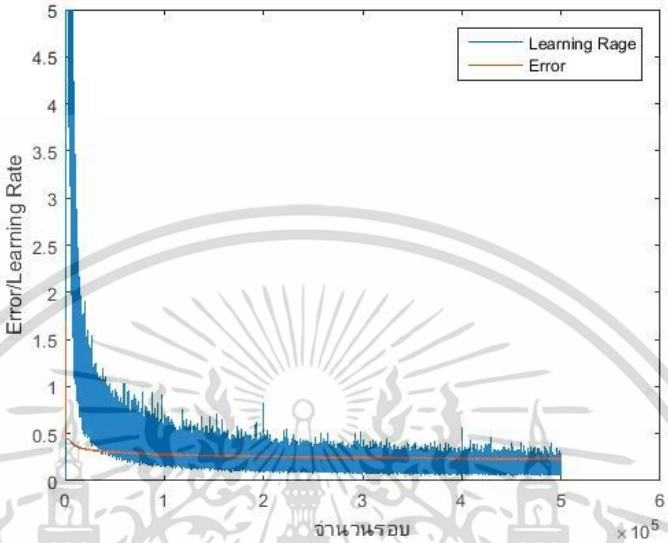
รูปที่ 5.5 อัตราการเรียนรู้ที่ปรับตัวเองได้

จากกราฟการทดลองจะเป็นการแสดงค่าอัตราการเรียนรู้ที่มีการปรับแต่ละครั้งตามรอบการเรียนรู้ของ batch ซึ่งสังเกตได้ข้อสรุปว่าใช้ช่วงการเรียนรู้ต้นๆจะอัตราการเรียนรู้จะมีการแกว่งมาจากความผิดพลาดยังไม่สามารถลดลงได้เหมือนเทียบกับความไวของความผิดพลาด แต่เมื่อจำนวน batch มาอยู่ในช่วงที่มีจำนวนรอบมากขึ้น ความไวของความผิดพลาดน้อยลง อัตราการเปลี่ยนแปลงของค่าน้ำหนักน้อยลงด้วยทำให้อัตราการเรียนรู้เริ่มลดลงอย่างช้า ๆ และไม่มีการแกว่งของค่าอัตราการเรียนรู้



รูปที่ 5.6 ความผิดพลาด ในกรณีที่อัตราการเรียนรู้ปรับตัวเองได้

จากรูปการทดลองโดยแนวตั้งคือความผิดพลาดจากการเรียนรู้และแนวนอนคือจำนวนรอบซึ่งจากการทดลองจะเห็นได้ว่าความผิดพลาดมีการลดลงเป็นแบบexponential ซึ่งแสดงให้เห็นว่าอัตราการเรียนรู้ที่ปรับตัวเองได้สามารถลดความผิดพลาดลงได้



รูปที่ 5.7 ความผิดพลาด เทียบกับ อัตราการเรียนรู้ปรับตัวเองได้

นำผลการทดลองวาดกราฟของอัตราการเรียนรู้และความผิดพลาดนำมาแสดงเข้ามาอยู่ในกราฟเดียวกันโดยแนวตั้งจะประกอบไปด้วย ความผิดพลาดและอัตราการเรียนรู้ โดยแนวนอนจะเป็นจำนวนรอบ เมื่อทำการพิจารณาในจำนวนรอบที่ 0 – 200 จะเห็นว่าความผิดพลาดมีการลดลงอย่างรวดเร็วและเมื่อเทียบกับอัตราการเรียนรู้ก็จะเห็นว่ามีการแกว่างขึ้นลงซึ่งแสดงให้เห็นว่าอัตราการเรียนรู้ที่ปรับตัวเองได้นั้นจะมีการปรับทุกรอบเพื่อให้ความผิดพลาดมีการลดลงอย่างรวมเร็ว แต่เมื่อความผิดพลาดเริ่มเข้าสู่สถาวะอิ่มตัวหรือค่าความผิดพลาดเริ่มคงที่ก็จะสังเกตได้ว่าอัตราการเรียนรู้ก็จะมีการปรับค่าเป็นค่าเล็ก ๆ ตามแนวโน้นของความผิดพลาดในการเรียนรู้เช่นกัน

เมื่อการเรียนรู้เข้าสู่สภาวะคงที่ นั้นก็ก็คือ ความไวของความผิดพลาด (gradient)และความผิดพลาด มีการเปลี่ยนแปลงน้อย ทำให้อัตราการเรียนรู้ที่ปรับตัวเองได้จะเป็นค่าคงที่ค่าใดค่าหนึ่ง นั้นสามารถสรุปได้ว่าการเรียนรู้ได้เข้าสู่คำตอบแล้ว

5.3 สรุปความผิดพลาดในการทดสอบ

เมื่อทำการเรียนรู้ครบตามเงื่อนไขที่กำหนด อีกขั้นตอนที่สำคัญคือการทดสอบในชุดข้อมูลที่ไม่ได้นำไปใช้การเรียนรู้ เพื่อเป็นตรวจสอบว่าสามารถแก้ปัญหากับชุดข้อมูลที่ไม่เคยเรียนได้ได้หรือไหม ซึ่งการทดสอบของข้อมูลชุด MNIST และ CIFAR-10 ที่แบ่งไว้ใช้ในการทดสอบ สามารถสรุปผลของการทดสอบได้ในตาราง

ตารางที่ 5.5 สรุปผลความผิดพลาดในการทดสอบ

ชุดข้อมูล	จำนวนข้อมูลที่ทดสอบ	ความผิดพลาดในการทดสอบ
MNIST	10,000	0.091
CIFAR-10	10,000	0.221

จากตารางที่ 5.3 เหมือนเทียบกับตารางที่ 3.2 และ ตารางที่ 3.5 จะแสดงให้เห็นว่า ความผิดพลาดในการทดสอบ จะมีค่าน้อย กว่าความผิดพลาดในการ เรียนรู้ซึ่งแสดงให้เห็นว่า ความผิดพลาดในการทดสอบ ที่มีอัตราการเรียนรู้ที่ปรับตัวเองได้นั้น สามารถลดค่าความผิดพลาดได้น้อยกว่า วิธีดังเดิมที่เป็นการกำหนดค่าอัตราการเรียนรู้ที่กำหนดค่าเรียนรู้เป็นค่าคงที่

บทที่ 6
สรุปและข้อเสนอแนะ

6.1 ข้อสรุป

จากการทดลองในบทที่ 3 เมื่อการทดลองมีการเปลี่ยนค่าอัตราการเรียนรู้ค่าต่างๆ จะเห็นได้ว่าถ้าอัตราการเรียนรู้ที่มีค่าน้อยเกินไปซึ่งในการทดลองได้กำหนดอัตราการเรียนรู้เท่ากับ0.01 จากผลการทดลองจะเห็นว่า ความผิดพลาดก็จะไม่สามารถลดลงได้ ทำให้การเรียนรู้ไม่สามารถลู่เข้าสู่คำตอบได้ และในทำนองเดียวกันถ้ากำหนดค่าสูงเกินไป ในการทดลองนั้นได้กำหนดให้อัตราการเรียนรู้เท่ากับ 10 ผลการทดลองแสดงเห็นแล้วว่าไม่สามารถลงความผิดพลาดในการเรียนรู้ลงเลยทำให้ไม่สามารถลู่เข้าสู่คำตอบได้เช่นกัน สำหรับผลการทดลอง อัตราการเรียนรู้ที่ให้ผลความผิดพลาดน้อยที่สุดคือการทดลองที่กำหนดอัตราการเรียนรู้เท่ากับ 2 จากบทที่ 3 จะสรุปได้ว่าการเลือกอัตราการเรียนรู้ก็เรื่องสำคัญสำหรับการเรียนรู้ว่าจะลู่เข้าสู่คำตอบหรือไม่

การเลือกอัตราเรียนรู้ที่เหมาะสมกับโครงสร้าง หรือ dataset ที่จะใช้ในการทดลองนั้นขึ้นอยู่กับการลองผิดลองถูกและประสบการณ์ของผู้ทำวิจัยแต่ละท่าน ด้วยเหตุนี้จึงได้นำเสนอวิธีการให้อัตราการเรียนรู้ปรับตัวเองได้ จากบทที่ 4 จะแสดงวิธีมาได้ซึ่งอัตราการเรียนรู้ที่ปรับตัวเองได้ โดยอาศัยหลักการณ์ประมาณค่าฟังก์ชันความผิดพลาดจากอนุกรมเทร์เลอร์ เพื่อประมาณค่าความผิดพลาดและคำนวณกลับเป็นค่าอัตราการเรียนรู้ จากผลการทดลองในบทที่ 5 แสดงให้เห็นว่า อัตราการเรียนรู้มีการรปรับค่าทำให้ความผิดพลาดมีการลดลงและสามารถลู่เข้าหาคำตอบได้

6.2 ข้อเสนอแนะ

ในการทำวิจัย ผู้วิจัยได้ทำการประมวลผลผ่าน CPU ซึ่งทำให้ใช้เวลาการประมวลใช้เวลานาน ดังการทดลองในบทที่ 3 ตารางที่ 3.3 ที่ได้ทดลองคอมพิวเตอร์ 2 เครื่องที่มีคุณสมบัติที่แตกต่างกันส่งผลการทดลองได้แสดงให้เห็นแล้วว่า เวลาที่ใช้ในการคำนวณนอกจากจะขึ้นอยู่กับโครงสร้างของการเรียนรู้แล้วยังรวมถึงคุณสมบัติของเครื่องคอมพิวเตอร์ จากผลการทดลองนี้ทางผู้วิจัยอยากนำเสนอให้ทำการประมวลผลผ่าน GPU เพื่อเป็นการประหยัดเวลาในการทดลอง

เอกสารอ้างอิง

[1] Robert Hecht-Nielsen, “Theory of the Backpropagation Neural Network,” International 1989 Joint Conference on Neural Networks, IEEE Press,1989, pp. 593 – 605.

[2] Soniya, Sandeep Paul, Lotika Singh, “A Review on Advances in Deep Learning,”2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI),14-17 Dec. 2015.

[3] Neena Aloysius and Geetha M, “A Review on Deep Convolutional Neural Networks,” International Conference on Communication and Signal Processing, April 6-8, 2017, India, pp.588–592.

[4] Francis Quintal Lauzon, “An introduction to deep learning,” The 11th International Conference on Information Sciences, Signal Processing and their Applications: Special Sessions, pp.1438–1439.

[5] Xuedan du, Yinghao Cai, Shuo Wang, and Leijie Zhang, “Overview of Deep Learning,” 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp.159–164.

[6] U. Bhattacharya , S.K. Parui, “Self-adaptive learning rates in backpropagation algorithm improve its function approximation performance,” Proceedings of ICNN'95 - International Conference on Neural Networks.

[7] C. W. Kuo and J. S. Shi, “Cryptand/metalion coated piezoelectric quartz crystal sensors with artificial back propagation neural networkanalysis for nitrogen dioxide and carbon monoxide,” Sensors and Actuators B: Chemical, vol. 106, Apr. 2005, pp.468–476

[8] Yong Li ; Yang Fu ; Hui Li ; Si-Wen Zhang, “The Improved Training Algorithm of Back Propagation Neural Network with Selfadaptive Learning Rate,” 2009 International Conference on Computational Intelligence and Natural Computing, pp.73–76.

[9] Li Deng, “The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web],” IEEE Signal Processing Magazine (Volume: 29 , Issue: 6 , Nov. 2012).

[10] Zhiwen Shao, Shouhong Ding, Hengliang Zhu, Chengjie Wang and Lizhuang Ma, “Face alignment by deep convolutional network with adaptive learning rate”, 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 20-25 March 2016,PP1283-1287

ภาคผนวก

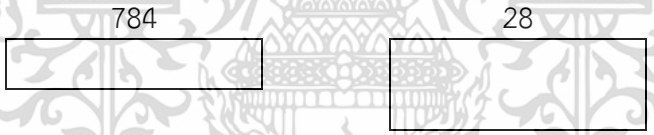
ภาคผนวก ก

ชุดข้อมูล MST

รูปที่ ก.1 ตัวอย่างชุดข้อมูล MNIST

ชุดข้อมูลMNIST เป็นข้อมูลที่เขียนด้วยมือ เป็นตัวเลข 0- 9 ซึ่งข้อมูลจะเป็นเป็น 2 ชุด คือสำหรับเรียนรู้และทดสอบ ซึ่งแต่ละชุดก็แบ่งเป็น 2 กลุ่มคือกลุ่มที่เป็น input และ output ซึ่งก่อนอื่นต้องทำความเข้าใจรูปแบบของข้อมูลและจัดรูปแบบของข้อมูลใหม่ดังนี้

อินพุต: การจัดเตรียมข้อมูลเพื่อใช้ในเขียนโปรแกรม เนื่องจากข้อมูล dataset ที่ได้มาจะมีการจัดเรียง 1รูปเป็น 1 array ที่มีความยาว 784 เพราเฉพาะนั้นในการเขียนโปรแกรมจะมีจัดรูปแบบใหม่เป็น Array 2 มิติ มีขนาดเป็น 28 X 28



รูปที่ ก.2 แปลงข้อจาก 1x784 เป็น 28x28

เอาต์พุต: จะมีกลุ่มอยู่ 10 กลุ่มคือ 0- 9 ดังนั้นเลยมีการสร้าง output เป็น 10note ดังในตาราง

ตารางที่ ก.1 แสดงความสัมพันธ์ระหว่าง note กับชื่อกลุ่ม

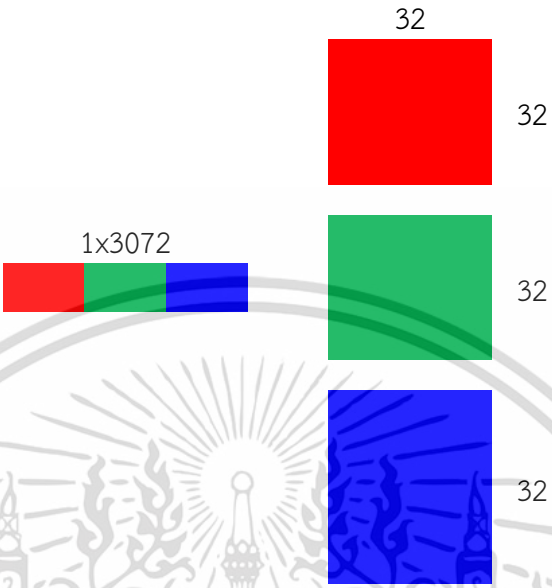
Note 1	Note 2	Note 3	Note 4	Note 5	Note 6	Note 7	Note 8	Note 9	Note 10	กลุ่ม
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	2
0	0	0	1	0	0	0	0	0	0	3
0	0	0	0	1	0	0	0	0	0	4
0	0	0	0	0	1	0	0	0	0	5
0	0	0	0	0	0	1	0	0	0	6
0	0	0	0	0	0	0	1	0	0	7
0	0	0	0	0	0	0	0	1	0	8
0	0	0	0	0	0	0	0	0	1	9

ชุดข้อมูล CIFAR-10 dataset

รูปที่ ก.3 ตัวอย่างชุดข้อมูล CIFAR-10 dataset

ชุดข้อมูล CIFAR-10 dataset

อินพุต: การจัดเตรียมข้อมูลเพื่อใช้ในเขียนโปรแกรม เนื่องจากข้อมูล dataset ที่ได้มาจะมีการจัดเรียง 1 รูปเป็น 1 array มีความยาว 3072 เพราเฉพาะนั้นในการเขียนโปรแกรมจะมีจัดรูปแบบใหม่เป็น 32 X 32 x3 โดยจัดรูปแบบ input ที่จากเดิมมีแค่ 1 note เป็น 3 note เพราะต้องการจัดรูปแบบ input เป็นแบบ array2 มิติดังรูป



รูปที่ ก.4 การจัดรูปแบบ input เป็นแบบarray2 มิติ

เอาต์พุต: จะมีกลุ่มอยู่ 10 กลุ่มคือ 0- 9 ดั่งนั้นเลยมีการสร้าง output เป็น 10 note ดังในตาราง

ตารางที่ ก.2 แสดงความสัมพันธ์ระหว่าง note กับชื่อกลุ่ม

Note 1	Note 2	Note 3	Note 4	Note 5	Note 6	Note 7	Note 8	Note 9	Note 10	กลุ่ม
1	0	0	0	0	0	0	0	0	0	airplane
0	1	0	0	0	0	0	0	0	0	automobile
0	0	1	0	0	0	0	0	0	0	bird
0	0	0	1	0	0	0	0	0	0	cat
0	0	0	0	1	0	0	0	0	0	deer
0	0	0	0	0	1	0	0	0	0	Dog
0	0	0	0	0	0	1	0	0	0	Frog
0	0	0	0	0	0	0	1	0	0	Horse
0	0	0	0	0	0	0	0	1	0	Ship
0	0	0	0	0	0	0	0	0	1	truck

ภาคผนวก ข

งานวิจัยที่ได้รับการตีพิมพ์

ITC IT 2018

"Communication and IT for Smart City"

KMUTNB 60 2019 INVENTION TO INNOVATION

Proceedings of
The 18th International Symposium on Communications and Information Technologies

September 26-29, 2018

at Sukosol Hotel, Bangkok, Thailand

KMUTNB

ECTI Association

IEEE THAILAND SECTION

IEEE Computational Intelligence Society Thailand Chapter

The 18th International Symposium on Communications and Information Technologies (ISCIT 2018)

The Improved Training Algorithm of Deep Learning with Self-Adaptive Learning Rate

1st Sutit Ongart
Department of Computer Engineering,
Faculty of Engineering,
King Mongkut's Institute of
Technology Ladkrabang
Bangkok, Thailand
sutit.ongart@gmail.com

2nd Kietikul Jearanaitanakij
Department of Computer Engineering,
Faculty of Engineering,
King Mongkut's Institute of
Technology Ladkrabang
Bangkok, Thailand
kietikul.je@kmitl.ac.th

3rd Jirapat Sangthong
Department of Telecommunication
Engineering, Faculty of Engineering,
Mahanakorn University of Technology
Bangkok, Thailand
jirapats@mutacth.com

***Abstract*— This paper proposed the improvement of convergence performance for deep learning. For traditional algorithm, the learning rate is depended on experience and experiment. In this work, the learning rate can be adaptived based on Taylor's formula. This formula has the relationship between the root mean square errors changed, connection template, weights and biases changes are obtained. The proposed self-adaptive learning rate is depended on neural network structure, root mean square error and error curve surface gradient. From the results, this proposed system has iteration times less than the traditional algorithm with constant learning rate.**

Keywords*— *back-propagation, deep learning, self-adaptive, learning rate.

I. Introduction

The back-propagation (BP) algorithm is generally used in training process. However, it has many challenges such as more learning late, less local training and connection weights that will be increased with the number of input. The input data have many types namely in binary number, text and image. There are many input data of image because it have a variety attributes with size and three primary colors. The input node of image can be calculated with width * length * 3 that provides a lots of input nodes.

Nowadays, the deep learning algorithm is already introduced. The structure of deep learning can be used to solve the input nodes problem. The process is the same method with back-propagation neural networks that adjusts the connection weights of each node. The deep learning can be divided into two processes. Forward process, the data and weights will be evaluated by using convolution, max-pooling and fully connected. Backward process, the weights will be adjusted for compared the speed of the error with the weights.

For fully connected of deep learning, the learning rate is defined by constant number. The much learning rate provides the good error results at the beginning because it adjusts weights many times. But, when it is convergence it maybe sways and cannot find the answer. The less learning rate provides the learning converges slowly that take a long training time. Kuo and Shi proposed the self-adaptive learning rate by using the comparison of loss function and the speed of error in neural network [1].

This work proposed the improved training algorithm of deep learning by using self-adaptive learning rate. The learning rate can be adaptived based on Taylor's formula. The MNIST database is used as the sample input data. The input data are the 70,000 images of 0-9 numbers. From the result, this proposed system has iteration times less than the traditional algorithm with constant learning rate.

II. Deep Learning Algorithm

The deep learning algorithm structure is shown in figure 1. It is including convolution, max-pooling and fully connected.

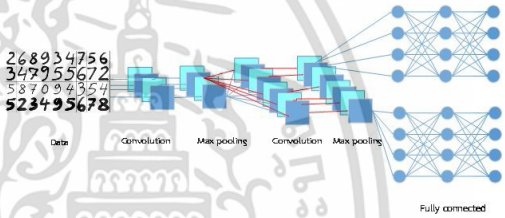


Fig.1. The deep learning algorithm structure.

A. Convolution

The convolution is including two parameters that are the number and size of templates. The template is used to converse the image data by defined the different number of templates. The number of template is very important. The less template cause it cannot recognize the image. The much template take a long iteration times. Besides that, it also take a long evaluation time in testing process. For the size of template, the small size template provided the good learning result but take a long time. While, the big size template take the uneffective convergence results because of swaying of the gradient.

The generally image filtering method uses the mean of the set of images or used the mean of the points that around the focus point. The mean of images is reduce the change of input data. It can be useful for remove the noise of high frequency signal. So, the signal filtering is used to emphasize the required image properties and reduce the unwanted image properties. For the depth level change, it is the same concept with high pass filtering. This method must use convolution instead mean number.

For image processing, the convolution is used between the template and image. The template is n*m matrix of number of the image for convolution. If K(x,y) is the n*m template and I(x,y) is the n*m image. The convolution between the template and the image can be represented by:

$$I'(X,Y) = K * I = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} K(i,j)\cdot I(X-i,Y-j) \quad (1)$$

978-1-5386-8458-0/18/$31.00 ©2018 IEEE

$$O(X,Y) = f(I'(X,Y)) \tag{2}$$

$$f(x) = \max(0,x) \tag{3}$$

where $I'(X,Y)$ is the image after convolution process, $f(x)$ is the Relu function and $O(X,Y)$ is the convolution output.

B. Max Pooling Layer

Max pooling layer is define the size of max pooling matrix. Max pooling find the layout output by considering the max value after the convolution method. The big matrix takes the convergence results sway from the answer. But, the small matrix takes converge slowly.

1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

6	8
3	4

Fig.2. Max pooling matrix 2*2 size.

C. Fully Connected

Fully connected of deep learning algorithm is the same concept as forward process of training process in neural network. This process must define number of node in hidden layer and output layer for group the data follow the number of output.

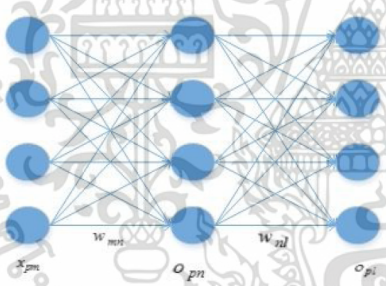


Fig.3. Fully connected structure.

$$o_j = f(\sum_{i=1}^{M} w_{ji} x_i + b_j) \tag{4}$$

$$o_{pk} = f(\sum_{j=1}^{N} w_{kj} x_j + b_k) \tag{5}$$

where o_j, o_{pk} is the output of hidden nodes and output nodes respectively, x_i is input, b_j, b_k is the biases of hidden nodes and output nodes respectively, w_{ji} is the weight between input nodes and hidden nodes, w_{kj} is the weight between hidden nodes and output nodes and M, N, L is the size of input nodes, hidden nodes and output nodes respectively.

D. Back propagation algorithm

Back propagation algorithm is included in to three phases: loss function, backward and weight update. For loss function phases, the square errors of one set data (E_p) can be defined as:

$$E_P = \frac{1}{2}\sum_{l=1}^{L}(t_l - o_l)^2 \tag{6}$$

Then, the square errors of all data (E) can be defined as:

$$E = \sum_{p=1}^{P} E_P = \frac{1}{2}\sum_{P=1}^{P}\sum_{l=1}^{L}(t_{pl} - o_{pl})^2 \tag{7}$$

After that, the root mean square errors of all data (E_{rms}) can be defined as:

$$E_{rms} = \frac{1}{P}\sum_{P=1}^{P} E_P = \frac{1}{2P}\sum_{P=1}^{P}\sum_{l=1}^{L}(t_{pl} - o_{pl})^2 \tag{8}$$

Where t_{Pl} is the target output and o_{Pl} is the output from l_{th} nodes in output layer that small different with sample data p_{th}.

The batch algorithm is different from the standard back propagation algorithm by weight update. For standard algorithm, the weight is updated after the data of each set is evaluated. But, the weight is updated after the data of all set are evaluated in batch algorithm. The weights and biases of batch algorithm between output layer and hidden layer can be calculated by:

$$\Delta w_{nl} = -\eta \frac{\partial E}{\partial w_{nl}} \tag{9}$$

$$\Delta b_l = -\eta \frac{\partial E}{\partial b_l} \tag{10}$$

Where η is the learning rate , w_{nl} is the weights between l_{th} nodes in output layer and n_{th} nodes in hidden layer, b_l is biases of l_{th} nodes in output layer

The template and biases-template of batch algorithm between input layer and hidden or convolution layer can be calculated by:

$$\Delta K_{nl} = -\eta \frac{\partial E}{\partial K_{nl}} \tag{11}$$

$$\Delta b_l = -\eta \frac{\partial E}{\partial b_l} \tag{12}$$

where K_{nl} is the template between n_{th} nodes in input nodes and l_{th} nodes in output nodes, b_l is biases of l_{th} nodes in output nodes

III. Proposed Training Algorithm

The traditional back propagation algorithm is very difficult to define the learning rate because it does not have the theory supported. Generally, the learning rate is selected from testing. If η is big, learning rate is fast but root mean square error will be swayed. On the other hand, learning rate and converge become slow if η is small. In training process, the weight and biases will be adjusted continuously for reduce E_m until the error is closed to the neural network structure. The sample of training process and target output, the all mean square error is the function of weights and biases.

$$E = f(w_{nl}, b_{ll}, K_{nl}, b_l) \tag{13}$$

Taylor's formula is not considered higher terms. The all mean square error can be evaluated by:

$$E(i+1) = E(i) + \frac{\partial E(o)}{\partial w_{nl}}\Delta w_{nl} + \frac{\partial E(i)}{\partial b_l}\Delta b_l + \frac{\partial E(i)}{\partial K_{nl}}\Delta K_{nl} + \frac{\partial E(i)}{\partial b_l}\Delta b_l \tag{14}$$

where

$$\Delta E = E(i+1) - E(i) \tag{15}$$

Then, the Taylor's formula will be evaluated with equation (9)-(12) and can be represented by:

$$\Delta E = -\eta\left[(\frac{\partial E(i)}{\partial w_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 + (\frac{\partial E(i)}{\partial k_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2\right] \le 0 \tag{16}$$

From increase the speed of convergence, the learning rate can be defined by:

$$\eta = \frac{-\Delta E}{\left[(\frac{\partial E(i)}{\partial w_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 + (\frac{\partial E(i)}{\partial k_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2\right]} \tag{17}$$

Then, the equation (17) can be represented by:

$$\eta = \frac{-\Delta E}{\delta} \tag{18}$$

where

$$\delta = \delta_1 + \delta_2 \tag{19}$$

$$\delta_1 = (\frac{\partial E(i)}{\partial w_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 \tag{20}$$

$$\delta_2 = (\frac{\partial E(i)}{\partial k_{nl}})^2 + (\frac{\partial E(i)}{\partial b_l})^2 \tag{21}$$

δ_1 is gradient in fully connected layer, δ_2 is gradient in Convolution layer

Gradient is used to shows the error of curvature gradient. It considers by using error signal at considered node and all output value.

The value of ΔE for iteration at $E(i+1)$ cannot be calculated. Then, equation (15) will be evaluated by using equation (7) and can be represented by:

$$E(i) = \frac{1}{2}\sum_{p=1}^{P}\sum_{l=1}^{L}\{[t_{pl} - o_{pl}(i)]^2\} \tag{22}$$

$$E(i+1) = \frac{1}{2}\sum_{p=1}^{P}\sum_{l=1}^{L}\{[t_{pl} - o_{pl}(i+1)]^2\} \tag{23}$$

$$-\Delta E = \frac{1}{2}\sum_{p=1}^{P}\sum_{l=1}^{L}\{[t_{pl} - y_{pl}(i)]^2 - [t_{pl} - y_{pl}(i+1)]^2\} \tag{24}$$

Use $\frac{P}{P}$ in $[t_{pl} - y_{pl}(i)]^2$ then

$$[t_{pl} - y_{pl}(i)]^2 \frac{P}{P} = [t_{pl} - y_{pl}(i)]^2 \frac{1}{P} + [t_{pl} - y_{pl}(i)]^2 \frac{p-1}{P}$$

Then

$$-\Delta E = \frac{1}{2}\sum_{p=1}^{P}\sum_{l=1}^{L}\{[t_{pl} - y_{pl}(i)]^2 \frac{1}{P} + [t_{pl} - y_{pl}(i)]^2 \frac{p-1}{P} - [t_{pl} - y_{pl}(i+1)]^2\}$$

If the sum of sample data is large, it can be estimated by:

$$(p-1)/p \approx 1 \tag{28}$$

For the more iteration training process, it can be estimated by:

$$[t_{pl} - o_{pl}(i)]^2 \approx [t_{pl} - o_{pl}(i+1)] \tag{29}$$

Then, the equation (24) can be reform and can be represented by:

$$-\Delta E = \frac{1}{2P}\sum_{p=1}^{P}\sum_{l=1}^{L}[t_{pl} - o_{pl}(k)]^2 = E_{rms} \tag{30}$$

Finally, the equation (30) can be reform and can be represented by:

$$\eta = \frac{E_{rms}}{\delta} \tag{31}$$

Now we can derive the deep learning training algorithm by Self-adaptive learning rate

Step 1 Initialization
Step 2 Activation
- Convolution, Max-poling, and fully-connection
Step 3 convolution and weight training
-Calculate the root mean square errors of all data (E_{rms})
- Calculate the error gradient δ
- Calculate the learning rate $\eta = \frac{E_{rms}}{\delta}$
-Update the Template on convolution layer and weight on fully Connected
Step 4 Increase iteration, go back to step 2 repeat until the error criterion is satisfied

IV. Results

In this section, the proposed training algorithm will be tested. We used MNIST database as the sample input data. The input data are the 70,000 images of 0-9 numbers. The 60,000 images are used for training process. The less 10,000 images are used for testing process. The table I shows the result by comparing between the traditional training algorithm and proposed training algorithm.

TABLE I. THE COMPARISON BETWEEN TRADITIONAL AND PROPOSED TRAINING ALGORITHM

Iteration	Traditional Training Algorithm		Proposed Training Algorithm	
	Time (m)	*Error*	*Time (m)*	*Error*
10	34	0.0556	35	0.0436

From the result, the time of both algorithms is not significant different. However the error of proposed training algorithm is better than the traditional algorithm with 0.0436.

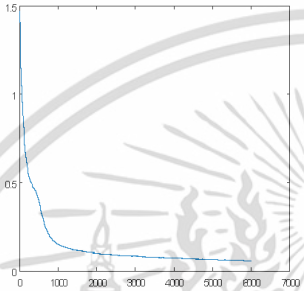


Fig.4. The training error of self-adaptive learning rate.

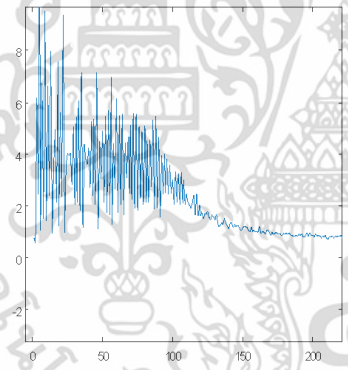


Fig.5. The self-adaptive learning rate in each iteration.

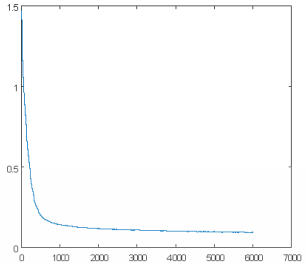


Fig.6. The training error of 0.75 learning rate.

V. CONCLUSION

This work proposed the improved training algorithm of deep learning by using self-adaptive learning rate. The learning rate can be adaptived based on Taylor's formula. The MNIST database is used as the sample input data. The input data are the 70,000 images of 0-9 numbers. From the result, this proposed system has iteration times less than the traditional algorithm with constant learning rate.

REFERENCES

[1] C. W. Kuo and J. S. Shi, "Cryptand/metalion coated piezoelectric quartz crystal sensors with artificial back propagation neural networkanalysis for nitrogen dioxide and carbon monoxide," Sensors and Actuators B: Chemical, vol. 106, Apr. 2005, pp.468–476.

[2] H. Cai , D. Q. Sun, Q. Y. Cao and F. Pu, "A novel BP algorithm based on self-adaptive parameters and performance analysis,"Proceedings of the First International Conference on Innovative Computing, Information and Control(ICICIC06), IEEE Press, Sep. 2007, pp. 383 – 387.

[3] C. Zhang and T. H. Keung Kwan, "Parameter effects on convergence speed and generalization capability of backpropagation algorithm," International Journal of Electronics, vol. 74, Jan. 1993, pp. 35–46.

[4] N. Mounir and C. Mohamed, "A hybrid training algorithm for feedforward neural networks," Neural Processing Letters, vol. 24, Oct. 2006, pp. 107–117.

[5] J. Wen, J.L. Zhao, S.W.Luo and Z. Han, "The improvements of bp neural network learning algorithm," Prodeedings of 5th International Conference on Signal Processing (WCCC-ICSP 2000), IEEE Press, Aug. 2000, pp. 1647–1649.

[6] D. E. Rumelhart, G. E. Hinton and R.J. Williams, "learning internal representations by error propagation," Parallel Distributed Processing, vol.1, MA: MIT Press, 1986, pp. 318–362.

[7] W. Wu, H. M. Shao and Z. X. Li, "Convergence of batch bp algorithm with penalty for FNN training," Proceedings of the Fifth Mexican International Conference on Artificial Intelligence (MICAI2006), IEEE Press, Nov. 2006, pp. 245–252.

ประวัติผู้เขียน

ชื่อ-นามสกุล	นาย สุทิศ องอาจ
ที่อยู่	53 ม.1 ต.รุง อ.กันทรลักษ์ จ.ศรีสะเกษ
ประวัติการศึกษา	2553วิศวกรรมศาสตรบัณฑิตสาขาวิชาวิศวกรรมวัดคุม
	(เกียรตินิยมอันดับ1) มหาวิทยาลัยเทคโนโลยีมหานคร
ความชำนาญเฉพาะด้าน	1.) ระบบควบคุมอัตโนมัติ,IOT
	2.) Mysql, PHP, Asp.net , C#, Angular js,Vue js
	3) windows form ,android ,IOS, Xamarin
	4.) สมองกลฝั่งตัว,Drone 4 ใบพัด
	5) Matlab ,Scilat ,Labview

ประสบการณ์การทำงานและผลงานวิจัย

พ.ศ.2554 - 30 ก.ค. 2562	อาจารย์ประจำภาควิชาวิศวกรรมรโทรคมนาคม
	คณะ วิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีมหานคร
1 สิงหาคม 2562 – ปัจจุบัน	ตำแหน่ง senior software Engineer
	บริษัทสามารถคอร์ปอเรชั่น จำกัด