บทที่ 2
ผลงานวิจัยที่ผ่านมา

บทนี้จะกล่าวถึงประวัติของการพัฒนาข่ายงานนิวรัลและผลงานวิจัยที่ผ่านมาของอัลกอริธึมที่จะทำการวิจัยและผลงานทางด้านข่ายงานนิวรัลที่เกี่ยวกับวิศวกรรมเคมี ซึ่งเป็นการประยุกต์ใช้ในกระบวนการผลิตต่างๆ และสุดท้ายกล่าวถึงผลงานข่ายงานนิวรัลที่เกี่ยวกับงานวิจัยครั้งนี้

2.1 จุดกำเนิดผลงานด้านข่ายงานนิวรัล

ค.ศ. 1943 Warren McCulloch และ Walter Pitts [1] ได้เป็นผู้บุกเบิกในการออกแบบข่ายงานนิวรัลแบบแรก โดยใช้ตรรกเทร์สโฮลด์ (threshold logic) ด้วยหลายอินพุทและหนึ่งเอาท์พุท นั้นคือเอาท์พุทมีค่าเท่ากับ 0 ถ้าระบบเซลประสาทไม่ทำงาน หรือมีค่าเท่ากับ 1 ถ้าเซลล์ประสาททำงาน [1],[2] นิวรัลจะทำงานได้ต่อเมื่อผลรวมของอินพุทเกินค่าเทรสโฮตส์ที่กำหนดไว้ ในรูปของฟังก์ชัน f(x) = 1 ถ้า x มีค่ามากกว่าเทรสโฮตส์ และ f(x) = 0 ถ้า x มีค่าน้อยกว่าเทรสโฮตส์ (ฟังก์ชันนี้จะเรียกว่าเป็นฟังก์ชันในการบ่งชี้, Indicator function) ซึ่งข่ายงานนี้สามารถใช้ได้กับระบบเชิงเส้นเท่านั้นเนื่องจากมีเพียงนิวรัลเพียงหน่วยเดียวไม่มีการเชื่อมโยงกับนิวรัลอื่น และค่าน้ำหนักถูกกำหนดให้มีค่าคงที่ด้วย

ค.ศ. 1949 Donald Hebb [3] ได้สร้างกฎการเรียนรู้สำหรับข่ายงานนิวรัล คือถ้านิวรัล 2 หน่วยสามารถทำงานได้เหมือนกัน จะทำให้ค่าน้ำหนักที่เป็นสัญญาณในการเชื่อมต่อจะดีขึ้น

ค.ศ. 1959 Frank Rosenblatt [4] ได้พัฒนาแบบจำลองของ McCulloch and Pitts ต่อ ข่ายงานนี้มีหน่วยที่เรียกว่าเปอร์เซปตรอน (perceptron) การเรียนรู้จะมีการปรับค่าน้ำหนักจนได้ค่าน้ำหนักที่ถูกต้อง ซึ่งจะดีกว่ากฎของ Hebb และข่ายงานนี้มีชั้นของน้ำหนักเพียงชั้นเดียว ฟังก์ชันมูลฐานที่ใช้เป็นฟังก์ชันเชิงเส้น และฟังก์ชันกระตุ้นเป็นแบบขั้นบันไดซึ่งเป็นแบบฟังก์ชันไม่ต่อเนื่อง ค่าเอาท์พุทที่ให้ออกมาอยู่ในช่วง -1 หรือ 1 จะขึ้นอยู่กับค่าน้ำหนัก และการรวมกันเชิงเส้นของอินพุท

ค.ศ. 1962 Bernard Widrow และ Marcian Hoff [3] ได้เสนอแบบจำลองข่ายงานนิวรัลที่เรียกว่า ADALINE มาจากคำว่า adaptive linear neuron หรือ adaptive linear system ที่ใช้กฎการเรียนรู้แบบเดลตา คือใช้การลดความแตกต่างระหว่างค่าผลลัพธ์จากข่ายงานกับค่าเป้าหมายโดยใช้การปรับน้ำหนักของข่ายงาน ซึ่งข่ายงานมีชั้นน้ำหนักเพียง 1 ชั้น

ค.ศ. 1969 Minsky และ Papert [5] เสนอว่าเปอร์เซปตรอนไม่สามารถเป็นตัวแก้ปัญหาของฟังก์ชันอย่างง่ายที่ไม่เป็นเชิงเส้น ซึ่งรวมทั้งกรณี "exclusive or " (XOR) เนื่องมาจากเหตุผลนี้การศึกษาข่ายงานนิวรัลก็เงียบหายไปในช่วงค.ศ. 1970-1979 อย่างไรก็ตามในช่วงต้นปี ค.ศ. 1980 ข่ายงานนิวรัลก็กลับมาแพร่หลายอีกครั้ง Schalkoff [6] กล่าวไว้ว่า หลังจากยุคของเปอร์เซปตรอน (post-perceptron) เริ่มด้วยการเพิ่มชั้นซ่อนในข่ายงานซึ่งเป็นการเพิ่มผลดีในการคำนวณและผลดีสำหรับโครงสร้างข่ายงานนิวรัลแบบป้อนข้างหน้าหลายชั้น โดยนำมาใช้จนถึงปัจจุบันนี้

ค.ศ. 1972 Teuvo Kohonen [3] พัฒนาข่ายงานแบบ Self – Organizing Feature Maps (SOM) ซึ่งใช้โครงสร้างการจัดกลุ่ม (clustering layer)

ค.ศ. 1974 Werbos [3] ได้เสนออัลกอริธึมการกระจายความผิดพลาดย้อนกลับ ซึ่งเป็นวิธีการฝึกข่ายงานนิวรัลที่ใช้กฎการเรียนรู้แบบเดลตา แต่ในช่วงนั้นยังไม่เป็นที่รู้จักกันอย่างกว้างขวาง หลังจากนั้น ค.ศ. 1986 Rumelhart และคณะได้เสนออัลกอริธึมการกระจายความผิดพลาดย้อนกลับอีกครั้งจนเป็นที่รู้จักกันอย่างแพร่หลาย

ค.ศ. 1976 Stephen Grossberg และ Gail carpenter [3] ได้พัฒนาทฤษฎีของ Self-Organizing ต่อ โดยเรียกข่ายงานนี้ว่า adaptive resonance theory

ค.ศ. 1990 Leonard และ Kramer [3] เสนอการใช้โมเมนตัมช่วยเร่งการเรียนรู้ของอัลกอริธึมการกระจายความผิดพลาดย้อนกลับ

2.2 ผลงานของอัลกอริธึมการกระจายความผิดพลาดย้อนกลับ [7]

ค.ศ. 1995 Bertsekas และค.ศ. 1996,1987 Polyak ได้กล่าวถึงอัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับแบบมาตรฐานนั้น (standard backpropagation) คือวิธีการใช้กฎความแตกต่างทั่วไป (generalized delta rule) ซึ่งอัลกอริธึมนี้ได้ค้นพบโดย Rumelhart, Hintonand William ในปี 1986 โดยมีฟังก์ชันโมเมนตัมเข้ามาร่วมด้วยเรียกว่าเป็นวิธีของ "heavy ball method" ที่ใช้ในการวิเคราะห์เชิงตัวเลข

อัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับแบบมาตรฐานสามารถใช้กับการฝึกข่ายงานเป็นช่วงๆ (batch training) นั้นคือค่าน้ำหนักจะถูกปรับทุกครั้งเมื่อสิ้นสุดทั้งชุดของข้อมูล(epoach) ที่ใช้ในการฝึกข่ายงานสำหรับในแต่ละรอบ(epoach) หรือการฝึกข่ายงานจะดีขึ้นถ้ามีการปรับทุกชุดของข้อมูลที่ใช้ฝึกข่ายงาน (incremental training) สำหรับการฝึกข่ายเป็นช่วงๆ ของอัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับแบบมาตรฐาน ค่าผิดพลาดจะเข้าสู่ค่าความผิดพลาดต่ำสุดเฉพาะที่ (local minima) ก่อนที่จะเข้าสู่ความผิดพลาดต่ำสุด

ของข่ายงาน (global minima) ดังนั้นการที่จะให้เข้าสู่ค่าผิดพลาดรวมได้คือต้องมีอัตราการเรียนรู้เพิ่มเข้ามาและต้องทำการปรับให้ลดลงอย่างช้าๆ

ค.ศ. 1995 McCelland, McNaughton และ O'Reilly ได้กล่าวถึงการเรียนรู้แบบลำดับ (sequence learning) ซึ่งเป็นการฝึกข่ายงานเฉพาะชุดข้อมูลฝึกข่ายงานในปัจจุบันแต่ทำการปรับน้ำหนักทั้งชุดฝึกข่ายงาน (ชุดข้อมูลในปัจจุบันรวมกับชุดในอดีต) ดังนั้นจึงเป็นแบบจำลองที่มีการเรียนรู้เหมือนมนุษย์ นั่นคือ มนุษย์และสัตว์จะสังเกตจดจำสิ่งใหม่ แต่ยังไม่ลืมสิ่งเก่า ซึ่งสามารถใช้กับข้อมูลที่มีสัญญาณรบกวนได้

การฝึกข่ายงานในการออฟติไมซ์ฟังก์ชันเป้าหมาย ได้มีผู้ศึกษามากมาย เช่น Fletcher (1987), Grill, Murray and Wright (1981) และ Masters (1995) กล่าวไว้ว่าการที่จะออฟติไมซ์ลักษณะที่ไม่เป็นเชิงเส้นได้ดีที่สุดจะขึ้นอยู่กับลักษณะปัญหา สำหรับอัลกอริธึมแบบ Levenberg-Marquardt เป็นอัลกอริธึมหนึ่งในการแก้ปัญหาลักษณะที่ไม่เป็นเชิงเส้นซึ่งอัลกอริธึมชนิดนี้จะเป็นอนุพันธ์อันดับ 2 โดยจะแบ่งเป็น 3 ประเภทดังนี้คือ

(1) *Stabilized Newton and Gauss-Newton algorithms* ใช้กับจำนวนน้ำหนักน้อย แต่ต้องการหน่วยความจำเท่ากับจำนวนน้ำหนักยกกำลังสอง

(2) *Quasi-Newton and Gauss-Newton algorithms* ใช้กับจำนวนน้ำหนักปานกลาง แต่ต้องการหน่วยความจำเท่ากับจำนวนน้ำหนักยกกำลังสอง แต่จะพบปัญหามากเนื่องจากมีหน่วยความจำจำกัด

(3) *Conjugate gradient* ใช้กับจำนวนน้ำหนักมากๆ แต่ต้องการหน่วยความจำเป็นสัดส่วนโดยตรงกับจำนวนน้ำหนัก

สำหรับการหาฟังก์ชันเป้าหมายที่มีอนุพันธ์ไม่ต่อเนื่องเช่น ฟังก์ชันกระตุ้นแบบแรมพ์ (ramp) ขาดอนุพันธ์ที่จุดสูงสุดและต่ำสุด หรือใช้กับค่าความผิดพลาดของอัลกอริธึมแบบเทรสโฮสต์ สำหรับข่ายงานขนาดใหญ่จะมีความช้ามากในการแบ่งประเภทข้อมูล แต่ถ้าใช้กับฟังก์ชันกระตุ้นแบบอนุพันธ์ต่อเนื่องจะดีมากเพราะสามารถเข้าสู่ความผิดพลาดจริงได้ (global minima)

ค.ศ. 1996 Bertsekas และ Tsitsiklis ได้กล่าวถึงการประมาณแบบสโตชาสติก (stochastic approximation) ว่าสำหรับการฝึกอัลกอริธึมแบบการกระจายย้อนกลับแบบช่วงๆ (batch training) สามารถเข้าสู่ค่าความผิดพลาดรวมได้ช้าและเป็นสิ่งยากในการปรับค่าอัตราการเรียนรู้และโมเมนตัม ซึ่งมีงานวิจัยมากมายที่จะเพิ่มความเร็วสำหรับข้อบกพร่องนี้ วิธีส่วนใหญ่จะไม่ค่อยแพร่หลาย แต่จะมี 2 อัลกอริธึมที่มีประสิทธิภาพคือ QUICKPROP (Fahlman 1989) และ RPROP (Riedmiller and Broun, 1993) ซึ่ง Joust and Werner (1994) ได้กล่าวไว้ว่าเป็นการออฟติไมซ์ลักษณะที่ไม่เป็นเชิงเส้นได้เร็วกว่าและน่าเชื่อถือกว่าอัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับแบบมาตรฐานมาก ดังนั้นอัลกอริธึมทั้งสองนี้ช่วยปรับปรุงอัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับแบบมาตรฐานให้มีประสิทธิภาพสูงขึ้นและ Bertsekas ยังกล่าวอีกว่าการเรียนรู้แบบสโตชาสติกจะใช้

เมื่อคำนวณฟังก์ชันเป้าหมายที่มีสัญญาณรบกวน ซึ่งส่วนใหญ่แล้วจะนำมาใช้ในการเรียนรู้แบบออนไลน์ ซึ่งอัลกอริธึมจะทำการสุ่มชุดข้อมูลขึ้นมาในการนำไปปรับน้ำหนัก เพื่อทำการลดเกรเดียนเดสเซนท์

2.3 งานที่เกี่ยวกับการปรับปรุงอัลกอริธึมการกระจายความผิดพลาดย้อนกลับ

งานวิจัยดังต่อไปนี้เป็นงานวิจัยที่เกี่ยวกับการปรับปรุงอัลกอรธึมการกระจายความผิดพลาดย้อนกลับให้ดีขึ้นสำหรับการสร้างแบบจำลองโดยจะแบ่งตามหัวข้อวิธีการในการช่วยปรับปรุงข่ายงานให้ดีขึ้น

2.3.1 การฝึกข่ายงานให้เร็วขึ้น

ข่ายงานนิวรัลที่ใช้อัลกอริธึมการกระจายความผิดพลาดย้อนกลับมีข้อเสียคือการฝึกข่ายงานช้า ดังนั้นจึงมีผู้วิจัยเกี่ยวกับการฝึกข่ายงานให้เร็วขึ้นโดยแบ่งเป็นดังนี้

2.3.1.1 หลีกเลี่ยงเทอมที่เป็นอนุพันธ์ (Fudge Derivative Term)

การปรับปรุงสิ่งแรกในด้านการเพิ่มความเร็วคือหลีกเลี่ยงเทอมอนุพันธ์ในชั้นเอาท์พุท เช่นถ้าใช้ฟังก์ชันกระตุ้นแบบซิกมอยด์คือ 1/(1+exp(-D*X)) อนุพันธ์คือ S*(1-S) ซึ่ง S คือค่าฟังก์ชันกระตุ้นของเอาท์พุท โดยส่วนใหญ่แล้ว D=1 ซึ่งอนุพันธ์จะเกิดค่ามากที่สุดเมื่อ S=(1/2) และจะทำให้ค่าน้ำหนักเกิดการเปลี่ยนแปลงมากที่สุด แต่ถ้าค่า S มีค่าใกล้ 0 หรือ 1 ค่าเทอมอนุพันธ์จะมีค่าน้อยมาก ในความเป็นจริงถ้าข่ายงานส่งค่าการตอบสนอง (ผลลัพธ์จากการผ่านฟังก์ชันมูลฐาน) คือ 1 เอาท์พุทจะเท่ากับ 0 ซึ่งข่ายงานก็จะมีค่าออกนอกช่วง การปรับน้ำหนักจะทำโดยนำค่าน้ำหนักเดิมมาบวกกับค่าน้ำหนักที่เปลี่ยนแปลงไป แต่ค่าน้ำหนักที่เปลี่ยนแปลงน้อยมากจึงทำให้การปรับน้ำหนักน้อยมาก นั้นคือยิ่งใช้เวลานานมากในการฝึกข่ายงานที่ถูกต้อง Fahlman [8] ได้ทำการเพิ่มความเร็วโดยทำการเพิ่มเทอม 0.1 รวมเข้าไปกับเทอมอนุพันธ์ คือ 0.1+S*(1-S) และ Chen and Mars [9] ก็ได้เพิ่มเทอม 1 เข้าไปในเทอมอนุพันธ์เหมือนกันซึ่งวิธีนี้ค่าความผิดพลาดยิ่งมากขึ้นเมื่อชั้นข่ายงานน้อยลง และอัตราการเรียนรู้ยิ่งมีค่าน้อยลง ในการทดลองนี้ได้ใช้จำนวนนิวรัลในแต่ละชั้นคือ 10-5-10 ในการแก้ปัญหาซึ่งพบว่าค่าอัตราการเรียนรู้เท่ากับ 0.1 เท่าของค่าอัตราการเรียนรู้ที่มากที่สุดจะให้ผลดีที่สุด ซึ่งเรียกวิธีนี้ว่า "Differential step size" แต่ว่าค่า 0.1 ไม่ใช่ค่าที่ดีที่สุดเสมอไป ซึ่งจะต้องทำการทดลองหาจากค่าอัตราการเรียนรู้สูงสุดจนถึงค่าอัตราการเรียนรู้ต่ำสุดเพื่อหาค่าดีที่สุด นอกจากนี้ค่าอัตราการเรียนรู้ที่ใช้สำหรับชั้นเอาท์พุทต้องมีค่ามากกว่าค่าอัตราการ

เรียนรู้ที่ไม่ใช้วิธีนี้เล็กน้อย ซึ่ง Schiffman และคณะ [10] ได้แสดงวิธีการวัดความผิดพลาดของวิธีนี้นั้นคือเมื่อนำค่า 1 มารวมกับอนุพันธ์ชั้นเอาท์พุทจะให้ค่าความผิดพลาดน้อยที่สุด

2.3.1.2 การเชื่อมต่อระหว่างอินพุทและเอาท์พุทโดยตรง (Direct Input-Output Connection)

Eduardo Somtag [11] ได้เสนอแนะไว้ว่าการเชื่อมต่อโดยตรงจากชั้นซ่อนไปที่ชั้นเอาท์พุทจะช่วยเพิ่มการฝึกข่ายงานให้เร็วขึ้น และจะช่วยเพิ่มความเร็วอีกถ้าฟังก์ชันกระตุ้นเป็นเชิงเส้น และถ้ามีจำนวนนิวรัลในชั้นซ่อนน้อยโดยตัดจำนวนนิวรัลในชั้นซ่อนที่ไม่ต้องการออก แต่ไม่เสนอแนะเมื่อใช้กับจำนวนเอาท์พุทมากๆเพราะว่าอาจไม่ทำให้เกิดการเรียนรู้

2.3.1.3 การปรับความชัน (Sharpness gain)

Izui and Pentland [12] ได้เสนอการลดเวลาในการฝึกข่ายงานโดยการเพิ่มความชันหรือเกน (sharpness or gain, D) ในฟังก์ชันซิกมอยด์ 1/(1+exp(-D*X)) แต่ถ้า D มากๆ จะเสี่ยงต่อการถึงจุดต่ำสุดเฉพาะที่ (local minimum) บางครั้งค่าที่ดีที่สุดของ D อาจน้อยกว่า 1 ก็ได้

2.3.1.4 การเลือกใช้อัลกอริธึมในการช่วยเพิ่มความเร็ว

ก. Rprop Algorithm

Rprop เป็นอัลกอริธึมหนึ่งที่ฝึกข่ายได้ดีและมีงานวิจัยมากมายได้กล่าวไว้ว่าในบางครั้ง Rprop สามารถแก้ปัญหาได้ดีในแบบหนึ่งแต่ไม่ได้ดีที่สุดสำหรับทุกกรณี

ข. Quickprop Algorithm

Scott Fahlman [8] ได้กล่าวถึง Quickprop ว่าเป็นอัลกอริธึมที่ดีวิธีหนึ่งและอาศัยวิธีของนิวตันสำหรับการหารากของควอทดราติกฟังก์ชัน (root of a quadratic function)

ค. Super Sab Algorithm

Super Self-adjusting Backpropagation Algorithm (Super SAB Algorithm) ถูกพัฒนาโดย Tom Tollenaere [13] ได้กล่าวว่า Super SAB ไม่เร็วเท่า Quickprop หรือ Rprop สำหรับปัญหาทั่วไป แต่ใน

บางครั้งจะดีสำหรับปัญหาในบางกรณี ซึ่งจากการทดลองจะให้ผลดีถ้าคือต้องจำกัดค่าอัตราการเรียนรู้ให้อยู่ในช่วง 1-5

ง. Conjugate Gradient Algorithm

Jonathan Shewcheek และ Patrick van der Smagt [14] ได้ทำการทดลอง XOR โดยใช้ข้อมูลอินพุทคือ sin(x)*cos(2*x) และ tan(x) และได้เสนอแนะว่าอัลกอริธึมชนิดนี้มีความเร็วสูง และสามารถแก้ปัญหาสำหรับการประมาณบนเส้นโค้งได้ดี ซึ่งสามารถใช้ได้ดีทั้งข้อมูลที่มีการรบกวนและไม่มีการรบกวน

จ. Cascade Correlation Algorithm

Fahlman และ Lebieve [15] เสนอวิธีการฝึกข่ายงานแบบ Cascade Correlation Algorithm ซึ่งเป็นการฝึกข่ายงานแบบการเพิ่มนิวรัลในชั้นซ่อน ณ เวลาหนึ่งๆ ซึ่งจะเร็วกว่าข่ายงานการกระจายย้อนกลับแบบเดิม เพราะว่าค่าน้ำหนักใหม่จะถูกฝึก ส่วนน้ำหนักที่เหลือจะถูกกำหนดให้เป็นค่าคงที่ Lutz Prechelt [16] ได้เปรียบเทียบอัลกอริธึมนี้กับอัลกอริธึมมาตรฐาน โดยทำการเปลี่ยนแปลงโครงสร้างในการฝึก 5 แบบ โดยใช้ชุดข้อมูลอินพุท 42 ชุดซึ่งพบว่าในบางกรณีก็เหมาะกับปัญหา regression และไม่เหมาะกับปัญหาการแบ่งแยกประเภท (classification)

Recurrent Cascade Correlation (RCC) ปรับปรุงมาจาก Cascade correlation และ RCC ที่ถูกเสนอในช่วงแรกไม่สามารถเป็นตัวแทน Cascade Correlation ได้ และชนิดการเรียนรู้จะเป็นแบบการทำงานเป็นลำดับ (temporal sequence) เช่นระบบรถยนต์ ซึ่งการแก้ปัญหาของ Cascade Correlation คือทำการนำค่าเอาท์พุทย้อนกลับเข้ามาในชั้นซ่อน ซึ่งจะแก้ปัญหาความช้าสำหรับข่ายงานขนาดใหญ่ของแบบเดิมได้ด้วย

ฉ. อัลกอริธึมอื่นๆ ที่เร็ว

Shiffmann และคณะ [17] ได้ทำการศึกษาหาอัลกอริธึมที่เร็วโดยการใช้ cross-entropy error measure แทนค่าผิดพลาดของผลรวมกำลังสองและใช้การสเกลช่วงข้อมูลอินพุทโดยวิธีค่าเฉลี่ยและค่าเบี่ยงเบนมาตรฐาน Dugan Alpsan และคณะ [18] ให้ความสนใจในการศึกษาอัลกอริธึมแบบต่างๆสำหรับแก้ปัญหาทางด้านยา คือพยายามใช้วิธีต่างๆในการเร่งให้เร็วขึ้นและปรับปรุงการเรียนรู้ของปัญหา พบว่าอัลกอริธึมการกระจายความผิดพลาดย้อนกลับไม่สามารถมาถึงจุดนี้ได้และนั้นคือข้อเสียของข่ายงานแบบการกระจายความผิดพลาดย้อนกลับ แต่ในระหว่างที่อัลกอริธึมอื่นที่เร็วกว่าสามารถมาถึงจุดต่ำสุดจริงได้ (global error)

2.3.2 การเลือกตัวแทนชุดข้อมูล

การเลือกตัวแทนข้อมูลที่ถูกต้องมีผลกระทบโดยตรงกับเวลาที่ใช้ในการฝึก

ก. การลดขนาดช่วงของข้อมูล

ข. ตัวแทนเอาท์พุทที่ใช้สำหรับปัญหาการแยกประเภท

ค. การแทนที่ข้อมูลที่หายไป

จากงานวิจัยมากมายได้กล่าวว่าอินพุทที่ใช้ในข่ายงานอัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับต้องถูกลดขนาดช่วงข้อมูลให้อยู่ในช่วง 0 ถึง 1 ก่อน ในกรณีของฟังก์ชันกระตุ้นที่ใช้เป็นซิกมอยด์หรือ tanh ซึ่งสิ่งนี้ก็มีงานวิจัยอีกส่วนหนึ่งที่โต้แย้งว่าไม่เป็นความจริง อินพุทสามารถเป็นค่าจริงได้แต่ข่ายงานจะเรียนรู้ได้ช้าถ้าช่วงของอินพุทมีขนาดใหญ่มาก ส่วนค่าเอาท์พุทถ้าเกิดใช้ซิกมอยด์ค่าเอาท์พุทก็จะอยู่ในช่วง 0 ถึง 1 หรือถ้าใช้ tanh ก็จะได้ค่าเอาท์พุทอยู่ในช่วง -1 ถึง 1 ซึ่งถ้าเอาท์พุทอยู่นอกช่วงก็ต้องทำการขยายขนาดช่วงข้อมูลให้อยู่ในช่วงของค่าเอาท์พุทจริง แต่ก็สามารถหลีกเลี่ยงโดยใช้ฟังก์ชันเชิงเส้น

2.3.3 ฟังก์ชันกระตุ้น

Leshno และคณะ [19] กล่าวว่าฟังก์ชันกระตุ้นมีหลายฟังก์ชันให้เลือกใช้ ตัวอย่างเช่น

ก. ฟังก์ชันกระตุ้นที่เป็นเชิงเส้น

$$D.x \qquad ...(2.1)$$

โดยทั่วไป D = 1, x = อินพุทของนิวรัล แต่ฟังก์ชันเชิงเส้นไม่เหมาะสมสำหรับการประมาณฟังก์ชันทั่วไปส่วนใหญ่ จะเป็นแบบไม่เป็นเชิงเส้น เช่นฟังก์ชันกระตุ้นแบบ ซิกมอยด์ (logistic) มีค่าเอาท์พุท 0-1

ข. ฟังก์ชันซิกมอยด์

$$1/(1+\exp(-D.x)) \qquad ...(2.2)$$

โดยทั่วไป D = 1, x = ค่าอินพุทนิวรัล ในบางการทดลองแสดงว่าค่านิวรัลในชั้นซ่อนที่มีค่าอยู่รอบๆ ค่า 0 จะฝึกเร็วกว่าดังนั้นการใช้งานโดยทั่วไปจึงลบด้วย 0.5 ดังนั้น ฟังก์ชันซิกมอยด์สามารถใช้ฟังก์ชัน

$$
\begin{array}{l}
x \mid f(x) \\
x \geq 4.1 \mid 1 \\
-4.1 < x < 4.1 \mid 0.5 + x(1 - abs(x)/2) \\
x \leq -4.1 \mid 0
\end{array}
\qquad \ldots(2.3)
$$

ค. ฟังก์ชัน tanh มีค่าเอาท์พุทอยู่ในช่วง -1 ถึง 1

$$
2/(1+\exp(-2*D*x))-1 \qquad \ldots(2.4)
$$

ค่านี้มีจุดศูนย์กลางอยู่ที่ 0 นั้นคือ tanh จะเร็วกว่าฟังก์ชันซิกมอยด์ Kalman and Kwasny [20] กล่าวว่าในบางข่ายงานก็เร็วกว่าแต่บางข่ายงานก็ช้ากว่า

การประมาณเชิงตัวเลขอย่างง่ายโดยฟังก์ชัน tanh กล่าวโดย Anguita Parodi and Zunino [21] ว่า

$$
\begin{array}{l}
x \mid f(x) \\
x > 1.92033 \mid 0.96016 \\
0 < x \leq 1.92033 \mid 0.96016 - 0.26037 - (x - 1.92033)^2 \\
-192033 < x < 0 \mid 0.26037 * (x + 1.92033)^2 - 0.96016 \\
x \leq -1.92033 \mid 0.96016
\end{array}
\qquad \ldots(2.5)
$$

นอกจากจะเร็วขึ้นที่จะหลีกเลี่ยงในการคำนวณเทอมอนุพันธ์ ยังสามารถประหยัดเวลาในการประมวลผลกว่าการประมาณด้วยฟังก์ชันซิกมอยด์ที่เป็นอนุกรมเชิงเส้น เช่น piecewise - linear function ซึ่งต้องการจำนวนในการวนซ้ำมากในการแก้ปัญหา แต่สามารถประหยัดเวลาได้เล็กน้อยเท่านั้น

ง ฟังก์ชันเกาส์เสียน (Gaussian function)

$$\exp(-D*D*x*x) \quad ...(2.6)$$

ใช้ในกรณีจะให้การฝึกข่ายงานที่เร็ว สำหรับตัวอย่างในข่ายงาน -2-1-1 กับอินพุทที่เชื่อมต่อโดยตรงกับค่าเอาท์พุท จะให้การฝึกข่ายงานที่เร็ว

David Elliot [22] กล่าวถึงฟังก์ชันซิกมอยด์ที่มีค่าจาก -1 ถึง 1 ดังสมการ (2.7) และจาก 0 ถึง 1 ดังสมการ (2.8)

$$D*x/(1+|D*x|) \quad ...(2.7)$$

$$(D*x/2)/(1+|D*x|)+0.5 \quad ...(2.8)$$

ฟังก์ชันซิกมอยด์ทั้งสองจะเข้าสู่เป้าหมายช้ามาก นั้นหมายความว่า ถ้าพยายามที่ให้ได้ค่าเอาท์พุทจะต้องใช้จำนวนการวนรอบซ้ำมากกว่าที่จะมาถึงค่าเป้าหมาย แต่สำหรับปัญหาการแบ่งประเภท สามารถตรวจสอบจากการให้ค่าเอาท์พุทที่ถูกต้องมากกว่าค่าเอาท์พุทอื่น และสิ่งนี้จะช่วยประหยัดเวลาในการประมวลผลในการวนรอบซ้ำจำนวนมากของปัญหาการแบ่งประเภท

2.3.4 การปรับปรุงผลของค่าเอาท์พุท

การฝึกข่ายงานที่เร็วจะเป็นสิ่งที่ดีแต่ต้องได้ค่าผลของค่าเอาท์พุทในชุดทดสอบข้อมูลที่ดีด้วย ในหัวข้อต่อไปนี้เป็นการปรับปรุงให้ถึงเป้าหมายได้ดีขึ้น

ก. ขนาดของข่ายงาน

ทฤษฎีนี้กล่าวไว้ว่าฟังก์ชันปกติส่วนใหญ่เป็นการประมาณโดยใช้ชั้นซ่อนชั้นเดียวและฟังก์ชันกระตุ้นที่เป็นค่าที่ไม่ใช่โพลีโนเมียล (non-polynomial) และเอาท์พุทเชิงเส้น ซึ่งค่านิวรัลในชั้นซ่อนส่วนใหญ่จะเป็น N-1 โดย N คือจำนวนชุดข้อมูล แต่ในทางตรงข้ามก็สามารถออกแบบให้ใช้จำนวนนิวรัลน้อยๆถ้าสามารถเป็นไปได้ Eduardo Sontag [23] และ Daniel L.Chester [24] ได้กล่าวถึงเหตุผลในการที่จะต้องมีชั้นซ่อน 2 ชั้น และ ผลการทดลองที่แสดงว่าชั้นซ่อน 2 ชั้นดีกว่า ชั้นเดียว Patrick M. Shea and Felix Liu [25] และ Alvin J.Surkan

และ J.Clay Singleton [26] และ Jacques de Villiers and Etienne Barnard [27] กล่าวว่าจำนวนชั้นซ่อน 2 ชั้นสามารถเข้าสู่ความผิดพลาดต่ำสุดเฉพาะที่ได้มากกว่า Tamura และ Tateishi [28] กล่าวไว้ว่าส่วนใหญ่สามารถประมาณด้วยชั้นซ่อน 2 ชั้น ที่มีนิวรัล เท่ากับ N/2+3 ซึ่ง N คือจำนวนข้อมูล ซึ่งเป็นการบอกทางนัยว่าชั้นซ่อน 2 ชั้นดีกว่าชั้นเดียว และทางที่ดีควรฝึกข่ายงานที่ชั้นซ่อนชั้นเดียวก่อนถ้าจำเป็นจึงทำการเพิ่มชั้นซ่อนอีกหนึ่งชั้น

งานวิจัยจำนวนมากในการเสนอแนะสูตรของการหาจำนวนนิวรัลในชั้นซ่อน สูตรเหล่านี้ไม่จำเป็นเพราะแต่ละข่ายงานมีความซับซ้อนไม่เหมือนกัน สำหรับปัญหาที่ไม่ต้องการความใกล้เคียงกับเส้นโค้งหรือพื้นผิวก็สามารถใช้จำนวนนิวรัลในชั้นซ่อนน้อยๆ แต่ถ้าปัญหาซับซ้อนก็อาจจะต้องจำนวนนิวรัลในชั้นซ่อนมากขึ้น ซึ่งควรที่จะเริ่มใช้จำนวนนิวรัลน้อยๆไปหามาก เนื่องจากจำนวนน้อยสามารถฝึกข่ายงานได้เร็วกว่า

ข. การรวมเอาท์พุท

เทคนิคในการปรับปรุงผลคือการรวมตัวประมาณ (เอาท์พุท) ของหลายๆข่ายงาน โดยทั่วไปตามจินตนาการปัญหาจะเป็นเชิงเส้น แต่ถ้าข่ายงานไม่ได้เป็นเส้นตรง ก็สามารถหาเส้นโค้งที่ใกล้เคียงกับเส้นตรงได้ อาจจะเป็นไปได้ที่ข่ายงานอื่นๆจะได้เส้นโค้งที่แตกต่างกันโดยมาจากการเฉลี่ยค่าเอาท์พุทตามเส้นโค้งเพื่อให้ได้ใกล้เคียงกับเส้นตรง ในบางการทดลองตามเส้นตรงนี้สามารถสรุปว่าการเฉลี่ยเอาท์พุทของหลายๆข่ายงาน ในบางครั้งจะสามารถแก้ปัญหาได้แต่บางครั้งไม่สามารถแก้ปัญหาได้ Perone and Cooper [29] ได้กล่าวไว้ว่าวิธีนี้เป็นวิธีหนึ่งที่มีประโยชน์คือจะให้ผลที่ดีสำหรับข่ายงานที่มีการฝึกแล้วผลลัพธ์ไม่ดีมากๆ เช่นการทำนายเกี่ยวกับสต็อกในการตลาดในการเฉลี่ยของค่าเอาท์พุทในหลายๆข่ายงานจะให้ผลดีกว่าข่ายงานเดียวที่ให้ผลดีที่สุด แต่สำหรับบางงานวิจัยเช่นงานวิจัยที่เกี่ยวกับโซนาร์จะไม่เป็นไปตามทฤษฎีนี้

ค . การลดลงของค่าน้ำหนัก (Weight decay)

ถ้ารูปของฟังก์ชันที่ใช้ในการประมาณเป็นเส้นตรงก็สามารถใช้การวิเคราะห์แบบการประมาณค่าน้อยสุดได้ (least square analysis) รวมกับการใช้อัลกอริธึมแบบกระจายความผิดพลาดย้อนกลับจะให้ผลเร็วกว่าการใช้อัลกอริธึมกระจายความผิดพลาดย้อนกลับชนิดเดียว ดังนั้นถ้าพยายามประมาณค่าให้เป็นเส้นตรงคือถ้าจุดกระจายด้านเหนือหรือต่ำกว่าเส้นตรง ข่ายงานจะต้องทำให้อยู่ในเส้นตรงซึ่งผ่านจุดของข้อมูลมากหรือใกล้ที่สุดสำหรับแต่ละจุด อีกวิธีหนึ่งที่จะหลีกเลี่ยงในการใช้จำนวนนิวรัลในชั้นซ่อนน้อยที่สุดเท่าที่อาจจะเป็นไปได้ คือการใช้วิธีการลดลงของค่าน้ำหนัก วิธีนี้จะพยายามรักษาค่าน้ำหนักให้น้อยที่สุด นั่นคือใช้การลบเศษส่วนของค่าน้อยๆออกจากค่าน้ำหนัก ถ้า ค่าน้ำหนัก = w และ เศษส่วนของค่าน้อยๆ คือ lambda*w

$$w = w - \text{lambda}*w \qquad \text{....(2.9)}$$

ซึ่ง Finnoff Hergert และคณะ [30] กล่าวว่าผลที่ดีที่สุดของการเริ่มใช้การลดลงของค่าน้ำหนักคือทำให้ข่ายงานมาถึงจุดต่ำสุดของชุดฝึกข่ายงานได้

ง . อัลกอริธึมที่เร่งการเรียนรู้ (Acceleration Algorithm) เช่น Quickprop , Rprop จะเร็วกว่าข่ายงานการกระจายความผิดพลาดย้อนกลับมาตรฐาน

จ . การตัดค่าน้ำหนักที่เกินออก

Utans and Moody [31] ได้เสนอว่าการตัดค่าน้ำหนักที่เกินออกจะให้ผลของข่ายงานที่ดีขึ้น

2.4 ผลงานด้านข่ายงานนิวรัลในทางวิศวกรรมเคมี

ค.ศ.1989 Bhat และ McAvoy [32] ได้ศึกษาแบบจำลองเครื่องปฏิกรณ์ถังกวนแบบต่อเนื่อง (isothermal CSTR reactor) โดยใช้ข่ายงานนิวรัลแบบมีการป้อนไปข้างหน้าหลายชั้น (multilayered feedforward networks) ซึ่งสามารถหาจุดออฟติไมซ์ของปริมาณผลิตภัณฑ์ต่อวัตถุดิบได้

ค.ศ. 1991 Willis,Massimo, Montagur, Tham, และ Morris [33] ศึกษาการสร้างแบบจำลองโดยใช้ข่ายงานนิวรัลสร้างแบบจำลอง 2 กระบวนคือกระบวนการหมัก ซึ่งใช้ข่ายงานนิวรัลช่วยในการประมาณค่าความเข้มข้นของมวลจุลินทรีย์ และกระบวนการกลั่น โดยการประมาณค่าขององค์ประกอบของผลผลิตที่ยอดหอ

ค.ศ. 1992 Dawson และ Schopflocher [34] ศึกษาฟังก์ชันกระตุ้นแบบเกาส์เสียน (non-monotonic gaussian activation funciton) และศึกษาวิธีการเรียนรู้ที่เร็วกว่าการใช้ฟังก์ชันกระตุ้นแบบซิกมอยด์ โดยใช้การฝึกในข่ายงานที่ใช้อัลกอริธึมแบบการกระจายความผิดพลาดย้อนกลับ

ค.ศ. 1992 Nahas,Henson, และ Seborg [35] ใช้ข่ายงานนิวรัลที่มีการป้อนไปข้างหน้า 3 ชั้น ใช้ฝึกแบบจำลองของเครื่องปฏิกรณ์แบบถังกวนต่อเนื่อง (CSTR) และการทำระบบการทำให้เป็นกลาง (pH neutralization)

ค.ศ. 1992 Pollard และคณะ [36] ใช้ข่ายงานนิวรัลแบบป้อนไปข้างหน้าหลายชั้นในการแก้ปัญหากระบวนการในอุตสาหกรรม นั้นคือนำไปใช้ในหอกลั่นเพื่อทำนายอุณหภูมิของชั้นเทรย์โดยทำการป้อนอัตราการไหลย้อนกลับ (column reflux flow rate) เป็นอินพุท

ค.ศ. 1995 สุรพล คำสุภา [37] ศึกษาการสร้างแบบจำลองข่ายงานนิวรัล โดยศึกษาโครงสร้างของข่ายงานนิวรัลแบบกลับกระแสและไม่กลับกระแส พบว่าแบบจำลองแบบกลับกระแสเป็นโครงสร้างที่ข่ายงานเรียนรู้ได้

รวดเร็วกว่าเนื่องจากค่าเป้าหมายในอดีตเป็นตัวช่วยกำหนดแนวทางในการเรียนรู้โดยทำให้มีทิศทางในการหาคำตอบที่รวดเร็ว โดยมีพารามิเตอร์ที่เกี่ยวข้องคือ ค่าอัตราการเรียนรู้, ค่าโมเมนตัมและจำนวนนิวรัลในชั้นซ่อน และพบว่าโครงสร้างของข่ายงานนิวรัลที่มีจำนวนชั้นซ่อน 2 ชั้นไม่ได้ช่วยในเรียนรู้ให้รวดเร็วเมื่อเปรียบเทียบกับจำนวนชั้นซ่อนเพียงหนึ่งชั้นเนื่องจากต้องเสียเวลาในการคำนวณค่าน้ำหนักที่เชื่อมโยงระหว่างนิวรัลมากขึ้น แต่อย่างไรก็ตามโครงสร้างที่มีชั้นซ่อนมากกว่า 1 ชั้นอาจจะเหมาะสมกับกระบวนการที่มีความซับซ้อนมากกว่าก็ได้

ค.ศ. 1995 R.Baratti, G.Uacca และ A.Servida [38] นำข่ายงานนิวรัลไปประยุกต์ใช้กับการมอนิเตอร์และการควบคุมบนหอกลั่น 2 หอ ได้แก่หอกลั่นบิวเทน (Butane splitter tower) และ gasoline stabilizer (หอกลั่นทั้ง 2 ชนิดอยู่ที่ SARAS Refinery) โดยข้อมูลที่ใช้ในการฝึกข่ายใช้ข้อมูล 4 วัน

ค.ศ.1995 C.J. Quek ., R.Balasubramanian, and Rangaiah [39] ได้พัฒนาและประเมินอุปกรณ์การทำนายค่าที่ต้องการโดยใช้คอมพิวเตอร์(soft sensor) บนพื้นฐานของข่ายงานนิวรัลเพื่อใช้ในการควบคุมซัลเฟอร์ที่จะแยกออกมาจากหน่วยการนำซัลเฟอร์กลับไปใช้ใหม่ (Sulfur Recovery Unit ,SRU) ในโรงกลั่นปิโตรเคมี เครื่องมือวิเคราะห์แบบออนไลน์ถูกใช้เพื่อหาความแตกต่างระหว่างความเข้มข้นของไฮโดรเจนซัลไฟด์ และ ซัลเฟอร์ไดออกไซด์ SO_2-H_2S-$2SO_2$ เพื่อออฟติไมซ์อัตราส่วนของอากาศต่ออัตราการป้อนในหน่วย SRU ในการทำงานเมื่อก่อนจะใช้เครื่องมือวิเคราะห์ซึ่งต้องการความถี่สูงและเครื่องมือไม่น่าเชื่อถือ การทำนายความเข้มข้นของ SO_2-$2SO_2$ สำหรับ SRU ได้ทำการศึกษาด้วยกัน 3 แบบจำลอง แบบจำลองแรกศึกษาโดยใช้ linear regression แบบที่สองใช้ nonlinear regression แบบสุดท้ายใช้ ข่ายงานนิวรัล (Neural network) ซึ่งผลจากการทดลองพบว่าความถูกต้องที่ใช้แบบจำลองข่ายงานนิวรัลถูกต้องที่สุด แต่ต้องมีข้อมูลอินพุทที่ถูกต้องและต้องมีหน่วยความจำมากพอที่สามารถคำนวณได้ แต่แบบจำลอง linear regression มีข้อดีคือสามารถใช้ได้กับโปรแกรม Excel spreadsheet แต่ในขณะที่ nonlinear regression ต้องใช้ซอฟท์แวร์ซึ่งต้องการหน่วยความจำในการคำนวณสูง เช่น Pentium PC เป็นอย่างน้อย และข่ายงานนิวรัลใช้เวลามากในการสร้างและการติดตั้งอุปกรณ์การทำนายค่าที่ต้องการโดยใช้คอมพิวเตอร์ (soft sensor) ในโรงงาน ถ้าถูกนำไปใช้กับการมอนิเตอร์แบบออนไลน์ต้องง่ายต่อการดำเนินการในระหว่างการควบคุม ถ้าไม่สามารถนำเข้าใช้งานได้ ระบบควบคุมก็ควรจะมีการคำนวณในคอมพิวเตอร์ตัวหลัก (server) แล้วผลต้องสามารถใช้ได้ดีกับระบบควบคุม ทั้ง 3 แบบจำลอง แบบ linear regression ง่ายในการใช้งานที่ DCS (Distributed Control System) แต่ข่ายงานนิวรัลไม่สามารถใช้งานได้เนื่องจากหน่วยความจำถูกจำกัด ส่วนใหญ่จะถูกติดตั้งบน VAX system แต่ยากในการเชื่อมต่อกับระบบควบคุม สังเกตว่าทุกแบบจำลองจะง่ายถ้าเป็น PC

ค.ศ. 1995 K.W. Lee and H.N.Lam [40] ได้วิจัยว่าข่ายงานนิวรัลแบบป้อนไปข้างหน้าที่ใช้ฟังก์ชันกระตุ้นแบบซิกมอยด์ในการเป็นตัวแทนของฟังก์ชันต่อเนื่องจะมีค่าความถูกต้องสูง ซึ่งไม่มีทฤษฎีที่แน่นอนในการหาจำนวนของนิวรัลในชั้นซ่อน

ค.ศ. 1995 Zvi Boger [41] ข่ายงานนิวรัลส่วนใหญ่ถูกใช้สำหรับแบบจำลองในอุตสาหกรรม ขนาดกลางหรือขนาดเล็ก มีเหตุผลหนึ่งของการไม่ถูกนำไปใช้ในโรงงานขนาดใหญ่คือการเรียนรู้ช้า หรือไม่มีการเข้าสู่เป้าหมาย (non convergence) การพัฒนาข่ายงานนิวรัลจะถูกออฟติไมซ์หลังจากกรองอินพุทและหาจำนวนชั้นซ่อนที่เหมาะสม งานวิจัยนี้ได้กล่าวถึงประสบการณ์ที่ใช้อัลกอริธึมทีสร้างขึ้นใหม่นี้ใช้เวลาประมาณ 6 ปีในการพัฒนาข่ายงานนิวรัลในอุตสาหกรรม ซึ่งตัวอย่างโรงงานอุตสาหกรรม ได้แก่โรงงานการบำบัดน้ำเสีย และ การผลิตสารออร์-กานิกส์ในเครื่องปฏิกรณ์แบบครั้งคราว (batch reactor)

ค.ศ. 1996 A.L. Riddele [42] ได้นำข่ายงานนิวรัลช่วยในการปรับปรุงกระบวนการผลิตที่โรงกลั่นน้ำมันโมบิล (Mobil's Beaumont,Texas) โดยทำนายค่า raffinate refractive index (RI) ซึ่งใช้ข้อมูลจากกระบวนการผลิตและข้อมูลจากห้องทดสอบคุณภาพ ความถี่ในการเก็บข้อมูลคือรายชั่วโมง และประโยชน์ที่ได้รับคือสามารถเพิ่มผลิตภัณฑ์ได้ถึง 8.4%

ค.ศ. 1997 Srdjan Nesic and Miran Vrhovac [43] การศึกษาและพัฒนาแบบจำลองการกัดกร่อนของคาร์บอนไดออกไซด์โดยการใช้ข่ายงานนิวรัล ประสิทธิภาพของแบบจำลองจะถูกทดสอบด้วยข้อมูลจากการทดลองและมีการแสดงให้เห็นถึงการเลียนแบบข้อมูลโดยการเปรียบเทียบกับแบบ semi-empirical models

ค.ศ. 1998 A.Barsamian และ J.Macias [44] ได้สร้างโปรแกรมสำหรับการนำไปใช้ในโรงงานจริง โดยนำไปใช้และประสบผลสำเร็จ 2 โรงงานได้แก่โรงกลั่นน้ำมันในทาง U.S. 1 โรงงาน ในโรงงาน U.S ทำการปรับปรุง 2 หน่วยการผลิต คือหน่วยกลั่นน้ำมันดิบและหน่วยการปรับปรุงค่าออกแทนของแก๊สโซลีน ในส่วนหน่วยกลั่นน้ำมันดิบ ได้ใช้ข้อมูลในหน่วยกลั่นน้ำมันดิบเพื่อการอ้างอิงคุณสมบัติต่างๆ โดยใช้ข่ายงานนิวรัลในการทำนายผลของ 90% ของจุดกลั่น , จุดติดไฟ, จุดแข็งตัว ผลสรุปพบว่ามีความถูกต้อง +/- 3 °C ซึ่งถือว่าดีมาก และในโครงการอนาคตคาดว่าจะใช้เป็น APC (Autonomous Process Control) และในหน่วยการปรับปรุงค่าออกเทนของแก๊สโซลีน (gasoline blending octane) ใช้ข่ายงานนิวรัลในการปรับปรุงสูตร เช่น ethyl RT*70 octane correlation ซึ่งผลสรุปแสดงความถูกต้อง +/- 0.3-0.6 ของค่าออกเทน ส่วนอีก 1 โรงงานเป็นโรงงานทางยุโรป ได้แก่ CEPSA ได้ใช้ข่ายงานนิวรัลในการทำนายค่าการซึมแพร่จากเครื่องมือวิเคราะห์ความหนืด

ค.ศ. 1999 H.H.Chen, M.T Manry, และ Hema Chandrasekaran [45] ได้เสนอการใช้เปอร์เซปตรอนแบบป้อนไปข้างหน้าหลายชั้น (Feedforward Multilayer Perceptrons) ในการตรวจสอบหาตำแหน่งโดยใช้ข้อมูลจากเครื่องมือตรวจสอบอนุภาค

2.5 ผลงานที่เกี่ยวข้องกับงานวิจัย

ค.ศ. 1991 K.B. McAuley และ J.F.MacGroger [46] ทำนายดัชนีอัตราการหลอมเหลวของพอลีเมอร์ (Melt Flow Rate) และความหนาแน่น ในเครื่องปฏิกรณ์แบบฟลูอิไดซ์เบดสำหรับกระบวนการผลิตโพลีเอททิลีน โดยใช้เทคนิคคาลมานฟิลเตอร์ (kalman filter) ในการทำนายดัชนีเหล่านี้

จากงานวิจัยที่ผ่านมากพบว่าข่ายงานนิวรัลสามารถนำไปประยุกต์ใช้ได้มากมายหลายด้าน ดังนั้นการนำข่ายงานนิวรัลไปประยุกต์ใช้ในการทำนายค่าพารามิเตอร์ของกระบวนการผลิตต่างๆในอุตสาหกรรมจึงเป็นแนวทางที่น่าสนใจในการนำไปประยุกต์ใช้ในการวิจัยครั้งนี้เพราะสามารถทำนายค่าเอาท์พุทที่ต้องการได้โดยไม่ต้องทราบสมการคณิตศาสตร์ (Black box) การศึกษางานวิจัยนี้จะใช้ข่ายงานแบบป้อนไปข้างหน้าหลายชั้นที่มีการเรียนรู้แบบอัลกอริธึมการกระจายความผิดพลาดย้อนกลับชนิด Levenberg – Marquardt ในการแก้ปัญหา