นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง: การจัดการกับโปรแกรมตรรกะอันดับที่หนึ่ง
ด้วยนิวรอลเน็ตเวิร์ก

นายธนุพล เลิศลำเนาชัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2547
ISBN 974-17-7108-8

FIRST-ORDER LOGICAL NEURAL NETWORKS: MAKING NEURAL NETWORKS HANDLE FIRST-ORDER LOGIC PROGRAMS

Mr. Thanupol Lerdlamnaochai

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Computer Engineering
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2004

ISBN 974-17-7108-8

หัวข้อวิทยานิพนธ์	นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง: การจัดการกับโปรแกรมตรรกะอันดับที่หนึ่งด้วยนิวรอลเน็ตเวิร์ก
โดย	นายธนุพล เลิศลำเนาชัย
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษา	รองศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

…….. คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.ดิเรก ลาวัณย์ศิริ)

คณะกรรมการสอบวิทยานิพนธ์

…… ประธานกรรมการ
(รองศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

…….. อาจารย์ที่ปรึกษา
(รองศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล)

…… กรรมการ
(อาจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์)

…… กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ณชล ไชยรัตนะ)

นายธนุพล เลิศลำเนาชัย : นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง: การจัดการกับโปรแกรมตรรกะอันดับที่หนึ่งด้วยนิวรอลเน็ตเวิร์ก. (FIRST-ORDER LOGICAL NEURAL NETWORKS: MAKING NEURAL NETWORKS HANDLE FIRST-ORDER LOGIC PROGRAMS) อ. ที่ปรึกษา : รศ.ดร.บุญเสริม กิจศิริกุล, 57 หน้า. ISBN 974-17-7108-8.

การโปรแกรมตรรกะเชิงอุปนัยหรือไอแอลพีเป็นวิธีการเรียนรู้ที่มีข้อดีที่สามารถนำความรู้ภูมิหลังมาใช้ในการเรียนรู้ได้และแนวคิดหรือกฎที่ได้จากการเรียนรู้ก็อยู่ในรูปแบบที่มนุษย์เข้าใจได้ง่าย อย่างไรก็ตามระบบไอแอลพีก็มีข้อจำกัดเนื่องจากไม่สามารถจำแนกตัวอย่างได้ในกรณีที่ตัวอย่างนั้นไม่ตรงกับกฎข้อใดเลยในเซตของกฎที่ได้จากการเรียนรู้ วิทยานิพนธ์นี้จึงนำเสนอวิธีการแก้ปัญหาโดยการนำแนวคิดของระบบไอแอลพีรวมเข้ากับนิวรอลเน็ตเวิร์กได้เป็นระบบใหม่ที่มีความยืดหยุ่นมากขึ้น ระบบที่ได้มีชื่อว่า นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง (First-Order Logical Neural Network: FOLNN) ระบบนี้มีโครงสร้างพื้นฐานมาจากนิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้า ที่ประยุกต์ให้สามารถรับอินพุตที่เป็นตัวอย่างและความรู้ภูมิหลังที่อยู่ในรูปแบบของโปรแกรมตรรกะอันดับที่หนึ่งมาใช้ในการเรียนรู้ได้โดยตรง ในการทดลองได้ใช้ชุดข้อมูลที่เป็นตรรกะอันดับที่หนึ่งจำนวน 2 ชุด ผลการทดลองเปรียบเทียบระหว่างวิธีที่นำเสนอกับระบบ PROGOL ซึ่งเป็นระบบที่รู้จักอย่างแพร่หลายของระบบไอแอลพี ปรากฏว่าวิธีที่นำเสนอให้ค่าเปอร์เซ็นต์ความถูกต้องสูงกว่าระบบ PROGOL ทั้ง 2 ชุดข้อมูล นอกจากนี้ยังได้ทดลองเพื่อแสดงให้เห็นถึงความทนทานต่อข้อมูลที่มีสัญญาณรบกวน ผลการทดลองปรากฏว่าค่าเปอร์เซ็นต์ความถูกต้องของวิธีที่ใช้ในงานวิจัยนี้ลดลงช้ากว่าค่าเปอร์เซ็นต์ความถูกต้องของระบบ PROGOL

ภาควิชา.... วิศวกรรมคอมพิวเตอร์.....ลายมือชื่อนิสิต.....
สาขาวิชา....วิศวกรรมคอมพิวเตอร์.....ลายมือชื่ออาจารย์ที่ปรึกษา.....
ปีการศึกษา .....2547.....ลายมือชื่ออาจารย์ที่ปรึกษาร่วม.....

4670327421 : MAJOR Computer Engineering

KEY WORD: FIRST-ORDER LOGIC / INDUCTIVE LOGIC PROGRAMMING / NEURAL NETWORKS

THANUPOL LERDLAMNAOCHAI : (FIRST-ORDER LOGICAL NEURAL NETWORKS: MAKING NEURAL NETWORKS HANDLE FIRST-ORDER LOGIC PROGRAMS. THESIS ADVISOR : ASSOC. PROF. BOONSERM KIJSIRIKUL, Ph.D., 57 pp. ISBN 974-17-7108-8.

The main advantages of Inductive Logic Programming (ILP) are the ability of employing background knowledge and inducing human readable representations in form of a set of first-order rules. Nevertheless, ILP systems have the restriction to the classification of imperfect data such as noisy unseen data which may not be covered by any learned rules. This thesis proposes a novel flexible learning method called First-Order Logical Neural Network (FOLNN) to alleviate the restriction of rule-based systems. FOLNN is based on the feedforward neural network that integrates inductive learning from examples and background knowledge. In addition, the proposed method enables neural networks to process first-order logic programs directly. In the experiments, FOLNN has been evaluated on two domains of first-order learning problems and compared with PROGOL, the state-of-the-art ILP system. The experimental results show that our proposed method provides more accurate results than PROGOL in both datasets. Furthermore, we also evaluate FOLNN on noisy domain to see how well the learner is robust to noisy data. The results show that the accuracy of our method decreases much slower and is much higher than that of PROGOL.

Department…. Computer Engineering…. Student's signature.....
Field of study…. Computer Engineering…Advisor's signature.....
Academic year …2004…….Co-advisor's signature.....

กิตติกรรมประกาศ

การวิจัยและวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้เป็นอย่างดีด้วยความอนุเคราะห์อย่างยิ่งของรองศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล อาจารย์ที่ปรึกษา โดยท่านได้ให้ความรู้ คำแนะนำและข้อคิดเห็นอันเป็นประโยชน์ในการวิจัยมาโดยตลอด จนทำให้มีวิทยานิพนธ์ฉบับนี้เกิดขึ้น รวมถึงผลงานทางวิชาการที่ได้รับการตีพิมพ์ในที่ต่างๆ รวมทั้งคณะกรรมการสอบวิทยานิพนธ์ทุกท่านที่ได้ให้ข้อเสนอแนะอันเป็นประโยชน์ยิ่งในการทำวิจัย

ขอขอบคุณสมาชิกห้องปฏิบัติการอัจฉริยภาพเครื่องจักรและการค้นพบความรู้ (MIND LAB) เพื่อนๆ พี่ๆ น้องๆทุกคนในห้องปฏิบัติการชั้น 20 และเพื่อนร่วมรุ่นของข้าพเจ้าที่ให้คำแนะนำและช่วยแก้ปัญหาเบ็ดเตล็ดในงานวิจัย และทำให้ชีวิตในการเรียนระดับปริญญาโทของข้าพเจ้าเต็มเปี่ยมไปด้วยความปีติ

ท้ายที่สุดนี้ผู้วิจัยขอกราบขอบพระคุณบิดา มารดา และทุกคนในครอบครัว ที่ให้การสนับสนุน ห่วงใยและให้กำลังใจข้าพเจ้าเสมอมา

สารบัญ

หน้า

สารบัญภาพ

หน้า

สารบัญตาราง

หน้า

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การเรียนรู้ของเครื่อง (Machine Learning) [1] มีแนวคิดมาจากความต้องการที่จะให้คอมพิวเตอร์มีการเรียนรู้และสามารถพัฒนาตนเองได้ด้วยประสบการณ์เหมือนดังเช่นมนุษย์ จึงได้เกิดความพยายามที่จะสร้างโปรแกรมคอมพิวเตอร์ที่สามารถตอบสนองแนวคิดนี้ได้ หรืออาจจะกล่าวอีกนัยหนึ่งคือ เป็นแนวคิดที่ต้องการสร้างโปรแกรมคอมพิวเตอร์ให้สามารถเรียนรู้จากตัวอย่างที่ผู้สอนได้จัดเตรียมไว้ให้ โดยโปรแกรมที่ได้สร้างขึ้นนั้นจะต้องสามารถเรียนรู้เพื่อสร้างแนวคิดที่ครอบคลุมกับตัวอย่างที่ผู้สอนต้องการ และสามารถนำแนวคิดที่ได้นั้นไปใช้ในการจำแนกตัวอย่างใหม่ที่ระบบไม่ได้นำมาใช้ในการเรียนรู้ได้

การเรียนรู้ของเครื่องมีหลายประเภทด้วยกัน เช่น การโปรแกรมตรรกะเชิงอุปนัย (Inductive Logic Programming: ILP) นิวรอลเน็ตเวิร์ก (Neural Networks) เน็ตเวิร์กเบย์ (Bayesian Networks) การเรียนรู้แบบต้นไม้ตัดสินใจ (Decision Tree Learning) เป็นต้น และจากความสามารถของการเรียนรู้ของเครื่องทำให้ในปัจจุบันได้มีการนำการเรียนรู้ของเครื่องไปประยุกต์ใช้กับงานในด้านต่างๆ มากมาย เช่น การรู้จำตัวอักษร การรู้จำเสียงพูด การทำเหมืองข้อมูล การตรวจโรคของผู้ป่วย การวิเคราะห์โครงสร้างของโมเลกุล เป็นต้น

การโปรแกรมตรรกะเชิงอุปนัยหรือไอแอลพีเป็นวิธีการเรียนรู้ของเครื่องประเภทหนึ่งซึ่งมีการนำแนวคิดทางตรรกะมาประยุกต์ใช้ ทำให้มีข้อดีที่แตกต่างจากการเรียนรู้ของเครื่องแบบอื่นๆ เนื่องจากแนวคิดที่ได้จากการเรียน (concept) จะอยู่ในรูปแบบของตรรกะอันดับที่หนึ่ง (first-order logic) ซึ่งเป็นลักษณะของการอธิบายความรู้ของมนุษย์เป็นส่วนใหญ่ (if-then rules) ทำให้สามารถเข้าใจได้ง่ายกว่าแนวคิดที่ได้จากวิธีอื่น นอกจากนี้ตรรกะอันดับที่หนึ่งยังสามารถอธิบายแนวความคิดได้ซับซ้อนกว่าตรรกศาสตร์ประพจน์ (propositional logic) ด้วย ในการเรียนรู้ระบบไอแอลพีจะรับอินพุตเป็น ความรู้ภูมิหลัง (background knowledge) ตัวอย่างบวก (positive examples) และตัวอย่างลบ (negative examples) เพื่อสร้างกฎที่ครอบคลุมตัวอย่างบวกแต่ไม่ครอบคลุมตัวอย่างลบ โดยกฎที่ได้ ความรู้ภูมิหลัง ตัวอย่างบวก และตัวอย่างลบ ทั้งหมดจะอยู่ในรูปแบบตรรกะอันดับที่หนึ่ง

ตัวอย่างของระบบไอแอลพี ได้แก่ PROGOL [2], GOLEM [3] และ FOIL [4] เป็นต้น ระบบไอแอลพีเหล่านี้ได้ถูกนำไปประยุกต์ใช้กับงานในปัจจุบันได้แก่ การออกแบบไฟไนต์เอลิเมนต์

การเรียนรู้กฎจากฐานข้อมูลเกมหมากรุก การวิเคราะห์ความสามารถในการก่อกลายพันธุ์ของโมเลกุล การรู้จำตัวอักษรภาษาไทยเป็นต้น

โดยปกติแล้วระบบไอแอลพีมักจะนำมาใช้ในการจำแนกตัวอย่างที่มี 2 ประเภท (class) โดยพยายามสร้างกฎหรือแนวคิดที่ครอบคลุมตัวอย่างบวกและไม่ครอบคลุมตัวอย่างลบที่ใช้ในการเรียนรู้ จากนั้นเมื่อนำกฎที่ได้ไปใช้ในการจำแนกตัวอย่างทดสอบ ก็จะจำแนกว่าตัวอย่างนั้นเป็นตัวอย่างบวก ถ้าตัวอย่างนั้นตรงกับกฎ แต่ถ้าไม่ตรงกับกฎก็จะจำแนกว่าเป็นตัวอย่างลบ ซึ่งถ้าเรานำระบบไอแอลพีไปใช้ในการจำแนกตัวอย่างแบบหลายประเภท (multi-class classification) ตัวอย่างนั้นต้องตรงกับกฎจึงจะสามารถจำแนกได้ว่าตัวอย่างนั้นอยู่ประเภทใด แต่ในบางครั้งตัวอย่างที่นำมาจำแนกอาจมีความผิดพลาด เช่น ถูกสัญญาณรบกวนทำให้ข้อมูลบางส่วนผิดพลาดหรือหายไป (noisy data) หรือมีความรู้ภูมิหลังที่ใช้ในการสอนที่ไม่สมบูรณ์ ทำให้ตัวอย่างนั้นไม่ตรงกับกฎข้อใดข้อหนึ่งในเซตของกฎทั้งหมด และไม่สามารถที่จะจำแนกได้

เนื่องด้วยข้อจำกัดของระบบไอแอลพีที่ไม่สามารถจำแนกตัวอย่างในลักษณะนี้ได้ จำเป็นต้องมีวิธีการอื่นเข้ามาช่วยเพื่อทำให้ระบบไอแอลพีสามารถจำแนกตัวอย่างที่มีลักษณะเป็นหลายประเภท รวมถึงสามารถจำแนกตัวอย่างที่ไม่ตรงกับกฎได้ ในงานวิจัยนี้จะนำเสนอระบบการเรียนรู้แบบใหม่ที่ยืดหยุ่นขึ้น โดยนำนิวรอลเน็ตเวิร์กที่สามารถจำแนกตัวอย่างที่มีลักษณะแบบหลายประเภท และสามารถทนต่อข้อมูลที่มีสัญญาณรบกวนได้ดี มาประยุกต์เข้ากับการโปรแกรมตรรกะเชิงอุปนัยที่มีการนำความรู้ภูมิหลังมาใช้ในการเรียนรู้และสามารถอธิบายความรู้ได้กว้างขวาง ระบบที่พัฒนาขึ้นนี้สามารถจัดการกับข้อจำกัดของโปรแกรมตรรกะเชิงอุปนัยได้และให้ผลในการจำแนกที่แม่นยำมากขึ้น รวมทั้งสามารถรับอินพุตในรูปแบบของตรรกะอันดับที่หนึ่งมาเรียนรู้ได้โดยตรง ไม่ต้องใช้ระบบไอแอลพี โดยเรียกระบบนี้ว่า นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง (First-Order Logical Neural Network: FOLNN)

1.2 วัตถุประสงค์

เพื่อจัดการกับข้อจำกัดของการโปรแกรมตรรกะเชิงอุปนัยในการจำแนกตัวอย่างที่ไม่ตรงกับกฎ โดยใช้นิวรอลเน็ตเวิร์ก

1.3 ขอบเขตของการวิจัย

1. พัฒนานิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งซึ่งสามารถรับอินพุตในรูปแบบของตรรกะอันดับที่หนึ่งมาใช้ในการเรียนรู้
2. เปรียบเทียบประสิทธิภาพกับระบบไอแอลพี PROGOL [2] เป็นอย่างน้อย

1.4 ขั้นตอนและวิธีดำเนินงานวิจัย

1. ศึกษาแนวคิดและทฤษฎีการเรียนรู้ของวิธีการโปรแกรมตรรกะเชิงอุปนัย นิวรอลเน็ตเวิร์ก และการเรียนรู้แบบหลายตัวอย่างย่อย
2. ศึกษางานวิจัยที่เกี่ยวข้อง
3. ออกแบบอัลกอริทึมและพัฒนานิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง
4. ทดสอบประสิทธิภาพของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งกับกลุ่มตัวอย่างทดสอบ
5. วิเคราะห์ผลเปรียบเทียบกับระบบไอแอลพี PROGOL และปรับปรุงประสิทธิภาพของอัลกอริทึม
6. สรุปผลการทดลอง และจัดทำวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้วิธีการเรียนรู้แบบใหม่ที่ใช้จัดการกับปัญหาของการโปรแกรมตรรกะเชิงอุปนัยในการจำแนกตัวอย่างที่ไม่ตรงกับกฎ
2. ได้นิวรอลเน็ตเวิร์กแบบใหม่ที่นำข้อดีของการโปรแกรมตรรกะเชิงอุปนัยมาประยุกต์ทำให้สามารถรับความรู้ภูมิหลังมาใช้ในการเรียนรู้และสามารถรับอินพุตในรูปแบบของตรรกะอันดับที่หนึ่งได้

1.6 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์

วิทยานิพนธ์นี้แบ่งเนื้อหาออกเป็น 5 บทดังนี้ บทที่ 1 เป็นบทนำซึ่งกล่าวถึงที่มาและความสำคัญของปัญหา รวมทั้งวัตถุประสงค์ของงานวิจัยชิ้นนี้ บทที่ 2 กล่าวถึงทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้องในงานวิจัยนี้ บทที่ 3 กล่าวถึงรายละเอียดทั้งหมดของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง (First-Order Logical Neural Network: FOLNN) ซึ่งเป็นวิธีที่นำเสนอเพื่อใช้แก้ปัญหาในงานวิจัยนี้ บทที่ 4 แสดงรายละเอียดของการทดลองและผลการทดลอง และบทที่ 5 จะเป็นข้อสรุปและข้อเสนอแนะจากการวิจัย

1.7 ผลงานที่ตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้รับการตีพิมพ์เป็นบทความทางวิชาการและนำเสนอในงานประชุมวิชาการ มีรายละเอียดดังนี้

1. Thanupol Lerdlamnaochai and Boonserm Kijsirikul. 2004. A New Framework for Learning First-Order Representation. The 8th Annual National Symposium

on Computational Science and Engineering (ANSCSE8), July, Nakhon Ratchasima, Thailand.

2. Thanupol Lerdlamnaochai and Boonserm Kijsirikul. 2004. Learning Logic Programs by First-Order Neural Networks. The 8th National Computer Science and Engineering Conference (NCSEC2004), October, Songkhla, Thailand.

3. Thanupol Lerdlamnaochai and Boonserm Kijsirikul. 2004. First-Order Logical Neural Networks. The Fourth International Conference on Hybrid Intelligent Systems (HIS'04), December, Kitakyushu, Japan.

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

บทนี้จะแบ่งเนื้อหาออกเป็น 2 ส่วนด้วยกัน โดยเนื้อหาในส่วนแรกจะกล่าวถึงทฤษฎีและแนวคิดพื้นฐาน มีเนื้อหาประกอบด้วย การโปรแกรมตรรกะเชิงอุปนัย นิวรอลเน็ตเวิร์ก และการเรียนรู้แบบหลายตัวอย่างย่อย และเนื้อหาในส่วนที่สองจะกล่าวถึงงานวิจัยที่เกี่ยวข้องกับงานวิจัยนี้

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 การโปรแกรมตรรกะเชิงอุปนัย

การโปรแกรมตรรกะเชิงอุปนัย หรือไอแอลพี (Inductive Logic Programming: ILP) [1, 5, 6] เป็นการนำวิธีการเรียนรู้ของเครื่องมาใช้ร่วมกับวิธีการโปรแกรมตรรกะ (Logic Programming) โดยยังมีแนวคิดหลักเป็นไปตามแนวคิดการเรียนรู้ของเครื่อง คือ พยายามสร้างแนวคิดหรือสมมติฐานที่ครอบคลุมตัวอย่างบวกแต่ไม่ครอบคลุมตัวอย่างลบ แต่ว่าการโปรแกรมตรรกะเชิงอุปนัยมีความแตกต่างจากการเรียนรู้ของเครื่องแบบอื่นในส่วนที่ได้มีการนำเอาโปรแกรมตรรกะมาใช้ในการอธิบายตัวอย่างและความรู้ภูมิหลัง ระบบจะทำการเรียนรู้เพื่อสร้างแนวคิดในรูปแบบของโปรแกรมตรรกะจากตัวอย่างและความรู้ภูมิหลังที่ได้รับ แนวคิดที่ได้นี้สามารถนำไปใช้ในการจำแนกตัวอย่างใหม่ที่ระบบยังไม่เคยเห็นหรือไม่ได้ใช้ในการสอนได้ ซึ่งการใช้โปรแกรมตรรกะเชิงอุปนัยในการอธิบายสมมติฐานและตัวอย่าง ทำให้มีข้อดีที่เหนือกว่าวิธีอื่นๆ คือ

1. เนื่องจากแนวคิดที่ได้จากการเรียนของโปรแกรมตรรกะเชิงอุปนัยจะอยู่ในรูปแบบของกฎที่เป็นเงื่อนไข (if-then rule) ซึ่งความรู้ส่วนใหญ่ของมนุษย์ก็อธิบายได้ด้วยกฎ ลักษณะนี้ ทำให้แนวคิดที่ได้สามารถอ่านและเข้าใจได้ง่าย (human readable) รวมทั้งอธิบายความรู้ของมนุษย์ได้ดีกว่าการเรียนรู้ของเครื่องแบบอื่น
2. การโปรแกรมตรรกะเชิงอุปนัยสามารถใช้ความรู้ภูมิหลังในการเรียนรู้ได้ง่ายกว่าวิธีการเรียนรู้ของเครื่องแบบอื่น ที่ไม่ได้ใช้การโปรแกรมตรรกะ ทำให้ผู้สอนสามารถใส่ความรู้ที่เป็นประโยชน์เพิ่มเข้าไปในการเรียนรู้ได้

โดยปกติแล้วตรรกะที่นำมาใช้อธิบายแนวคิดหรือตัวอย่างมีอยู่ 2 ประเภทด้วยกัน คือ ตรรกะอันดับที่หนึ่ง (first-order logic) และตรรกศาสตร์ประพจน์ (propositional logic) ตรรกะทั้ง 2 ประเภทมีความแตกต่างกันตรงที่การอธิบายด้วยตรรกศาสตร์ประพจน์นั้นจะไม่มีการนำตัวแปร

มาใช้ ในขณะที่ถ้าอธิบายด้วยตรรกะอันดับที่หนึ่งจะสามารถใช้ตัวแปรมาช่วยอธิบายได้ เพื่อแสดงให้เห็นถึงความแตกต่างระหว่างการอธิบายแนวคิดด้วยตรรกศาสตร์ประพจน์เปรียบเทียบกับการอธิบายด้วยตรรกะอันดับที่หนึ่ง จะยกตัวอย่างประกอบดังนี้

ในการเรียนรู้ของเครื่องเพื่อให้สร้างแนวคิดของ Daughter(x,y) ซึ่งใช้อธิบายความสัมพันธ์ระหว่างบุคคล 2 คน คือ x และ y ซึ่งคนหนึ่งเป็นบุตรสาวของอีกคนหนึ่ง โดยค่าของ Daughter(x,y) จะเป็นจริง ก็ต่อเมื่อ x มีบุตรสาวเป็น y นอกเหนือจากนี้จะให้ค่าเป็นเท็จ เพื่อให้ระบบสามารถสร้างแนวคิดได้จะต้องใส่ข้อมูลที่ใช้ในการเรียนรู้ให้แก่ระบบ กำหนดให้ข้อมูลของแต่ละคนประกอบไปด้วยคุณสมบัติดังนี้ Name, Mother, Father, Male, Female ในขั้นตอนการเรียนรู้เพื่อสร้างแนวความคิด Daughter(x,y) อินพุตที่ใช้จะประกอบด้วยข้อมูลของบุคคล 2 คน และค่าของแนวคิดเป้าหมาย Daugther ดังตัวอย่างบวกที่แสดงในรูปที่ 2.1 ซึ่งเป็นความสัมพันธ์ระหว่าง Sharon กับ Bob โดย Sharon เป็นลูกสาวของ Bob

$Name_1$ = Sharon,	$Mother_1$ = Louise,	$Father_1$ = Bob,
$Male_1$ = False,	$Female_1$ = True,	
$Name_2$ = Bob,	$Mother_2$ = Nora,	$Father_2$ = Victor,
$Male_2$ = True,	$Female_2$ = False,	$Daughter_{1,2}$ = True

รูปที่ 2.1 ตัวอย่างบวกที่ให้แก่ระบบเพื่อทำการเรียนรู้แนวคิด Daughter

หลังจากทำการป้อนตัวอย่างดังแสดงในรูปที่ 2.1 ให้แก่ระบบในการเรียนรู้ ในกระบวนการเรียนรู้ของตัวเรียนรู้แบบตรรกศาสตร์ประพจน์ (propositional learner) เช่น CN2 หรือ C4.5 [7] จะได้ผลที่มีลักษณะดังรูปที่ 2.2

IF $(Father_1 = Bob) \wedge (Name_2 = Bob) \wedge (Female_1 = True)$

THEN $Daughter_{1,2}$ = True

รูปที่ 2.2 กฎที่อธิบายด้วยตรรกศาสตร์ประพจน์

จะเห็นได้ว่ากฎที่ได้นี้มีความถูกต้อง สอดคล้องกับตัวอย่างบวกที่ได้รับและสามารถนำไปใช้ได้ แต่กฎนี้เป็นกฎที่มีข้อจำกัดมากเกินไป เนื่องจากมีการระบุค่าเฉพาะลงไปในกฎด้วย ทำให้กฎที่ได้ไม่สามารถใช้จำแนกความสัมพันธ์ของคนสองคนอื่นๆที่ไม่ใช่ Sharon กับ Bob ได้ ต่างกับกฎที่ได้จากการเรียนด้วยการโปรแกรมตรรกะเชิงอุปนัยซึ่งอธิบายแนวคิดด้วยตรรกะอันดับที่หนึ่งมีลักษณะเป็นดังรูปที่ 2.3

IF Father(y, x) ∧ Female(y)

THEN Daughter(x, y)

รูปที่ 2.3 กฎที่อธิบายด้วยตรรกะอันดับที่หนึ่ง

เมื่อ x และ y เป็นตัวแปรซึ่งสามารถใช้แทนบุคคลใดๆได้ กฎที่ได้มีความหมายว่า ถ้า y มีพ่อเป็น x และ y เป็นเพศหญิงแล้ว x จะมีลูกสาวเป็น y

กฎที่ได้เมื่อมีการใช้การอธิบายด้วยตรรกะอันดับที่หนึ่งจะทำให้มีความยืดหยุ่นมากขึ้น เพราะอธิบายความสัมพันธ์ในรูปของตัวแปรที่ไม่ได้ระบุค่าเฉพาะลงไปเหมือนกับกฎที่เป็นลักษณะของตรรกศาสตร์ประพจน์ ซึ่งกฎที่ได้ในลักษณะนี้สามารถนำไปใช้ในการจำแนกความสัมพันธ์ของคนอื่นๆในอนาคตได้ดีกว่า

จะเห็นได้ว่าตรรกะอันดับที่หนึ่งที่มีการใช้ตัวแปรทำให้กฎที่ได้มีความยืดหยุ่นมากกว่าตรรกศาสตร์ประพจน์ที่ไม่มีการใช้ตัวแปร ทั้งยังสามารถอธิบายแนวคิดที่ซับซ้อนได้ดีกว่าการอธิบายด้วยตรรกศาสตร์ประพจน์ด้วย

หลักการสำคัญของการโปรแกรมตรรกะเชิงอุปนัย คือ การสร้างแนวคิด กฎ หรือ สมมติฐาน (h) ที่สามารถอธิบายตัวอย่าง (E) ได้ โดยประกอบกับความรู้ภูมิหลัง (B) ดังความสัมพันธ์

$$B \wedge h \vdash E$$

โดยที่นิพจน์ $X \vdash Y$ มีความหมายว่า Y สามารถอิมพลาย (imply) ได้โดยใช้ X ดังนั้นนิพจน์ข้างต้นจึงหมายความว่า ตัวอย่าง E สามารถอิมพลายได้โดยใช้ความรู้ภูมิหลัง B และสมมติฐาน h

จุดมุ่งหมายสำคัญของวิธีการโปรแกรมตรรกะเชิงอุปนัย คือ การสร้างสมมติฐานที่สามารถอิมพลายตัวอย่างได้ จากกลุ่มตัวอย่างและความรู้ภูมิหลังที่ได้ป้อนให้เพื่อใช้ในการเรียนรู้ ซึ่งมีทฤษฎีที่เกี่ยวข้องที่ใช้ในการสร้างสมมติฐานหลายวิธี ในที่นี้จะนำเสนอวิธีพื้นฐาน คือ วิธีการอนุมานเชิงอุปนัย (Inductive Inference) [1] และวิธีการไออาร์ (Inverse Resolution: IR) [1]

2.1.1.1 วิธีการเรียนรู้ของโปรแกรมตรรกะเชิงอุปนัย

วิธีการอนุมานเชิงอุปนัย

ในการสร้างสมมติฐานขึ้นจากตัวอย่างและความรู้ภูมิหลัง จะกำหนดให้ D เป็นเซตของตัวอย่างที่ใช้สอนเพื่อทำการเรียนรู้ซึ่งมีรูปแบบ $<x_i, f(x_i)>$ โดย x_i คือ ตัวอย่างที่ i และ $f(x_i)$ เป็นค่าเป้าหมายที่ต้องการ (target value) ของตัวอย่างที่ i และกำหนดให้ B คือ ความรู้ภูมิหลัง นิยามของการเรียนรู้ คือ การสร้างสมมติฐาน h ซึ่งค่า $f(x_i)$ ของตัวอย่าง x_i แต่ละตัวสามารถอิมพลายได้ด้วยสมมติฐาน h ลักษณะของตัวอย่าง x_i และความรู้ภูมิหลัง B หรือเขียนในรูป

$$(\forall(x_i, f(x_i)) \in D)\ B \wedge h \wedge x_i \vdash f(x_i) \qquad (1)$$

ตัวอย่างเช่น ในการสร้างแนวคิดของความสัมพันธ์ของคนสองคน (u, v) เมื่อ u มีลูกเป็น v โดยอาจเขียนแทนด้วยสัญพจน์ Child(u, v) และสมมติให้มีตัวอย่างบวกเพียงตัวอย่างเดียวคือ Child(Bob, Sharon) ซึ่งหมายถึง Sharon เป็นลูกของ Bob โดยอธิบายตัวอย่างด้วยสัญพจน์ต่อไปนี้ คือ Male(Bob), Female(Sharon) และ Father(Bob, Sharon) มีความรู้ภูมิหลังเป็น Parent(u, v) ← Father(u, v) จะสามารถอธิบายโดยใช้นิพจน์ (1) ได้ดังนี้

$f(x_i)$: Child(Bob, Sharon)

x_i : Male(Bob), Female(Sharon), Father(Sharon, Bob)

B : Parent(u, v) ← Father(u, v)

ในกรณีนี้สามารถสร้างสมมติฐานซึ่งครอบคลุมตัวอย่างได้หลายแบบ ตามคำนิยามในนิพจน์ (1) $(B \wedge h \wedge x_i) \vdash f(x_i)$ ตัวอย่างของสมมติฐานที่ได้ เช่น

h_1: Child(u, v) ← Father(v, u)

h_2: Child(u, v) ← Parent(v, u)

จากสมมติฐานที่ได้จะเห็นได้ว่าสัญพจน์เป้าหมาย Child(Bob, Sharon) ถูกครอบคลุมด้วยเงื่อนไข $h_1 \wedge x_i$ โดยไม่ต้องใช้ความรู้ภูมิหลัง B แต่สำหรับในกรณีของสมมติฐาน h_2 เมื่อนำ h_2 ไปรวมกับ x_i โดยไม่ใช้ความรู้ภูมิหลัง B จะพบว่าไม่ครอบคลุมสัญพจน์ Child(Bob, Sharon) ทำให้จำเป็นต้องใช้ความรู้ภูมิหลัง B ประกอบด้วยจึงจะทำให้ครอบคลุม ซึ่งเป็นไปตามเงื่อนไขของนิพจน์ $(B \wedge h \wedge x_i) \vdash f(x_i)$ ตัวอย่างนี้แสดงให้เห็นถึงบทบาทสำคัญของความรู้ภูมิหลังในการขยายเซตของสมมติฐานจากตัวอย่าง ดังเช่นสมมติฐาน h_2 ซึ่งต้องใช้ความรู้ภูมิหลังประกอบด้วยจึงจะสามารถครอบคลุมตัวอย่างได้

การสร้างสมมติฐานตามนิยาม $(\forall(x_i, f(x_i)) \in D)\ B \wedge h \wedge x_i \vdash f(x_i)$ มีลักษณะที่น่าสนใจหลายอย่างดังนี้

1. นิยามนี้สอดคล้องกับหลักการพื้นฐานของการเรียนรู้ คือ การสร้างสมมติฐานที่สามารถอธิบายตัวอย่างที่ใช้ในการเรียนรู้ได้
2. การใช้ความรู้ภูมิหลังมาช่วยในการสร้างสมมติฐานทำให้ได้สมมติฐานที่ครอบคลุมตัวอย่างมากขึ้น
3. ความรู้ภูมิหลังช่วยให้กระบวนการเรียนรู้ค้นหาสมมติฐานได้ง่ายขึ้น

วิธีการไออาร์

วิธีการไออาร์หรืออินเวอร์สรีโซลูชันเป็นวิธีที่ได้รับความนิยมในการนำมาพิจารณาตรรกะอันดับที่หนึ่งวิธีหนึ่ง โดยวิธีนี้สามารถสร้างสมมติฐานจากตัวอย่าง และความรู้ภูมิหลังที่รับเข้าไปได้ โดยจะแบ่งออกเป็นการนำไปใช้กับตรรกศาสตร์ประพจน์และตรรกะอันดับที่หนึ่ง

การนำไปใช้กับตรรกศาสตร์ประพจน์

การทำรีโซลูชันเป็นการหาอนุประโยคที่สามจากอนุประโยคที่กำหนดให้ 2 อนุประโยค ตัวอย่างเช่น ถ้ากำหนดให้ L เป็นประพจน์ P และ R เป็นอนุประโยคที่เกิดจากประพจน์หลายประพจน์รวมกัน จะได้กฎรีโซลูชันเป็นดังนี้

$$\frac{P \vee L \qquad \neg L \vee R}{P \vee R}$$

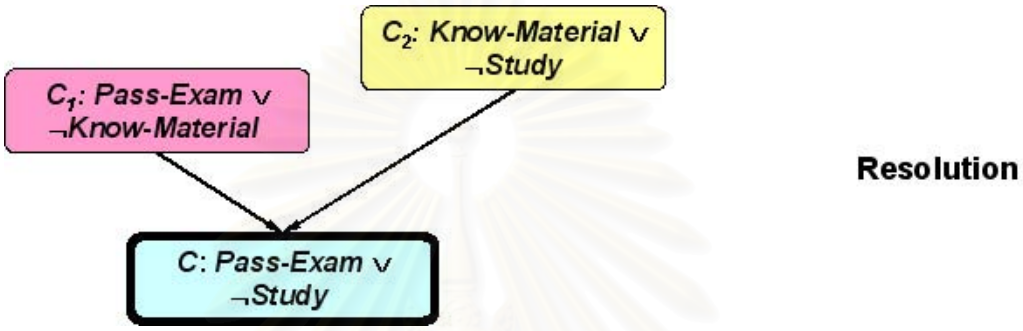
กฎรีโซลูชันข้างต้นแสดงให้เห็นว่า จากอนุประโยคที่กำหนดให้ 2 ประโยค คือ P V L และ ¬L V R สามารถที่จะนำมาสรุปอนุประโยคที่สาม P V R ได้ โดยสามารถเขียนเป็นหลักการได้ดังนี้

1. ให้อนุประโยค C_1 และ C_2เป็นอนุประโยคที่กำหนดให้ จากนั้นหาสัญพจน์ L จากอนุประโยค C_1 ซึ่งมีสัญพจน์ ¬L ปรากฏอยู่ในอนุประโยค C_2
2. ทำการสร้างอนุประโยค C ซึ่งเป็นผลจากการทำรีโซลูชัน โดยรวมสัญพจน์ทุกตัวจากอนุประโยค C_1และ C_2 ยกเว้นสัญพจน์ L และ ¬L ที่หาได้จากหลักการในข้อ 1 อนุประโยค C สามารถเขียนได้ดังดังสมการ

$$C = (C_1 - \{L\}) \cup (C_2 - \{\neg L\})$$

การใช้กฎรีโซลูชันในการสรุปสมมติฐานจากตัวอย่าง โดยการใช้อนุประโยคที่มีอยู่ C_1 และ C_2 ให้ทำการกำหนดสัญพจน์ L ซึ่งเป็นสัญพจน์แบบบวก (positive literal) ในอนุประโยคหนึ่ง และสัญพจน์แบบลบ (negative literal) ในอีกอนุประโยคหนึ่ง ก็จะสามารถสร้างอนุประโยค C ที่

เป็นเป้าหมายได้ดังรูปที่ 2.4 กำหนดให้อนุประโยค C_1 คือ Pass-Exam V ¬Know-Material และอนุประโยค C_2 คือ Know-Material V ¬Study ในขั้นแรกต้องหาสัญพจน์ L ได้เป็น L= ¬Know-Material ปรากฏอยู่ในอนุประโยค C_1 และสัญพจน์ที่ตรงกันข้ามกับ L นั่นคือ ¬ (¬Know-Material) หรือมีค่าเป็น Know-Material ซึ่งปรากฏอยู่ในอนุประโยค C_2 ดังนั้นในขั้นที่ 2 จะได้อนุประโยค C ซึ่งเป็นผลสรุปของสองอนุประโยคนี้ คือ การรวมกันของสัญพจน์ C_1 - {L} = Pass-Exam และ C_2 – {¬L} = ¬Study ทำให้ได้ C เป็น Pass-Exam V ¬Study



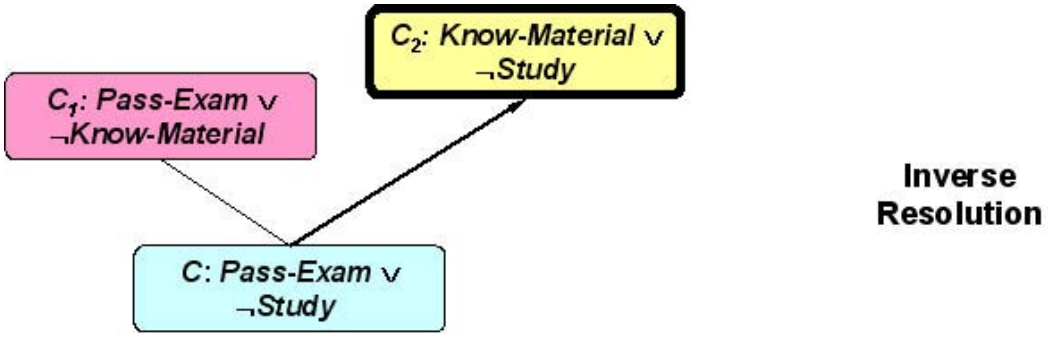
รูปที่ 2.4 การใช้กฎรีโซลูชันเพื่อหาข้อสรุปจากอนุประโยค 2 อนุประโยค

หลังจากที่ได้ทำการอธิบายการใช้กฎรีโซลูชันในการหาข้อสรุป ต่อไปจะทำการอธิบายวิธีไออาร์ซึ่งสามารถทำการอนุมานโดยอุปนัยได้ โดยกำหนดอนุประโยคเริ่มต้น C_1 และข้อสรุป C มาให้เพื่อใช้ในการสร้างอนุประโยคเริ่มต้น C_2 หรือนั่นคือ การพยายามสร้างอนุประโยค C_2 ซึ่งตรงตามนิพจน์ $C_1 \wedge C_2 \vdash C$ นั่นเอง หลักการของวิธีไออาร์จะเป็นการมองย้อนกลับของกฎรีโซลูชัน นั่นคือ สัญพจน์ใดๆที่ปรากฏอยู่ในข้อสรุป C แต่ไม่ปรากฏในอนุประโยค C_1 แสดงว่าสัญพจน์นั้นๆ ต้องปรากฏอยู่ในอนุประโยค C_2 ส่วนสัญพจน์ที่ปรากฏในอนุประโยค C_1 แต่ไม่ปรากฏใน C แสดงว่าสัญพจน์นั้นต้องถูกตัดทิ้ง (L) และต้องมีนิเสธของสัญพจน์นี้ (¬L) ปรากฏอยู่ใน C_2 ด้วย สามารถสรุปเป็นขั้นตอนในการทำวิธีไออาร์ได้ดังนี้

1. กำหนดอนุประโยค C_1และ C มาให้ ให้ทำการหาสัญพจน์ L ที่ปรากฏอยู่ในอนุประโยค C_1 แต่ไม่ปรากฏอยู่ในอนุประโยค C
2. ทำการสร้างอนุประโยค C_2 ตามสมการ

$$C_2 = (C - (C_1 - \{L\})) \cup \{\neg L\}$$

จากตัวอย่างในรูปที่ 2.5 แสดงวิธีการไออาร์ เพื่อหาสมมติฐาน C_2 จากอนุประโยค C_1 และ C ที่กำหนดมาให้ โดยที่สามารถหาสัญพจน์ L ใน C_1 ได้เป็น ¬Know-Material และท้ายที่สุดจะได้สมมติฐาน C_2 เป็น Know-Material V ¬Study



รูปที่ 2.5 การใช้วิธีไออาร์ในการสร้างสมมติฐานที่เป็นอนุประโยคเริ่มต้น

การนำไปใช้กับตรรกะอันดับที่หนึ่ง

การนำกฎรีโซลูชันมาใช้กับตรรกะอันดับที่หนึ่งจะมีความแตกต่างกับการนำไปใช้กับตรรกศาสตร์ประพจน์ เนื่องจากสัญพจน์ในตรรกะอันดับที่หนึ่งจะมีการใช้ตัวแปรในการอธิบายด้วยเพื่อทำให้สมมติฐานที่ได้มีความยืดหยุ่นมากขึ้น ดังนั้นจะต้องมีการใช้การแทนที่ตัวแปรเข้ามาร่วมด้วย โดยกำหนดให้ θ = {x/Bob, y/z} หมายถึง การแทนค่าของตัวแปร x ด้วยค่าคงที่ Bob และ แทนตัวแปร y ด้วยตัวแปร z โดยจะใช้ Wθ เพื่อแสดงการแทนค่าตาม θ ลงในนิพจน์ W ยกตัวอย่างเช่น ถ้าให้ L แทนสัญพจน์ Father(x, Bill) และใช้ θ ตามความหมายข้างบน จะได้ Lθ = Father(Bob,Bill) เนื่องจากมีการแทนค่าตัวแปร x ในสัญพจน์ Father(x,Bill) ด้วยค่าคงที่ Bob ตาม θ

ตัวแทน θ จะเป็นตัวแทนรวม (unifying substitution) ของสัญพจน์ L_1 และ L_2 เมื่อ $L_1\theta = L_2\theta$ ตัวอย่างเช่น ถ้า L_1 = Father(x, y), L_2 = Father(Bill, z) และให้ θ = {x/Bill, z/y} จะเห็นว่า θ นี้เป็นตัวแทนรวมของ L_1 และ L_2 เนื่องจาก $L_1\theta = L_2\theta$ = Father(Bill, y) ซึ่งหลักการของตัวแทนรวมนี้จะนำไปใช้แทน L ในตรรกศาสตร์ประพจน์ สามารถสรุปเป็นหลักการในการหาข้อสรุป C โดยการใช้กฎรีโซลูชันอันดับที่หนึ่งได้ดังนี้

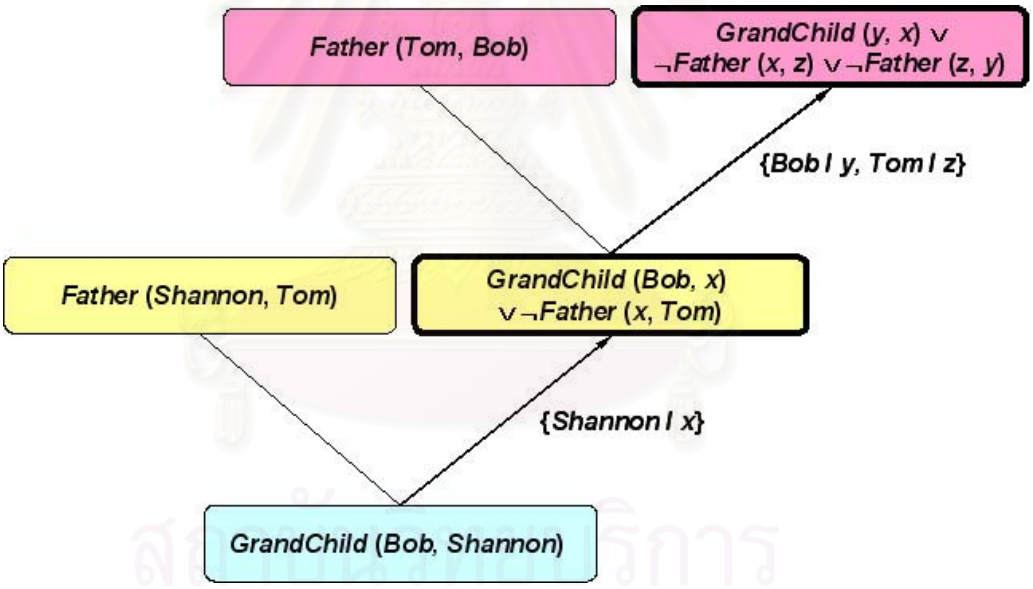
1. จากอนุประโยคที่กำหนดให้ C_1 และ C_2 ทำการหาสัญพจน์ L_1 จาก C_1 และสัญพจน์ L_2 จาก C_2 พร้อมทั้ง θ ที่ทำให้ $L_1\theta = L_2\theta$
2. ทำการสร้างอนุประโยคสรุป C โดยรวมสัญพจน์ทุกตัวจาก $C_1\theta$ และ $C_2\theta$ ยกเว้น $L_1\theta$ และ $\neg L_2\theta$ ดังสมการ

$$C = (C_1 - \{L_1\})\theta \cup (C_2 - \{L_2\})\theta$$

จากนั้นจะนำหลักการนี้มาทำวิธีไออาร์ เมื่อกำหนดให้อนุประโยค C_2 ไม่มีสัญพจน์ที่เหมือนกับสัญพจน์ในอนุประโยค C_1 และใช้นิยามของกฎการรีโซลูชัน $L_2 = \neg L_1\theta_1\theta_2^{-1}$ จะสามารถสรุปอนุประโยค C_2 ได้ดังสมการ

$$C_2 = (C - (C_1 - \{L_1\})\theta_1)\theta_2^{-1} \cup \{\neg L_1\theta_1\theta_2^{-1}\}$$

จากหลักการข้างต้น นำมาใช้ในตัวอย่างดังรูปที่ 2.6 ซึ่งต้องการเรียนรู้กฎสำหรับสัญพจน์เป้าหมาย GrandChild(y,x) โดยให้ข้อมูลที่ใช้ในการเรียนรู้ D = GrandChild(Bob, Shannon) ความรู้ภูมิหลัง B = { Father(Shannon, Tom), Father(Tom, Bob) } จากรูปเริ่มต้นที่ส่วนล่างที่สุด ซึ่งให้ข้อสรุป C เป็นตัวอย่างที่ใช้ในการเรียนรู้ GrandChild(Bob, Shannon) และเลือกอนุประโยค C_1 = Father(Shannon, Tom) จากความรู้ภูมิหลัง และได้ L_1 คือ Father(Shannon, Tom) เลือกตัวแทนย้อนกลับ θ_1^{-1} = { } และ θ_2^{-1} = {Shannon/x} ทำให้สามารถสร้างอนุประโยค C_2 จากการยูเนียนกันของ 2 อนุประโยค นั่นคือ อนุประโยค $(C - (C_1 - \{L_1\})\theta_1)\theta_2^{-1} = (C\theta_1)\theta_2^{-1}$ = GrandChild(Bob/x) และอนุประโยค $\neg L_1\theta_1\theta_2^{-1}$ = ¬Father(x,Tom) ดังนั้นจึงได้ผลลัพธ์ในขั้นต้นเป็น GrandChild(Bob/x) V ¬Father(x,Tom) หรืออาจเขียนได้ในรูป GrandChild(Bob/x) ← Father(x,Tom) และด้วยวิธีการนี้ก็จะสามารถสร้างอนุประโยคสุดท้าย GrandChild (y, x) V ¬Father (x, z) V ¬Father (z, y) ได้



รูปที่ 2.6 ตัวอย่างการใช้วิธีไออาร์ในการสร้างกฎในรูปแบบตรรกะอันดับที่หนึ่ง

2.1.1.2 เครื่องมือของระบบโปรแกรมตรรกะเชิงอุปนัย

การโปรแกรมตรรกะเชิงอุปนัยมีเครื่องมือที่สามารถนำมาใช้ในการเรียนรู้ได้มากมาย โดยประยุกต์หลักการของการเรียนรู้ดังที่อธิบายไปในหัวข้อข้างต้น เนื้อหาในส่วนนี้จะนำเสนอเครื่องมือต่างๆที่เป็นที่รู้จักของระบบโปรแกรมตรรกะเชิงอุปนัย ได้แก่ PROGOL, GOLEM, TILDE และ LINUS ซึ่งแต่ละระบบก็มีเทคนิคในการเรียนรู้และจุดเด่นที่แตกต่างกันออกไป รายละเอียดของแต่ละระบบมีดังนี้

PROGOL

ระบบ PROGOL [2] เป็นระบบโปรแกรมตรรกะเชิงอุปนัยที่มีประสิทธิภาพ และนิยมใช้กันอย่างแพร่หลาย ระบบ PROGOL รับอินพุตเป็นเซตของตัวอย่างบวก ตัวอย่างลบ และกลุ่มความรู้ภูมิหลัง การใช้งานระบบ PROGOL ผู้ใช้จะต้องประกาศลักษณะการทำงานของสัญพจน์ (mode declaration) เพื่อแสดงให้ระบบรู้ว่าแต่ละอาร์กิวเมต์ในแต่ละสัญพจน์มีคุณสมบัติอย่างไร โดยอาร์กิวเมนต์ของสัญพจน์สามารถเป็นได้ทั้งตัวแปรอินพุต ตัวแปรเอาต์พุต และค่าคงที่ นอกจากหน้าที่ของอาร์กิวเมนต์ต่างๆแล้ว การประกาศลักษณะการทำงานของสัญพจน์ยังสามารถระบุได้ถึงจำนวนครั้งที่ต้องการให้สัญพจน์นี้ปรากฏในกฎที่สร้างได้จากระบบอีกด้วย ในการสร้างกฎของระบบ PROGOL จะเริ่มตั้งแต่สุ่มตัวอย่าง และสร้างอนุประโยคที่เจาะจงที่สุด (most-specific) สำหรับตัวอย่างนั้น แล้วทำการค้นหาด้วยวิธีการ A* (A*-like search) โดยมีจุดมุ่งหมายเพื่อให้ได้การบีบอัด (compression) สูงสุดในการค้นหานั้น ทำให้ระบบ PROGOL ใช้เวลาและเนื้อที่หน่วยความจำมากในการสร้างกฎเมื่อเทียบกับระบบอื่นๆ

เนื่องจากระบบ PROGOL เป็นระบบที่ทำงานกับปัญหาที่มีลักษณะเป็นสองประเภท คือ ประเภทของตัวอย่างบวก และประเภทของตัวอย่างลบ ดังนั้นเมื่อต้องการนำระบบ PROGOL ไปใช้กับปัญหาที่มีลักษณะเป็นหลายประเภท จึงต้องทำการสร้างกฎไปทีละประเภท โดยกำหนดให้ตัวอย่างบวกในการเรียนรู้เป็นตัวอย่างในประเภทที่ต้องการสร้างกฎ และตัวอย่างลบเป็นตัวอย่างของประเภทอื่นๆ เมื่อทำซ้ำจนครบทุกประเภท ก็จะได้กฎสำหรับทุกๆประเภท

GOLEM

ระบบ GOLEM [3] เป็นระบบที่รับอินพุตเป็นตัวอย่างบวก ตัวอย่างลบ และกลุ่มความรู้ภูมิหลังในการเรียนรู้ นอกจากนี้ยังเป็นระบบที่ทำงานกับปัญหาที่มีลักษณะเป็นสองประเภทเช่นเดียวกับระบบ PROGOL แต่ว่าจะแตกต่างกันในกระบวนการเรียนรู้ โดยระบบ GOLEM จะเริ่มต้นสร้างอนุประโยคด้วยการสุ่มเลือกตัวอย่างขึ้นมาเป็นคู่ แล้วหาอาร์แอลจีจีของตัวอย่างคู่นั้น เลือกผลการทำอาร์แอลจีจีที่ครอบคลุมตัวอย่างบวกมากที่สุด จากนั้นจึงนำอนุประโยคที่ได้มาทำอาร์แอลจีจีกับตัวอย่างบวกตัวใหม่ซึ่งสุ่มเลือกมาจากเซตของตัวอย่างบวกจนกระทั่งไม่สามารถหาอนุประโยคที่ครอบคลุมตัวอย่างได้มากขึ้น

ในการใช้งานระบบ GOLEM ผู้ใช้สามารถระบุจำนวนคู่ของตัวอย่างที่จะสุ่มมาทำอาร์แอลจีจีเพื่อหาอนุประโยคที่ครอบคลุมตัวอย่างมากที่สุดได้ ซึ่งถ้ากำหนดให้มีค่ามาก โอกาสที่ระบบจะพบอนุประโยคที่ครอบคลุมตัวอย่างได้มากก็จะมีค่าสูงด้วย แต่ว่าก็จะทำให้ใช้เวลาในการเรียนรู้สูงขึ้นด้วยเช่นกัน โดยระบบได้กำหนดค่าของจำนวนคู่เริ่มต้นไว้ที่ 8 คู่ นอกจากนี้การนำ

ระบบ GOLEM ไปใช้กับปัญหาที่มีลักษณะเป็นหลายประเภท ต้องใช้วิธีการสร้างกฎให้กับแต่ละประเภทเช่นเดียวกับระบบ PROGOL

TILDE

ระบบ TILDE [8] เป็นระบบที่มีการเรียนรู้ต่างจากระบบ PROGOL และ GOLEM โดยการเรียนรู้ของระบบ TILDE จะสร้างต้นไม้ตัดสินใจ (decision tree) ขึ้นมาเพื่อใช้ในการจำแนกตัวอย่าง การสร้างต้นไม้ตัดสินใจจะเริ่มจากสร้างโนด (node) ทดสอบขึ้นมาและเปรียบเทียบค่าฮิวริสติก (heuristic) ของแต่ละโนด ระบบจะเลือกโนดทดสอบที่ให้ค่าฮิวริสติกดีที่สุดมาสร้างเป็นโนดจริงในต้นไม้ จากนั้นจะนำตัวอย่างทั้งหมดมาทำการทดสอบกับโนดนี้เพื่อแบ่งตัวอย่างออกเป็นสองพวก คือ พวกที่ให้ค่าความจริงเป็นจริงกับพวกที่ให้ค่าความจริงเป็นเท็จ แล้วระบบจะเริ่มสร้างโนดทดสอบขึ้นมาใหม่สำหรับทดสอบกับตัวอย่างทั้งสองพวกแล้วทำการทดสอบไปเรื่อยๆ จนกระทั่งมีตัวอย่างเพียงกลุ่มเดียวที่ให้ค่าความจริงกับโนดนั้น

ข้อดีของระบบ TILDE ที่แตกต่างจากระบบ PROGOL และ GOLEM คือ ระบบ TILDE สามารถนำไปใช้กับปัญหาที่มีลักษณะเป็นหลายประเภทได้โดยตรง เนื่องจากการสร้างต้นไม้ตัดสินใจจะได้โนดใบ (leaf node) แต่ละโนดแทนตัวอย่างในแต่ละประเภท ดังนั้นเมื่อจำแนกแล้วตัวอย่างไปตกอยู่ที่โนดใดก็จะถูกจำแนกเป็นประเภทนั้น

LINUS

ระบบ LINUS [9] เป็นระบบที่ทำการเรียนรู้โดยเปลี่ยนรูปแบบของปัญหาจากตรรกะอันดับที่หนึ่งไปเป็นการแก้ปัญหาในรูปแบบของค่าคุณสมบัติ (attribute value) การทำงานของระบบจะเริ่มจากการสร้างลักษณะสำคัญจากตัวอย่างและความรู้ภูมิหลังซึ่งอยู่ในรูปตรรกะอันดับที่หนึ่ง แล้วใช้ลักษณะสำคัญเหล่านั้นเป็นค่าคุณสมบัติของตัวอย่างแต่ละตัว จากนั้นจึงใช้ระบบที่สามารถเรียนรู้ได้จากค่าคุณสมบัติได้ เช่น CN2 และ C4.5 มาทำการรู้จำตัวอย่างจากคุณสมบัติที่ได้จากระบบ LINUS อีกทีหนึ่ง

2.1.2 นิวรอลเน็ตเวิร์ก

นิวรอลเน็ตเวิร์ก หรือที่เรียกว่า นิวรอลเน็ตเวิร์กเทียม (Artificial Neural Network) [1, 10] เป็นการเรียนรู้ของเครื่องรูปแบบหนึ่ง ซึ่งเลียนแบบมาจากจากลักษณะทั่วไปทางชีววิทยาของระบบสมองมนุษย์ และพัฒนาขึ้นโดยมีสมมติฐานว่า

1. การประมวลผลข้อมูลต่างๆเกิดขึ้นที่ส่วนประกอบเล็กๆจำนวนมาก ที่เรียกว่านิวรอน (neuron)

2. การทำงานของนิวรอนจะใช้การส่งสัญญาณผ่านส่วนที่เชื่อมต่อระหว่างนิวรอนที่เรียกว่า คอนเนกชันลิงค์ (connection link)
3. เส้นเชื่อมแต่ละเส้นจะมีค่าน้ำหนักที่ต่างกัน เพื่อแสดงว่านิวรอนนั้นได้รับอิทธิพลลักษณะใดจากนิวรอนอื่น
4. แต่ละนิวรอนจะได้รับข้อมูลจากนิวรอนอื่น โดยผ่านฟังก์ชันกระตุ้น (activation function)

นิวรอลเน็ตเวิร์กประกอบไปด้วยกลุ่มของหน่วยประมวลผล หรือ นิวรอน ในบางครั้งก็อาจเรียกว่าหน่วย (unit) หรือ โนดก็ได้ การทำงานของนิวรอลเน็ตเวิร์กจะแบ่งเป็นชั้นๆ (layer) แต่ละชั้นก็จะประกอบไปด้วยนิวรอนจำนวนหนึ่ง โดยที่แต่ละนิวรอนจะเชื่อมต่อกันด้วยค่าน้ำหนักค่าหนึ่ง ในการทำงานแต่ละนิวรอนจะทำการประมวลผลในชั้นตัวเอง เมื่อเสร็จแล้วก็จะทำการส่งผลลัพธ์ต่อเพื่อนำไปคำนวณในชั้นถัดไป จนกระทั่งถึงชั้นสุดท้ายเมื่อคำนวณเสร็จก็จะได้เป็นผลลัพธ์ของเน็ตเวิร์ก โดยหลักๆแล้วเน็ตเวิร์กจะถูกแบ่งชั้นการทำงานออกเป็น 3 ส่วนด้วยกัน คือชั้นนำข้อมูลเข้า (input layer) เป็นชั้นแรกของเน็ตเวิร์ก ถัดไปจะเป็นชั้นซ่อน (hidden layer) และชั้นสุดท้ายเป็นชั้นผลลัพธ์ (output layer) โดยชั้นซ่อนนั้นอาจมีมากกว่า 1 ชั้นก็ได้

วิธีการเรียนรู้ของนิวรอลเน็ตเวิร์กมีข้อดีและข้อด้อยดังนี้

ข้อดี

1. มีความแม่นยำในการจำแนกตัวอย่างสูง
2. ทนทานต่อชุดข้อมูลที่มีสัญญาณรบกวนได้ดี
3. ใช้ได้ดีกับการจำแนกตัวอย่างแบบหลายประเภท
4. อินพุตที่ใช้ในการเรียนรู้เป็นได้ทั้งค่าที่ต่อเนื่อง (continuous value) และไม่ต่อเนื่อง (discrete value)

ข้อด้อย

1. ผลลัพธ์ที่ได้จากการเรียนรู้จะอยู่ในรูปของเวกเตอร์ของค่าน้ำหนักเส้นเชื่อมซึ่งเข้าใจหรือแปลความหมายได้ค่อนข้างยาก
2. ใช้เวลาในการสอนเน็ตเวิร์กค่อนข้างนานเมื่อเทียบกับการเรียนรู้ของเครื่องวิธีอื่นๆ

นิวรอลเน็ตเวิร์กสามารถแบ่งประเภทออกตามจำนวนชั้นของการทำงานได้เป็น 2 แบบ คือ แบบข่ายงานชั้นเดียว (single layer) และ ข่ายงานหลายชั้น (multi-layer network) แต่ว่าการนับ

จำนวนชั้นนั้นอาจมีการนับไม่เหมือนกัน ในบางที่ก็นับว่าเน็ตเวิร์กที่มีชั้นนำเข้าและชั้นผลลัพธ์เท่านั้นเป็นข่ายงาน 2 ชั้น แต่ในบางที่ก็จะไม่นับชั้นนำเข้าเนื่องจากในชั้นนี้ไม่ได้มีการประมวลผล ดังนั้นข่ายงาน 2 ชั้นจึงเป็นเน็ตเวิร์กที่มีชั้นนำเข้า ชั้นซ่อน 1 ชั้น และชั้นผลลัพธ์ ในที่นี้เพื่อให้เข้าใจตรงกันจะขอทำการนับจำนวนชั้นของข่ายงานแบบไม่นับชั้นนำข้อมูลเข้าหรือทำการนับตามจำนวนชั้นของเส้นเชื่อมก็ได้

2.1.2.1 ข่ายงานชั้นเดียว

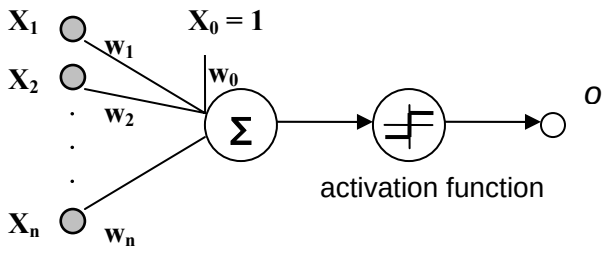
นิวรอลเน็ตเวิร์กที่มีลักษณะเป็นข่ายงานชั้นเดียวนั้น จะมีการรับสัญญาณเข้าและทำการประมวลผล รวมทั้งหาผลลัพธ์ที่ชั้นนั้นเลย ตัวอย่างของข่ายงานชั้นเดียว เช่น เพอร์เซปตรอนอย่างง่าย (Simple Perceptron) ฮอปฟิลด์เน็ตเวิร์ก (Hopfield Network) เป็นต้น เพอร์เซปตรอนจะมีการเชื่อมต่อกันระหว่างชั้นนำเข้าข้อมูลและชั้นผลลัพธ์ ไม่มีการเชื่อมต่อกันระหว่างนิวรอนในชั้นเดียวกัน แต่ฮอปฟิลด์เน็ตเวิร์ก ทุกๆนิวรอนในชั้นเดียวกันจะเชื่อมต่อกันด้วย

การทำงานของเพอร์เซปตรอนจะเริ่มจากการรับค่าเวกเตอร์ซึ่งประกอบด้วยอินพุตจำนวนจริง แล้วคำนวณหาผลรวมเชิงเส้น (linear combination) ของอินพุต จากนั้นนำผลรวมที่ได้มาเปรียบเทียบกับค่าขีดแบ่ง (threshold) ที่กำหนดไว้ โดยถ้าผลรวมมีค่ามากกว่าค่าขีดแบ่งก็จะให้เอาต์พุตเป็น 1 แต่ถ้าได้ผลรวมไม่ถึงค่าที่กำหนดก็จะให้เอาต์พุตเป็น -1 ดังสมการ (2) ข่ายงานเพอร์เซปตรอนแสดงดังรูปที่ 2.7

$$o(x_1,...,x_n) = \begin{cases} 1 & if\ w_0 + w_1x_1 + ... + w_nx_n > 0 \\ -1 & if\ w_0 + w_1x_1 + ... + w_nx_n < 0 \end{cases} \quad (2)$$

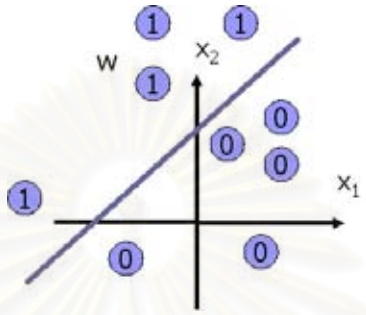
เมื่อ

- $-w_0$ เป็นค่าขีดแบ่ง (threshold)
- w_i เป็นค่าน้ำหนักของแต่ละอินพุต มีค่าเป็นจำนวนจริง
- x_i เป็นค่าอินพุต
- o เป็นค่าเอาต์พุตของเพอร์เซปตรอน มีค่าเป็น 1 หรือ -1



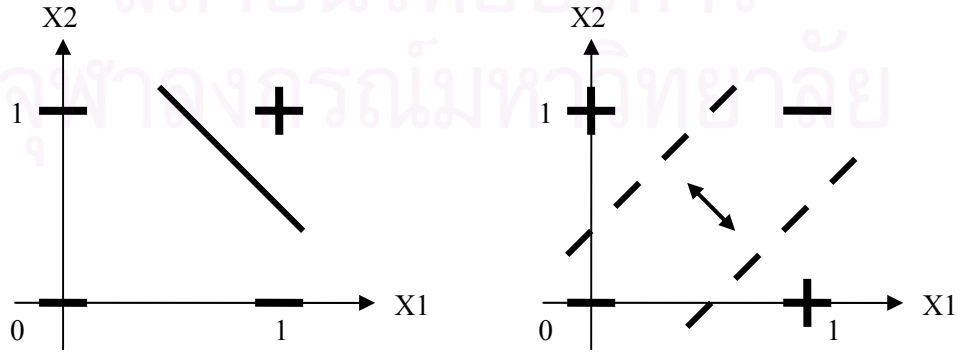
รูปที่ 2.7 ข่ายงานเพอร์เซปตรอน

เราอาจมองได้ว่าระนาบตัดสินใจ (decision surface) ของเพอร์เซปตรอน คือ ระนาบใน n มิติ โดยเพอร์เซปตรอนจะให้เอาต์พุตสำหรับตัวอย่างบวกอยู่ด้านหนึ่งของเส้นแบ่ง และให้เอาต์พุตสำหรับตัวอย่างลบอยู่อีกด้านหนึ่งของเส้นแบ่งดังรูปที่ 2.8 ในกรณีที่สามารถใช้ระนาบในการแบ่งเซตของตัวอย่างออกเป็นตัวอย่างบวกและตัวอย่างลบได้ จะเรียกเซตของตัวอย่างที่มีลักษณะนี้ว่าเซตของตัวอย่างที่สามารถแยกได้แบบเชิงเส้น (linearly separable examples)



รูปที่ 2.8 ระนาบตัดสินใจของเพอร์เซปตรอนในการแบ่งเซตของตัวอย่าง

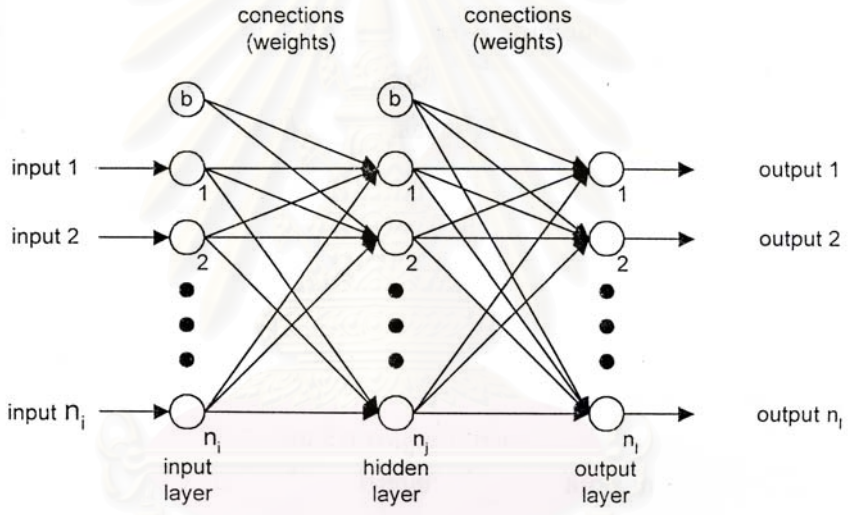
โดยปกติแล้วเดี่ยวสามารถใช้แสดงฟังก์ชันบูลีน (Boolean Function) ได้หลายตัว เช่น AND, OR, NOR, NAND เป็นต้น โดยกำหนดให้ค่าของบูลีนที่เป็นจริงแทนด้วย 1 และแทนด้วย -1 เมื่อให้ค่าเป็นเท็จ แต่อย่างไรก็ตามก็มีบางฟังก์ชันบูลีนที่ไม่สามารถแทนด้วยเพอร์เซปตรอนเดี่ยวได้ เช่น XOR เนื่องจากเส้นแบ่งของเพอร์เซปตรอนเดี่ยวไม่สามารถที่จะแบ่งเซตของตัวอย่างได้ดังแสดงในรูปที่ 2.9 โดยในขั้นตอนการเรียนรู้ค่าน้ำหนักของเส้นเชื่อมจะแกว่งไปมา ไม่ลู่เข้าสู่ค่าใดค่าหนึ่งและไม่อาจหาเส้นแบ่งที่แบ่งตัวอย่างได้อย่างถูกต้อง ที่เป็นเช่นนี้เพราะว่าในกรณีของ XOR นั้นเราไม่สามารถที่จะแบ่งตัวอย่างด้วยเส้นตรงเส้นเดียวได้อยู่แล้ว ทำให้เพอร์เซปตรอนไม่สามารถเรียนรู้ได้ จึงจำเป็นต้องใช้เพอร์เซปตรอนมาต่อกันให้มีลักษณะเป็นหลายชั้น เพื่อให้สามารถแยกตัวอย่างเหล่านี้ได้ เพอร์เซปตรอนที่นำมาต่อกันจะเรียกว่านิวรอลเน็ตเวิร์กแบบหลายชั้น ซึ่งเน็ตเวิร์กชนิดนี้มีความสามารถในการแบ่งตัวอย่างที่ซับซ้อนได้มากขึ้น



รูปที่ 2.9 บูลีนฟังก์ชัน AND และบูลีนฟังก์ชัน XOR เมื่อแบ่งด้วยเพอร์เซปตรอน

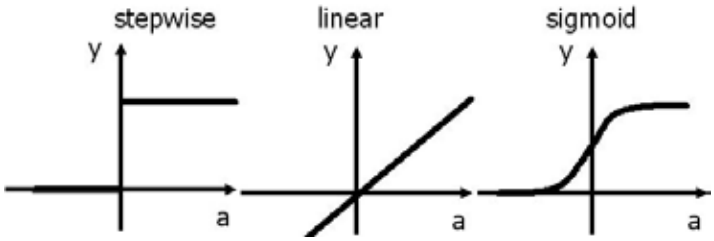
2.1.2.2 ข่ายงานหลายชั้น

เนื่องจากข้อจำกัดของเน็ตเวิร์กแบบข่ายงานชั้นเดียวที่ไม่สามารถนำมาใช้ในการแบ่งตัวอย่างที่มีลักษณะเป็นแบบไม่สามารถแบ่งได้ด้วยเชิงเส้น (linearly non-separable examples) ได้ ดังเช่นปัญหาการสร้างเส้นแบ่งในฟังก์ชันบูลีน XOR เป็นต้น ทำให้มีการเสนอวิธีการใหม่ที่แก้ปัญหานี้ได้ คือ เปลี่ยนแปลงโครงสร้างจากข่ายงานชั้นเดียวเป็นข่ายงานหลายชั้นดังแสดงในรูปที่ 2.10 ซึ่งเน็ตเวิร์กแบบหลายชั้นนี้จะสร้างเส้นแบ่งได้หลายเส้น และทำให้สามารถแก้ปัญหาซับซ้อนต่างๆที่ไม่สามารถแก้ได้ด้วยเน็ตเวิร์กที่มีข่ายงานชั้นเดียวได้ เช่น งานในการจำแนกใบหน้า หรือ ระบบขับรถยนต์อัตโนมัติ เป็นต้น และทำให้นิวรอลเน็ตเวิร์กได้รับความสนใจมากยิ่งขึ้น ตัวอย่างของข่ายงานหลายชั้น เช่น แบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กแบบหลายชั้น (multi-layer Backpropagation Neural Network) บอลทซ์แมนแมชชีน (Boltzman Machine) เป็นต้น



รูปที่ 2.10 นิวรอลเน็ตเวิร์กที่มีข่ายงานหลายชั้น

แบ็กพรอพาเกชันนิวรอลเน็ตเวิร์ก เป็นขั้นตอนวิธีแบบหนึ่งของนิวรอลเน็ตเวิร์กที่ได้รับความนิยมมาก มีความสามารถในการสร้างระนาบตัดสินใจแบบไม่เป็นเชิงเส้น (nonlinear decision surface) ซึ่งสามารถแบ่งแยกตัวอย่างได้ดีกว่าระนาบตัดสินใจแบบเชิงเส้น (linear decision surface) ความสามารถนี้เนื่องมาจากแบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กมีฟังก์ชันกระตุ้นที่แตกต่างจากเพอร์เซปตรอน องค์ประกอบนี้สามารถที่จะทำงานกับฟังก์ชันแบบไม่เชิงเส้น (nonlinear function) โดยใช้องค์ประกอบซิกมอยด์ (sigmoid) ดังรูปที่ 2.11



รูปที่ 2.11 องค์ประกอบแบบต่างๆ

องค์ประกอบซิกมอยด์จะทำงานโดยหาผลคูณเวกเตอร์ของอินพุตเวกเตอร์กับค่าน้ำหนักของเส้นเชื่อมระหว่างอินพุตนิวรอนกับนิวรอนในชั้นถัดไป จากนั้นนำผลที่ได้ผ่านค่าขีดแบ่งเอาต์พุตขององค์ประกอบซิกมอยด์จะเป็นฟังก์ชันต่อเนื่อง (continuous function) ที่ให้ค่าระหว่าง 0 กับ 1 ในขณะที่เอาต์พุตของเพอร์เซปตรอนนั้นจะให้เอาต์พุตเป็นค่าไม่ต่อเนื่องเช่นดังสมการ (2) ให้เอาต์พุตเป็น 1 กับ -1 เท่านั้น นอกจากนี้องค์ประกอบซิกมอยด์ยังมีลักษณะที่สามารถหาอนุพันธ์ได้ ซึ่งเป็นลักษณะที่ต้องการของการทำแบ็กพรอพาเกชันด้วย (กรณีขององค์ประกอบแบบขั้นบันได (stepwise) ซึ่งใช้ในสมการ (2) นั้นไม่สามารถหาอนุพันธ์ได้) นิยามของฟังก์ชันซิกมอยด์เป็นดังสมการ

$$\sigma(y) = \frac{1}{1+e^{-y}}$$

ขั้นตอนวิธีแบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กจะทำการเรียนรู้เพื่อปรับค่าน้ำหนักสำหรับนิวรอลเน็ตเวิร์ก แบบหลายชั้นจากชุดข้อมูลที่นำเข้าเพื่อใช้ในการเรียนรู้ การปรับค่าจะอาศัยหลักการของเกรเดียนต์เดสเซนต์ (gradient descent) เพื่อลดค่ากำลังสองของค่าความผิดพลาด (squared error) ระหว่างเอาต์พุตที่ได้จากเน็ตเวิร์กและค่าเป้าหมายของอินพุตนั้นๆ โดยถ้าค่าความผิดพลาดยังไม่ถึงจุดต่ำสุดก็จะทำการปรับปรุงค่าน้ำหนักใหม่แล้วผ่านการคำนวณหาผลลัพธ์ใหม่ในรอบการสอนถัดไป จนทำให้มีค่าความผิดพลาดน้อยที่สุดหรืออยู่ในเกณฑ์ที่ยอมรับได้ โดยขั้นตอนวิธีสำหรับการปรับค่าน้ำหนักในแบบแบ็กพรอพาเกชันเป็นดังนี้

กำหนดให้ตัวอย่างที่ใช้ในการเรียนรู้อยู่ในรูป $(\vec{x},\vec{t})$ เมื่อ $\vec{x}$ เป็นอินพุตเวกเตอร์ของนิวรอลเน็ตเวิร์ก และ $\vec{t}$ เป็นเวกเตอร์เป้าหมายของนิวรอลเน็ตเวิร์ก ให้ η เป็นค่าอัตราการเรียนรู้ (learning rate) โดยที่อินพุตขององค์ประกอบ j ซึ่งมาจากเอาต์พุตขององค์ประกอบ i แทนด้วย x_{ji} และค่าน้ำหนักที่เชื่อมจากองค์ประกอบ i ไปยังองค์ประกอบ j แทนด้วย w_{ji}

1. สร้างนิวรอลเน็ตเวิร์กตามโครงสร้างที่ต้องการ กำหนดจำนวนชั้นและจำนวนนิวรอนของแต่ละชั้น
2. กำหนดค่าน้ำหนักของเส้นเชื่อมเริ่มต้นแบบสุ่ม

3. ปรับค่าน้ำหนักด้วยวิธีดังนี้

สำหรับแต่ละอินพุตเวกเตอร์ $(\vec{x},\vec{t})$ ในเซตตัวอย่างที่ใช้เรียนรู้

- ใช้ตัวอย่าง x เป็นอินพุต คำนวณหาค่าผลลัพธ์จากทุกๆนิวรอนในแต่ละชั้น แล้วส่งต่อไปคำนวณในชั้นต่อไป จนได้ผลลัพธ์ o ของทุกๆนิวรอนในชั้นผลลัพธ์
- คำนวณค่าความคลาดเคลื่อน δ_k ของทุกนิวรอน k ในชั้นเอาต์พุตจาก

$$\delta_k \leftarrow o_k(1-o_k)(t_k-o_k)$$

- คำนวณค่าความคลาดเคลื่อน δ_h ของทุกนิวรอน h ในชั้นซ่อนจาก

$$\delta_h \leftarrow o_h(1-o_h)\sum_{k\in outputs} w_{kh}\delta_k$$

- ปรับค่าน้ำหนักของเส้นเชื่อม w_{ji} ในแต่ละชั้นโดย

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

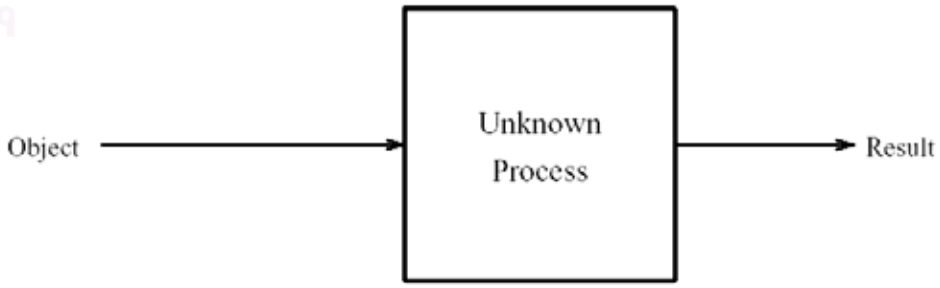
เมื่อ

$$\Delta w_{ji} = \eta\delta_j x_{ji}$$

2.1.3 การเรียนรู้แบบหลายตัวอย่างย่อย

ในปัจจุบันระบบการเรียนรู้สามารถแบ่งออกได้เป็น 3 ประเภทใหญ่ๆ ด้วยกัน คือ การเรียนรู้แบบสอน (supervised learning) การเรียนรู้แบบไม่สอน (unsupervised learning) และการเรียนแบบเสริมความแกร่ง (reinforcement learning)

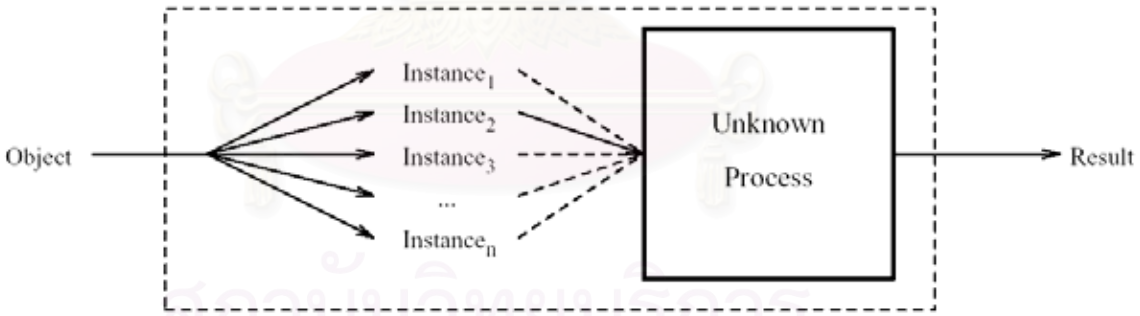
การเรียนรู้แบบสอนเป็นระบบการเรียนรู้ที่ผู้สอนต้องให้ตัวอย่างในการเรียนรู้แก่ระบบ โดยตัวอย่างที่ให้นั้นจะต้องสามารถระบุได้ว่าเป็นตัวอย่างในประเภทใด (class) เมื่อระบบเรียนรู้เสร็จแล้วจะสามารถนำมาใช้ในการจำแนกตัวอย่างทดสอบได้ โดยระบบจะบอกว่าตัวอย่างที่นำมาทดสอบนั้นอยู่ในประเภทใดดังแสดงในรูปที่ 2.12



รูปที่ 2.12 ลักษณะของระบบการเรียนรู้แบบสอน

การเรียนรู้แบบไม่สอนเป็นการเรียนที่ผู้สอนไม่จำต้องระบุว่าแต่ละตัวอย่างอยู่ในประเภทใด แต่เมื่อระบบเรียนเสร็จแล้วจะสามารถบอกได้แค่เพียงว่าตัวอย่างใดบ้างที่เป็นประเภทเดียวกัน ไม่สามารถบอกได้ว่าแต่ละตัวอย่างเป็นประเภทใด [11] ส่วนการเรียนแบบเสริมความแกร่งตัวผู้สอนจะมีการให้รางวัลและทำโทษแก่ระบบ โดยให้รางวัลเมื่อระบบทำงานได้ถูกต้องและทำโทษเมื่อระบบทำงานผิด [12]

อย่างไรก็ตามในบางปัญหานั้นก็ไม่สามารถที่จะเรียนได้อย่างมีประสิทธิภาพด้วยการเรียนรู้ที่มีอยู่ทั้ง 3 ประเภทนี้ ตัวอย่างของปัญหาในลักษณะนี้จะเป็นตัวอย่างที่มีลักษณะคลุมเครือ (ambiguous example) ซึ่งไม่สามารถระบุเจาะจงลงไปได้ว่าแต่ละตัวอย่างนั้นเป็นตัวอย่างบวกหรือลบ แต่ว่าสามารถที่จะบอกได้ว่าในกลุ่มตัวอย่างนั้นๆ (ประกอบด้วยตัวอย่างหลายๆตัวดังรูปที่ 2.13) มีตัวอย่างบวกอยู่ด้วยหรือไม่ ตัวอย่างเช่น มีประตูบานหนึ่งซึ่งถูกล็อกเอาไว้ และเรามีพวงกุญแจอยู่จำนวนหนึ่ง ซึ่งเรารู้แค่ว่าพวงกุญแจแต่ละพวงนั้นสามารถใช้เปิดประตูได้หรือไม่ แต่ไม่รู้ว่ากุญแจดอกไหนเป็นกุญแจที่ถูกต้อง เราต้องการรู้ว่าถ้าได้พวงกุญแจอันใหม่มา พวงกุญแจนั้นจะสามารถใช้เปิดประตูได้หรือไม่ หรืออย่างเช่น ปัญหาการจำแนกรูปภาพ เช่น เราต้องการให้ระบบเรียนรู้ว่าในภาพมีน้ำตกอยู่หรือไม่ โดยที่ตัวอย่างที่เราให้แก่ระบบเรียนก็จะบอกแค่ว่าแต่ละรูปเป็นรูปที่มีน้ำตกอยู่หรือไม่ แต่ไม่ทราบว่าน้ำตกอยู่ที่ตำแหน่งใดในภาพ (ไม่สามารถหรือจำเป็นต้องบอกว่าจุดภาพใดเป็นน้ำตก) เป็นต้น



รูปที่ 2.13 ลักษณะของระบบการเรียนรู้แบบหลายตัวอย่างย่อย

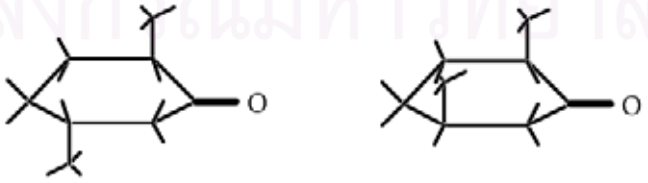
การเรียนรู้แบบหลายตัวอย่างย่อย [13, 14] เป็นระบบการเรียนรู้ที่ได้ถูกนำเสนอเพื่อนำมาใช้แก้ปัญหาในกรณีตัวอย่างที่มีลักษณะเช่นนี้ ซึ่งนับว่าเป็นทางเลือกหนึ่งของการเรียนรู้เมื่อผู้สอนไม่สามารถที่จะกำหนดค่าเป้าหมายให้กับตัวอย่างทุกตัวที่นำมาใช้ในการเรียนได้

ปัญหาที่ทำให้เกิดการเรียนรู้แบบหลายตัวอย่างย่อยขึ้นนั้น มาจากปัญหาที่นำเสนอโดย Dietterich และคณะ [15] เป็นปัญหาของการนำโมเลกุลมาใช้ทำยา (MUSK problem) จุดมุ่งหมายของปัญหานี้ คือ ต้องการที่จะหาแนวคิดที่จะทำนายได้ว่าโมเลกุลหนึ่งๆสามารถ

นำมาใช้ทำเป็นตัวยาได้หรือไม่ แต่เนื่องจากในการตัดสินว่าโมเลกุลหนึ่งๆนั้นจะสามารถนำมาใช้ทำเป็นตัวยาได้ เราทราบเพียงแค่ว่าโครงสร้างหลักควรมีลักษณะเป็นเช่นไร แต่ว่าในโครงสร้างหลักหนึ่งๆนั้นพันธะของอะตอมสามารถเปลี่ยนลักษณะไปได้หลากหลายแบบดังรูปที่ 2.14 ทำให้ไม่สามารถหาได้ว่าลักษณะเฉพาะแบบใดเป็นลักษณะเฉพาะที่ถูกต้องและสามารถนำมาใช้ทำยาได้ ซึ่งปัญหาลักษณะนี้ไม่ตรงกับปัญหาที่จะใช้ระบบการเรียนรู้ที่มีอยู่ทั้ง 3 แบบมาเรียนได้ จึงทำให้เกิดการเรียนรู้แบบหลายตัวอย่างย่อยขึ้น โดยชุดข้อมูล MUSK ได้ถูกนำมาใช้เป็นชุดข้อมูลทดสอบในการทดสอบประสิทธิภาพของระบบการเรียนรู้แบบหลายตัวอย่างย่อยที่พัฒนาขึ้นต่อๆมาเพื่อแก้ปัญหาตัวอย่างที่มีลักษณะคลุมเครือ

การเรียนรู้แบบหลายตัวอย่างย่อยจะนิยามให้เซตของตัวอย่างในการเรียนรู้ประกอบไปด้วยเซตของถุง (bag) โดยที่แต่ละถุงจะประกอบไปด้วยตัวอย่างย่อย (instance) ดังรูปที่ 2.13 ซึ่งจำนวนตัวอย่างย่อยในแต่ละถุงอาจมีจำนวนที่เท่ากันหรือไม่เท่ากันก็ได้ ชนิดของถุงตัวอย่างแบ่งเป็น 2 ประเภท คือ ถุงตัวอย่างบวกและถุงตัวอย่างลบ ถุงตัวอย่างหนึ่งๆจะจัดว่าเป็นถุงตัวอย่างบวกก็ต่อเมื่อในถุงนั้นมีตัวอย่างย่อยอย่างน้อย 1 ตัวที่เป็นตัวอย่างบวก และจะจัดว่าเป็นถุงตัวอย่างลบก็ต่อเมื่อตัวอย่างย่อยทุกตัวในถุงนั้นเป็นตัวอย่างลบ อย่างไรก็ตามแม้ว่าจะทราบว่าแต่ละถุงนั้นเป็นประเภทใด (บวกหรือลบ) แต่ว่าตัวอย่างย่อยแต่ละตัวนั้นจะไม่ทราบว่าเป็นประเภทใด จุดมุ่งหมายของการเรียนรู้แบบหลายตัวอย่างย่อยจะเป็นการหาแนวคิด (concept) จากถุงตัวอย่างที่ใช้ในการเรียนรู้ โดยแนวคิดนั้นจะต้องสามารถจำแนกตัวอย่างที่ใช้ในกระบวนการเรียนรู้ได้อย่างถูกต้อง และแนวคิดที่ได้นั้นจะถูกนำไปใช้ในการจำแนกถุงตัวอย่างที่ไม่เคยพบมาก่อนในการเรียน

Single Bonds Rotate



Multiple Stereoisomers Exist

รูปที่ 2.14 ตัวอย่างการเปลี่ยนลักษณะการเชื่อมต่อพันธะในโครงสร้างหลัก

อย่างไรก็ตามปัญหาของการเรียนรู้แบบหลายตัวอย่างย่อยนั้นจัดว่ามีความยากมากกว่าการเรียนรู้แบบสอนที่มีสัญญาณรบกวนในข้อมูลที่ใช้ในการเรียนรู้ (การเรียนรู้แบบสอนผู้สอนจะต้องสามารถระบุได้ว่าทุกตัวอย่างที่นำมาใช้ในการเรียนจัดอยู่ในประเภทใด) เนื่องจากว่าในถุงตัวอย่างบวกนั้นไม่ได้มีแค่เพียงตัวอย่างย่อยบวกแต่เพียงอย่างเดียว ทำให้มองได้เหมือนว่าตัวอย่างย่อยลบนั้นเป็นสัญญาณรบกวน และโดยปกติแล้วอัตราส่วนของตัวอย่างย่อยลบต่อตัวอย่างย่อยบวกในถุงตัวอย่างบวกก็มีค่าค่อนข้างสูง การเรียนรู้แบบนี้จึงอาจจะมองว่าคล้ายกับเป็นการเรียนรู้แบบสอนที่มีสัญญาณรบกวนเป็นจำนวนมาก แต่ว่าในกรณีที่ถุงตัวอย่างมีตัวอย่างย่อยเพียงแค่ตัวเดียว ปัญหาการเรียนรู้แบบหลายตัวอย่างย่อยสามารถลดรูปไปเป็นปัญหาการเรียนรู้แบบสอนได้

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 งานวิจัยทางด้านโปรแกรมตรรกะอันดับที่หนึ่งและนิวรอลเน็ตเวิร์ก

การโปรแกรมตรรกะเชิงอุปนัยหรือระบบไอแอลพีเป็นระบบการเรียนรู้ที่มีประสิทธิภาพระบบหนึ่ง ข้อดีของการเรียนรู้ของระบบนี้คือ การที่สามารถนำความรู้ภูมิหลังมาใช้ในการเรียนรู้ได้ และแนวคิดที่เรียนรู้ได้จะเป็นลักษณะของกฎที่มนุษย์สามารถอ่านและเข้าใจได้ง่าย ด้วยข้อดีเหล่านี้ระบบไอแอลพีจึงถูกนำไปประยุกต์ใช้กับงานในด้านต่างๆ เช่น การวิเคราะห์ไฟไนต์เอลิเมนต์ (Finite Element Mesh Design) การวิเคราะห์ความสามารถก่อกลายพันธุ์ของโมเลกุล (Mutagenesis) การจำแนกคุณภาพทางด้านชีววิทยาของน้ำในแม่น้ำ การสร้างแบบจำลองของโมเลกุลทางชีวภาพ (Biomolecular Modeling) การปรับปรุงคุณภาพของข้อมูลในการออกแบบโปรแกรม เป็นต้น ภาพรวมของการเรียนรู้ของระบบนี้ได้ถูกกล่าวไว้ในงานวิจัยของ Ivan Bratko และ Stephen H. Muggleton เมื่อ ปี 1995 [16]

อย่างไรก็ตามระบบการเรียนรู้แบบอ้างอิงความรู้นั้นก็มีข้อจำกัดเนื่องจากเป็นระบบที่ไม่ทนทานต่อข้อมูลที่มีสัญญาณรบกวน และได้มีงานวิจัยและแนวคิดต่างๆที่ถูกนำเสนอออกมาเพื่อแก้ปัญหานี้ หนทางหนึ่งที่เป็นที่น่าสนใจ คือ การนำระบบการเรียนรู้อื่นเข้ามาประยุกต์กับระบบการเรียนแบบนี้

งานวิจัยของ Alfred Ultsch และ Dieter Korus [17] เป็นงานวิจัยหนึ่งที่สนับสนุนแนวความคิดนี้ นักวิจัยทั้งสองท่านนำเสนอระบบการเรียนรู้ที่ผนวกนิวรอลเน็ตเวิร์กเข้าด้วยกันกับระบบการเรียนแบบอ้างอิงความรู้ โดยเสนอว่านิวรอลเน็ตเวิร์กเป็นระบบที่แนวคิดที่ได้จากการเรียนไม่สามารถแปลความหมายให้มนุษย์เข้าใจได้ง่าย ส่วนระบบการเรียนแบบอ้างอิงความรู้ก็เป็นระบบที่ไม่ทนทานต่อสัญญาณรบกวน แต่ว่าเมื่อนำระบบการเรียนรู้ทั้งสองมาผนวกเข้า

ด้วยกันแล้ว ข้อดีของแต่ละระบบจะสามารถช่วยแก้ปัญหาให้กับอีกระบบหนึ่งได้ และทำให้ได้ระบบที่สมบูรณ์มากขึ้น โดยได้แสดงวิธีต่างๆในการผนวกทั้งสองระบบเข้าด้วยกันแบบคร่าวๆ

มีงานวิจัยบางส่วนที่แสดงให้เห็นว่าการนำนิวรอลเน็ตเวิร์กเข้ามารวมด้วยนั้นจะทำให้ระบบมีประสิทธิภาพมากขึ้นและให้ผลการเรียนรู้ที่ดีขึ้น ตัวอย่างเช่น งานวิจัยของ Geoffrey G. Towell และ Jude W. Shavlik นักวิจัยสองท่านนี้นำเสนอแนวคิดของการผนวกนิวรอลเน็ตเวิร์กเข้ากับความรู้เบื้องต้น โดยมีสมมติฐานว่านิวรอลเน็ตเวิร์กจะช่วยให้การเรียนรู้มีประสิทธิภาพมากขึ้นกว่าการนำความรู้ที่มีมาใช้เลยโดยตรง ระบบที่นำเสนอเรียกว่า Knowledge-Based Artificial Neural Networks (KBANN) [18] ขั้นแรกของการเรียนรู้จะสร้างเน็ตเวิร์กที่สอดคล้องกับความรู้เบื้องต้นที่มีอยู่ (ความรู้เบื้องต้นอาจเป็นกลุ่มความรู้ภูมิหลัง หรือความรู้ที่ผ่านการเรียนรู้มาแล้วก็ได้) และทำการกำหนดค่าน้ำหนักให้กับเส้นเชื่อมของเน็ตเวิร์กโดยดูจากชนิดของประพจน์ (มีนิเสธหรือไม่มีนิเสธ) จากนั้นจึงทำการสอนเน็ตเวิร์กโดยใช้วิธีแบ็กพรอพาเกชันนิวรอลเน็ตเวิร์ก ในการทดลอง พบว่า KBANN ที่ผ่านกระบวนการเรียนรู้แล้วให้เปอร์เซ็นต์ความถูกต้องสูงกว่าเมื่อเปรียบเทียบกับการเรียนรู้ด้วยวิธีอื่น งานวิจัยนี้นับว่าเป็นงานวิจัยแรกๆที่เสนอการแก้ปัญหาของระบบการเรียนแบบใช้ความรู้ภูมิหลัง (ระบบไอแอลพีก็จัดว่าเป็นระบบการเรียนแบบใช้ความรู้ภูมิหลังแบบหนึ่ง) ด้วยการนำเทคนิคการเรียนรู้ของนิวรอลเน็ตเวิร์กมาประยุกต์ หรืองานวิจัยของ Rajesh Parekh และ Vasant Honavar ที่ผนวกนิวรอลเน็ตเวิร์กเข้ากับระบบการเรียนแบบอ้างอิงความรู้คล้ายกับระบบ KBANN โดยการสร้างเน็ตเวิร์กเริ่มต้นจะใช้หลักการของ KBANN โดยความรู้เบื้องต้นจะอยู่ในรูปแบบของตรรกศาสตร์ประพจน์ แต่ว่างานวิจัยนี้ได้พัฒนาให้สามารถนำมาใช้กับสัญพจน์ที่มีจำนวนจริงได้ และสามารถปรับขนาดของข่ายงานตามความเหมาะสมได้ โดยเรียกระบบนี้ว่า Tiling-Pyramid [19] ซึ่งประยุกต์มาจากวิธีการ Pyramid และ MTTiling จากการทดลอง Tiling-Pyramid มีค่าเปอร์เซ็นต์ความถูกต้องในการจำแนกสูงขึ้นกว่าการนำกฎเพียงอย่างเดียวมาใช้ในการจำแนก งานวิจัยเหล่านี้ได้แสดงให้เห็นถึงแนวคิดที่ว่าการรวมระบบนิวรอลเน็ตเวิร์กเข้ากับการเรียนแบบอ้างอิงความรู้นั้นจะให้ผลในการจำแนกที่ดีขึ้น

งานวิจัยที่แสดงให้เห็นถึงการรวมระบบที่สมบูรณ์นั้นเป็นของ Artur S. d'Avila Garcez, Krysia B. Broda และ Dov M. Gabbay ระบบที่นำเสนอมีชื่อว่า C-IL^{2}P [20] ระบบการเรียนรู้นี้ผนวกนิวรอลเน็ตเวิร์กเข้ากับการโปรแกรมตรรกะแบบประพจน์ งานวิจัยนี้ได้แสดงให้เห็นวิธีตั้งแต่การสร้างโครงสร้างของนิวรอลเน็ตเวิร์กจากข้อมูลตรรกะที่รับเข้ามา การใส่ตัวอย่างให้แก่เน็ตเวิร์กเพื่อทำการเรียนรู้ และการปรับค่าน้ำหนักเส้นเชื่อมให้เหมาะสมกับตัวอย่างที่ใช้สอน นอกจากนี้ยังได้แสดงการเปลี่ยนรูปจากเน็ตเวิร์กกลับไปเป็นกฎเหมือนเดิมเมื่อทำการเรียนรู้เสร็จสิ้นแล้วเพื่อให้สามารถเข้าใจถึงสิ่งที่เรียนมาและนำไปประยุกต์ใช้ได้ง่ายขึ้น ในส่วนของการทดลองนั้นได้ทำทั้งใน

ส่วนของกระบวนการเรียนรู้ของเน็ตเวิร์กและการแปลงจากเน็ตเวิร์กที่ผ่านกระบวนการเรียนรู้แล้วกลับไปเป็นกฎ โดยเปรียบเทียบผลของ C-IL^{2}P กับ อัลกอริทึมอื่นๆ เช่น แบ็กพรอพาเกชันนิวรอลเน็ตเวิร์ก และ ID3 เป็นต้น

ระบบ C-IL^{2}P นี้นับว่าเป็นระบบการเรียนรู้ระบบหนึ่งที่นำนิวรอลเน็ตเวิร์กมาประยุกต์เข้ากับโปรแกรมตรรกะโดยสมบูรณ์ แต่อย่างไรก็ตามตรรกะที่นำมารวมเข้ากับนิวรอลเน็ตเวิร์กนั้นก็เป็นลักษณะของตรรกศาสตร์ประพจน์ ซึ่งไม่สามารถอธิบายแนวคิดที่ซับซ้อนได้เหมือนกับตรรกะอันดับที่หนึ่ง สำหรับตัวอย่างของงานวิจัยที่นำนิวรอลเน็ตเวิร์กมาประยุกต์เข้ากับตรรกะอันดับที่หนึ่งนั้น ได้แก่ งานวิจัยของ Boonserm Kijsirikul, Sukree Sinthupinyo และ Kongsak Chongkasemwongse นักวิจัยกลุ่มนี้ใช้วิธีนำกฎที่ได้จากระบบไอแอลพีมาผ่านการดึงลักษณะสำคัญ (feature extraction) ด้วยหลักการของสายสัญพจน์ปิด (closed chain) และ สายสัญพจน์เปิด (open chain) จากนั้นนำลักษณะสำคัญที่ได้มาหาค่าความจริงแล้วนำไปเป็นอินพุตของนิวรอลเน็ตเวิร์ก ให้เน็ตเวิร์กทำการเรียนรู้ด้วยขั้นตอนแบ็กพรอพาเกชัน เมื่อเรียนรู้เสร็จแล้วเน็ตเวิร์กที่ได้สามารถนำไปใช้ในการจำแนกตัวอย่างที่ไม่เคยเห็นมาก่อนได้ ระบบที่นำเสนอนั้นมีชื่อว่า BANNAR [21] โดยในการทดลองพบว่าเมื่อใช้แบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กมาช่วยประมาณกฎจะให้เปอร์เซ็นต์ความถูกต้องสูงกว่าการใช้กฎจากระบบไอแอลพีเพียงอย่างเดียว

อย่างไรก็ตามแม้ว่างานวิจัยของ Kijsirikul และคณะ จะสามารถนำนิวรอลเน็ตเวิร์กมาใช้ร่วมกับตรรกะอันดับที่หนึ่งได้ แต่ว่าการนำมาใช้นั้นยังจำเป็นต้องพึ่งระบบไอแอลพีอยู่ เนื่องจากในขั้นแรกนั้น BANNAR ต้องใช้ระบบไอแอลพีเพื่อสร้างกฎที่จะนำมาเป็นอินพุตให้กับนิวรอลเน็ตเวิร์ก งานวิจัยนี้จึงต้องการนำเสนอวิธีการเรียนรู้ที่สามารถประยุกต์นิวรอลเน็ตเวิร์กเข้ากับโปรแกรมตรรกะอันดับที่หนึ่งได้ โดยให้เป็นระบบการเรียนรู้ที่สมบูรณ์ในตัวเองเหมือนดังเช่นระบบ C-IL^{2}P และไม่จำเป็นต้องพึ่งระบบไอแอลพีในการเรียนรู้

2.2.2 งานวิจัยทางด้านการเรียนรู้แบบหลายตัวอย่างย่อย

การเรียนรู้แบบหลายตัวอย่างย่อยเป็นระบบการเรียนรู้ที่เกิดขึ้นเพื่อใช้แก้ปัญหาตัวอย่างที่มีลักษณะคลุมเครือ ซึ่งไม่สามารถแก้ได้อย่างมีประสิทธิภาพโดยระบบการเรียนรู้ที่มีอยู่ในปัจจุบัน อันได้แก่ การเรียนรู้แบบสอน การเรียนรู้แบบไม่สอน และการเรียนแบบเสริมความแกร่ง

ปัญหาที่ตัวอย่างมีลักษณะคลุมเครือนั้นถูกคิดค้นขึ้นเมื่อปี 1997 โดย Thomas G. Dietterich, Richard H. Lathrop และ Tomas Lozano-Perez โดยนักวิจัยกลุ่มนี้ต้องแก้ปัญหาของการนำโมเลกุลมาใช้ทำยา (MUSK problem) ซึ่งปัญหานี้มีลักษณะเป็นแบบคลุมเครือ (รายละเอียดกล่าวไว้แล้วในหัวข้อ2.1.3) วิธีการที่นำเสนอมีชื่อว่า Axis-Parallel Rectangles [15]

โดยใช้การวัดระยะห่างของแต่ละค่าคุณสมบัติเทียบกับแกนอ้างอิง วิธีที่นำเสนอนี้ให้ค่าเปอร์เซ็นต์ความถูกต้องเป็น 92.4% ในชุดข้อมูล MUSK1 และ 90.2% ในชุดข้อมูล MUSK2 ซึ่งนับว่าเป็นเปอร์เซ็นต์ความถูกต้องที่สูงเมื่อเทียบกับวิธีที่ใช้กับระบบการเรียนรู้แบบอื่นๆที่ไม่สนใจตัวอย่างย่อย เช่น C4.5 หรือ นิวรอลเน็ตเวิร์ก นอกจากนี้ชุดข้อมูล MUSK ยังได้ถูกนำมาใช้เป็นชุดข้อมูลทดสอบกับระบบการเรียนรู้แบบหลายตัวอย่างย่อยที่ได้พัฒนาขึ้นมาในภายหลังอีกด้วย

หลังจากนั้นได้มีงานวิจัยจำนวนมากที่ได้นำเสนอเพื่อแก้ปัญหาลักษณะนี้ เช่น วิธี Diverse Density [14] และ ซับพอร์ตเวกเตอร์แมชชีนสำหรับการเรียนรู้แบบหลายตัวอย่างย่อย [22] เป็นต้น นิวรอลเน็ตเวิร์กเป็นวิธีหนึ่งที่ถูกนำเสนอเพื่อมาใช้กับการเรียนรู้แบบหลายตัวอย่างย่อย เช่น งานของ Zhi-Hua Zhou และ Min-Ling Zhang ได้นำนิวรอลเน็ตเวิร์กมาประยุกต์ใช้กับการเรียนรู้แบบหลายตัวอย่างย่อย [23] มีการนิยามค่าความผิดพลาดขึ้นมาใหม่และเสนออัลกอริทึมเพื่อใช้ในการเรียนรู้ของเน็ตเวิร์ก โดยการปรับค่าน้ำหนักของเส้นเชื่อมจะยังคงใช้หลักการพื้นฐานของขั้นตอนแบ็กพรอพาเกชัน แต่ว่าผลการทดลองยังคงให้ค่าเปอร์เซ็นต์ความถูกต้องน้อยกว่าวิธีของ Dietterich และคณะ ในการทดสอบกับชุดข้อมูล MUSK หลังจากนั้น Pakaket Wattuya, Arnon Rungsawang และ Boonserm Kijsirikul ได้พัฒนาการเรียนรู้แบบหลายตัวอย่างย่อยด้วยนิวรอลเน็ตเวิร์กให้ดีขึ้นจากงานของ Zhi-Hua Zhou และ Min-Ling Zhang โดยวิธีที่นำเสนอมีชื่อว่า MINN-SBC [24] อัลกอริทึมที่ใช้ของ MINN-SBC เป็นขั้นตอนของแบ็กพรอพาเกชัน แต่ว่ามีการตั้งค่าขอบเขตในการจำแนกในการเรียนรู้ใหม่ จากโดยทั่วไปที่จะใช้ค่า 0.5 เป็นตัวแบ่ง (ในกรณีที่ถุงตัวอย่างบวกมีค่าเป้าหมายเป็น 1 และถุงตัวอย่างลบมีค่าเป็น 0) ก็ลองตั้งค่าขอบเขตในกระบวนการเรียนรู้ให้มีค่าสูงขึ้น โดยในการทดลองพบว่าผลที่ได้มีค่าความถูกต้องสูงขึ้นจากงานของ Zhi-Hua Zhou และ Min-Ling Zhang

อย่างไรก็ตามจากผลการทดลองพบว่าวิธีเหล่านี้ยังให้ค่าเปอร์เซ็นต์ความถูกต้องน้อยกว่าวิธีของ Dietterich และคณะ ในการทดสอบกับชุดข้อมูล MUSK แต่ว่าก็ทำให้ได้วิธีการเรียนรู้ที่หลากหลายมากยิ่งขึ้น

ในปี 2003 Xin Huang, Shu-Ching Chen และ Mei-Ling Shyu ได้เสนอมุมมองใหม่ในการแก้ปัญหาของการเรียนรู้แบบหลายตัวอย่างย่อย โดยพยายามที่จะมองปัญหาของการเรียนรู้แบบหลายตัวอย่างย่อยไปเป็นปัญหาของการหาค่าเหมาะสมที่สุด (optimization)[13] โดยการทำให้ค่าความผิดพลาดมีค่าน้อยที่สุด ซึ่งถ้าทำได้เช่นนั้นแล้ว ปัญหานี้ก็จะเป็นแค่ปัญหาในการค้นหาค่าเหมาะสมที่สุดเท่านั้นและสามารถนำวิธีการต่างๆที่มีอยู่แล้วในการแก้ปัญหาประเภทการหาค่าเหมาะสมที่สุด มาใช้แก้ปัญหาของการเรียนรู้แบบหลายตัวอย่างย่อยได้ โดยในการทดลองได้นำ

วิธี Quasi-Newton มาใช้ทดสอบกับชุดข้อมูลทดสอบ MUSK ให้เปอร์เซ็นต์ความถูกต้องสูงสุดเป็น 92.4% เทียบเท่ากับเปอร์เซ็นต์ความถูกต้องของ Dietterich

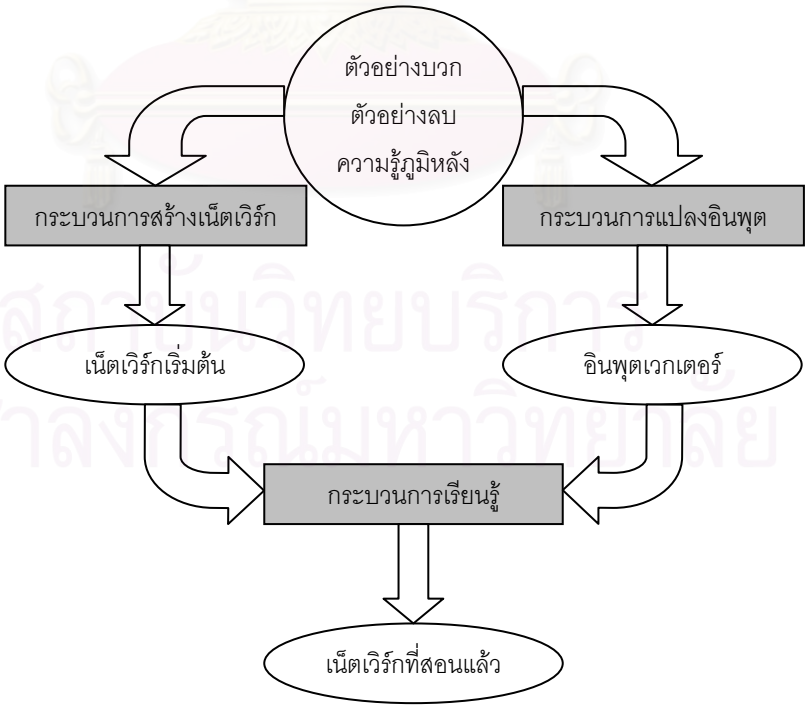
บทที่ 3

นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง

บทนี้จะกล่าวถึงรายละเอียดของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง ประกอบไปด้วยโครงสร้างของระบบซึ่งจะอธิบายภาพรวมของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง ต่อมากล่าวถึงการสร้างเน็ตเวิร์กจากอินพุตที่เป็นตรรกะอันดับที่หนึ่ง การป้อนอินพุตให้กับเน็ตเวิร์กและหัวข้อสุดท้ายกล่าวถึงการปรับค่าน้ำหนักเส้นเชื่อมของเน็ตเวิร์ก

3.1 โครงสร้างของระบบ

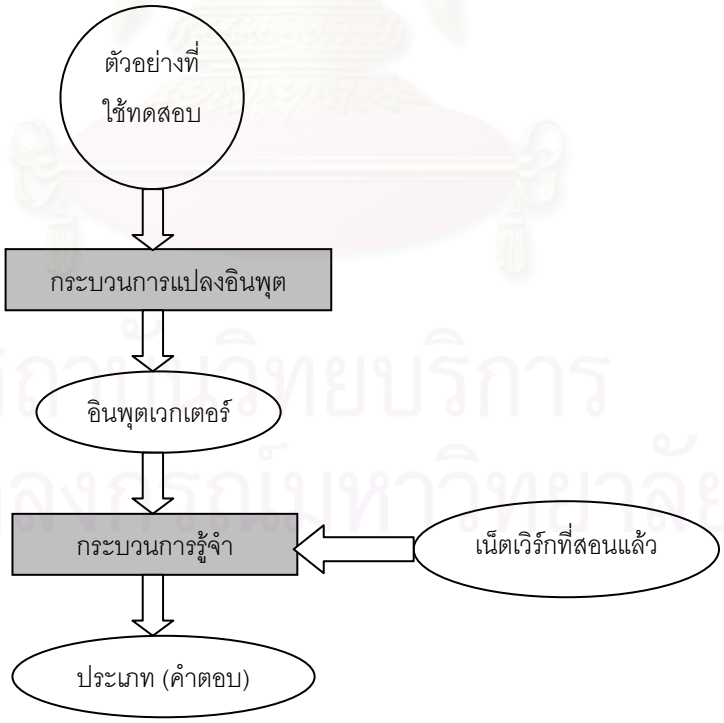
นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งเป็นระบบที่ถูกพัฒนาขึ้นเพื่อเพิ่มประสิทธิภาพในการจำแนกตัวอย่างให้สูงขึ้น โดยรวมข้อดีของนิวรอลเน็ตเวิร์กและการโปรแกรมตรรกะเชิงอุปนัยเข้าไว้ด้วยกัน คือ ความสามารถในการทนต่อข้อมูลที่มีสัญญาณรบกวนและจำแนกตัวอย่างแบบหลายประเภทได้อย่างมีประสิทธิภาพของนิวรอลเน็ตเวิร์กกับความสามารถในการนำความรู้ภูมิหลังมาใช้และอธิบายความรู้ให้มนุษย์เข้าใจได้ง่ายของระบบไอแอลพี เพื่อทำให้ระบบมีความยืดหยุ่นสูงขึ้นและสามารถจัดการกับข้อจำกัดของระบบไอแอลพีได้ การทำงานของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งจะแบ่งออกเป็น 2 ส่วน คือ ขั้นตอนการเรียนรู้ และขั้นตอนการรู้จำ



รูปที่ 3.1 ขั้นตอนการเรียนรู้

ขั้นตอนการเรียนรู้ เริ่มจากระบบจะรับอินพุตเข้ามาเป็นตัวอย่างบวก ตัวอย่างลบ และความรู้ภูมิหลัง ซึ่งทั้งหมดจะอยู่ในรูปของตรรกะอันดับที่หนึ่ง นำมาสร้างเป็นเน็ตเวิร์กเริ่มต้น จากนั้นจะทำการป้อนอินพุตให้กับเน็ตเวิร์กเพื่อทำการสอนเน็ตเวิร์กที่ได้สร้างขึ้น โดยในส่วนนี้จะมีขั้นตอนการแปลงอินพุต เพื่อทำให้อินพุตที่อยู่ในรูปของตรรกะอันดับที่หนึ่งเปลี่ยนมาเป็นอินพุตเวกเตอร์จำนวนจริงที่ใช้เรียนได้สำหรับนิวรอลเน็ตเวิร์กก่อน จากนั้นอินพุตที่ผ่านการแปลงแล้วจะนำไปใช้ในกระบวนการเรียนรู้เพื่อสอนเน็ตเวิร์ก กระบวนการเรียนรู้เป็นส่วนที่ทำหน้าที่ในการปรับค่าน้ำหนักของเส้นเชื่อมของเน็ตเวิร์กเพื่อทำให้เน็ตเวิร์กสามารถจำแนกตัวอย่างที่ใช้สอนได้ถูกต้อง การปรับค่าน้ำหนักของเส้นเชื่อมจะใช้วิธีของแบ็กพรอพาเกชันที่มีฟังก์ชันกระตุ้นเป็นองค์ประกอบซิกมอยด์ [25] ขั้นตอนในการเรียนรู้แสดงในรูปที่ 3.1

เมื่อเสร็จสิ้นขั้นตอนการเรียนรู้แล้ว เน็ตเวิร์กที่ได้นั้นสามารถนำไปใช้ในการรู้จำตัวอย่างใหม่ โดยในขั้นตอนการรู้จำนั้น ตัวอย่างที่ต้องการจำแนกจะถูกแปลงเป็นอินพุตเวกเตอร์ให้กับเน็ตเวิร์กที่ผ่านการเรียนรู้แล้ว โดยการแปลงอินพุตจะทำเช่นเดียวกับในขั้นตอนของการเรียนรู้ จากนั้นอินพุตจะถูกคำนวณหาผลลัพธ์โดยเน็ตเวิร์กที่สอนแล้ว ในการจำแนกประเภทจะเลือกประเภทที่เป็นคำตอบจากนิวรอนในชั้นผลลัพธ์ที่ให้ค่าเอาต์พุตสูงที่สุด ขั้นตอนในการรู้จำแสดงในรูปที่ 3.2



รูปที่ 3.2 ขั้นตอนการรู้จำ

3.2 การสร้างเน็ตเวิร์ก

ขั้นตอนนี้เป็นส่วนแรกของขั้นตอนการเรียนรู้เพื่อทำการสร้างเน็ตเวิร์กเริ่มต้นที่สอดคล้องกับอินพุตที่เป็นความรู้ภูมิหลังและตัวอย่างที่ใช้สอน ก่อนที่จะนำเน็ตเวิร์กที่ได้นี้ไปใช้ในขั้นตอนต่อๆไป

เนื่องจากนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งนั้นเป็นระบบการเรียนรู้ที่พัฒนามาจากนิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้า (feedforward neural network) ดังนั้นตัวโครงสร้างของเน็ตเวิร์กที่จะสร้างก็จะมีพื้นฐานมาจากนิวรอลเน็ตเวิร์ก แต่ว่าประยุกต์ให้สามารถรับอินพุตที่เป็นตัวอย่างและความรู้ภูมิหลังที่อยู่ในรูปแบบของตรรกะอันดับที่หนึ่งได้ อย่างไรก็ตามการสร้างเน็ตเวิร์กจากอินพุตที่เป็นตรรกะอันดับที่หนึ่งนั้นมีความแตกต่างจากการสร้างเน็ตเวิร์กจากอินพุตที่เป็นตรรกศาสตร์ประพจน์ เนื่องจากในตรรกศาสตร์ประพจน์นั้นไม่มีตัวแปรที่นำมาใช้สร้างความสัมพันธ์ระหว่างสัญพจน์ได้ ทำให้สามารถแปลงอินพุตที่เป็นสัญพจน์แต่ละสัญพจน์มาเป็นนิวรอนของเน็ตเวิร์กได้โดยตรงดังเช่นงานวิจัยของ Garcez และคณะ [20] ในขณะที่อินพุตที่เป็นตรรกะอันดับที่หนึ่งซึ่งมีการใช้ตัวแปรไม่สามารถทำได้

ดังนั้นเพื่อให้สามารถสร้างเน็ตเวิร์กจากอินพุตที่เป็นตรรกะอันดับที่หนึ่งได้ จึงทำการลดรูปปัญหาโดยใช้วิธีการคลี่อินพุตที่เป็นตรรกะอันดับที่หนึ่งออกมาเป็นตรรกศาสตร์ประพจน์ทุกกรณีเสียก่อน แล้วจึงใช้อินพุตที่คลี่แล้วมาสร้างเน็ตเวิร์ก โดยการคลี่นั้นจะต้องกำหนดความลึกของตัวแปรในสัญพจน์ (จำนวนตัวแปร) ที่จะนำมาใช้อธิบายแนวคิดที่ต้องการเรียนรู้ด้วยเพื่อให้กำหนดได้ว่าต้องคลี่สัญพจน์ถึงระดับใด โดยปกติแล้วการกำหนดค่าความลึกนั้นเป็นค่าโครงแบบ (configuration) ตัวหนึ่งที่สามารถกำหนดได้ในการเรียนด้วยระบบไอแอลพี อย่างไรก็ตามถึงแม้ว่าการคลี่สัญพจน์นั้นจะคลี่ออกมาทุกกรณีก็ตาม แต่ว่าหลังจากที่คลี่แล้วจะพบว่าในบางสัญพจน์นั้นเราสามารถที่จะทำการตัดเล็ม (prune) ได้ เมื่อพบว่าสัญพจน์นั้นมีค่าความจริงเป็นเท็จในทุกๆกรณี ดังเช่น สัญพจน์ father(x,y) เมื่อทำการคลี่โดยกำหนดจำนวนตัวแปรมีค่าเป็น 3 จะได้ผลออกมาเป็นดังรูปที่ 3.3

father(x,x)	father(x,y)	father(x,z)
father(y,x)	father(y,y)	father(y,z)
father(z,x)	father(z,y)	father(z,z)

รูปที่ 3.3 สัญพจน์ father(x,y) ที่ทำการคลี่แล้ว

ในจำนวนสัญพจน์ทั้ง 9 สัญพจน์ที่ได้คลี่ออกมาแล้วนั้น จะเห็นได้ว่าสัญพจน์ father(x,x), father(y,y) และ father(z,z) มีค่าความจริงเป็นเท็จเสมอเนื่องจากไม่ว่าจะแทนที่ตัวแปร x,y และ z

ด้วยใครก็ตาม ก็ไม่มีทางที่คนๆนั้นจะเป็นพ่อของตัวเองได้ จึงสามารถทำการตัดเล็มสัญพจน์เหล่านี้ทิ้งได้ สัญพจน์ที่เหลืออีก 6 ตัวจะถูกนำไปใช้ในการสร้างเน็ตเวิร์กต่อไป

สำหรับการสร้างเน็ตเวิร์กนั้น ได้กำหนดโครงสร้างของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งออกเป็น 3 ส่วนด้วยกัน คือ ชั้นนำเข้า (input layer) ชั้นซ่อน (hidden layer) และชั้นผลลัพธ์ (output layer) โดยกำหนดให้ในแต่ละชั้นมีความหมายและหน้าที่ดังต่อไปนี้

1. ชั้นนำเข้า เป็นชั้นที่แสดงถึงสัญพจน์ (predicate) ต่างๆที่จะนำมาใช้เป็นตัวแทนของกฎหรือแนวคิดในการเรียน โดยจะเป็นชั้นแรกในการรับข้อมูล ทำการประมวลผลก่อนที่จะส่งผ่านไปยังชั้นต่อๆไป จำนวนนิวรอนในชั้นนี้จะขึ้นอยู่กับจำนวนสัญพจน์จากความรู้ภูมิหลัง โดยสัญพจน์ 1 สัญพจน์จะถูกแทนด้วยนิวรอน 1 นิวรอนในชั้นนำเข้า ถ้าสัญพจน์นั้นมีจำนวนอาร์กิวเมนต์ (arity) เพียง 1 ตัว แต่ถ้าสัญพจน์นั้นมีจำนวนอาร์กิวเมนต์มากกว่า 1 ตัวแล้ว จำนวนนิวรอนสำหรับสัญพจน์นั้นจะมีค่าเท่ากับจำนวนที่เป็นไปได้ทั้งหมดในการสลับที่ของตัวแปร (permutation) สำหรับอาร์กิวเมนต์ของสัญพจน์นั้นๆ
2. ชั้นซ่อน เป็นชั้นที่เชื่อมต่อระหว่างชั้นนำเข้าและชั้นผลลัพธ์ ชั้นนี้จะช่วยเพิ่มความสามารถในการเรียนกฎหรือแนวคิดที่มีความซับซ้อนได้ จำนวนนิวรอนในชั้นนี้มักจะกำหนดโดยดูจากความซับซ้อนของแนวคิดที่ต้องการเรียนรู้ ในงานวิจัยนี้ใช้ชั้นซ่อนจำนวน 1 ชั้นและจำนวนนิวรอนในชั้นซ่อนจะถูกกำหนดโดยดูจากผลการทดลอง
3. ชั้นผลลัพธ์ เป็นชั้นสุดท้ายของเน็ตเวิร์ก ชั้นผลลัพธ์จะแสดงถึงแนวคิดที่ต้องการให้ระบบทำการเรียนรู้ โดยชั้นนี้จะคำนวณหาผลลัพธ์สุดท้ายของเน็ตเวิร์กเพื่อนำมาใช้ในการจำแนกตัวอย่าง จำนวนนิวรอนในชั้นผลลัพธ์จะขึ้นอยู่กับจำนวนแนวคิดทั้งหมดที่ต้องการเรียนหรือจำนวนกลุ่มที่ต้องการจำแนก

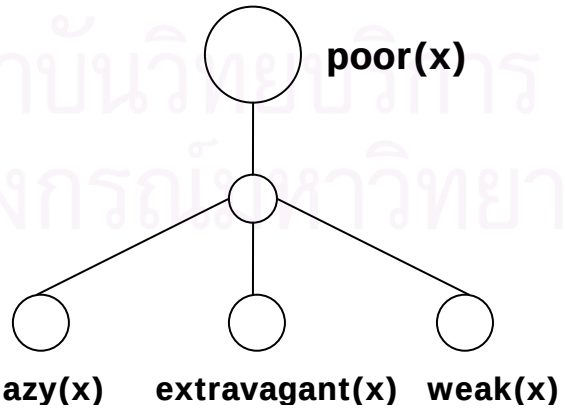
หลักของข้อกำหนดเหล่านี้ จะถูกนำมาใช้ในการสร้างเน็ตเวิร์ก ยกตัวอย่างดังเช่น การเรียนแนวคิดของ poor(x) ซึ่งมีอินพุตที่ใช้ในการเรียนเป็นดังรูปที่ 3.4

- **ความรู้ภูมิหลัง** : lazy(John), extravagant(John), lazy(Peter), weak(Peter), extravagant(Bob), weak(Bob)
- **ตัวอย่างบวก** : poor(John)
- **ตัวอย่างลบ** : poor(Peter), poor(Bob)

รูปที่ 3.4 อินพุตสำหรับการเรียนแนวคิด poor(x)

อินพุตที่ได้รับนั้นประกอบด้วย 3 ส่วนด้วยกัน คือ ความรู้ภูมิหลัง ตัวอย่างบวกและตัวอย่างลบ ในที่นี้มีตัวอย่างบวก 1 ตัว คือ poor(John) ซึ่งหมายความว่า John เป็นคนจน ลักษณะของ John ได้ถูกอธิบายไว้ในความรู้ภูมิหลัง คือ John เป็นคนเกียจคร้าน (lazy(John)) และฟุ่มเฟือย (extravagant(John)) ส่วนตัวอย่างลบมี 2 ตัว คือ poor(Peter) และ poor(Bob) ซึ่งหมายความว่า Peter และ Bob ไม่ได้เป็นคนจน โดยที่ Peter เป็นคนเกียจคร้านและไม่แข็งแรง (weak(Peter)) Bob เป็นคนฟุ่มเฟือยและไม่แข็งแรง

อินพุตที่ได้รับมีจำนวนสัญพจน์ที่ปรากฏในความรู้ภูมิหลังอยู่ 3 ตัวด้วยกัน คือ lazy(x), extravagant(x) และ weak(x) (กำหนดให้จำนวนตัวแปรมีค่าเป็น 1) ซึ่งสัญพจน์เหล่านี้จะถูกนำมาใช้อธิบายกฎที่ต้องการเรียนจึงนำสัญพจน์เหล่านี้มาสร้างเป็นนิวรอนในชั้นนำเข้า และเนื่องจากสัญพจน์ทั้ง 3 สัญพจน์นี้มีจำนวนอาร์กิวเมนต์เพียง 1 ตัวเท่านั้น ดังนั้นเมื่อสร้างเน็ตเวิร์กจากอินพุตชุดนี้จะได้ว่า ในชั้นนำเข้าจะมีนิวรอนอยู่ 3 นิวรอน เพื่อแทนลักษณะทั้ง 3 สัญพจน์นั้น ส่วนจำนวนนิวรอนในชั้นผลลัพธ์จะมีเพียง 1 นิวรอนเท่านั้น เนื่องจากแนวคิดที่ต้องการเรียนมีเพียงแนวคิดเดียวเท่านั้น คือ poor(x) เน็ตเวิร์กที่สร้างแล้วแสดงในรูปที่ 3.5



รูปที่ 3.5 ลักษณะโครงสร้างของเน็ตเวิร์กโดยกำหนดให้ในชั้นซ่อนมีนิวรอน 1 นิวรอน

3.3 การป้อนอินพุตให้กับเน็ตเวิร์ก

เมื่อสร้างเน็ตเวิร์กแล้ว ขั้นตอนต่อไปก็จะทำการป้อนอินพุตให้กับเน็ตเวิร์กเพื่อทำการเรียนรู้ ซึ่งโดยปกติแล้วอินพุตของนิวรอลเน็ตเวิร์กจะอยู่ในรูปของจำนวนจริง แต่ว่าอินพุตของระบบไอแอลพีจะอยู่ในรูปของตรรกะ (ความรู้ภูมิหลังและตัวอย่าง) ซึ่งไม่สามารถป้อนให้กับนิวรอลเน็ตเวิร์กเพื่อใช้เรียนได้โดยตรง จำเป็นต้องเปลี่ยนอินพุตที่เป็นลักษณะของตรรกะไปเป็นอินพุตที่สามารถเรียนรู้ได้โดยนิวรอลเน็ตเวิร์กเสียก่อน

ปกติแล้วตัวอย่างที่ใช้สอนนิวรอลเน็ตเวิร์ก 1 ตัว จะประกอบไปด้วยค่าอินพุตของนิวรอนทุกตัวในชั้นนำเข้าและค่าเป้าหมายที่ต้องการของตัวอย่างนั้นๆ ดังนั้นในขั้นตอนการแปลงอินพุตจากรูปแบบของตรรกะมาเป็นรูปแบบของจำนวนจริงจึงมี 2 ส่วนด้วยกัน คือ แปลงค่าอินพุตให้กับนิวรอนในชั้นนำเข้า และแปลงค่าเป้าหมายให้กับนิวรอนในชั้นผลลัพธ์

ในกระบวนการแปลงอินพุตให้กับนิวรอนในชั้นนำเข้าจะทำทีละ 1 ตัวอย่างและทีละ 1 นิวรอน ตามนิยามต่อไปนี้

$$X_{ij} = \begin{cases} 1 & \text{if } L_i\theta_j \text{ is true in background knowledge} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

เมื่อ

X_{ij} คือ ค่าอินพุตของนิวรอนที่ i ในชั้นนำเข้าเมื่อทำการป้อนตัวอย่างที่ j

L_i คือ สัญพจน์ประจำนิวรอนที่ i ในชั้นนำเข้า

θ_j คือ การแทนค่าตัวแปรด้วยค่าคงที่จากตัวอย่างที่ j

หรือกล่าวอีกนัยหนึ่งว่า ตัวอย่าง 1 ตัวจะถูกเปลี่ยนให้เป็นค่าจำนวนจริง 1 เพื่อเป็นอินพุตของนิวรอน ถ้าแทนค่าตัวแปรในสัญพจน์ประจำนิวรอนด้วยค่าคงที่ที่ปรากฏในตัวอย่างแล้วมีค่าความจริงเป็นจริงในความรู้ภูมิหลัง แต่ถ้าไม่เป็นจริงในความรู้ภูมิหลังก็จะกำหนดให้อินพุตของนิวรอนมีค่าเป็น 0 สำหรับค่าเป้าหมายของนิวรอนในชั้นผลลัพธ์จะกำหนดตามนิยามต่อไปนี้

$$T_{kj} = \begin{cases} 1 & \text{if } L_k\theta_j \text{ is positive example} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

เมื่อ

T_{kj} คือ ค่าเป้าหมายของนิวรอนที่ k ในชั้นผลลัพธ์เมื่อทำการป้อนตัวอย่างที่ j

L_k คือ สัญพจน์ประจำนิวรอนที่ k ในชั้นผลลัพธ์

นั่นคือ ค่าเป้าหมายของนิวรอนจะมีค่าเป็น 1 ถ้าแทนค่าตัวแปรในสัญพจน์ประจำนิวรอนด้วยค่าคงที่ที่ปรากฏในตัวอย่างแล้ว เป็นสัญพจน์ที่เป็นตัวอย่างบวก แต่ถ้าไม่ใช่ตัวอย่างบวกก็จะกำหนดให้ค่าเป้าหมายของนิวรอนนั้นมีค่าเป็น 0

ตัวอย่างเช่น จากตัวอย่างอินพุตในรูปที่ 3.4 เมื่อป้อนตัวอย่างบวก poor(John) ให้กับเน็ตเวิร์กแล้ว จะได้ค่าอินพุตของนิวรอน lazy(x), extravagant(x) และ weak(x) เป็น 1, 1 และ 0 ตามลำดับ เนื่องจากเมื่อแทนตัวแปร x ในสัญพจน์ของแต่ละนิวรอนด้วย John ซึ่งเป็นค่าคงที่ของตัวอย่าง poor(John) แล้ว จะได้แต่ละสัญพจน์มีค่าเป็น lazy(John), extravagant(John) และ weak(John) ซึ่ง lazy(John) และ extravagant(John) เป็นจริงในความรู้ภูมิหลังจึงให้ค่าอินพุตของนิวรอนเป็น 1 ในขณะที่ weak(John) ไม่เป็นจริงในความรู้ภูมิหลัง จึงมีค่าเป็น 0 ส่วนค่าเป้าหมายของนิวรอน poor(x) จะมีค่าเป็น 1 เนื่องจาก poor(John) เป็นตัวอย่างบวก สำหรับตัวอย่างลบทั้งสองตัว คือ poor(Peter) และ poor(Bob) จะได้ค่าอินพุตของนิวรอนเป็น 1,0,1 และ 0,1,1 ตามลำดับ และมีค่าเป้าหมายเป็น 0 ทั้งคู่ ค่าอินพุตของทุกนิวรอนและค่าเป้าหมายของทุกตัวอย่างสรุปได้ดังแสดงในตารางที่ 3.1

ตารางที่ 3.1 ค่าอินพุตและค่าเป้าหมายจากตัวอย่างในรูปที่ 3.4

	lazy(x)	extravagant(x)	weak(x)	**ค่าเป้าหมาย**
poor(John)	1	1	0	1
poor(Peter)	1	0	1	0
poor(Bob)	0	1	1	0

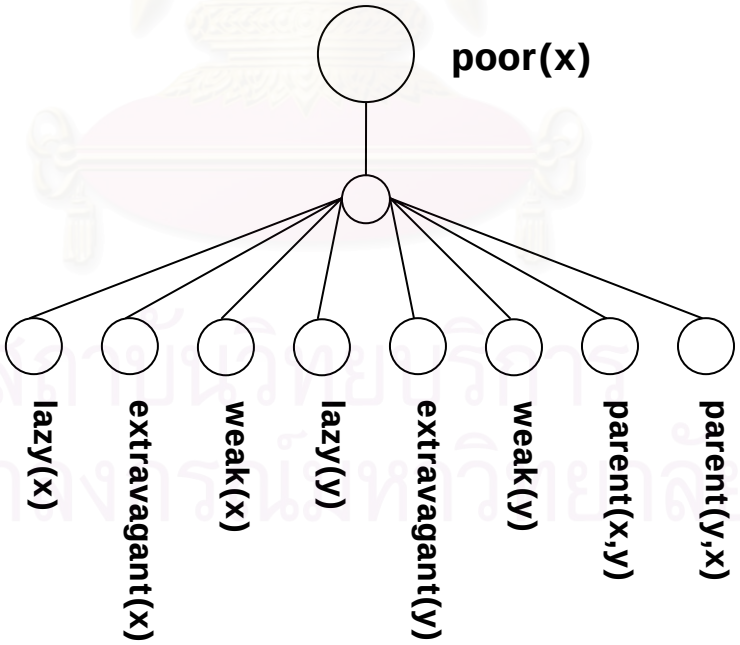
อย่างไรก็ตามเนื่องจากแนวคิดที่เรียนได้จากระบบไอแอลพีอาจมีการสร้างตัวแปรใหม่ขึ้นในสัญพจน์ที่อยู่ในส่วนของตัวกฎโดยที่ตัวแปรนั้นไม่ปรากฏอยู่ในส่วนหัวของกฎ ตัวแปรที่สร้างขึ้นนั้นมีประโยชน์เพื่อนำมาใช้อธิบายความสัมพันธ์ที่เกี่ยวข้อง (relation) กันของแต่ละสัญพจน์ อย่างเช่น แนวคิด mother(x,y) ← father(z,y), husband(z,x) ซึ่งหมายความว่า x จะเป็นแม่ของ y ก็ต่อเมื่อ z เป็นพ่อของ y และ z เป็นสามีของ x ในกรณีนี้จะเห็นว่าตัวแปร z ซึ่งไม่ปรากฏในสัญพจน์เป้าหมาย (mother(x,y)) แต่ถูกสร้างขึ้นเพื่อนำมาใช้ในการอธิบายความสัมพันธ์ของแต่ละสัญพจน์ในส่วนของตัวกฎ และเนื่องจากโดยปกติแล้วตัวอย่างที่รับเข้ามาเพื่อให้ระบบทำการเรียนรู้จะอยู่ในรูปของสัญพจน์เป้าหมายที่มีการแทนค่าของตัวแปรลงไปแล้ว เช่น mother(Jane,Joe) ตัวอย่างนี้มีการแทนค่า Jane และ Joe ลงในตัวแปร x และ y แล้ว ทำให้ไม่สามารถทราบได้ว่าสัญพจน์ที่จะนำมาใช้อธิบายแนวคิดต้องมีการสร้างตัวแปรใหม่เพื่อนำมาใช้อธิบายความเกี่ยวข้องระหว่างสัญพจน์หรือไม่ หรือถ้ามีจะต้องแทนค่าตัวแปรนั้นด้วยอะไร ทำให้

การสร้างอินพุตสำหรับนิวรอนเกิดปัญหาความไม่แน่นอนขึ้น เช่น ถ้าความรู้ภูมิหลังและตัวอย่างที่รับเข้ามามีลักษณะเป็นดังรูปที่ 3.6

- **ความรู้ภูมิหลัง** : lazy(John), **parent(Peter,John)**, lazy(Peter), extravagant(Peter), extravagant(Bob), weak(Bob)
- **ตัวอย่างบวก** : poor(John), poor(Peter)
- **ตัวอย่างลบ** : poor(Bob)

รูปที่ 3.6 อินพุตที่มีสัญพจน์ที่ใช้อธิบายความสัมพันธ์ที่เกี่ยวข้องในความรู้ภูมิหลัง

ในกรณีนี้ความรู้ภูมิหลังมีสัญพจน์ที่เหมือนกับในรูปที่ 3.4 อยู่ 3 ตัว คือ lazy(x), extravagant(x) และ weak(x) และมีสัญพจน์ที่เพิ่มเข้ามาใหม่ คือ parent(x,y) ซึ่งสัญพจน์นี้มีอาร์กิวเมนต์ 2 ตัว และถ้ากำหนดให้ใช้จำนวนตัวแปรเป็น 2 ก็จะสามารถคลี่สัญพจน์นี้ออกมาได้เป็น 4 ตัว คือ parent(x,x), parent(x,y), parent(y,x) และ parent(y,y) ซึ่งสามารถตัดเล็มได้ 2 ตัว คือ parent(x,x) และ parent(y,y) ดังนั้นในกรณีนี้จะได้เน็ตเวิร์กที่มีจำนวนนิวรอนในชั้นนำเข้า 8 นิวรอน และจำนวนนิวรอนในชั้นผลลัพธ์เป็น 1 นิวรอน ดังแสดงในรูปที่ 3.7



รูปที่ 3.7 เน็ตเวิร์กที่สร้างจากอินพุตในรูปที่ 3.6

ในการแปลงอินพุตนั้น นิวรอนของสัญพจน์ lazy(x), extravagant(x) และ weak(x) สามารถหาค่าอินพุตได้โดยใช้วิธีเดียวกันกับที่หาในตัวอย่างที่แล้ว แต่ว่าสำหรับนิวรอนที่เหลืออีก

5 นิวรอน ซึ่งในสัญพจน์มีตัวแปร y ปรากฏอยู่นั้น จะมีปัญหาในการกำหนดค่าอินพุตให้กับนิวรอน เนื่องจากเมื่อเรารับตัวอย่างเข้ามา เราจะทราบแค่เพียงว่าตัวแปรที่ปรากฏในส่วนหัวของกฎต้องแทนที่ด้วยค่าคงที่ใด แต่ไม่ทราบว่าตัวแปรอื่นๆต้องแทนที่ด้วยค่าคงที่ใด เนื่องจากไม่ได้ระบุมาในตัวอย่างที่ป้อนเข้ามา นั่นคือในกรณีนี้เราทราบแค่เพียงว่าตัวแปร x ต้องแทนด้วยค่าคงที่ใด แต่ว่าไม่ทราบว่าตัวแปร y ต้องแทนที่ด้วยค่าคงที่ใดจึงจะเป็นการแทนที่ที่ถูกต้อง เพราะมีค่าคงที่ที่สามารถแทนลงในตัวแปร y ได้หลายตัว

ตัวอย่างเช่น เมื่อเราป้อนตัวอย่างบวก poor(John) ให้กับเน็ตเวิร์ก การหาค่าอินพุตของนิวรอน parent(y,x) จะแทนค่า x ด้วย John แต่ว่าตัวแปร y สามารถแทนได้ด้วย John, Peter หรือ Bob ก็ได้ ถ้าเราแทนที่ตัวแปร y ด้วย Peter ก็จะได้สัญพจน์ parent(y,x) มีค่าเป็น parent(Peter,John) ซึ่งมีค่าความจริงเป็นจริงในความรู้ภูมิหลังและทำให้ได้ค่าอินพุตเป็น 1 แต่ถ้าเราแทนที่ตัวแปร y ด้วย John หรือ Bob ก็จะทำให้ค่าอินพุตเป็น 0 ส่วนกรณีของ parent(x,y) ไม่เกิดปัญหาเนื่องจากเมื่อแทนค่าตัวแปร x ด้วย John แล้วไม่ว่าจะแทนค่าตัวแปร y ด้วยค่าคงที่ใดก็ไม่เป็นจริงในความรู้ภูมิหลัง ทำให้ค่าอินพุตของนิวรอน parent(x,y) เป็น 0 ในทุกกรณี ดังแสดงในตารางที่ 3.2

ตารางที่ 3.2 ค่าอินพุตของนิวรอนเมื่อแทนตัวแปร y ด้วยค่าคงที่ต่างๆ

ค่าอินพุตของนิวรอน	parent(x,y)	parent(y,x)
แทน x ด้วย John แทน y ด้วย John	0	0
แทน x ด้วย John แทน y ด้วย Peter	0	1
แทน x ด้วย John แทน y ด้วย Bob	0	0

จากตัวอย่างข้างต้นแสดงให้เห็นว่าการกำหนดค่าอินพุตให้กับนิวรอนในกรณีที่สัญพจน์ประจำนิวรอนนั้นมีตัวแปรที่ใช้แสดงความสัมพันธ์กับสัญพจน์อื่น โดยตัวแปรนั้นไม่ปรากฏอยู่ในส่วนหัวของกฎด้วย ในบางกรณีจะทำให้ไม่สามารถกำหนดค่าที่แน่นอนให้กับนิวรอนได้ ดังเช่นการกำหนดค่าอินพุตให้กับนิวรอน parent(y,x) เนื่องจากไม่ทราบว่าควรแทนค่าให้กับตัวแปรที่แสดงความสัมพันธ์ด้วยค่าใด เพื่อจัดการกับปัญหานี้จึงได้นำหลักการของการเรียนรู้แบบหลายตัวอย่างย่อย [13] มาประยุกต์ เพื่อทำการสร้างอินพุตให้กับนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง

การเรียนรู้แบบหลายตัวอย่างย่อยจะนิยามตัวอย่างในการเรียนรู้เป็นเซตของถุง โดยที่แต่ละถุงประกอบไปด้วยตัวอย่างย่อย ถุงตัวอย่างจะถูกกำหนดว่าเป็นถุงตัวอย่างบวกก็ต่อเมื่อในถุงนั้นมีตัวอย่างย่อยอย่างน้อย 1 ตัวที่เป็นบวก และจะกำหนดให้เป็นถุงตัวอย่างลบก็ต่อเมื่อตัวอย่างย่อยทุกตัวในถุงนั้นเป็นตัวอย่างลบทั้งหมด ด้วยหลักการนี้เราจะนิยามตัวอย่างในการเรียนรู้ของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งเป็นเซตของตัวอย่าง $\{B_1, B_2, \ldots B_n\}$ โดยที่ n เป็นจำนวนของ

ตัวอย่าง (รวมทั้งตัวอย่างบวกและตัวอย่างลบ) และแต่ละถุงตัวอย่างจะประกอบไปด้วยตัวอย่างย่อย m ตัว $\{B_{i1}, B_{i2}, \ldots, B_{imi}\}$ เมื่อ B_{ij} หรือตัวอย่างย่อยแต่ละตัวเป็นการแทนค่าคงที่ในตัวแปรที่เป็นไปได้แต่ละแบบ ซึ่งจุดนี้เป็นส่วนสำคัญเพราะทำให้เราสามารถนำการแทนค่าตัวแปรทุกกรณีที่เป็นไปได้มาใช้ในการเรียนได้ในรูปแบบของถุงตัวอย่าง

ถุงตัวอย่างจะถูกกำหนดให้เป็นถุงตัวอย่างบวกถ้าตัวอย่างที่นำมาแปลงเป็นถุงตัวอย่างนั้นเป็นตัวอย่างบวก แต่ถ้าแปลงมาจากตัวอย่างลบก็จะได้เป็นถุงตัวอย่างลบ (ในกรณีจำแนกตัวอย่างแบบหลายประเภท จะถือว่าทุกถุงเป็นถุงตัวอย่างบวกของประเภทนั้นๆ) ในการกำหนดค่าเป้าหมายถุงตัวอย่างบวกจะถูกกำหนดให้มีค่าเป็น 1 ส่วนถุงตัวอย่างลบจะมีค่าเป็น 0 ดังแสดงในสมการที่ (4)

ตารางที่ 3.3 แสดงตัวอย่างการแปลงตัวอย่าง poor(John) จากอินพุตในรูปที่ 3.6 ไปเป็นถุงตัวอย่าง โดยอินพุตนี้จะมีถุงตัวอย่างบวกทั้งหมด 2 ถุง คือ poor(John) และ poor(Peter) ถุงตัวอย่างลบ 1 ถุง คือ poor(Bob) ในแต่ละถุงจะประกอบไปด้วยตัวอย่างย่อยทั้งหมด 3 ตัว จากการแทนค่าตัวแปร y ด้วยค่า John, Peter และ Bob และตัวอย่างย่อยแต่ละตัวจะมีขนาดอินพุตเวกเตอร์เท่ากับจำนวนนิวรอนในชั้นนำเข้า (มีขนาดเป็น 8)

ตารางที่ 3.3 การแปลงตัวอย่าง poor(John) ไปเป็นอินพุตให้นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง

ถุงตัวอย่าง ของ poor(John)	**แทน** x **ด้วย** John **แทน** y **ด้วย** John	**แทน** x **ด้วย** John **แทน** y **ด้วย** Peter	**แทน** x **ด้วย** John **แทน** y **ด้วย** Bob
lazy(x)	1	1	1
extravagant(x)	0	0	0
weak(x)	0	0	0
lazy(y)	1	1	0
extravagant(y)	0	1	1
weak(y)	0	0	1
parent(x,y)	0	0	0
parent(y,x)	0	1	0

จากนั้นจึงนำเซตของถุงตัวอย่างที่ได้ไปใส่เป็นอินพุตให้กับเน็ตเวิร์กและทำการเรียนรู้โดยอาศัยวิธีของแบ็คพรอพาเกชัน (Backpropagation Algorithm) สำหรับการเรียนรู้แบบหลายตัวอย่างย่อย [23]

ในกรณีที่ไม่มีการสร้างตัวแปรใหม่ขึ้นในสัญพจน์ที่นำมาใช้อธิบายแนวคิด แต่ละถุงตัวอย่างก็จะประกอบไปด้วยตัวอย่างย่อยเพียง 1 ตัวเท่านั้น และสามารถมองได้ว่าเป็นปัญหาการเรียนรู้แบบสอนแทนการเรียนรู้แบบหลายตัวอย่างย่อยได้

3.4 การปรับค่าน้ำหนักของเส้นเชื่อม

การเรียนรู้ของเน็ตเวิร์กเป็นขั้นตอนสุดท้ายในขั้นตอนการเรียนรู้ ขั้นตอนนี้ทำหน้าที่ปรับค่าน้ำหนักของเส้นเชื่อมให้สามารถจำแนกตัวอย่างได้อย่างถูกต้อง การปรับค่าน้ำหนักจะทำทั้งส่วนที่เชื่อมระหว่างชั้นนำเข้ากับชั้นซ่อน และระหว่างชั้นซ่อนกับชั้นผลลัพธ์ การปรับค่าน้ำหนักจะนำหลักการของแบ็กพรอพาเกชัน มาใช้ร่วมกับฟังก์ชันกระตุ้นองค์ประกอบซิกมอยด์ หลักการของแบ็กพรอพาเกชันเป็นการปรับค่าน้ำหนักโดยมีจุดมุ่งหมายที่จะทำให้ค่าความผิดพลาดกำลังสอง (squared error) รวมระหว่างค่าผลลัพธ์และค่าเป้าหมายมีค่าน้อยที่สุด โดยค่าความผิดพลาดรวม (global error: E) ของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งซึ่งมีนิวรอนชั้นนำเข้า p ตัว นิวรอนชั้นผลลัพธ์ o ตัว และมีชั้นซ่อน 1 ชั้น จะนิยามดังนี้

$$E = \sum_{i=1}^{n} E_i \tag{5}$$

โดยที่ E_i เป็นค่าความผิดพลาดของถุงตัวอย่างแต่ละถุง ซึ่งนิยามโดยขึ้นอยู่กับประเภทของถุงตัวอย่างนั้นๆ คือ

$$E_i = \begin{cases} \min\limits_{1 \le j \le m_i} \sum\limits_{k=1}^{o} E_{ijk} & if\ B_i = + \\ \max\limits_{1 \le j \le m_i} \sum\limits_{k=1}^{o} E_{ijk} & if\ B_i = - \end{cases} \tag{6}$$

และค่าความผิดพลาดของตัวอย่างย่อย (E_{ijk}) แต่ละตัว นิยามดังนี้

$$E_{ijk} = \begin{cases} 0 & if\ (B_i = +) \text{ and } (0.5 \le o_{ijk}), l_{ik} = 1 \\ 0 & if\ (B_i = -) \text{ and } (o_{ijk} < 0.5), for\ all\ k \\ \frac{1}{2}(l_{ik} - o_{ijk})^2 & otherwise \end{cases} \tag{7}$$

เมื่อ B_i=+ หมายถึง ถุงตัวอย่างบวก

B_i=- หมายถึง ถุงตัวอย่างลบ

o_{ijk} หมายถึง ผลลัพธ์ของนิวรอนที่ k ในชั้นผลลัพธ์ที่ได้จากถุงตัวอย่างที่ i ตัวอย่างย่อยที่ j (B_{ij})

l_{ik} หมายถึง ค่าเป้าหมายของนิวรอนที่ k ในชั้นผลลัพธ์ของถุงตัวอย่างที่ i

ค่าความผิดพลาดข้างต้นจะถูกปรับด้วยอัลกอริทึมแบ็คพรอพาเกชันดังที่กล่าวไปแล้วในหัวข้อ 2.1.2.2 โดยในการเรียนแต่ละรอบ ตัวอย่างที่ใช้ในการสอนจะถูกป้อนให้กับเน็ตเวิร์กทีละถุงตัวอย่างทีละตัวอย่างย่อย จากนั้นจะทำการคำนวณหาค่าความผิดพลาดของแต่ละตัวอย่างย่อยในถุงนั้นๆตามสมการ (7) คือถ้าเน็ตเวิร์กจำแนกตัวอย่างได้ถูกต้องแล้ว ก็จะกำหนดให้ค่าความผิดพลาดสำหรับตัวอย่างย่อยนั้นมีค่าเป็น 0 โดยในกรณีที่ถุงตัวอย่างที่ป้อนให้กับเน็ตเวิร์กเป็นถุงตัวอย่างบวกและพบว่ามีตัวอย่างย่อยที่ให้ค่าความผิดพลาดเป็น 0 แล้ว ตัวอย่างย่อยที่เหลือไม่จำเป็นต้องป้อนให้กับเน็ตเวิร์กและไม่ต้องทำการปรับค่าน้ำหนักเส้นเชื่อมใดๆ เพราะถือว่าตัวอย่างย่อยตัวนั้นเป็นตัวอย่างที่ทำให้ถุงตัวอย่างเป็นบวกและเน็ตเวิร์กจำแนกได้ถูกต้องแล้ว แต่ถ้าเน็ตเวิร์กจำแนกผิดพลาดก็จะทำการหาค่าความผิดพลาดสำหรับตัวอย่างย่อยนั้น เพื่อนำไปหาค่าความผิดพลาดของถุงตัวอย่างตามสมการที่ (6) จากนั้นเมื่อป้อนครบทุกตัวอย่างย่อยในถุงแล้วก็จะนำค่าความผิดพลาดของถุงที่ได้ไปทำการปรับค่าน้ำหนักของเส้นเชื่อมตามวิธีของแบ็กพรอพาเกชัน แล้วจึงทำการป้อนถุงตัวอย่างใหม่เข้าไปและทำตามลำดับขั้นตอนเดิม โดยจะหยุดกระบวนการเรียนรู้เมื่อค่าความผิดพลาดรวมในสมการ (5) มีค่าลดลงจนถึงจุดที่กำหนดไว้ หรือจำนวนรอบในการเรียนรู้ครบจำนวนที่ได้กำหนดไว้ เมื่อเสร็จสิ้นกระบวนการเรียนรู้แล้วเน็ตเวิร์กที่ได้สามารถนำไปใช้จำแนกตัวอย่างทดสอบได้

บทที่ 4

การทดลอง

บทนี้กล่าวถึงการทดลองและผลการทดลองที่ได้จากนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง เปรียบเทียบกับระบบ PROGOL ซึ่งเป็นระบบไอแอลพีตัวหนึ่ง โดยการวัดผลจะสนใจที่ค่าเปอร์เซ็นต์ความถูกต้องในการจำแนกตัวอย่างทดสอบ

เนื้อหาในบทนี้จะแบ่งออกเป็น รายละเอียดของข้อมูลที่ใช้ในการทดลอง วิธีการทดลอง จากนั้นจะเป็นส่วนของผลการทดลองและวิเคราะห์ผลการทดลอง ซึ่งได้แบ่งการทดลองออกเป็น 2 ส่วน คือ การทดลองกับชุดข้อมูลปกติและการทดลองกับชุดข้อมูลที่มีสัญญาณรบกวน และส่วนท้ายที่สุดเป็นสรุปผลการทดลองทั้งหมด

4.1 ชุดข้อมูลที่ใช้ในการทดลอง

ชุดข้อมูลหรือกลุ่มของปัญหาที่ใช้ในการทดลองครั้งนี้เป็นชุดข้อมูลที่ใช้ทดสอบระบบไอแอลพี เนื่องจากเราต้องการทดสอบว่านิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งสามารถแก้ปัญหาของระบบไอแอลพีและทำให้เปอร์เซ็นต์ความถูกต้องในการจำแนกสูงขึ้นได้หรือไม่ ซึ่งชุดข้อมูลที่นำมาใช้ทดสอบมี 2 ชุดด้วยกัน คือ การวิเคราะห์ไฟไนต์เอลิเมนต์ (Finite Element Mesh Design: FEM) และการวิเคราะห์ความสามารถในการก่อกลายพันธุ์ (Mutagenesis: MUTA) ซึ่งมีรายละเอียดของแต่ละชุดข้อมูลเป็นดังนี้

4.1.1 การวิเคราะห์ไฟไนต์เอลิเมนต์ (Finite Element Mesh Design: FEM)

วิธีไฟไนต์เอลิเมนต์เป็นวิธีที่ใช้กันอย่างแพร่หลายในงานของวิศวกรและนักออกแบบโครงสร้าง ปัญหา FEM [26] นี้มีจุดมุ่งหมายเพื่อหากฎที่ใช้ในการวิเคราะห์หาไฟไนต์เอลิเมนต์ในโครงสร้างสำหรับงานทางด้านวิศวกรรม ชุดข้อมูล FEM ที่นำมาใช้ทดสอบประกอบไปด้วยชุดตัวอย่างที่เป็นโครงสร้างหลักๆ 5 โครงสร้างและแบ่งประเภทของตัวอย่างเป็น 13 ประเภทด้วยกัน โดยจะแบ่งประเภทตามจำนวนองค์ประกอบ (element) ที่เหมาะสมของโครงสร้างนั้น ตัวอย่างแต่ละตัวถูกจัดอยู่ในรูปแบบ mesh(Edge, Number_of_Element) เมื่อ Edge คือชื่อโครงสร้างและ Number_of_Element คือ จำนวนองค์ประกอบภายในโครงสร้างนั้น รวมจำนวนตัวอย่างทั้งหมด 278 ตัวอย่าง มีกลุ่มความรู้ภูมิหลังซึ่งอธิบายลักษณะต่างๆของโครงสร้างที่มีผลต่อการออกแบบไฟไนต์เอลิเมนต์ ประกอบไปด้วยลักษณะของเส้นเชื่อม เช่น circuit และ short ฯลฯ เงื่อนไข

ขอบเขต เช่น free และ fix ฯลฯ การโหลด เช่น not_loaded และ one_side_loaded ฯลฯ และความสัมพันธ์ระหว่างวัตถุ เช่น neighbour และ opposite ฯลฯ

4.1.2 การวิเคราะห์ความสามารถในการก่อกลายพันธุ์ (Mutagenesis: MUTA)

ปัญหา MUTA [27] เป็นปัญหาการเรียนรู้แบบ 2 ประเภท มีจุดมุ่งหมายเพื่อทำการสร้างกฎมาวิเคราะห์ความสามารถก่อกลายพันธุ์ของโมเลกุล ว่าโมเลกุลหนึ่งๆมีฤทธิ์ (acitve) ในการกลายพันธุ์หรือไม่ ปัญหานี้มีความสำคัญเนื่องจากมีความเกี่ยวข้องกับปัญหาการทำนายการก่อมะเร็งด้วย ชุดข้อมูล MUTA ประกอบไปด้วยกลุ่มตัวอย่างที่เป็นตัวอย่างของข้อมูลของโมเลกุลทั้งหมด 188 โมเลกุล แบ่งเป็น 125 โมเลกุลอยู่ในกลุ่มตัวอย่างบวก (active) และ 63 โมเลกุลอยู่ในกลุ่มตัวอย่างลบ (inactive) ในความรู้ภูมิหลังจะอธิบายลักษณะอะตอมของโมเลกุลหนึ่งๆในรูปแบบ atom(AtomID,Element,Type,Charge) และพันธะเชื่อมต่อระหว่างอะตอมในรูปแบบ bond(Atom1,Atom2,BondType) นอกจากนี้ยังมีคุณสมบัติอื่นๆที่บอกลักษณะภายในโมเลกุลนั้น เช่น ระดับพลังงาน โครงสร้างของโมเลกุล เป็นต้น

4.2 วิธีการทดลอง

ในการทดลองข้อมูลที่นำมาใช้ทดลองจะถูกแบ่งเป็นชุดข้อมูลฝึก (training set) และชุดข้อมูลทดสอบ (test set) ชุดข้อมูลฝึกจะถูกนำไปสอนนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง เพื่อทำการปรับค่าน้ำหนักของเส้นเชื่อมของเน็ตเวิร์กให้เหมาะสมตามขั้นตอนการเรียนรู้ในรูปที่ 3.1 และเมื่อเสร็จสิ้นกระบวนการเรียนรู้แล้ว จึงนำเน็ตเวิร์กที่ได้ไปทดสอบกับชุดข้อมูลทดสอบ การจำแนกชุดข้อมูลทดสอบจะทำตามขั้นตอนการรู้จำดังแสดงในรูปที่ 3.2 โดยประเภทของตัวอย่างจะตัดสินใจเลือกจากนิวรอนในชั้นผลลัพธ์ที่ให้ค่าผลลัพธ์สูงที่สุด

นอกจากนี้ในการทดลองได้ทำการกำหนดค่าพารามิเตอร์ (parameter) เริ่มต้นที่ใช้ในการสอนเน็ตเวิร์กแต่ละครั้งเป็นค่าที่แตกต่างกัน ซึ่งค่าเหล่านี้ประกอบไปด้วย ค่าอัตราการเรียนรู้ ค่าโมเมนตัม จำนวนนิวรอนในชั้นซ่อน และค่าน้ำหนักของเส้นเชื่อมเริ่มต้น

การทดลองเพื่อไม่ให้ผลการทดลองโน้มเอียงตามการแบ่งชุดข้อมูลฝึกและชุดข้อมูลทดสอบ จึงทำการทดลองโดยใช้วิธี k-fold cross validation [1] ทำโดยแบ่งข้อมูลทั้งหมดออกเป็น k ส่วนเท่าๆกันแบบสุ่ม แล้วทำการทดลองทั้งหมด k ครั้ง ในแต่ละครั้งจะเลือกหนึ่งส่วนใดๆเป็นชุดทดสอบและอีก k-1 ส่วนที่เหลือจะถูกนำมาใช้เป็นชุดข้อมูลฝึก ทำให้ข้อมูลทุกตัวจะได้เป็นข้อมูลทดสอบ 1 ครั้ง จากนั้นนำผลการทดลองที่ได้ทั้งหมด k ครั้งมาหาค่าเฉลี่ย ในการทดลองนั้นเมื่อกำหนดค่าพารามิเตอร์เป็นค่าๆหนึ่งแล้ว ก็จะใช้ค่าพารามิเตอร์นั้นๆในการทดลองสำหรับทุก fold ในรอบการทดลองนั้น

การทดลองในงานวิจัยนี้จะแบ่งเป็น 2 ส่วนด้วยกัน คือ ผลการทดลองกับชุดข้อมูลปกติ และผลการทดลองกับชุดข้อมูลที่มีสัญญาณรบกวน ผลการทดลองที่ได้จะเป็นค่าเฉลี่ยจาก 3-fold cross validation โดยจะทำการเปรียบเทียบกันระหว่างผลลัพธ์ของนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งและระบบ PROGOL ซึ่งเป็นระบบการเรียนรู้ตัวหนึ่งที่ได้รับการยอมรับกันอย่างแพร่หลายของระบบไอแอลพี ผลที่แสดงในตารางนั้นจะเลือกจากผลการทดลองที่ให้ค่าเฉลี่ยสูงที่สุดจากการกำหนดค่าพารามิเตอร์หลายๆกรณีทั้งสำหรับนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่งและระบบ PROGOL รายละเอียดและผลการทดลองจะแสดงในหัวข้อต่อๆไป

4.3 ผลการทดลองกับชุดข้อมูลปกติ

การทดลองในส่วนนี้จะทำการวัดค่าเปอร์เซ็นต์ความถูกต้องในการจำแนกจากชุดข้อมูลการวิเคราะห์ไฟไนต์เอลิเมนต์ และการวิเคราะห์ความสามารถในการก่อกลายพันธุ์ โดยใช้ชุดข้อมูลดั้งเดิมไม่ได้แก้ไขตัวข้อมูล รายละเอียดและข้อมูลของชุดข้อมูลทั้งสองสามารถดูได้ที่ http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/applications.html

ผลการทดลองเปรียบเทียบได้ทำการทดสอบค่าทีแบบทางเดียว (one-tailed paired t-test) และได้แสดงระดับความเชื่อมั่นด้วยตัวเลขยกในตาราง ตัวเลข 1, 2, 3 และ 4 แสดงระดับความเชื่อมั่นที่สูงกว่า 90.0%, 98.0%, 99.0% และ 99.5% ตามลำดับ

4.3.1 ผลการทดลองของชุดข้อมูลการวิเคราะห์ไฟไนต์เอลิเมนต์

โครงสร้างของเน็ตเวิร์กที่ใช้จะมีจำนวนนิวรอนในชั้นนำเข้าทั้งหมด 130 นิวรอน กำหนดจากความรู้ภูมิหลังที่ได้รับเข้ามา โดยใช้จำนวนตัวแปร 5 ตัว นิวรอนในชั้นผลลัพธ์ 13 นิวรอน กำหนดจากประเภทของตัวอย่าง และใช้จำนวนนิวรอนในชั้นซ่อน 80 นิวรอน (จากการทดลอง) ค่าเริ่มต้นของน้ำหนักของเส้นเชื่อมกำหนดแบบสุ่ม กำหนดค่าอัตราการเรียนรู้เป็น 0.0001 และค่าโมเมนตัมเป็น 0.97 โดยกระบวนการเรียนรู้จะหยุดเมื่อครบ 6000 รอบหรือค่าความผิดพลาดรวมมีค่าน้อยกว่า 0.05 ผลการทดลองเปรียบเทียบระหว่างนิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง (FOLNN) กับระบบ PROGOL ในตารางที่ 4.1 แสดงให้เห็นว่าระบบ FOLNN ให้ค่าความถูกต้องเป็น 59.18% ซึ่งสูงกว่าระบบ PROGOL ที่ให้ค่าความถูกต้องเป็น 57.80% (ผลการทดลองที่ได้เป็นผลที่ได้จากการเฉลี่ยแล้ว) โดยมีระดับความเชื่อมั่นที่สูงกว่า 90.0%

ตารางที่ 4.1 ผลเปรียบเทียบเปอร์เซ็นต์ความถูกต้องในการทดสอบกับชุดข้อมูล FEM

อัลกอริทึม	เปอร์เซ็นต์ความถูกต้อง
FOLNN	59.18[1]
PROGOL	57.80

4.3.2 ผลการทดลองของชุดข้อมูลการวิเคราะห์ความสามารถในการก่อกลายพันธุ์

การกำหนดโครงสร้างของเน็ตเวิร์กจะกำหนดโดยใช้วิธีเดียวกันกับการทดลองที่แล้ว โครงสร้างของเน็ตเวิร์กที่ใช้จะมีจำนวนนิวรอนในชั้นนำเข้าทั้งหมด 235 นิวรอน นิวรอนในชั้นผลลัพธ์ 1 นิวรอน และใช้จำนวนนิวรอนในชั้นซ่อน 100 นิวรอน ในกรณีนี้ใช้จำนวนตัวแปร 6 ตัวค่าเริ่มต้นของน้ำหนักของเส้นเชื่อมกำหนดแบบสุ่ม กำหนดค่าอัตราการเรียนรู้เป็น 0.0001 และค่าโมเมนตัมเป็น 0.97 ผลการทดลองเปรียบเทียบในตารางที่ 4.2 แสดงให้เห็นว่าระบบ FOLNN ให้ค่าความถูกต้องเป็น 88.27% และยังคงให้ค่าความถูกต้องสูงกว่าระบบ PROGOL ที่ให้ค่าความถูกต้อง 84.58% โดยมีระดับความเชื่อมั่นที่สูงกว่า 98.0%

ตารางที่ 4.2 ผลเปรียบเทียบเปอร์เซ็นต์ความถูกต้องในการทดสอบกับชุดข้อมูล MUTA

อัลกอริทึม	เปอร์เซ็นต์ความถูกต้อง
FOLNN	88.27[2]
PROGOL	84.58

4.4 วิเคราะห์ผลการทดลองกับชุดข้อมูลปกติ

จากผลการทดลองที่ได้ทั้งในชุดข้อมูล FEM และชุดข้อมูล MUTA จะเห็นได้ว่าระบบ FOLNN ให้ค่าเปอร์เซ็นต์ความถูกต้องสูงกว่าระบบ PROGOL ทั้งสองกรณี เนื่องจากแนวคิดที่เรียนได้จากระบบ PROGOL นั้นอยู่ในรูปของกฎ ซึ่งแม้จะมีข้อดีที่อยู่ในรูปแบบที่มนุษย์สามารถเข้าใจได้ แต่ว่าเนื่องจากกฎเหล่านี้เป็นลักษณะของฮอร์นคลอส (Horn clause) ซึ่งเขียนอยู่ในรูป

$$H \leftarrow (L_1 \wedge L_2 \wedge \ldots \wedge L_n)$$

กฎลักษณะนี้จะมีการเชื่อมกันระหว่างประโยคย่อย (clause) ด้วยตัวปฏิบัติการ “และ” (operation “and”) ทำให้มีจุดอ่อนที่เมื่อมีบางประโยคย่อยให้ค่าความจริงที่ผิดไปก็จะทำให้การจำแนกผิดไปด้วย ในขณะที่ระบบ FOLNN ไม่มีปัญหาในจุดนี้เนื่องจากตัวระบบอยู่ในรูปแบบของเน็ตเวิร์ก ซึ่งจะให้ค่าความสำคัญกับแต่ละนิวรอนด้วยค่าน้ำหนักของเส้นเชื่อม สัญพจน์ใดที่มีความสำคัญมากก็จะให้ค่าน้ำหนักของเส้นเชื่อมของนิวรอนที่แทนสัญพจน์นั้นมีค่ามาก ส่วนสัญพจน์ใดที่มีความสำคัญน้อยก็จะให้ค่าน้ำหนักของเส้นเชื่อมของนิวรอนนั้นมีค่าน้อย ทำให้แม้จะมีบางสัญพจน์ที่มีค่าความจริงที่ผิดไปบ้าง แต่ถ้าสัญพจน์นั้นเป็นตัวที่ไม่มีความสำคัญหรือมีความสำคัญไม่มากนักก็จะมีผลกระทบต่อเน็ตเวิร์กเพียงเล็กน้อยและยังมีโอกาสที่จะจำแนกตัวอย่างนั้นได้ถูกต้องอยู่

4.5 ผลการทดลองกับชุดข้อมูลที่มีสัญญาณรบกวน

การทดลองในส่วนนี้มีจุดประสงค์เพื่อทดสอบว่าระบบที่เราพัฒนาขึ้นมีความสามารถในการทนต่อสัญญาณรบกวนได้ดีเพียงใดเมื่อเปรียบเทียบกับระบบไอแอลพีที่มีข้อจำกัดในเรื่องการจำแนกตัวอย่างที่มีสัญญาณรบกวน โดยดูจากค่าเปอร์เซ็นต์ความถูกต้องที่ได้

เพื่อทำการทดลองในส่วนนี้ได้ใช้ชุดข้อมูล MUTA ที่ใช้ในการทดลองที่แล้ว (แบ่งเป็น 3 ส่วนแล้วเพื่อทำ 3-fold cross validation) มาทำการเติมสัญญาณรบกวนลงไปอย่างสุ่มจำนวน 10% และ 15% ที่ประเภทของตัวอย่าง การเติมสัญญาณรบกวนจะทำเฉพาะชุดข้อมูลฝึกเท่านั้น ไม่มีการเติมสัญญาณรบกวนลงในชุดข้อมูลทดสอบ การเติมสัญญาณรบกวน x% อย่างสุ่ม หมายความว่า ในตัวอย่าง 100 ตัวจะมีตัวอย่างอยู่ x ตัวที่ถูกเลือกมาอย่างสุ่ม และทำให้มีค่าของประเภทของตัวอย่างนั้นๆผิดไปจากเดิม

ตารางที่ 4.3 ผลเปรียบเทียบเปอร์เซ็นต์ความถูกต้องในการทดสอบกับชุดข้อมูล MUTA ที่มีสัญญาณรบกวน

Noise level in dataset	PROGOL 0% noise setting	PROGOL 10% noise setting	PROGOL 15% noise setting	FOLNN
10%	64.23	69.72	71.29	84.01[3]
15%	60.56	61.54	65.31	81.28[4]

เนื่องจากระบบ PROGOL มีตัวเลือก (option) ให้สามารถปรับตั้งค่าเพื่อรับมือกับสัญญาณรบกวนได้ด้วย ตัวเลือกนี้เป็นการอนุญาตให้ระบบ PROGOL สร้างกฎที่ครอบคลุมตัวอย่างลบได้ตามจำนวนค่าที่ตั้งไว้ ในการทดลองนี้จึงได้ทำการปรับค่าตัวเลือกเป็นค่าต่างๆ แล้วทำการทดลองและเปรียบเทียบผลการทดลองที่ได้กับระบบ FOLNN ดังแสดงในตารางที่ 4.3 โดยค่า x% noise setting ในตาราง หมายความว่า ได้ทำการปรับค่าตัวเลือกให้กับระบบ PROGOL ให้สามารถเรียนรู้กฎที่มีสัญญาณรบกวนได้ x%

ตารางที่ 4.3 แสดงให้เห็นว่าในกรณีที่ชุดข้อมูลมีสัญญาณรบกวนเป็นจำนวน 10% ระบบ PROGOL จะให้ค่าความถูกต้องสูงที่สุดเป็น 71.29% และระบบ FOLNN ให้ค่าความถูกต้องเป็น 84.01% และในกรณีที่เพิ่มปริมาณสัญญาณรบกวนเป็นจำนวน 15% ระบบ PROGOL จะให้ค่าความถูกต้องสูงที่สุดเป็น 65.31% และระบบ FOLNN ให้ค่าความถูกต้องเป็น 81.28% โดยทั้งสองกรณีระบบ PROGOL ให้ค่าความถูกต้องสูงที่สุดเมื่อปรับค่าการรับมือสัญญาณรบกวนเป็น 15% โดยในกรณีที่มีสัญญาณรบกวน 10% มีระดับความเชื่อมั่นสูงกว่า 99.0% และในกรณีที่มีสัญญาณรบกวน 15% มีระดับความเชื่อมั่นสูงกว่า 99.5%

4.6 วิเคราะห์ผลการทดลองกับชุดข้อมูลที่มีสัญญาณรบกวน

จากผลการทดลองที่ได้ในการทดลองกับชุดข้อมูลที่มีสัญญาณรบกวน จะเห็นได้ว่าค่าเปอร์เซ็นต์ความถูกต้องของระบบ PROGOL ลดลงอย่างมีนัยสำคัญจากกรณีที่ไม่มีสัญญาณรบกวน โดยค่าเปอร์เซ็นต์ความถูกต้องมีค่าลดลงมากขึ้นเมื่อสัญญาณรบกวนมีจำนวนมากขึ้น แม้ว่าระบบ PROGOL จะมีตัวเลือกให้ปรับเพื่อรับมือกับสัญญาณรบกวนได้และสามารถเพิ่มค่าเปอร์เซ็นต์ความถูกต้องให้มากขึ้นได้ตามลำดับเมื่อตั้งค่าการรับมือสัญญาณรบกวนให้มีค่าสูงขึ้น โดยให้ค่าเปอร์เซ็นต์ความถูกต้องสูงที่สุดเมื่อตั้งค่าให้เป็น 15% แต่ก็ยังพบว่ามีค่าเปอร์เซ็นต์ความถูกต้องลดลงไปมากจากกรณีที่ข้อมูลไม่มีสัญญาณรบกวน

สาเหตุที่ทำให้ความถูกต้องในการจำแนกของระบบ PROGOL ลดลงอย่างมากนั้นเนื่องมาจากกฎที่เรียนได้จากระบบไอแอลพีมีข้อจำกัดที่อธิบายความรู้ในลักษณะของฮอร์นคลอสดังที่ได้กล่าวมาแล้วในหัวข้อ 4.4 นอกจากนี้แล้วกฎที่อธิบายในลักษณะนี้ยังไม่สามารถทนต่อข้อมูลที่มีสัญญาณรบกวนได้ เนื่องจากเมื่อมีการเพิ่มสัญญาณรบกวนลงไปในข้อมูลแล้ว โอกาสที่จะทำให้แต่ละประโยคย่อยมีค่าความจริงที่ผิดไปจากเดิมมีมากขึ้น และเป็นผลให้โอกาสในการจำแนกประเภทได้ผิดก็ยิ่งมีสูงมากขึ้นตามไปด้วย

สำหรับค่าเปอร์เซ็นต์ความถูกต้องของระบบ FOLNN แม้ว่าจะมีค่าลดลงจากกรณีที่ทดลองกับชุดข้อมูลที่ไม่มีการเติมสัญญาณรบกวน แต่ว่าก็เป็นการลดลงอย่างช้าๆซึ่งจะเห็นความแตกต่างได้อย่างชัดเจนเมื่อเทียบกับค่าเปอร์เซ็นต์ความถูกต้องของระบบ PROGOL ที่เป็นเช่นนี้เนื่องมาจากระบบ FOLNN เกิดจากการรวมแนวคิดของระบบไอแอลพีเข้ากับนิวรอลเน็ตเวิร์ก ทำให้ระบบ FOLNN มีความสามารถในการทนต่อสัญญาณรบกวนได้ดีของนิวรอลเน็ตเวิร์ก ซึ่งทนได้ดีกว่ากฎที่เรียนได้จากระบบไอแอลพี นอกจากนี้ระบบ FOLNN ยังป้องกันการปรับเหมาะเกินไป (overfitting) กับข้อมูลที่มีสัญญาณรบกวนโดยใช้ความสามารถของนิวรอลเน็ตเวิร์กทำให้คุณสมบัติ (feature) ที่สำคัญมีค่าน้ำหนักของเส้นเชื่อมมากและคุณสมบัติที่ไม่สำคัญก็จะมีค่าน้ำหนักของเส้นเชื่อมน้อย

4.7 สรุปผลการทดลอง

ระบบ FOLNN สามารถรับอินพุตในรูปแบบของตรรกะอันดับที่หนึ่งมาใช้ในการเรียนรู้ได้ เนื่องจากการนำเทคนิคของการเรียนรู้แบบหลายตัวอย่างย่อยมาประยุกต์ใช้ทำให้สามารถแก้ปัญหาในส่วนของอินพุตที่มีค่าไม่แน่นอนเมื่อแทนค่าตัวแปรด้วยค่าคงที่ต่างๆได้

อย่างไรก็ตามในการสอนเน็ตเวิร์กนั้นก็ทำได้ค่อนข้างยากเนื่องจากการแทนค่าตัวแปรด้วยค่าคงที่นั้นมีจำนวนมาก ทำให้ในแต่ละถุงตัวอย่างก็จะมีจำนวนตัวอย่างย่อยมากตามไปด้วย

อัตราส่วนของจำนวนตัวอย่างย่อยที่เป็นบวกต่อจำนวนตัวอย่างย่อยที่เป็นลบก็จะมีค่าน้อย ซึ่งเปรียบเสมือนเป็นการเรียนจากข้อมูลที่มีสัญญาณรบกวนอยู่ เน็ตเวิร์กจึงเรียนได้ยาก นอกจากนี้ยังมีผลให้ใช้เวลาในขั้นตอนการเรียนรู้นานอีกด้วย

ในการทดลองได้ใช้จำนวนนิวรอนในชั้นซ่อนเป็นค่าต่างๆ ผลที่ได้จากการเปลี่ยนจำนวนนิวรอนให้ผลแตกต่างกันเพียงเล็กน้อยเท่านั้น จึงนำเสนอเฉพาะผลที่ดีที่สุดเท่านั้น

ผลการทดลองเปรียบเทียบกับระบบ PROGOL ปรากฏว่าระบบ FOLNN ให้ค่าเปอร์เซ็นต์ความถูกต้องสูงกว่าในทุกๆการทดลองและทุกชุดข้อมูล โดยเฉพาะในกรณีที่มีการเติมสัญญาณรบกวนลงในชุดข้อมูล ระบบ PROGOL ซึ่งไม่ทนต่อข้อมูลในลักษณะนี้มีค่าความถูกต้องลดลงไปอย่างมาก แม้ว่าจะทำการปรับค่าตัวเลือกเพื่อรับมือกับสัญญาณรบกวนแล้วก็ตาม ในขณะที่ FOLNN นั้นให้ค่าความถูกต้องลดลงไปเพียงเล็กน้อยเท่านั้น อันเป็นผลเนื่องมาจากการรวมเข้ากับนิวรอลเน็ตเวิร์ก ทำให้ระบบ FOLNN มีประสิทธิภาพที่สูงขึ้นในการจำแนกข้อมูลแบบหลายประเภทและข้อมูลที่มีสัญญาณรบกวน

บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การโปรแกรมตรรกะเชิงอุปนัยหรือระบบไอแอลพีเป็นระบบที่มีปัญหาในการจำแนกตัวอย่างเมื่อตัวอย่างที่นำมาจำแนกนั้นไม่ตรงกับกฎข้อใดในเซตของกฎที่เรียนรู้มา ทำให้ระบบไอแอลพีไม่สามารถจำแนกตัวอย่างนั้นได้ งานวิจัยนี้จึงนำเสนอวิธีการที่สามารถจัดการกับข้อจำกัดที่เกิดขึ้นในลักษณะนี้ของระบบไอแอลพี โดยนำแนวคิดของระบบไอแอลพีที่สามารถใช้ความรู้ภูมิหลังในการเรียนรู้และอธิบายความรู้ได้กว้างขวาง มนุษย์เข้าใจได้ง่ายมาประยุกต์เข้ากับนิวรอลเน็ตเวิร์กที่สามารถจำแนกตัวอย่างที่มีลักษณะแบบหลายประเภทได้อย่างมีประสิทธิภาพตัวหนึ่งและสามารถทนต่อข้อมูลที่มีสัญญาณรบกวนได้ดี เกิดเป็นระบบการเรียนรู้แบบใหม่ที่ยืดหยุ่นขึ้น มีชื่อว่า นิวรอลเน็ตเวิร์กตรรกะอันดับที่หนึ่ง (First-Order Logical Neural Networks: FOLNNs)

ระบบ FOLNN มีโครงสร้างพื้นฐานเป็นนิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้า (feedforward neural network) แต่ว่ามีข้อดีที่สามารถรับตัวอย่างและความรู้ภูมิหลังที่อยู่ในรูปแบบของโปรแกรมตรรกะมาทำการเรียนรู้ได้โดยตรง ขั้นตอนการเรียนรู้ของระบบ FOLNN จะแบ่งเป็น 3 ขั้นตอน คือ การสร้างเน็ตเวิร์ก การป้อนอินพุตให้กับเน็ตเวิร์กและการปรับค่าน้ำหนักของเส้นเชื่อม โดยในขั้นตอนการสร้างเน็ตเวิร์กจะต้องกำหนดจำนวนตัวแปรที่จะใช้ในส่วนของตัวกฎก่อนแล้วจึงคลี่สัญพจน์ที่เป็นตรรกะอันดับที่หนึ่งออกมาตามจำนวนตัวแปรที่กำหนด เพื่อมาสร้างเป็นนิวรอนในชั้นนำเข้า ขั้นตอนการป้อนอินพุตให้กับเน็ตเวิร์กได้นำวิธีการเรียนรู้แบบหลายตัวอย่างย่อยมาประยุกต์เพื่อแก้ปัญหากรณีอินพุตมีค่าไม่แน่นอนเมื่อแทนค่าตัวแปรด้วยค่าคงที่ต่างๆกัน อินพุตที่ป้อนให้กับเน็ตเวิร์กจะอยู่ในรูปแบบของถุงตัวอย่างที่ประกอบไปด้วยตัวอย่างย่อยที่เป็นการแทนค่าตัวแปรด้วยค่าคงที่ที่เป็นไปได้ทุกกรณี และการปรับค่าน้ำหนักของเส้นเชื่อมจะใช้วิธีของแบ็กพรอพาเกชันที่ประยุกต์ให้เข้ากับอินพุตที่อยู่ในรูปของถุงตัวอย่าง เมื่อเสร็จสิ้นกระบวนการเรียนรู้แล้ว เน็ตเวิร์กที่ได้สามารถนำไปใช้จำแนกประเภทของตัวอย่างทดสอบได้

ในการทดลองเพื่อทดสอบแนวคิดที่นำเสนอ ได้ทำการทดลองกับชุดข้อมูลของระบบไอแอลพี 2 ชุด คือ การวิเคราะห์ไฟไนต์เอลิเมนต์ และการวิเคราะห์ความสามารถในการก่อการกลายพันธุ์ เปรียบเทียบผลที่ได้กับระบบ PROGOL โดยการทดลองจะแบ่งเป็น 2 ส่วน คือ ทดลองกับชุดข้อมูลปกติและทดลองกับชุดข้อมูลที่เติมสัญญาณรบกวน จากผลการทดลองในชุดข้อมูล

ปกติพบว่าระบบ FOLNN ให้ค่าเปอร์เซ็นต์ความถูกต้องสูงกว่าระบบ PROGOL ทั้งในชุดข้อมูล FEM และชุดข้อมูล MUTA และในการทดลองกับชุดข้อมูลที่มีการเติมสัญญาณรบกวนลงไป ระบบ FOLNN ยังคงให้เปอร์เซ็นต์ความถูกต้องสูงกว่าระบบ PROGOL โดยเมื่อเปรียบเทียบผลการทดลองกับกรณีที่ชุดข้อมูลไม่มีสัญญาณรบกวนพบว่าระบบ PROGOL ให้ค่าเปอร์เซ็นต์ความถูกต้องลดลงไปอย่างมีนัยสำคัญ ในขณะที่ระบบ FOLNN มีค่าเปอร์เซ็นต์ความถูกต้องลดลงไปเพียงเล็กน้อยเท่านั้น ทั้งนี้เนื่องมาจากระบบ FOLNN มีข้อดีที่ได้รับจากนิวรอลเน็ตเวิร์ก คือ สามารถทนต่อข้อมูลที่มีสัญญาณรบกวนได้ดี และช่วยกำหนดความสำคัญให้กับแต่ละคุณสมบัติได้ด้วยค่าน้ำหนักของเส้นเชื่อม

งานวิจัยนี้แสดงให้เห็นว่าระบบ FOLNN ที่นำเสนอนั้นสามารถจัดการกับข้อจำกัดของระบบไอแอลพีได้และช่วยเพิ่มความแม่นยำในการจำแนกตัวอย่างให้สูงขึ้น นอกจากนี้วิธีที่นำเสนอนั้นยังมีจุดเด่นที่สามารถรับอินพุตที่อยู่ในรูปแบบของตรรกะอันดับที่หนึ่งมาใช้ในการเรียนรู้ได้โดยตรง ไม่จำเป็นต้องพึ่งระบบไอแอลพี

5.2 **ข้อเสนอแนะ**

1. งานวิจัยนี้สนใจแต่ในส่วนของการแปลงข้อมูลตรรกะมาเป็นเน็ตเวิร์กและทำการสอนเน็ตเวิร์กเท่านั้น ไม่ได้มุ่งเน้นทำการวิจัยเรื่องถอดเน็ตเวิร์กกลับไปเป็นกฎ (extracting rule) อย่างละเอียด อย่างไรก็ตามผู้วิจัยได้ทำการทดลองในส่วนนี้ไปเล็กน้อย และจะนำเสนอรายละเอียดในหัวข้อ 5.3 สำหรับข้อดีของการแปลงเน็ตเวิร์กกลับไปเป็นกฎนั้นจะช่วยให้สามารถนำแนวคิดที่เรียนได้มาประยุกต์ใช้กับงานต่างๆได้ง่ายขึ้น ตัวอย่างงานวิจัยในส่วนนี้ได้แก่ [18, 20, 28, 29]

2. จากผลการทดลองจะเห็นได้ว่าระบบ FOLNN เป็นระบบการเรียนรู้สำหรับตรรกะอันดับที่หนึ่งที่ให้ค่าความแม่นยำสูง อย่างไรก็ตามในขั้นตอนการสร้างเน็ตเวิร์กนั้น FOLNN ใช้จำนวนนิวรอนในชั้นนำเข้าค่อนข้างสูง โดยจำนวนนิวรอนสำหรับ 1 สัญพจน์มีจำนวนเท่ากับ

   จำนวนตัวแปร$^{\text{จำนวนพารามิเตอร์ของสัญพจน์}}$

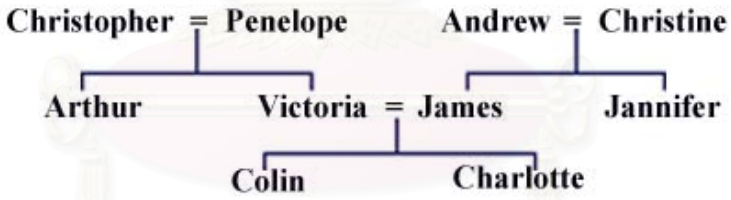
   ซึ่งถ้ามีการกำหนดตัวแปรที่จะใช้ในการเรียนให้มีค่ามาก หรือถ้าจำนวนพารามิเตอร์ของสัญพจน์มีค่ามาก ก็จะทำให้จำนวนนิวรอนยิ่งมีค่าเพิ่มมากตามไปด้วย โดยมีลักษณะการเพิ่มเป็นแบบฟังก์ชันเลขชี้กำลัง (exponential function) ผู้วิจัยได้

นำเสนอโครงสร้างของเน็ตเวิร์กอีกแบบหนึ่งซึ่งมีการใช้จำนวนนิวรอนให้น้อยลงจากเดิม โดยจะสรุปรายละเอียดไว้ในหัวข้อ 5.4

3. แม้ว่าระบบ FOLNN เป็นระบบที่ออกแบบมาสำหรับการเรียนรู้ตรรกะอันดับที่หนึ่ง แต่ว่าก็สามารถที่จะนำไปใช้กับการเรียนรู้ตรรกศาสตร์ประพจน์ได้ และน่าจะช่วยให้จำแนกข้อมูลที่มีค่าในบางคุณสมบัติหายไป (missing value) ได้ดีขึ้น ด้วยวิธีการของการเรียนรู้แบบหลายตัวอย่างย่อยใน FOLNN
4. การทดลองในงานวิจัยนี้ใช้เวลาค่อนข้างนานในกระบวนการเรียนรู้ของเน็ตเวิร์ก ถ้ามีการนำเทคนิคการโปรแกรมเชิงขนาน (parallel programming) หรือ การคำนวณแบบกระจาย (distributed computing) มาช่วย น่าจะทำให้ใช้เวลาน้อยลง

5.3 การถอดเน็ตเวิร์กกลับไปเป็นกฎ

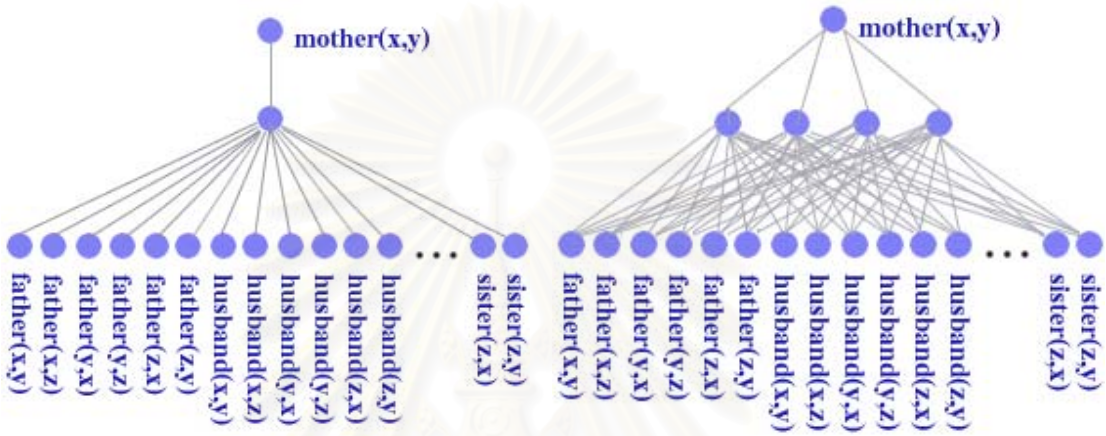
ในการทดลองในส่วนนี้ผู้วิจัยได้สร้างตัวอย่างขึ้นมาเองเพื่อใช้ในการเรียนรู้แนวคิด mother(x,y) โดยในความรู้ภูมิหลังมีเพรดิเคต (predicate) ที่ใช้อธิบายแนวคิดที่ต้องการเรียนอยู่ 4 ตัว คือ father, husband, grandmother และ sister ตัวอย่างและความรู้ภูมิหลังที่ป้อนให้กับเน็ตเวิร์กสอดคล้องกับสายความสัมพันธ์ในรูปที่ 5.1 ซึ่งถ้าเรียนรู้ด้วยระบบไอแอลพีจะได้กฎออกมาเป็น mother(x,y) ← father(z,y), husband(z,x)



รูปที่ 5.1 สายความสัมพันธ์ของครอบครัวหนึ่ง

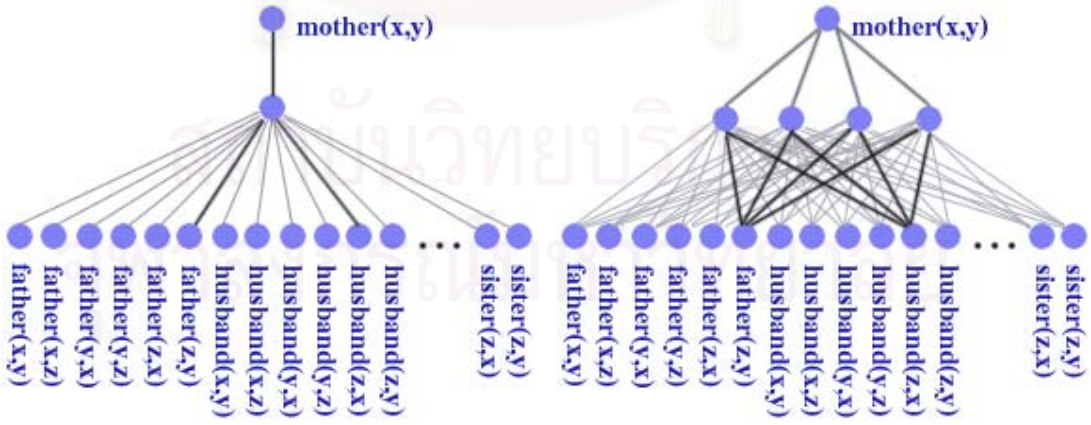
ในสายความสัมพันธ์ เมื่อ A=B หมายถึง A แต่งงานกับ B และจากสายสัมพันธ์นี้จะมีตัวอย่างบวกทั้งหมด 6 ตัวด้วยกัน คือ mother(Penelope,Arthur), mother(Penelope,Victoria), mother(Christine,James), mother(Christine,Jannifer), mother(Victoria,Colin) และ mother(Victoria,Charlotte) โดยที่แต่ละตัวสามารถแปลงเป็นถุงตัวอย่างบวกที่มีตัวอย่างย่อยจำนวน 10 ตัว ที่มาจากการแทนค่าตัวแปร z ทุกกรณีที่เป็นไปได้ (ใช้ตัวแปรจำนวน 3 ตัว) สำหรับตัวอย่างลบก็จะเป็นกรณีอื่นๆที่ไม่เป็นจริง เช่น mother(Christopher,Penelope), mother (Penelope,Colin), mother(Victoria,Jannifer) เป็นต้น โดยถุงตัวอย่างลบที่ใช้ในการสอนจะมาจากตัวอย่างลบทุกตัวที่เป็นไปได้

สำหรับสัญพจน์ที่นำมาสร้างนิวรอนในชั้นนำเข้าจะมาจากสัญพจน์ที่อธิบายในความรู้ภูมิหลังทั้ง 4 ตัว โดยเมื่อกำหนดตัวแปรที่ใช้เป็น 3 ตัว จะสามารถคลี่แต่ละสัญพจน์ออกมาได้เป็น 9 กรณี คือ สัญพจน์สำหรับ (x,x), (x,y), (x,z), (y,x), (y,y), (y,z), (z,x), (z,y) และ (z,z) แต่ว่าสามารถตัดเล็มทิ้งได้ 3 กรณีคือ (x,x), (y,y) และ (z,z) ทำให้เน็ตเวิร์กที่สร้างได้จะมีจำนวนนิวรอนในชั้นนำเข้าเป็น 24 นิวรอน จำนวนนิวรอนในชั้นผลลัพธ์จำนวน 1 นิวรอนและกำหนดจำนวนนิวรอนในชั้นซ่อนเป็น 1 นิวรอนและ 4 นิวรอนดังแสดงในรูปที่ 5.2



รูปที่ 5.2 เน็ตเวิร์กสำหรับเรียนแนวคิด mother(x,y)

หลังจากที่ผ่านการสอนเน็ตเวิร์ก เน็ตเวิร์กทั้งคู่สามารถจำแนกตัวอย่างได้อย่างถูกต้องและพบว่าค่าน้ำหนักของเส้นเชื่อมของนิวรอนในบางสัญพจน์มีค่าเพิ่มสูงมากขึ้นแตกต่างจากนิวรอนอื่นๆ โดยจะแสดงค่าน้ำหนักเส้นเชื่อมที่มีค่ามากด้วยเส้นเชื่อมสีดำเข้ม และเส้นเชื่อมที่มีค่าน้ำหนักไม่สูงจะแสดงด้วยเส้นเชื่อมสีอ่อน ดังรูปที่ 5.3



รูปที่ 5.3 เน็ตเวิร์กที่ผ่านการสอนแล้ว

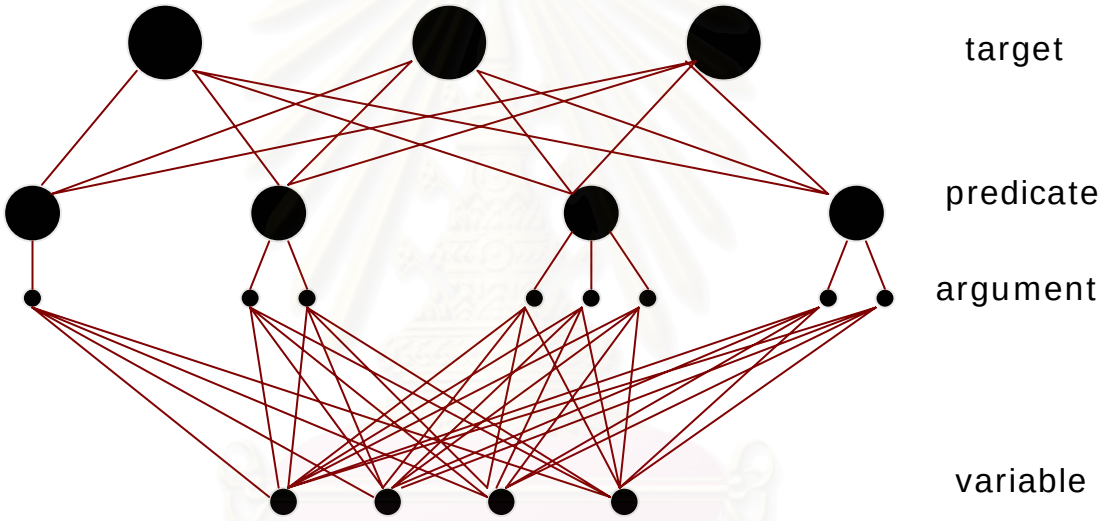
การเรียนของเน็ตเวิร์กเมื่อสัญพจน์ใดที่มีความสำคัญมาก นิวรอนประจำสัญพจน์นั้นก็จะมีค่าน้ำหนักของเส้นเชื่อมสูง ซึ่งเมื่อสังเกตจะพบว่านิวรอนที่มีความสำคัญมากนั้นเป็นตัวแทน

ของสัญพจน์ father(z,y) และ husband(z,x) ในเน็ตเวิร์กทั้งคู่ เมื่อเรามองกลับไปเป็นกฎโดยง่าย แล้วก็จะพบว่าแนวคิดหรือกฎที่เรียนได้จากเน็ตเวิร์กมีความหมายว่า mother(x,y) ← father(z,y), husband(z,x) ซึ่งตรงกับกฎที่เรียนได้จากระบบไอแอลพี

อย่างไรก็ตามในกรณีนี้เนื่องจากเน็ตเวิร์กที่ใช้ไม่ค่อยมีความซับซ้อน จึงทำให้การถอดกลับมาเป็นกฎทำได้ไม่ยากนัก แต่ถ้าเน็ตเวิร์กมีความซับซ้อนมากขึ้นก็จะทำให้ถอดกลับมาเป็นกฎได้ยากขึ้นและคงต้องใช้อัลกอริทึมที่มีประสิทธิภาพมากขึ้นในการแปลงเน็ตเวิร์ก

5.4 โครงสร้างใหม่ของ FOLNN

ในการพัฒนาโครงสร้างใหม่ของ FOLNN นั้น มีจุดประสงค์เพื่อทำให้อัตราการเติบโตของจำนวนนิวรอนในชั้นนำเข้ามีค่าลดลง โดยโครงสร้างที่พัฒนาขึ้นใหม่แสดงดังรูปที่ 5.4



รูปที่ 5.4 โครงสร้างใหม่ของ FOLNN

โครงสร้างใหม่นี้แบ่งเน็ตเวิร์กออกเป็น 4 ชั้นด้วยกัน คือ ชั้นตัวแปร ชั้นอาร์กิวเมนต์ ชั้นเพรดิเคตและชั้นเป้าหมาย การออกแบบโครงสร้างใหม่นี้ให้ชั้นเป้าหมายแทนแนวคิดที่ต้องการเรียนเช่นเดียวกับในโครงสร้างเก่า ถัดมาคือชั้นเพรดิเคต ที่ใช้อธิบายแนวคิดที่ต้องการเรียน แต่ในชั้นนี้จะแตกต่างจากโครงสร้างเก่าตรงที่ในโครงสร้างเก่านั้นจะเป็นลักษณะที่เป็นสัญพจน์ คือ มีตัวแปรรวมอยู่ด้วย เช่น father(x,y) และ sister(z,x) เป็นต้น ซึ่งแจกแจงออกมาทุกกรณีตามจำนวนตัวแปรที่ใช้ในการเรียน ในขณะที่ชั้นเพรดิเคตในโครงสร้างนี้จะอธิบายเฉพาะเพรดิเคตเท่านั้นไม่ได้รวมตัวแปรเข้าไปด้วย เช่น father และ sister เป็นต้น โดยจะแยกส่วนของตัวแปรไว้ในอีกชั้นนึง การออกแบบเช่นนี้เพื่อลดจำนวนนิวรอนให้น้อยลงกว่าโครงสร้างเดิม สำหรับตัวแปรที่จะใช้อธิบายร่วมกับเพรดิเคตจะแยกมาอยู่ที่ชั้นล่างสุด ซึ่งตัวแปรทุกตัวจะใช้ร่วมกันกับทุกเพรดิเคต

นอกจากนี้เพื่อให้การใช้เพรดิเคตเป็นไปอย่างสมบูรณ์ จึงได้สร้างชั้นของอาร์กิวเมนต์เพิ่มขึ้นมาอีกหนึ่งชั้น ชั้นนี้มีเพื่อไว้ช่วยกำหนดลำดับของตัวแปรในสัญพจน์ เนื่องจากถ้ามีแต่เพียงชั้นของเพรดิเคตและชั้นของตัวแปรเท่านั้น จะทำให้เกิดความสับสนในการใช้สัญพจน์ ตัวอย่างเช่น ถ้ามีเพรดิเคต father และตัวแปร x และ y เมื่อเราเชื่อมต่อนิวรอนของเพรดิเคตเข้ากับนิวรอนของตัวแปรทั้งสองโดยตรง เราจะไม่สามารถทราบได้ว่าต้องการจะอธิบายสัญพจน์ใดระหว่างสัญพจน์ father(x,x), father(x,y), father(y,x) และ father(y,y) เนื่องจากเราไม่ทราบลำดับของตัวแปร ซึ่งชั้นของอาร์กิวเมนต์จะช่วยแก้ปัญหานี้ได้ แต่ว่าเส้นเชื่อมที่ต่อระหว่างชั้นอาร์กิวเมนต์และชั้นเพรดิเคตจะไม่มีค่าน้ำหนักประจำเส้นเชื่อมเพราะว่าแค่ทำหน้าที่เป็นตัวบอกลำดับให้เพรดิเคตเท่านั้น

หน้าที่ของแต่ละชั้นในโครงสร้างใหม่สามารถจะนิยามโดยสรุปได้ดังนี้

1. ชั้นตัวแปร ชั้นนี้จะแสดงจำนวนตัวแปรที่ผู้สอนกำหนดเพื่อใช้ในการเรียนรู้ โดยเส้นเชื่อมจากนิวรอนในชั้นนี้จะเชื่อมต่อไปยังทุกนิวรอนในชั้นอาร์กิวเมนต์
2. ชั้นอาร์กิวเมนต์ เป็นชั้นที่ไว้กำหนดลำดับของตัวแปรในแต่ละเพรดิเคต จำนวนนิวรอนในส่วนนี้จะแยกเป็นนิวรอนของแต่ละเพรดิเคต ซึ่งแต่ละเพรดิเคตจะมีจำนวนนิวรอนเท่ากับจำนวนอาร์กิวเมนต์ของเพรดิเคตนั้นๆ
3. ชั้นเพรดิเคต เป็นชั้นที่แสดงถึงเพรดิเคตต่างๆที่จะนำมาใช้เป็นตัวแทนของแนวคิดหรือกฎที่ต้องการเรียน จำนวนนิวรอนในชั้นนี้จะขึ้นอยู่กับจำนวนเพรดิเคตในความรู้ภูมิหลังที่รับเข้ามาในการเรียนรู้
4. ชั้นเป้าหมาย ชั้นนี้เปรียบเทียบได้กับชั้นผลลัพธ์ในโครงสร้างเก่า ชั้นเป้าหมายจะแสดงถึงแนวคิดที่ต้องการเรียนรู้ โดยชั้นนี้จะคำนวณหาผลลัพธ์สุดท้ายของเน็ตเวิร์กเพื่อใช้จำแนกตัวอย่าง จำนวนนิวรอนในชั้นนี้จะขึ้นอยู่กับจำนวนแนวคิดทั้งหมดที่ต้องการเรียนหรือจำนวนกลุ่มที่ต้องการจำแนก

ในการทดลองเบื้องต้นเพื่อทดสอบประสิทธิภาพของโครงสร้างแบบใหม่ของ FOLNN นั้นได้ทำการทดสอบกับชุดข้อมูล FEM โดยวัดผลการทดลองเพียงแค่ 1 fold เท่านั้น (ผลการทดลองปกติที่แสดงในงานวิจัยนี้จะเฉลี่ยจากการทดลอง 3 fold) พบว่าค่าเปอร์เซ็นต์ความถูกต้องในการจำแนกด้วยโครงสร้างใหม่มีค่าลดลงจากผลที่ได้ในการทดลองด้วยโครงสร้างเก่า โดยให้ค่าความถูกต้องอยู่ในช่วงประมาณ 40-50 เปอร์เซ็นต์เท่านั้น จากเดิมที่มีค่าเปอร์เซ็นต์ความถูกต้องเป็น 59.18

ที่ผลการทดลองออกมาเป็นดังนี้เนื่องมาจากว่าในโครงสร้างใหม่มีการลดรูปการกระจายออกมาเป็นสัญพจน์ให้เก็บไว้อยู่ในรูปของเพรดิเคตและตัวแปรแทน ทำให้มีข้อมูลบางส่วนหายไป นอกจากนี้ในส่วนของการปรับค่าน้ำหนักเส้นเชื่อมก็พบว่าในการปรับค่าน้ำหนักเส้นเชื่อมสำหรับนิวรอนในบางสัญพจน์นั้นกลับไปมีผลกระทบกับนิวรอนประจำสัญพจน์อื่นที่ไม่ได้ปรับค่าน้ำหนักไปด้วย ตัวอย่างเช่น ถ้าต้องการปรับค่าน้ำหนักของสัญพจน์ father(x,y) ซึ่งต้องไปปรับค่าน้ำหนักของเส้นเชื่อมที่เชื่อมระหว่างตัวแปร x กับอาร์กิวเมนต์ตัวแรกของเพรดิเคต father และเส้นเชื่อมที่เชื่อมระหว่างตัวแปร y กับอาร์กิวเมนต์ตัวที่สองของเพรดิเคต อย่างไรก็ตามการปรับค่าน้ำหนักในลักษณะนี้กลับไปมีผลกระทบต่อสัญพจน์อื่นๆ เช่น father(x,z) และ father(z,y) ด้วย ที่เป็นเช่นนี้เพราะเมื่อเราปรับค่าน้ำหนักของ father(x,y) ที่ปรับค่าของเส้นเชื่อมที่เชื่อม x กับ อาร์กิวเมนต์ตัวแรก สัญพจน์อื่นๆที่มี x เป็นอาร์กิวเมนต์ตัวแรกย่อมจะได้รับการปรับค่าน้ำหนักตามไปด้วย เช่นเดียวกันกับกรณีของตัวแปร y ที่จะกระทบกับสัญพจน์ที่มีตัวแปร y เป็นอาร์กิวเมนต์ตัวที่ 2

โครงสร้างใหม่ที่นำเสนอนี้แม้ว่าจะลดจำนวนนิวรอนที่ใช้ในการเรียนรู้ได้แต่ว่าผลลัพธ์ที่ได้ในการจำแนกยังไม่เป็นที่น่าพอใจสักเท่าไร ซึ่งอาจจะเป็นเพราะข้อจำกัดของโครงสร้างใหม่ที่ถูกลดรูปไปจนอาจทำให้ไม่สามารถเรียนแนวคิดที่มีความซับซ้อนได้ หรืออาจจะเป็นเพราะปัญหาในการปรับค่าน้ำหนักเส้นเชื่อมที่มีการใช้เส้นเชื่อมร่วมกัน ซึ่งถ้าสามารถหาวิธีในการปรับค่าน้ำหนักที่เหมาะสมกับโครงสร้างแบบนี้ได้ก็อาจจะทำให้ได้ระบบการเรียนรู้ที่มีประสิทธิภาพ ลดอัตราการโตของจำนวนนิวรอนและมีลักษณะการเรียนรู้ที่เป็นตรรกะอันดับที่หนึ่งมากกว่าเดิม

รายการอ้างอิง

[1] Mitchell, T.M. Machine Learning. New York: McGraw-Hill, 1997.

[2] Roberts, S. An Introduction to Progol. Technical Manual. University of York, 1997.

[3] Muggleton, S.H. and Feng, C. Efficient Induction of Logic Programs. In S.H. Muggleton (ed.), Inductive Logic Programming. Academic Press, 1992.

[4] Quinlan, J.R. Learning Logical Definition from Relations. Machine Learning. 5 (1990): pp. 239-266.

[5] Lavrac, N. and Dzeroski, S. Inductive Logic Programming Techniques and Applications. New York: Ellis Horwood, 1994.

[6] Nienhuys-Cheng, S.-H. and Wolf, R.d. Foundation of Inductive Logic Programming. New York: Springer-Verlag, 1997.

[7] Quinlan, J.R. C4.5:Programs for Machine Learning. Machine Learning. San Francisco: Morgan Kaufmann Publishers Inc., 1993.

[8] Blockeel, H. and Raedt, L.D. Experiments with Top-down Induction of Logical Decision Trees. Technical Report CW247. Department of Computer Sciences, K.U.Leuven, 1997.

[9] Lavrac, N., Dzeroski, s., and Grobelnik. Learning Nonrecursive Definitions of Relations with LINUS. Proceedings of the European Working Session on Learning. Porto, Portugal, 1991.

[10] Bishop, C.M. Neural Networks for Pattern Recognition. Oxford University Press, 1995.

[11] Chatfield, C. and Collins, A.J. Introduction to Multivariate Analysis. London: Chapman and Hall, 1980.

[12] Kaelbling, L.P., Littman, M.L., and Moore, A.W. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research. 4 (1996).

[13] Huang, X., Chen, S.-C., and Shyu, M.-L. An Open Multiple Instance Learning Framework and Its Application in Drug Activity Prediction Problems. Proceedings of the Third IEEE Symposium on BioInformatics and BioEngineering (BIBE'03). Bethesda, Maryland, 2003.

[14] Maron, O. and Lozanon, T. A Framework for Multiple-Instance Learning. Neural Information Processing Systems, pp. 570-576. MIT Press, 1998.

[15] Dietterich, T.G., Lathorp, R.H., and Lozano-Perez, T. Solving the Multiple-Instance Problem with Axis-Parallel Rectangles. Artificial Intelligence. 89 (1997): pp. 31-71.

[16] Bratko, I. and Muggleton, S.H. Applications of Inductive Logic Programming. Communications of the ACM, pp. 65-70. 1995.

[17] Ultsch, A. and Korus, D. Integration of neural networks with knowledge-based systems. Neural Networks, 1995. Proceedings., IEEE International Conference on, pp. 1828-1833, 1995.

[18] Towell, G.G. and Shavlik, J.W. Knowledge-based artificial neural networks. Artificial Intelligence, pp. 119-165. 1994.

[19] Parekh, R. and Honavar, V. Constructive Theory Refinement in Knowledge Based Neural Networks. Proceedings of the International Joint Conference on Neural Networks, pp. 2208-2213. Anchorage, Alaska, 1998.

[20] Garcez, A.S.d.A., Broda, K.B., and Gabbay, D.M. Neural-Symbolic Learning Systems. Perspectives in Neural Computing. London: Springer-Verlag, 2002.

[21] Kijsirikul, B., Sinthupinyo, S., and Chongkasemwongse, K. Approximate Match of Rules Using Backpropagation Neural Networks. Machine Learning. 44 (2001): pp. 273-299.

[22] Andrews, S., Tsochantaridis, I., and Hofmann, T. Support Vector Machines for Multiple-Instance Learning. Advances in Neural Information Processing Systems (NIPS 2002), 2002.

[23] Zhou, Z.-H. and Zhang, M.-L. Neural Network for Multi-Instance Learning. Proceedings of the International Conference on Intelligent Information Technology, pp. 455-459. Beijing, China, 2002.

[24] Wattuya, P., Rungsawang, A., and Kijsirikul, B. Multiple-Instance Neural Network with Strong Boundary Criteria. The 7th National Computer Science and Engineering Conference. Chonburi, Thailand, 2003.

[25] Rumelhart, D.E., Hinton, G.E., and Williams, R.J. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland (eds.), Parallel Distributed Processing. Cambridge, MA: The MIT Press, 1986.

[26] Dolsak, B. and Muggleton, S. The Application of Inductive Logic Programming to Finite Element Mesh Design. In S. Muggleton (ed.), Inductive Logic Programming, pp. 453--472. Academic Press, 1992.

[27] Srinivasan, A., Muggleton, S.H., Sternberg, M.J.E., and King, R.D. Theories for mutagenicity: a study in first-order and feature-based induction. Artificial Intelligence, pp. 277-299. Elsevier Science Publishers Ltd., 1996.

[28] Andrew, R., Diederich, J., and Tickle, A.B. Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks. Knowledge-Based Systems, pp. 373-389. 1995.

[29] Craven, M.W. Extracting Comprehensible Models from Trained Neural Networks. Doctoral dissertation, Department of Computer Science, University of Wisconsin-Madison, 1996.

ประวัติผู้เขียนวิทยานิพนธ์

นายธนุพล เลิศลำเนาชัย เกิดเมื่อวันพุธที่ 29 กันยายน พ.ศ. 2525 ที่จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2546 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ปีการศึกษา 2546