บทที่ 3

ทฤษฎีเครือข่ายประสาท

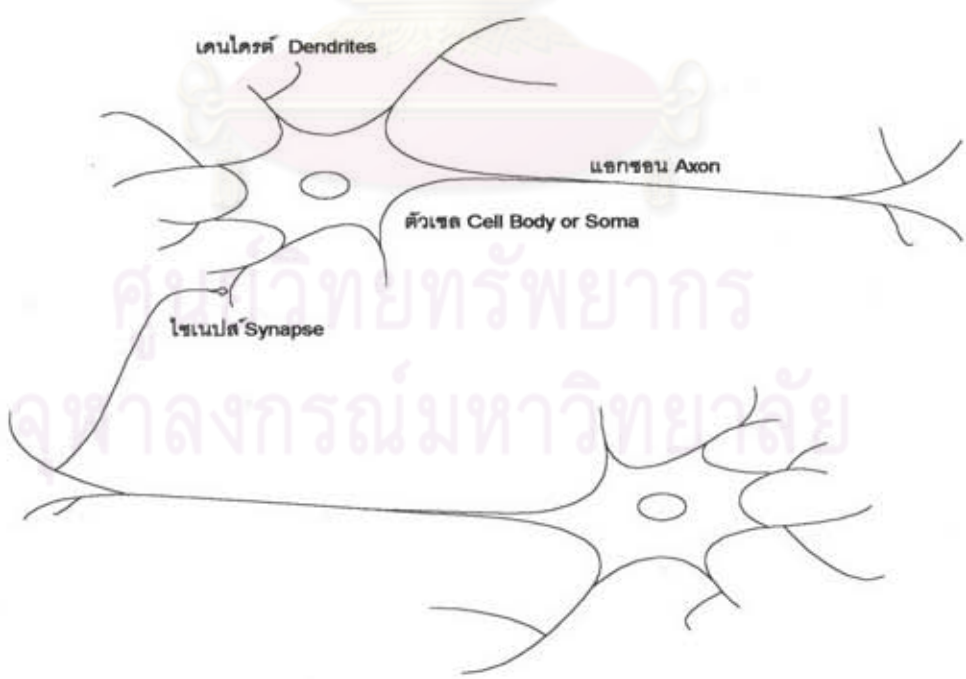
ในปัจจุบันนี้คอมพิวเตอร์ได้ถูกนำมาใช้งานอย่างกว้างขวางเกือบจะทุกด้านก็ว่าได้ แต่ยังคงมีขีดความสามารถจำกัดอยู่เมื่อเทียบกับสมองของมนุษย์ ซึ่งสมองสามารถจดจำและเรียนรู้จากประสบการณ์ในอดีตและนำมาปรับใช้กับสถานการณ์ปัจจุบัน ดังนั้นจึงได้มีการศึกษาการทำงานของสมอง เพื่อนำมาเป็นแบบจำลองของเซลล์ประสาท (Artificial Neuron) เครือข่ายประสาทเทียม (Artificial Neural Network) และศึกษากระบวนการเรียนรู้ (Learning algorithm) เพื่อนำมาประยุกต์ใช้กับเครื่องคอมพิวเตอร์ การพัฒนาเครือข่ายประสาทเริ่มตั้งแต่ปี พ.ศ. 2483 โดย McCulloch และ Pitts ซึ่งเป็นผู้ที่แสดงหลักการทำงานของเครือข่ายประสาทเทียมอย่างง่าย ในปลายปี พ.ศ. 2493 จึงได้เริ่มมีการนำเครือข่ายประสาทมาประยุกต์ใช้กับงานจริง ต่อมาในปี พ.ศ. 2501 Rosenblatt ได้คิดค้นกฎการเรียนรู้ให้กับเครือข่ายประสาทแต่ว่ายังแก้ปัญหาได้เฉพาะปัญหาที่ไม่ซับซ้อน ต่อมาปี พ.ศ. 2503 Widrow และ Hoff ได้พัฒนากฎการเรียนรู้ขึ้นมาใหม่ซึ่งยังใช้กันในปัจจุบันโดยเรียกกฎนี้ว่า "Widrow - Hofff learning rule" แต่ยังคงใช้ได้กับเครือข่ายอย่างง่าย จากนั้นในปี พ.ศ. 2512 Minsky และ Papert ได้เสนอเครือข่ายแบบใหม่เพื่อให้แก้ปัญหาได้ซับซ้อนมากขึ้นแต่ยังไม่ประสบความสำเร็จในการพัฒนากฎการเรียนรู้ให้ดีขึ้น งานการพัฒน่าที่สำคัญต่อมาคือในปี พ.ศ. 2515 Kohonen ได้คิดค้นเครือข่ายแบบใหม่ที่เรียกว่า Kohonen network ในปี พ.ศ. 2519 Grossberg ได้คิดค้นเครือข่ายที่เรียกว่า Self - organizing networks และในปี พ.ศ. 2523 เมื่อเครื่องคอมพิวเตอร์ได้รับการพัฒนาให้ทำงานได้เร็วขึ้นซึ่งส่งผลให้มีการคิดค้นกระบวนวิธีการเรียนรู้ใหม่ๆ เกิดขึ้นที่สำคัญคือกระบวนวิธีการแพร่กระจายกลับ (Back-propagation algorithm) โดยนำมาใช้กับเครือข่ายประสาทแบบหลายชั้น ซึ่งการเรียนรู้ชนิดนี้จะนำมาใช้ในวิทยานิพนธ์นี้

การนำทฤษฎีต่างๆ ของเครือข่ายประสาทที่ผ่านการพัฒนามาหลายสิบปีมาประยุกต์ใช้กับงานจริงมีอย่างมากมายหลายด้าน เช่น ด้านโทรคมนาคม นำมาใช้ในการบีบข้อมูลและภาพ (Image and data compression) การแปลภาษาพูด (Real - time translation of spoken language) การจดจำเสียง (Voice recognition) การจดจำภาพ (Image recognition) การจดจำรูปแบบ (Pattern recognition) ซึ่งจะจดจำรูปแบบเดิมและจำรูปแบบนั้นได้เมื่อพบอีกครั้ง

นอกจากนี้ยังมีการใช้งานแบบอื่นๆ เช่นการค้นหาเป้าหมาย (Trajectory control) การวิเคราะห์การตลาด (Market analysis) ฯลฯ และนอกจากนี้ยังได้นำมาประยุกต์ใช้กับงานด้านระบบไฟฟ้ากำลังเช่นการพยากรณ์ความต้องการไฟฟ้าระยะสั้น และการคำนวณโหลดโฟลว์ซึ่งจะได้กล่าวในบทต่อไป

3.1 การจำลองเซลล์ประสาท

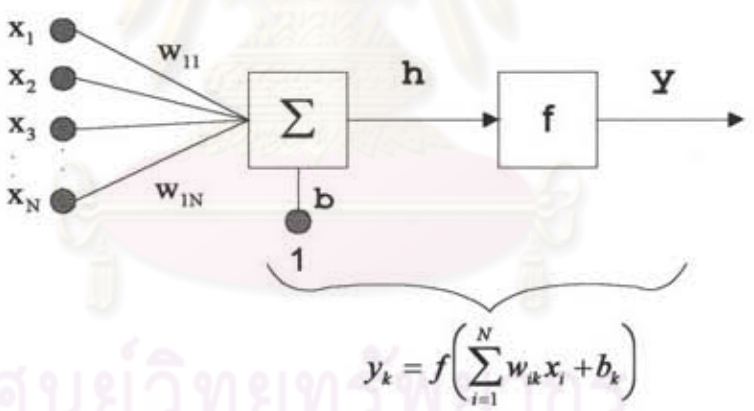
สมองมนุษย์ประกอบไปด้วยเซลล์ประสาท (Neuron) จำนวนมาก (ประมาณ 10^{11} ยูนิต) ที่มีการเชื่อมโยงกันอย่างหนาแน่น (การเชื่อมโยงแต่ละเซลล์ประสาทประมาณ 10^{4} ยูนิต) แต่ละเซลล์ประสาทประกอบไปด้วยเดนไดรต์ (Dendrites) ทำหน้าที่รับความรู้สึกจากเส้นใยประสาทโดยส่งข้อมูลเป็นสัญญาณไฟฟ้า (Electrical signals) มาที่เดนไดรต์ และที่เดนไดรต์แต่ละกิ่งจะรับรู้ได้ด้วยค่าถ่วงน้ำหนัก (Weight) ที่ต่างกันที่และแทนค่าถ่วงน้ำหนักด้วยความแข็งแรง (Strength) ของแต่ละไซเนปส์ (Synapse) จากนั้นจึงส่งข้อมูลให้ตัวเซลล์ (Cell body or soma) ซึ่งทำหน้าที่รวบรวมสิ่งที่ได้รับรู้แล้วส่งให้แอกซอน (Axon) แอกซอนจะส่งสัญญาณออกไป โดยสัญญาณที่ส่งออกมาจะเป็นฟังก์ชันของผลรวมของสิ่งที่ได้รับรู้จากตัวเซล รูปของเซลล์ประสาทสองเซลล์อย่างง่ายดังรูปที่ 3.1 [24,25]



รูปที่ 3.1 แสดงเซลล์ประสาทสองเซลล์อย่างง่าย

เมื่อทราบหลักการทำงานของเซลล์ประสาทแล้ว จะสามารถสร้างแบบจำลองของเซลล์ประสาทได้ ซึ่งเรียกว่าเซลล์ประสาทเทียม (Artificial neuron) หรือยูนิต (Unit) ดังรูปที่ 3.2 เซลล์ประสาทเทียมมีความเร็วในการประมวลผลสูงกว่าเซลล์ประสาท แต่สมองของมนุษย์สามารถทำงานหลายๆ งานที่ซับซ้อนพร้อมๆ กันได้เร็วกว่าคอมพิวเตอร์ เพราะโครงสร้างของเซลล์ประสาทมีโครงสร้างที่ขนานกันอย่างมาก (Massively parallel structure) [24] ซึ่งสามารถสรุปความสัมพันธ์ระหว่างเซลล์ประสาทกับเซลล์ประสาทเทียมได้ดังนี้

เซลล์ประสาท	**เซลล์ประสาทเทียม**
ตัวเซลล์ (Cell body)	ยูนิต (Unit)
เดนไดรต์ (Dendrites)	ตัวแปรด้านเข้า (Input)
แอกซอน (Axon)	ตัวแปรด้านออก (Output)
ไซแนปส์ (Synapse)	ค่าถ่วงน้ำหนัก (Weight)
ความเร็วช้ากว่า	ความเร็วสูงกว่า
มีเซลล์จำนวนมาก (ประมาณ 10^9 ยูนิต)	มีเซลล์จำนวนน้อย (ประมาณ 100 ยูนิต)



รูปที่ 3.2 แสดงเซลล์ประสาทเทียมอย่างง่าย

จากรูปที่ 3.2 ตัวแปรด้านเข้า x_i โดยที่ i = 1,2,3,...,N จะถูกส่งผ่านเข้าไปรวมกันด้วยค่าถ่วงน้ำหนักที่ไม่เท่ากัน จะได้ผลรวมเป็น

$$h_k = \sum_{i=1}^{N} w_{ik}x_i + b_k \qquad (3.1)$$

$$= w_{1k}x_1 + w_{2k}x_2 + \ldots + w_{Nk}x_N + b_k$$

จากนั้นส่งสัญญาณ h_k ผ่านฟังก์ชันที่เรียกว่าแอกติเวชัน (Activation function) เพื่อคำนวณค่าตัวแปรด้านออกที่ออกจากเซลล์ประสาทเทียมหรือยูนิต y_k

$$y_k = f(\sum_{i=1}^{N} w_{ik} x_i + b_k) \tag{3.2}$$

โดยที่ h_k เป็นค่าผลรวมของตัวแปรด้านเข้าของยูนิตที่ k

w_{ik} เป็นค่าถ่วงน้ำหนักระหว่างยูนิตที่ i และ k

b_k เป็นค่าไบอัส (Bias) ของยูนิตที่ k

ฟังก์ชันแอกติเวชัน f(h) มีอยู่หลายชนิดเช่น

1. ฟังก์ชันฮาร์ดลิมิต (Hard limit function)

   โดยที่ f (h) = hardlim (h)

   hardlim (h) = +1 เมื่อ $h \geq 0$ (3.3)

   = 0 เมื่อ $h < 0$

2. ฟังก์ชันเชิงเส้น (Linear function)

   โดยที่ f(h) = purelin (h)

   = h เมื่อ $-\infty < h < \infty$ (3.4)

3. ฟังก์ชันไบนารีซิกมอยด์ (Binary sigmoid function)

   โดยที่ $f(h) = \frac{1}{1+\exp(-n)}$ (3.5)

4. ฟังก์ชันไบโพลาร์ซิกมอยด์ (Bipolar sigmoid function)

   โดยที่ $f(h) = \frac{2}{1+\exp(-n)} - 1$ (3.6)

5. ฟังก์ชันไฮเพอร์โบลิกแทนเจนต์ (Hyperbolic tangent)

   โดยที่ $f(h) = \frac{1-\exp(-2x)}{1+\exp(-2x)}$ (3.7)

3.2 การเรียนรู้ของเครือข่ายประสาท (Neural network learning)

ในการเรียนรู้ของเครือข่ายประสาทจะใช้กฎการเรียนรู้ (Learning rule) เพื่อปรับเปลี่ยนค่าถ่วงน้ำหนักของเครือข่าย เพื่อให้ทำงานบางอย่าง กฎการเรียนรู้ของเครือข่ายประสาทมีหลายแบบ แต่ที่ใช้โดยทั่วไปแบ่งออกได้เป็น 3 แบบ [24-26] คือ

3.2.1 การเรียนรู้แบบมีการควบคุม (Supervised learning)

การเรียนรู้แบบนี้ต้องมีชุดของตัวอย่างในการปรับสอน (Training set) เพื่อแสดงพฤติกรรมที่แท้จริงของเครือข่ายประสาทนั้น ๆ ดังสมการที่ (3.8)

$$\{x_1, t_1\}, \{x_2, t_2\}, \ldots, \{x_P, t_P\} \qquad (3.8)$$

โดยมี x_p คือตัวแปรด้านเข้าของเครือข่ายประสาท

t_p คือผลลัพธ์ที่ต้องการ (Target)

P คือจำนวนรูปแบบ (Patterns) ที่ใช้ในการปรับสอนทั้งหมด

เมื่อใส่ตัวแปรด้านเข้าให้กับเครือข่ายประสาท ผลลัพธ์ที่ได้จากเครือข่ายจะได้ที่ตัวแปรด้านออกแล้วนำผลลัพธ์จากเครือข่ายประสาทไปเปรียบเทียบกับผลลัพธ์ที่ต้องการโดยกฎการเรียนรู้ เพื่อปรับค่าถ่วงน้ำหนักของเครือข่ายเพื่อให้ค่าของตัวแปรด้านออกมีค่าเข้าใกล้ผลลัพธ์ที่ต้องการ

3.2.2 การเรียนรู้แบบเสริม (Reinforcement Learning)

การเรียนรู้แบบนี้คล้ายกับ Supervised learning ยกเว้นแทนที่จะปรับค่าตัวด้านแปรออกให้เท่ากับผลลัพธ์ที่ต้องการในแต่ละตัวแปรด้านเข้า แต่จะปรับให้อยู่ในระดับคะแนน (Grade) ที่ต้องการแทนการปรับสอนแบบนี้เหมาะสำหรับการประยุกต์ใช้ในงานระบบควบคุม BaSu (1983)

3.2.3 การเรียนรู้แบบไม่มีการควบคุม (Unsupervised learning)

การเรียนรู้ชนิดนี้ค่าถ่วงน้ำหนักจะถูกปรับเพื่อให้สอดคล้องกับตัวแปรด้านเข้าเท่านั้น ไม่ต้องใช้ค่าของผลลัพธ์ที่ต้องการ ส่วนมากจะใช้ในงานการแยกออกเป็นส่วนย่อย (Clustering operation or network classifier) ซึ่งตัวอย่างของกฎการเรียนรู้แบบนี้คือ Adaptive resonance theory (ART) และ Kohonen self-organizing map

สำหรับในวิทยานิพนธ์นี้จะใช้วิธี Supervised learning เท่านั้นซึ่งจะได้กล่าวอีกครั้งในบทถัดไป

3.3 เครือข่ายเพอร์เซฟตรอน (Perceptron network)

เพอร์เซฟตรอน คือ รูปแบบของเครือข่ายประสาทอย่างง่ายซึ่งใช้สำหรับการแยกออกจากกันได้เชิงเส้น (Linearly separable) รูปแบบของเครือข่ายเพอร์เซฟตรอนดังแสดงในรูปที่ 3.3 ซึ่งประกอบไปด้วยเวกเตอร์ของตัวแปรด้านเข้า x_i มีมิติ Nx1 เมตริกซ์ค่าถ่วงน้ำหนัก w_{ik} มีมิติ NxM เวกเตอร์ของตัวแปรด้านออก y_k มีมิติ MX1 และไบอัส b_k มีมิติ Mx1

รูปที่ 3.3 แสดงเครือข่ายเพอร์เซฟตรอน

ค่าของตัวแปรด้านออกสามารถคำนวณได้จาก

$$y_k = \text{hard lim}\left(\sum_{i=1}^{N} w_{ik} x_i + b_k\right) \tag{3.9}$$

โดยที่เมตริกซ์ค่าถ่วงน้ำหนักสามารถเขียนกระจายได้เป็น

$$w_{ik} = \begin{bmatrix} w_{11} & w_{12} & \cdot & w_{1k} & \cdot\cdot & w_{1M} \\ w_{21} & w_{22} & \cdot & w_{2k} & \cdot\cdot & w_{2M} \\ \cdot & & & & & \cdot \\ w_{i1} & w_{i2} & \cdot & w_{ik} & \cdot\cdot & w_{iM} \\ \cdot & & & & & \cdot \\ w_{N1} & w_{N2} & \cdot & w_{Nk} & \cdot & w_{NM} \end{bmatrix} \tag{3.10}$$

จากฟังก์ชันแอกติเวชันที่อธิบายก่อนหน้านี้ จะได้ว่า

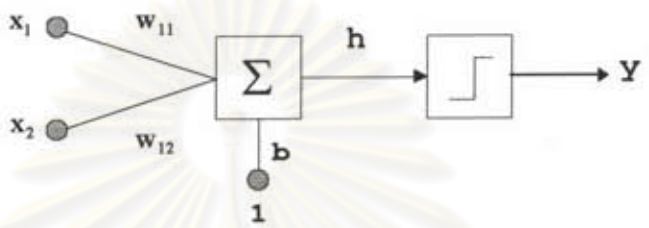
$$y_k = \text{hard lim}(h_k) = \begin{cases} 1 & ; \ h_k \geq 0 \\ 0 & ; \ h_k < 0 \end{cases} \tag{3.11}$$

ดังนั้นถ้า Inner product ของแถวที่ i^{th} ของเมตริกซ์ค่าถ่วงน้ำหนักกับเวกเตอร์ของตัวแปรด้านเข้ามีค่ามากกว่าหรือเท่ากับ $-b_k$ ค่าตัวแปรด้านออกจะมีค่าเป็น 1 นอกจากนั้นจะมีค่าเป็นศูนย์ ดังนั้นในแต่ละยูนิตหรือนิวรอน (Unit or neuron) ในเครือข่ายจะแบ่งตัวแปรด้านเข้า

ออกเป็นสองส่วน (Two regions) ซึ่งช่วยในการหาขอบเขต (Boundaries) ระหว่างสองบริเวณนี้ โดยเริ่มต้นศึกษาจากกรณีอย่างง่ายของเพอร์เซฟตรอนที่มี 1 ยูนิตและตัวแปรด้านเข้า 2 ตัว

3.3.1 เพอร์เซฟตรอน 1 ยูนิต (Single - unit perceptron)

พิจารณาเพอร์เซฟตรอนที่มีตัวแปรด้านเข้า 2 ตัว และมีตัวแปรด้านออก1 ยูนิต ดังแสดงในรูปที่ 3.4



รูปที่ 3.4 แสดงเพอร์เซฟตรอนที่มีตัวแปรด้านเข้า 2 ตัว และมีตัวแปรด้านออก 1 ยูนิต

ค่าของตัวแปรด้านออกของเครือข่ายนี้ หาได้จาก

$$y = \text{hard lim}(h) = \text{hard lim}(w_{11}x_1 + w_{12}x_2 + b) \qquad (3.12)$$

และสามารถหาขอบเขตการตัดสินใจ (Decision boundary) ของเพอร์เซฟตรอนได้จาก

$$\mathbf{W}.\mathbf{x} + \mathbf{b} = \mathbf{0} \qquad (3.13)$$

จากในเครือข่ายนี้ ขอบเขตการตัดสินใจคือ

$$h = \mathbf{wx} + \mathbf{b} = w_{11}x_1 + x_{12}x_2 + b = 0 \qquad (3.14)$$

เพื่อช่วยให้ง่ายขึ้นจะสมมติค่าให้กับค่าตัวถ่วงน้ำหนักและไบอัสมีค่าดังนี้

$$w_{11} = 1, w_{12} = 1, b = -1$$

ดังนั้นขอบเขตการตัดสินใจที่ได้คือ

$$h = \mathbf{wx} + \mathbf{b} = w_{11}x_1 + w_{12}x_2 + b = x_1 + x_2 - 1 = 0 \qquad (3.15)$$

จากสมการที่ (3.15) เป็นการนิยามสเปซของตัวแปรด้านเข้า (Input space) โดยที่ด้านหนึ่งของเส้นขอบเขตการตัดสินใจ ค่าของตัวแปรด้านออกของเครือข่ายมีค่าเป็น 0 อีกด้านหนึ่งมีค่าเป็น 1 หาจุดตัดแกนของเส้นขอบเขตการตัดสินใจ บนแกน x_1 และแกน x_2 การคำนวณหาจุดตัดแกน x_2 ทำได้โดยให้ $x_1 = 0$

$$x_2 = \frac{-b}{w_{12}} = -\frac{-1}{1} = 1 \quad \text{ถ้า} \quad x_1 = 0 \tag{3.16}$$

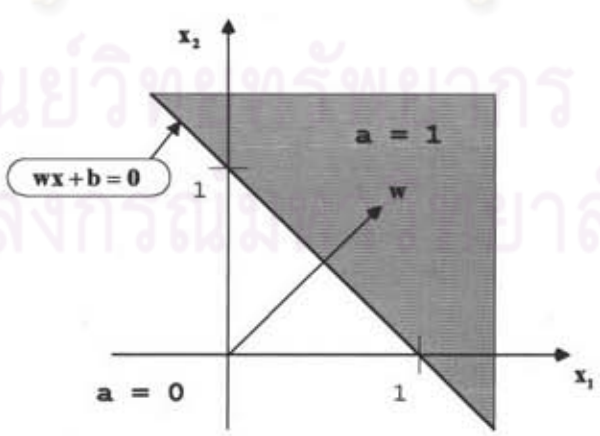
คำนวณหาจุดตัดแกน x_1 โดยให้ $x_2 = 0$

$$x_1 = \frac{-b}{w_{11}} = -\frac{-1}{1} = 1 \quad \text{ถ้า} \quad x_2 = 0 \tag{3.17}$$

เส้นขอบเขตการตัดสินใจดังแสดงในรูปที่ 3.5 และเพื่อหาว่าบริเวณใดจะให้ค่าของตัวแปรด้านออกของเครือข่ายเป็น 1 สามารถทำได้โดยใช้จุด 1 จุดในการทดสอบ สมมติว่าให้ค่าของตัวแปรด้านเข้าเป็น $x = \begin{bmatrix} 2 & 0 \end{bmatrix}^T$ ค่าของตัวแปรด้านออกของเครือข่ายคือ

$$y = \text{hard lim}(\mathbf{wx} + \mathbf{b}) = \text{hard lim}\left(\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} - 1 \right) = 1 \tag{3.18}$$

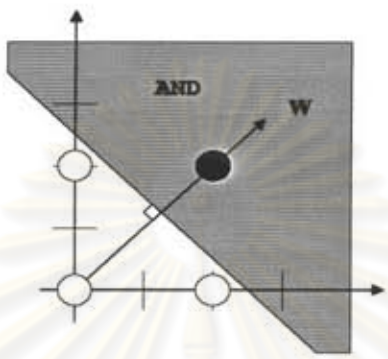
ดังนั้นบริเวณที่ค่าของตัวแปรด้านออกของเครือข่ายที่มีค่าเป็น 1 คือบริเวณที่อยู่ด้านบนทางขวามือของเส้นขอบเขตการตัดสินใจ บริเวณนี้แสดงไว้ด้วยการแรเงาดังในรูปที่ 3.5



รูปที่ 3.5 แสดงเส้นขอบเขตการตัดสินใจของเพอร์เซฟตรอนที่มีตัวแปรด้านเข้า 2 ตัว

เมื่อทดลองประยุกต์ใช้หลักการนี้ โดยใช้เครือข่ายเพอร์เซฟตรอนแก้ปัญหาของฟังก์ชันลอจิกอย่างง่ายของแอนเกท (AND gate) ซึ่งมีค่าของตัวแปรด้านเข้าและผลลัพธ์ที่ต้องการดังนี้

$$\left\{x_1 = \begin{bmatrix}0\\0\end{bmatrix}, t_1 = 0\right\}\left\{x_2 = \begin{bmatrix}0\\1\end{bmatrix}, t_2 = 0\right\}\left\{x_3 = \begin{bmatrix}1\\0\end{bmatrix}, t_3 = 0\right\}\left\{x_4 = \begin{bmatrix}1\\1\end{bmatrix}, t_4 = 1\right\}$$



รูปที่ 3.6 แสดงขอบเขตการตัดสินใจของแอนเกต (AND gate)

จากรูปที่ 3.6 แสดงเส้นขอบเขตการตัดสินใจของฟังก์ชันแอนเกท วงกลมสีดำแทนผลลัพธ์ที่ต้องการที่มีค่าเป็น 1 และที่ไม่มีสีแทนผลลัพธ์ที่ต้องการที่มีค่าเป็น O ถ้าต้องการแยกผลลัพธ์ที่ต้องการสีดำออกจากผลลัพธ์ที่ต้องการที่ไม่มีสี โดยใช้เส้นขอบเขตการตัดสินใจ จะเห็นได้ว่ามีมากมายหลายคำตอบในการแก้ปัญหานี้ เมื่อเลือกค่าถ่วงน้ำหนักที่ตั้งฉากกับเส้นขอบเขตการตัดสินใจมาหนึ่งค่าคือ $\mathbf{w} = \begin{bmatrix}2 & 2\end{bmatrix}$ ดังแสดงในรูปที่ 3.6 จากนั้นหาค่าไบอัส โดยใช้จุดทดสอบหนึ่งจุด ถ้าใช้ $\mathrm{x} = \begin{bmatrix}1.5 & 0\end{bmatrix}^{\mathrm{T}}$ จะได้ว่า

$$\mathbf{wx} + \mathbf{b} = \begin{bmatrix}2 & 2\end{bmatrix}\begin{bmatrix}1.5\\0\end{bmatrix} + b = 3 + b = 0 \Rightarrow b = -3$$

เมื่อหาค่า b ได้แล้วนำมาทดสอบกับค่าตัวแปรด้านเข้าและตัวแปรด้านออกแต่ละคู่ว่าเป็นจริงหรือไม่ สมมติว่าเราเลือกใช้ x_2 ในการทดสอบ ดังนั้นค่าตัวแปรด้านออกที่ได้คือ

$$y = \mathrm{hardlim}(\mathbf{wx_2} + \mathbf{b}) = \mathrm{hard\,lim}\left(\begin{bmatrix}2 & 2\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix} - 3\right)$$

$$y = \mathrm{hard\,lim}(-1) = 0$$

ซึ่งคำตอบที่ได้มีค่าเท่ากับเป้าหมายของ t_2 จริง

3.3.2 กฎการเรียนรู้ของเพอร์เซฟตรอน (Perceptron learning rule)

ขณะนี้ได้ทราบพฤติกรรมของเครือข่ายเพอร์เซฟตรอนแล้ว ซึ่งกฎการเรียนรู้ของเพอร์เซฟตรอนนี้เป็นกฎการเรียนรู้อย่างหนึ่งของ Supervised training ในที่นี้จะพิจารณาเฉพาะปัญหาที่แยกจากกันได้เชิงเส้นเท่านั้น ซึ่งมีคำตอบและวิธีการคำนวณค่าถ่วงน้ำหนักที่เหมาะสม ในกระบวนการเรียนรู้อย่างง่าย คือ การปรับค่าถ่วงน้ำหนักไปทีละยูนิตตามตัวแปรด้านออกที่เกิดขึ้น ถ้าตัวแปรด้านออกมีเครื่องหมายเหมือนกับผลลัพธ์ที่ต้องการแล้วจะไม่ปรับค่าถ่วงน้ำหนักนั้น แต่ถ้าตัวแปรด้านออกมีเครื่องหมายตรงกันข้ามแล้วจะปรับค่าถ่วงน้ำหนักนั้น โดยเป็นสัดส่วนเดียวกับผลคูณของตัวแปรด้านเข้ากับผลลัพธ์ที่ต้องการ ดังสมการที่ (3.19) ถึงสมการที่ (3.21)

$$w_{ik}^{new} = w_{ik}^{old} + \Delta w_{ik} \tag{3.19}$$

โดยที่

$$\Delta w_{ik} = (t_k - y_k)x_k \tag{3.20}$$

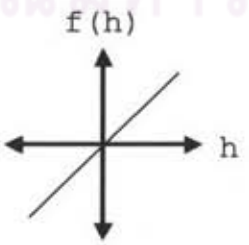
และ

$$b_k^{new} = b_k^{old} + (t_k - y_k) \tag{3.21}$$

3.4 ยูนิตเชิงเส้น (Linear units)

ในหัวข้อที่แล้วได้พิจารณาเครือข่ายเพอร์เซฟตรอนที่ใช้ฟังก์ชันแอกติเวชันแบบฟังก์ชันฮาร์ดลิมิต แต่ในหัวข้อนี้จะพิจารณาฟังก์ชันแอกติเวชันแบบฟังก์ชันเชิงเส้นที่ใช้ในยูนิตที่มีค่าต่อเนื่อง (Continuous-valued units)

ฟังก์ชันแอกติเวชันเชิงเส้นเป็นฟังก์ชันที่ต่อเนื่องและเป็นฟังก์ชันที่หาค่าอนุพันธ์ได้ ข้อดีของยูนิตชนิดนี้คือสามารถสร้างฟังก์ชันค่าผิดพลาด E ซึ่งเป็นเครื่องมือที่ใช้วัดค่าผิดพลาดที่เปลี่ยนแปลงของเครือข่ายเมื่อเทียบกับค่าถ่วงน้ำหนัก $\mathbf{W} = [\mathbf{w}_{ik}]$ จึงใช้วิธีการออปติไมซ์เซชันนี้ในรูปของเกรเดียนต์เดสเซนต์ (Gradient descent) [25] เพื่อที่จะลดค่าผิดพลาดให้น้อยที่สุด โดยจะเริ่มตัวอย่างในหัวข้อนี้ด้วยฟังก์ชันแอกติเวชัน $\mathbf{f(h)} = \mathbf{h}$ ซึ่งเป็นฟังก์ชันเชิงเส้นดังรูปที่ 3.7



รูปที่ 3.7 ฟังก์ชันแอกติเวชันชนิดเชิงเส้น

ตัวแปรด้านออกของเครือข่ายอย่างง่ายเมื่อมีตัวแปรด้านเข้า x_i คือ

$$y_k = f(h_k) = h_k = \sum_i w_{ik} x_i \tag{3.22}$$

โดยที่จะปรับค่าถ่วงน้ำหนักเพื่อให้ $y_k = t_k$ หรือ $t_k = \sum_i w_{ik} x_i$ ตามที่ต้องการ

3.4.1 การเรียนรู้แบบเกรเดียนต์เดสเซนต์ (Gradient descent learning)

เพื่อหากฎของการเรียนรู้และหาค่าถ่วงน้ำหนักทั้งชุด โดยการปรับค่าอย่างต่อเนื่องจากค่าถ่วงน้ำหนักตั้งต้น ในที่นี้เริ่มจากนิยามการวัดค่าผิดพลาดด้วยฟังก์ชัน

$$\begin{aligned} E &= \frac{1}{2}\sum_k (t_k - y_k)^2 \\ &= \frac{1}{2}\sum_k \left(t_k - \sum_i w_{ik} x_i\right)^2 \end{aligned} \tag{3.23}$$

ซึ่งถ้า E มีค่าน้อยลง ค่า w_{ik} จะปรับเข้าสู่ค่าถ่วงน้ำหนักที่ต้องการ โดยที่ E จะมีค่าเป็นบวกเสมอแต่จะมีค่าลู่เข้าสู่ศูนย์เมื่อผ่านกระบวนการทำซ้ำไปเรื่อยๆ

การปรับค่าถ่วงน้ำหนักทำได้โดยเลื่อนค่า E ลงมายังจุดต่ำสุดบนพื้นผิวที่ถูกนิยามใน W - space โดยปกติขั้นตอนของเกรเดียนต์เดสเซนต์จะปรับเปลี่ยนค่า w_{ik} แต่ละตัวด้วย Δw_{ik} ซึ่งเป็นสัดส่วนกับค่าเกรเดียนซ์ของ E ตามสมการที่ (3.24)

$$\begin{aligned} \Delta w_{ik} &= -a\frac{\partial E}{\partial w_{ik}} \\ &= \alpha \sum_P (t_k - y_k) x_i \end{aligned} \tag{3.24}$$

โดยที่ P คือจำนวนรูปแบบ (Patterns) ทั้งหมดที่ใช้ในการปรับสอน (Training)

α คืออัตราการเรียนรู้

ถ้าสามารถทำการเปลี่ยนค่าถ่วงน้ำหนักอย่างอิสระได้ทีละรูปแบบของตัวแปรด้านเข้า x_i จะได้

$$\Delta w_{ik} = \alpha (t_k - y_k) x_i \tag{3.25}$$

หรือ
$$\Delta w_{ik} = \alpha \delta_k x_i \tag{3.26}$$

โดยที่
$$\delta_k = t_k - y_k \tag{3.27}$$

และนิยาม δ_k คือค่าผิดพลาด

สมการที่ (3.25), (3.26) และ (3.27) นั้นรวมเป็นกฎที่เรียกว่ากฎเดลตา (delta rule, Widrow-Hoff rule หรือ Least mean square (LMS) rule)

จะสังเกตได้ว่าสมการที่ (3.25) เป็นผลมาจากเกรเดียนต์เดสเซนต์ ซึ่งพบว่าวิธีนี้จะง่ายต่อการนำไปใช้กับเครือข่ายที่มีหลายชั้น แต่ในขณะที่วิธีการของสมการที่ (3.20) นั้นใช้งานไม่สะดวก แต่วิธีดังกล่าวต้องใช้กับค่าต่อเนื่องและฟังก์ชันแอกติเวชันที่สามารถหาอนุพันธ์ได้

ฟังก์ชันค่าผิดพลาด (3.25) เป็นสมการควอดราติกในรูปของค่าถ่วงน้ำหนักกับค่าผิดพลาดมีพื้นผิวเป็นพาราโบลิกและประกอบด้วยค่าต่ำสุดอันหนึ่ง กฎเกรเดียนต์เดสเซนต์จะปรับเวกเตอร์ค่าถ่วงน้ำหนัก ในทิศทางของเวกเตอร์ตัวแปรด้านเข้า ขณะเดียวกันค่า α จะต้องมีค่าน้อยเพียงพอที่จะทำให้การเปลี่ยนแปลงของค่าถ่วงน้ำหนักเป็นไปในทิศทางที่ทำให้ค่าผิดพลาดลดลง ซึ่งจะได้แสดงตัวอย่างในหัวข้อถัดไป

3.4.2 การลู่เข้าของเกรเดียนต์เดสเซนต์ (Convergence of gradient descent) [27]

การวิเคราะห์การลู่เข้าของเกรเดียนต์เดสเซนต์นั้นจะทำให้รูปสมการควอดราติกจากสมการที่ (3.23) เป็นสมการไดอะโกนัลของค่าถ่วงน้ำหนักซึ่งเขียนเป็นสมการได้ดังนี้

$$E = \sum_{\lambda=1}^{M} a_\lambda \left(w_\lambda - w_\lambda^0\right)^2 \tag{3.28}$$

โดยที่ M คือจำนวนของค่าถ่วงน้ำหนักซึ่งเท่ากับจำนวนยูนิตของตัวแปรด้านเข้าคูณกับจำนวนยูนิตของตัวแปรด้านออก

a_λ และ w_λ^0 เป็นค่าคงที่ขึ้นอยู่กับเวกเตอร์ของค่าเจาะจง (Eigenvalues)

ค่าของ a_λ เป็นค่าบวกหรือศูนย์เพราะค่าผิดพลาดในสมการที่ (3.23) จะเป็นค่าบวกเสมอ และค่า w_λ คือค่าออปติมัมที่ทำให้ค่าผิดพลาดลดลงเมื่อ w_λ มีค่าเข้าใกล้ w_λ^0 สังเกตุว่าถ้า a_λ เป็นศูนย์แล้ว E จะเป็นอิสระจาก w_λ

จากสมการที่ (3.28) เมื่อใช้กฎของเกรเดียนต์เดสเซนต์จะได้

$$\begin{aligned}\Delta w_\lambda &= -\alpha \frac{\partial E}{\partial w_\lambda} \\ &= -2\alpha a_\lambda (w_\lambda - w_\lambda^0)\end{aligned} \tag{3.29}$$

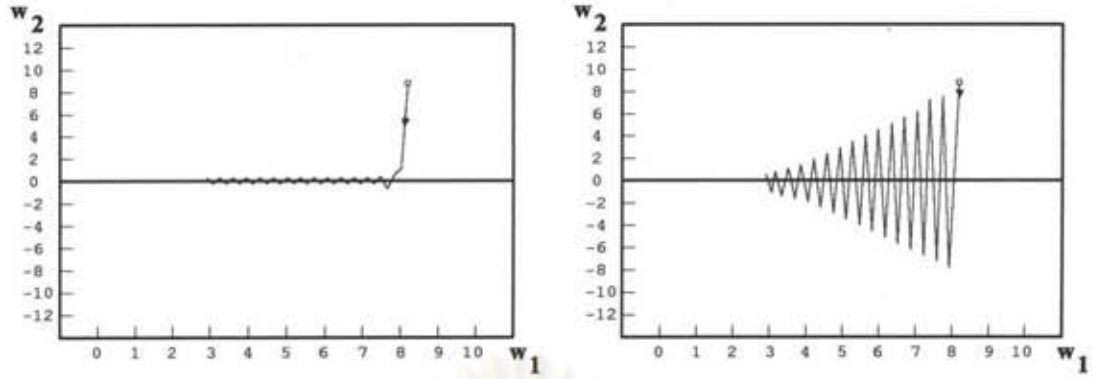
ดังนั้นระยะทาง $\delta w_\lambda = w_\lambda - w_\lambda^0$ และจะได้

$$\begin{aligned}
\delta w_\lambda^{new} &= w_\lambda^{new} - w_\lambda^0 \\
&= (w_\lambda^{old} + \Delta w_\lambda) - w_\lambda^0 \\
&= \left(w_\lambda^{new} - w_\lambda^0\right) + \Delta w_\lambda \\
&= \delta w_\lambda^{old} + \Delta w_\lambda \\
&= \delta w_\lambda^{old} + (-2\alpha a_\lambda \delta w_\lambda^{old}) \\
\delta w_\lambda^{new} &= \left(1 - 2\alpha a_\lambda\right)\delta w_\lambda^{old} \qquad (3.30)
\end{aligned}$$

สำหรับ $a_\lambda > 0$, การปรับค่าถ่วงน้ำหนักจะขยับเข้าไปใกล้กับค่า w_λ^0 ตราบเท่าที่ $|1 - 2\alpha a_\lambda| < 1$ เพราะระยะทาง δw_λ จะถูกคูณด้วยค่าคงที่นี้ทุกๆ กระบวนการทำซ้ำ ค่าของ α จะถูกจำกัดโดยค่าเจาะจง (Eigenvalue) ที่ใหญ่ที่สุด $\left(a_\lambda^{\max}\right)$ ซึ่งเกี่ยวข้องโดยตรงกับทิศทางของส่วนโค้งที่ลึกที่สุดบนพื้นผิวค่าผิดพลาด นั่นคือ η จะต้องมีค่าน้อยกว่า $1/a_\lambda^{\max}$ มิฉะนั้นค่าผิดพลาดจะมีค่าเพิ่มขึ้นเรื่อยๆ แต่อัตราเร็วในการเข้าสู่จุดที่เหมาะสมโดยทั่วไปจะถูกกำหนดโดยค่าเจาะจงที่น้อยที่สุดแต่ไม่เป็นศูนย์ $a_\lambda^{\min}$ ซึ่งเกี่ยวข้องโดยตรงกับทิศทางของส่วนโค้งที่ตื้นที่สุดบนพื้นผิวค่าผิดพลาด ถ้า $a_\lambda^{\max}/a_\lambda^{\min}$ มีค่ามากกระบวนการปรับค่าถ่วงน้ำหนักจะไปตามทิศทางของส่วนโค้งที่ตื้นอย่างช้ามาก

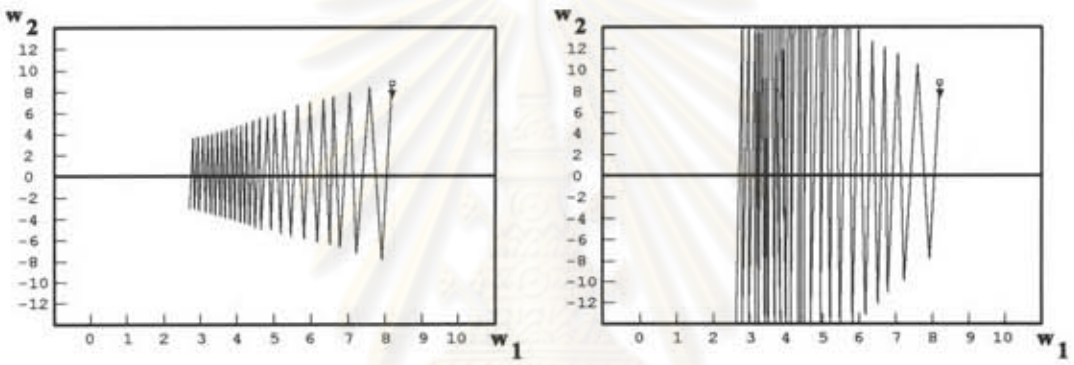
ตัวอย่างต่อไปนี้จะเป็นการแสดงวิธีเกรเดียนท์เดสเซนต์บนพื้นผิว $E = w_1^2 + 30w_2^2$ สำหรับ 20 รอบการคำนวณ ที่ค่าอัตราการเรียนรู้ (α) ต่างๆ กัน สมการในรูปไดอะโกนัลนี้มี a_1 คือ 1 และ a_2 คือ 30, w_1^0 และ w_2^0 เท่ากับศูนย์ ซึ่งเป็นค่าที่ E เข้าสู่จุดออปติมัม สมมติให้ α มีค่า 0.015 0.0322 0.033 และ 0.034 ตามลำดับ ดังแสดงในรูปที่ 3.8 และ 3.9

จากรูปที่ 3.8 และ รูปที่ 3.9 แสดงการปรับค่าถ่วงน้ำหนัก เมื่อสังเกตระยะทางจากจุดสุดท้ายถึงจุดต่ำสุด (Minimum) ที่ $\alpha = 0.015$ ค่า w_2 จะลงมาใกล้ศูนย์ค่อนข้างเร็ว แต่ w_1 จะปรับอย่างช้ามาก กรณีที่ $\alpha > 1/30 = 0.034$ การปรับค่าถ่วงน้ำหนักของ w_2 จะเกิดการแกว่งแบบลู่ออก การเข้าใกล้ที่เร็วที่สุดจะประมาณได้ว่าตัวคูณของ w_1 และ w_2 ซึ่งก็คือ $|1 - 2\alpha|$ และ $|1 - 60\alpha|$ ตามลำดับ แต่ต้องใช้ค่าเท่ากันคือ $\alpha = 1/31 = 0.0322$



$\alpha = 0.015$ $\alpha = 0.0322$

รูปที่ 3.8 เกรเดียนต์เดสเซนต์บนพื้นผิวควอดราติกที่มีค่า α เป็น 0.015 และ 0.0322 ตามลำดับ



$\alpha = 0.033$ $\alpha = 0.034$

รูปที่ 3.9 เกรเดียนต์เดสเซนต์บนพื้นผิวควอดราติกที่มีค่า α เป็น 0.033 และ 0.034 ตามลำดับ

อย่างไรก็ดีการวิเคราะห์นี้ได้สมมติว่าปรับค่าถ่วงน้ำหนักเป็นลำดับขั้นตามเกรเดียนต์เดสเซนต์ โดยปกติแล้วกระบวนการนี้จะปรับค่าถ่วงน้ำหนักอย่างช้าๆ ดังนั้นถ้าปรับเปลี่ยนค่าตัวแปร (เช่น α หรือ μ ซึ่งจะกล่าวในหัวข้อต่อไป) เป็นขั้นๆ ในทิศทางของเกรเดียนต์เดสเซนต์นี้แล้วจะช่วยเร่งการลู่เข้าให้เร็วขึ้นโดยเฉพาะเมื่อ $a_\lambda^{\max} / a_\lambda^{\min}$ มีค่ามาก แต่จะต้องควบคุมการปรับตัวแปรนั้นอย่างระมัดระวัง มิฉะนั้นจะเกิดการแกว่งเนื่องจากการเร่งค่ามากเกินไป

จากที่กล่าวมาแล้วนั้นได้สมมติว่าผลลัพธ์ที่ถูกต้องของการเรียนรู้ต้องการความอิสระเชิงเส้นของรูปแบบ (Pattern) แต่ถ้ารูปแบบไม่มีความเป็นอิสระเชิงเส้นแล้วจะให้สมการไดอะโกนัล (3.28) อยู่ในในรูปของสมการที่ (3.31)

$$E = E_0 + \sum_{\lambda=1}^{M} a_\lambda (w_\lambda - w_\lambda^0)^2 \qquad (3.31)$$

โดย $E_0 > 0$ ซึ่งเป็นจุดต่ำสุดของ E

3.5 ยูนิตไม่เชิงเส้น (Nonlinear units)

สำหรับที่ยูนิตไม่เชิงเส้นนี้จะใช้ฟังก์ชันแอกติเวชันที่ไม่เป็นเชิงเส้น เช่น ฟังก์ชันไบนารีซิกมอยด์ ไบโพลาร์ซิกมอยด์ หรือไฮเพอร์โปลิกแทนเจน แทนฟังก์ชันเชิงเส้นของยูนิตเชิงเส้น และใช้กฎการเรียนรู้แบบเดียวกับยูนิตเชิงเส้น นั่นคือเกรเดียนต์เดสเซนต์ ทำได้โดยหาค่าอนุพันธ์ของฟังก์ชันกำลังสองของค่าผิดพลาดดังนี้

$$E = \frac{1}{2}\sum_k \left(t_k - y_k\right)^2$$

$$= \frac{1}{2}\sum_k \left[t_k - f\left(\sum_i w_{ik} x_i\right)\right]^2 \qquad (3.32)$$

และพบว่า $$\frac{\partial E}{\partial w_{ik}} = -\sum_P \left[t_k - f\left(h_k\right)\right] f'\left(h_k\right) x_i \qquad (3.33)$$

ดังนั้นการปรับค่าถ่วงน้ำหนัก w_{ik} ด้วย $-\alpha \partial E / \partial w_{ik}$ จะเหมือนกับสมการ (3.26)

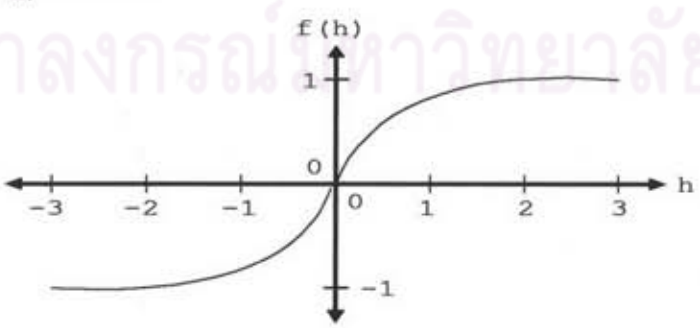
$$\Delta w_{ik} = \alpha \delta_k x_i \qquad (3.34)$$

แต่ค่าผิดพลาดจะเป็น $$\delta_k = \left[t_k - y_k\right] f'\left(h_k\right) \qquad (3.35)$$

จะต้องมีค่าแฟคเตอร์ $f'(h)$ จากการหาค่าอนุพันธ์ของฟังก์ชันแอกติเวชัน $f(h)$ ตัวอย่างเช่น

$$f(h) = \frac{2}{1+\exp(-h)} - 1 \qquad (3.36)$$

ฟังก์ชันแอกติเวชันนี้เป็นรูปฟอร์มของฟังก์ชันไบโพลาร์ซิกมอยด์ (Bipolar sigmoid function) ดังในรูปที่ 3.10 ค่าอนุพันธ์จะมากที่สุดเมื่อ $|h_k|$ มีค่าน้อย ดังนั้นการปรับค่าถ่วงน้ำหนักจะมีผลมากเมื่อยูนิตนั้น $|h_k|$ มีค่าน้อย



รูปที่ 3.10 แสดงลักษณะสมบัติของฟังก์ชันแอกติเวชันแบบฟังก์ชันไบโพลาร์ซิกมอยด์

ฟังก์ชันไบโพลาร์ซิกมอยด์เป็นฟังก์ชันที่คำนวณได้ง่าย เพราะอนุพันธ์ของฟังก์ชันนี้คือ $f'(h)=\frac{1}{2}\left(1-\left[f(h)\right]^2\right)$ โดยไม่จำเป็นต้องคำนวณอนุพันธ์ของ f(h) เพราะค่า $f\left(h_k\right)=y_k$ อยู่แล้ว ซึ่งจะได้ δ_k ดังสมการที่ (3.37)

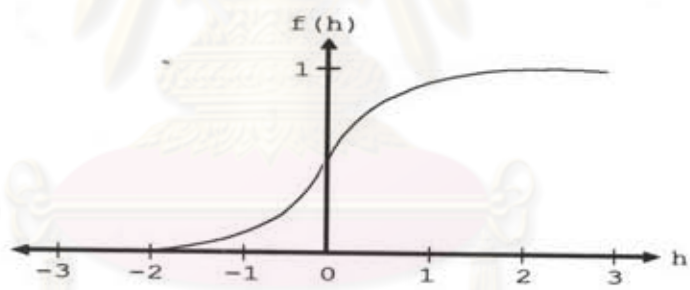
$$\begin{aligned}\delta_k &= \left[t_k-y_k\right]\left(1-\left(f(h)\right)^2\right)\\ &= \left[t_k-y_k\right]\left(1-\left(y_k\right)^2\right)\end{aligned} \qquad (3.37)$$

ทำนองเดียวกันฟังก์ชันไบนารีซิกมอยด์คือ

$$f(h)=\left[1+\exp(-h)\right]^{-1} \qquad (3.38)$$

สำหรับยูนิตที่มีตัวแปรด้านออกมีค่าระหว่าง 0 ถึง 1 ดังรูปที่ 3.11 และ $f'(h)=f(h)\left(1-f(h)\right)$ นั่นคือ

$$\begin{aligned}\delta_k &= \left[t_k-y_k\right]f\left(h_k\right)\left(1-f\left(h_k\right)\right)\\ &= \left[t_k-y_k\right]y_k\left(1-y_k\right)\end{aligned} \qquad (3.39)$$



รูปที่ 3.11 แสดงลักษณะสมบัติของฟังก์ชันแอกติเวชันฟังชันไบนารีซิกมอยด์

สำหรับเครือข่ายอย่างง่ายที่มีหนึ่งชั้น ข้อเด่นของฟังก์ชันแอกติเวชันแบบไม่เชิงเส้นคือ ฟังก์ชันเหล่านั้นสามารถจำกัดขอบเขตของตัวแปรด้านออกได้เช่น $(-1,+1)$ สำหรับฟังก์ชัน $f(h)=2\left[1+\exp(-h)\right]^{-1}-1$ หรือ $(0,+1)$ สำหรับฟังก์ชัน $f(h)=\left[1+\exp(-h)\right]^{-1}$ และมีบทบาทสำคัญสำหรับเครือข่ายประสาทแบบหลายชั้น (Multi-layer neural network) ซึ่งสามารถหาคำตอบสำหรับปัญหาที่ไม่สามารถหาคำตอบได้ด้วยกรณีฟังก์ชันแอกติเวชันแบบเชิงเส้น

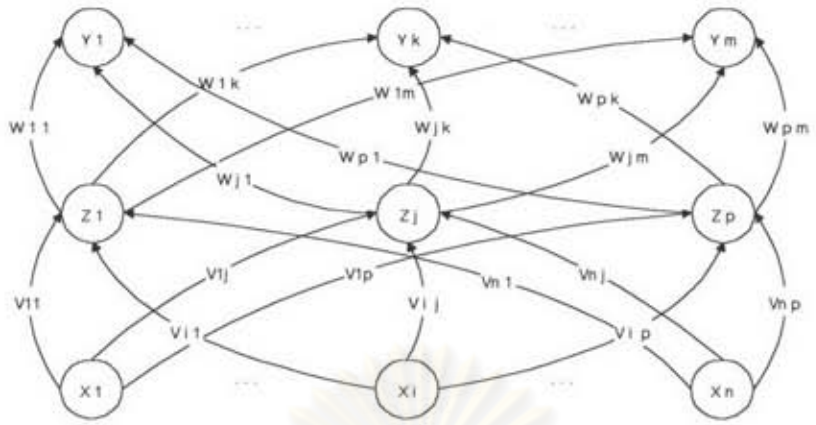
3.6 เครือข่ายประสาทแบบหลายชั้น (Multi-layer neural networks)

เครือข่ายประสาทหลายชั้นเป็นเครือข่ายประสาทที่สำคัญแบบหนึ่ง ประกอบด้วยส่วนรับข้อมูล (Sensory units or source nodes) ซึ่งโดยทั่วไปเรียกว่าชั้นตัวแปรด้านเข้า มีส่วนการคำนวณ (Computation nodes) เรียกว่าชั้นซ่อน (Hidden layers) ซึ่งอาจจะมีหนึ่งชั้นหรือมากกว่าก็ได้ และมีส่วนส่งข้อมูลออกจากส่วนการคำนวณเรียกว่าชั้นตัวแปรด้านออก (Output layer) [25,26]

เครือข่ายประสาทแบบหลายชั้นได้นำมาใช้ในแก้ปัญหาที่ซับซ้อนและปัญหาที่ไม่ลู่เข้า (Diverse preblems) ด้วยการปรับสอน (Training) ของเครือข่ายประสาทแบบชั้นเดียวได้ กระบวนวิธีที่นิยมใช้กันอย่างกว้างขวางคือ กระบวนวิธีแพร่กระจายความผิดพลาดกลับ (Error back-propagation algorithm) กระบวนวิธีนี้ขึ้นอยู่กับกฎการเรียนรู้เพื่อปรับค่าความผิดพลาด (Error-correction learning rule)

พื้นฐานกระบวนวิธีการแพร่กระจายความผิดพลาดกลับประกอบด้วย การส่งผ่านสองแบบระหว่างชั้นของเครือข่ายประสาท คือ การส่งผ่านไปข้างหน้า (Forward pass) และการส่งย้อนกลับ (Backward pass) ในการส่งผ่านไปข้างหน้าข้อมูลจะถูกป้อนเข้าสู่เครือข่ายประสาทให้แก่ชั้นซ่อนเพื่อส่งไปคำนวณต่อไปจนได้ผลลัพธ์ออกมาที่ชั้นตัวแปรด้านออก เมื่อมีตัวแปรด้านเข้าเข้าสู่เครือข่ายประสาท ในระหว่างการส่งผ่านไปข้างหน้าค่าถ่วงน้ำหนักที่เชื่อมต่อระหว่างชั้นของตัวแปรในเครือข่ายประสาททั้งหมดยังคงที่ และระหว่างการส่งย้อนกลับจะตรงกันข้ามคือค่าของตัวถ่วงน้ำหนักระหว่างชั้นของเครือข่ายประสาทจะถูกปรับเปลี่ยนค่าไปตามกฎการปรับค่าความผิดพลาด (Error-correction rule) โดยที่ค่าผลลัพธ์ที่ได้ของเครือข่ายประสาทไปลบกับค่าผลลัพธ์ที่ต้องการ (Desired or target) คือค่าความผิดพลาด (Error) ซึ่งค่าความผิดพลาดนี้จะแพร่กระจายกลับไปยังเครือข่ายตามทิศทางของการเชื่อมต่อ ดังนั้นจึงได้ชื่อว่า "การแพร่กระจายความผิดพลาดกลับ" (Error back-propagation) ค่าตัวถ่วงน้ำหนักที่เชื่อมต่อจะถูกปรับเพื่อให้ผลลัพธ์ที่ได้จากเครือข่ายเข้าใกล้ผลลัพธ์ที่ต้องการ กระบวนวิธีการกระจายค่าความผิดพลาดกลับนี้ใช้กระบวนวิธีการแพร่กระจายกลับ (Back-propagation algorithm) ดังนั้นจึงเรียกว่ากระบวนวิธีการแพร่กระจายกลับ และกระบวนวิธีการเรียนรู้สำหรับกระบวนวิธีนี้เรียกว่าการเรียนรู้แบบการแพร่กระจายกลับ (Back-propagation learning)

ในที่นี้จะพิจารณาเครือข่ายแบบป้อนไปสู่ข้างหน้าที่มี 2 ชั้น ดังรูปที่ 3.12 ซึ่งเป็นเครือข่ายประสาทที่ใช้โดยทั่วไป ชั้นบนสุดคือชั้นของตัวแปรด้านออก ชั้นถัดลงมาคือชั้นซ่อน ส่วนแถวล่างนั้นเป็นส่วนที่มียูนิตตัวแปรด้านเข้าต่ออยู่ซึ่งจะไม่นับเป็นชั้น



รูปที่ 3.12 แสดงเครือข่ายประสาทแบบป้อนไปสู่ข้างหน้าที่มี 2 ชั้น

สำหรับการคำนวณต่อจากนี้ไปจะกำหนดให้

T คือตัวแปรด้านออกที่เป็นผลลัพธ์ที่ต้องการ (Target) โดยที่ $T = (t_1, t_2, \ldots, t_k, \ldots, t_m)$

δ_k คือค่าการปรับเปลี่ยนความผิดพลาดของค่าถ่วงน้ำหนักสำหรับ w_{jk} ที่เกิดจากความคลาดเคลื่อนของยูนิตตัวแปรด้านออก Y_k ซึ่งก็คือค่าความคลาดเคลื่อนที่ของยูนิต Y_k ที่แพร่กระจายกลับไปยังยูนิตซ่อนที่เชื่อมอยู่ต่อกับยูนิต Y_k

δ_j คือค่าการปรับเปลี่ยนความผิดพลาดของค่าถ่วงน้ำหนัก (Error correction weight adjustment) สำหรับ v_{ij} ที่เกิดจากการแพร่กระจายกลับของความผิดพลาดจากชั้นตัวแปรด้านออกไปยังยูนิตของ Z_j

α คืออัตราการเรียนรู้

X_i คือยูนิตของตัวแปรด้านเข้า i โดยที่ $i = 1,2,\ldots,n$

v_{ij} คือค่าถ่วงน้ำหนักระหว่างยูนิตตัวแปรด้านเข้ากับยูนิตซ่อน

Z_j คือยูนิตซ่อน j โดยที่ $j = 1,2,\ldots,p$

ผลรวมสุทธิของข้อมูลด้านเข้า Z_j คือ z_in_j :

$$z\_in_j = \sum_i x_i v_{ij} \qquad (3.40)$$

สัญญาณที่ออกจากการผ่านฟังก์ชันแอกติเวชันของ Z_j คือ z_j:

$$z_j = f(z\_in_j) \qquad (3.41)$$

w_{jk} คือค่าถ่วงน้ำหนักระหว่างยูนิตซ่อนกับยูนิตตัวแปรด้านออก

Y_k คือยูนิตตัวแปรด้านออก k โดยที่ $k = 1,2,\ldots,m$

ผลรวมสุทธิของข้อมูลเข้า Y_k คือ y_in_k:

$$y\_in_k = \sum_j z_j w_{jk} \tag{3.42}$$

สัญญาณที่ออกจากการผ่านฟังก์ชันแอกติเวชันของ Y_k คือ y_k:

$$y_k = f(y\_in_k) \tag{3.43}$$

ในที่นี้จะกำหนดให้ฟังก์ชันเพื่อตรวจสอบค่าผิดพลาดคือ

$$E = \frac{1}{2}\sum_k \left[t_k - y_k\right]^2 \tag{3.44}$$

เขียนสมการที่ (3.44) จะเขียนใหม่ได้เป็น

$$E = \frac{1}{2}\sum_k \left[t_k - f\left(\sum_j w_{jk} f\left(\sum_i v_{ij} x_i\right)\right)\right]^2 \tag{3.45}$$

ด้วยการใช้กฎลูกโซ่ (Chain rule) จะได้ว่า

$$\begin{aligned}
\frac{\partial E}{\partial w_{jk}} &= \frac{\partial}{\partial w_{jk}}\left(\frac{1}{2}\sum_k \left[t_k - y_k\right]^2\right) \\
&= \frac{\partial}{\partial w_{jk}}\left(\frac{1}{2}\sum_k \left[t_x - f(y_in_k)\right]^2\right) \\
&= -\left[t_k - y_k\right]\frac{\partial}{\partial w_{jk}}\left(f\left(y_in_k\right)\right) \\
&= -\left[t_k - y_k\right] f'\left(y - in_k\right)\frac{\partial}{\partial w_{jk}}\left(y_in_k\right) \\
&= -\left[t_k - y_k\right] f'\left(y_in_k\right) z_j
\end{aligned} \tag{3.46}$$

เพื่อความสะดวกจะกำหนดให้

$$\delta_k = \left[t_k - y_k\right] f'(y\_in_k) \tag{3.47}$$

สำหรับค่าถ่วงน้ำหนักที่เชื่อมกับยูนิตซ่อน Z_j :

$$\frac{\partial E}{\partial v_{ij}} = -\sum_k \left[t_k - y_k\right]\frac{\partial}{\partial v_{ij}}\left(y_k\right)$$

$$= -\sum_k [t_k - y_k] f'(y - in_k) \frac{\partial}{\partial v_{ij}} (y\_in_k)$$

$$= -\sum_k \delta_k \frac{\partial}{\partial v_{ij}} (y\_in_k)$$

$$= -\sum_k \delta_k w_{jk} \frac{\partial}{\partial v_{ij}} (z_j)$$

$$= -\sum_k \delta_k w_{jk} f'(z\_in_j)[x_i] \quad (3.48)$$

และให้ $$\delta_j = -\sum_k \delta_k w_{jk} f'(z\_in_j) \quad (3.49)$$

จะสังเกตุได้ว่าค่า E เป็นฟังก์ชันที่สามารถหาอนุพันธ์อย่างต่อเนื่องทุกค่าของค่าถ่วงน้ำหนัก ดังนั้นจึงสามารถคำนวณหาค่าถ่วงน้ำหนักที่เหมาะสมโดยใช้กฎเกรเดียนต์เดสเซนต์ สำหรับการปรับค่าถ่วงน้ำหนักของยูนิตตัวแปรด้านออก ได้ดังนี้

$$\Delta w_{jk} = -\alpha \frac{\partial E}{\partial w_{jk}}$$

$$= \alpha [t_k - y_k] f'(y\_in_k) z_j$$

$$= \alpha \delta_k z_j \quad (3.50)$$

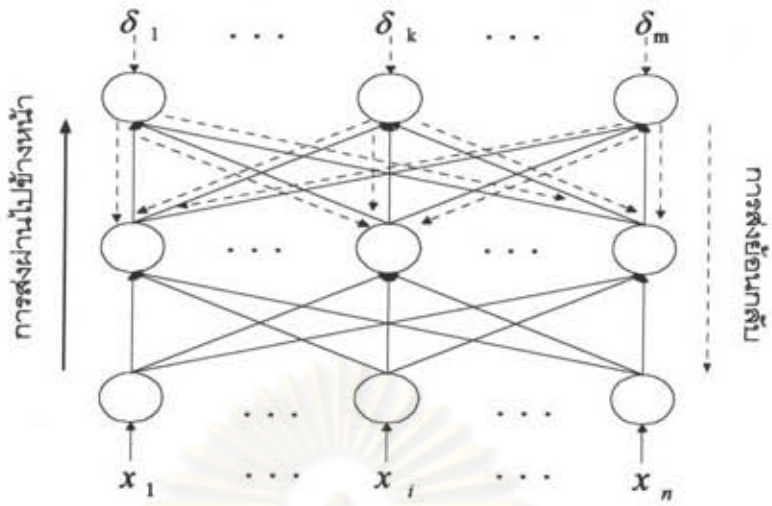
และสำหรับค่าถ่วงน้ำหนักของยูนิตซ่อนคือ

$$\Delta v_{ij} = -\alpha \frac{\partial E}{\partial v_{ij}}$$

$$= \alpha f'(z\_in_j) x_i \sum_k \delta_k w_{jk}$$

$$= \alpha \delta_j x_i \quad (3.51)$$

จะพบว่าค่า w_{jk} และ v_{ij} ถูกปรับค่าด้วย Δw_{jk} และ Δv_{ij} ตามลำดับจนกระทั่งค่าความผิดพลาด E มีค่าน้อยมากซึ่ง Δw_{jk} และ Δv_{ij} หามาจาก δ_k และ δ_j ตามลำดับ ซึ่งเป็นผลมาจากค่าผิดพลาด $(t_k - y_k)$ ในชั้นของตัวแปรด้านออกซึ่งเป็นค่าผิดพลาดที่ถูกนำกลับมาคำนวณหา Δw_{jk} และ Δv_{ij} จึงเรียกว่าการแพร่กระจายค่าผิดพลาดกลับ (Error back-propagation) รูปที่ 3.13 แสดงทิศทางการเคลื่อนที่ของค่าผิดพลาด โดยแสดงด้วยเส้นประ



รูปที่ 3.13 แสดงการเคลื่อนที่กลับของค่าผิดพลาด

จากที่กล่าวมาสามารถสรุปขั้นตอนการปรับสอนของกระบวนวิธีการแพร่กระจายกลับ (Standard backpropation) ได้ดังนี้ [26]

(1) สุ่มค่าน้อยๆ ให้แก่ค่าถ่วงน้ำหนัก

(2) ทดสอบเงื่อนไขการหยุดการปรับสอน (Training) โดยที่ค่าเฉลี่ยของความผิดพลาดต่อรูปแบบ (Average error per pattern, AEPP) [28] น้อยกว่าค่าความคลาดเคลื่อน (ε) ที่ยอมรับได้ ดังนี้

$$Average\ error\ per\ pattern = \frac{\sqrt{\sum_{i=1}^{P}\sum_{k=1}^{M} e_{ik}^2}}{P} \qquad (3.52)$$

โดยที่ P คือจำนวนรูปแบบ (Pattern) ที่ใช้ทั้งหมดในการปรับสอน

M คือจำนวนยูนิตของชั้นตัวแปรด้านออก

e_{ik} คือค่าความผิดพลาดระหว่างผลลัพธ์ที่ต้องการกับผลลัพธ์จากเครือข่ายประสาทที่ยูนิตตัวแปรด้านออก k คือ $(t_k - y_k)$ และรูปแบบข้อมูลการปรับสอนที่ i

ในระหว่างที่เงื่อนไขการหยุดการทำงานเป็นเท็จ ให้ทำขั้นตอนที่ 3 ถึง ขั้นตอนที่ 10

(3) สำหรับแต่ละคู่ของรูปแบบ (Pattern) ของการปรับสอนซึ่งประกอบไปด้วยเวกเตอร์ตัวแปรด้านเข้า และเวกเตอร์ตัวแปรด้านออกที่เป็นเป้าหมาย ให้ทำขั้นตอนที่ 4 ถึง 9

การป้อนไปสู่ข้างหน้า (Feedforward)

(4) ที่ชั้นซ่อนคำนวณหา z_in_j และ z_j จากสมการที่ (3.40) และ (3.41)

(5) ที่ชั้นตัวแปรด้านออกคำนวณหา y_in_k และ y_k จากสมการที่ (3.42) และ (3.43)

การแพร่กระจายความผิดพลาดกลับ (Back-propagation of error)

(6) ที่ชั้นซ่อนคำนวณหา δ_k จากสมการที่ (3.47) และหา Δw_{jk} จากสมการที่ (3.50)

(7) ที่ชั้นตัวแปรด้านออกคำนวณหา δ_j จากสมการที่ (3.49) และหา Δv_{ij} จากสมการที่ (3.51)

การปรับค่าถ่วงน้ำหนักใหม่ (Update weights)

(8) ค่าถ่วงน้ำหนักของยูนิตตัวแปรด้านออกคือ

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \tag{3.53}$$

ค่าถ่วงน้ำหนักของยูนิตซ่อนคือ

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij} \tag{3.54}$$

(9) ทดสอบเงื่อนไขการหยุดการทำงาน

3.6.1. กฎการเรียนรู้แบบการแพร่กระจายความผิดพลาดกลับร่วมกับโมเมนตัม (Standard back-propagation with momentum, SBM) [26]

จากการใช้กฎเกรเดียนต์เดสเซนต์ในการปรับค่าถ่วงน้ำหนักในระหว่างการปรับสอบนั้น พบว่าต้องใช้เวลานานในการปรับสอบถ้าค่า α มีขนาดเล็ก จะเกิดการแกว่งมากถ้า α มีขนาดใหญ่ และถึงแม้ว่าค่า α มีค่าพอดีที่ทำให้เครือข่ายเรียนรู้ได้เร็วแต่การปรับค่าถ่วงน้ำหนักจะเกิดการแกว่งที่ยังกว้างอยู่ จึงได้นำเอาหลักการของโมเมนตัมเข้าช่วย โดย Plaut และคณะ (1986) ได้คิดค้นขึ้นซึ่งมีผลมากในการปรับค่าถ่วงน้ำหนักและนอกจากนี้ยังช่วยแก้ปัญหาในเรื่องของ Local minimum ซึ่งสามารถทำการปรับค่าถ่วงน้ำหนักได้ดังนี้

$$\Delta w_{jk}(t+1) = \alpha \delta_k z_j + \mu \Delta w_{jk}(t) \tag{3.55}$$

และ

$$\Delta v_{ij}(t+1) = \alpha \delta_j x_i + \mu \Delta v_{ij}(t) \tag{3.56}$$

โดยที่ μ คือค่าโมเมนตัมซึ่งมีค่าอยู่ระหว่าง 0 ถึง 1

3.6.2. กฎการเรียนรู้แบบเดลตา-บาร์-เดลตา (Delta-Bar-Delta, DBD) [25,26]

ถึงแม้ว่าการใช้โมเมนตัมเข้ามาช่วยในการปรับค่าถ่วงน้ำหนักแล้วก็ตาม แต่การปรับสอนยังคงใช้เวลานานอยู่ จึงมีการวิจัยเพื่อหากระบวนวิธีการปรับสอนใหม่ๆ ซึ่งวิธี Delta-Bar-Delta ได้พัฒนาขึ้นมาโดย Jacobs (1988) และปรับปรุงแก้ไขอีกครั้งโดย Minai และ Williams (1990)

กฎของ Delta-Bar-Delta ประกอบด้วยกฎการปรับค่าถ่วงน้ำหนักและกฎการปรับอัตราการเรียนรู้ กำหนดให้ $\mathbf{w}_{jk}(t)$ คือค่าถ่วงน้ำหนักที่เวลา t ให้ $\alpha_{jk}(t)$ คืออัตราการเรียนรู้สำหรับค่าถ่วงน้ำหนักที่เวลา t และให้ E คือค่าความคลาดเคลื่อนกำลังสอง (Square error) สำหรับรูปแบบ (Pattern) ปัจจุบันที่เวลา t

กฎ Delta-Bar-Delta จะทำการปรับค่าถ่วงน้ำหนักสำหรับยูนิตตัวแปรด้านออกได้ดังนี้

$$\begin{aligned} w_{jk}(t+1) &= w_{jk}(t) - \alpha_{jk}(t+1)\frac{\delta E}{\delta w_{jk}} \\ &= w_{jk}(t) + \alpha_{jk}(t+1)\delta_k z_j \end{aligned} \tag{3.57}$$

สำหรับแต่ละยูนิตซ่อนคือ

$$\begin{aligned} w_{ij}(t+1) &= w_{ij}(t) - \alpha_{ij}(t+1)\frac{\delta E}{\delta w_{ij}} \\ &= w_{ij}(t) + \alpha_{ij}(t+1)\delta_j x_i \end{aligned} \tag{3.58}$$

สำหรับแต่ละยูนิตตัวแปรด้านออก จะนิยาม "Delta" เป็น

$$\Delta_{jk} = \frac{\delta E}{\delta w_{jk}} = -\delta_k z_j \tag{3.59}$$

สำหรับแต่ละยูนิตซ่อน

$$\Delta_{ij} = \frac{\partial E}{\partial v_{ij}} = -\delta_j x_i \tag{3.60}$$

กฎ Delta-Bar-Delta ใช้การรวมระหว่างค่าอนุพันธ์ปัจจุบันและค่าอนุพันธ์เดิมจึงเรียกว่า "Delta - Bar" ซึ่งมีวิธีการคำนวณ Delta - Bar ได้ดังนี้

สำหรับแต่ละยูนิตตัวแปรด้านออกจะได้

$$\overline{\Delta}_{jk}(t) = (1-\beta)\Delta_{jk}(t) + \beta\overline{\Delta}_{jk}(t-1) \tag{3.61}$$

และสำหรับแต่ละยูนิตซ่อนจะได้

$$\overline{\Delta}_{ij}(t) = (1-\beta)\Delta_{ij}(t) + \beta\overline{\Delta}_{ij}(t-1) \tag{3.62}$$

ค่าของตัวแปร β มีค่าอยู่ระหว่าง 0 ถึง 1

การเรียนรู้ของกฎ Delta-Bar-Delta คือจะปรับให้อัตราการเรียนรู้เพิ่มขึ้น ถ้า $\overline{\Delta}_{jk}(t-1)$ และ $\overline{\Delta}_{jk}(t)$ มีเครื่องหมายเหมือนกัน และจะลดอัตราการเรียนรู้ ถ้า $\overline{\Delta}_{jk}(t-1)$ และ $\overline{\Delta}_{jk}(t)$ มีเครื่องหมายตรงกันข้าม อัตราการเรียนรู้ใหม่สำหรับยูนิตของตัวแปรด้านออกคือ

$$\alpha_{jk}(t+1) = \begin{cases} \alpha_{jk}(t) + \kappa & ; \overline{\Delta}_{jk}(t-1)\Delta_{jk}(t) > 0 \\ (1-\gamma)\alpha_{jk}(t) & ; \overline{\Delta}_{jk}(t-1)\Delta_{jk}(t) < 0 \\ \alpha_{jk}(t) & ; \overline{\Delta}_{jk}(t-1)\Delta_{jk}(t) = 0 \end{cases} \tag{3.63}$$

สำหรับยูนิตซ่อนคือ

$$\alpha_{ij}(t+1) = \begin{cases} \alpha_{ij}(t) + \kappa & ; \overline{\Delta}_{ij}(t-1)\Delta_{ij}(t) > 0 \\ (1-\gamma)\alpha_{ij}(t) & ; \overline{\Delta}_{ij}(t-1)\Delta_{ij}(t) < 0 \\ \alpha_{ij}(t) & ; \overline{\Delta}_{ij}(t-1)\Delta_{ij}(t) = 0 \end{cases} \tag{3.64}$$

โดยที่ค่าของตัวแปร κ และ γ มีค่าระหว่าง 0 ถึง 1

กฎ Delta-Bar-Delta นี้เป็นวิธีได้รับความนิยมมาก เพราะค่าความผิดพลาดจะลดลงเร็วกว่ากฎการเรียนรู้แบบดั้งเดิม (Standard back-propagation) เนื่องจากอัตราการเรียนรู้จะถูกปรับเปลี่ยนไปแต่ละยูนิตที่เชื่อมต่อกันเลย

3.6.4. วิธีการปรับสอนแบบ Modified back-propagation (MBP) [25]

Haykin และคณะ (1991) ได้พัฒนาวิธี Modified back-propagation ขึ้นมาโดยใช้การรวมกันระหว่างกฎการเรียนรู้ของเดลตา-บาร์-เดลตา และโมเมนตัม การปรับค่าถ่วงน้ำหนักสำหรับยูนิตตัวแปรด้านออกทำได้โดย

$$w_{jk}(t+1) = w_{jk}(t) + \mu\,\Delta w_{jk}(t-1) + \alpha_{jk}(t+1)\delta_k z_j \tag{3.65}$$

สำหรับยูนิตซ่อนคือ

$$w_{ij}(t+1) = w_{ij}(t) + \mu \Delta w_{ij}(t-1) + \alpha_{ij}(t+1)\delta_j x_i \qquad (3.66)$$

โดยที่การคำนวณ $\Delta w_{jk}(t-1)$ และ $\Delta w_{ij}(t-1)$ ดังที่ได้กล่าวไว้ในหัวข้อ 3.6.2 และการคำนวณ $\alpha_{jk}(t+1)$ และ $\alpha_{ij}(t+1)$ ดังที่ได้กล่าวไว้ในหัวข้อ 3.6.3

วิธี Modified back-propagation เป็นวิธีที่ดีวิธีหนึ่งซึ่งเหมาะสมกับงานบางอย่าง แต่การใช้ค่าโมเมนตัมต้องเลือกค่าให้เหมาะสมจึงจะทำให้ค่าความผิดพลาดลดลงได้เร็วกว่าวิธีแบบดั้งเดิม (Standard back-propagation) หากเลือกค่าไม่เหมาะสมจะเกิดการแกว่งไม่ลู่เข้าของค่าถ่วงน้ำหนัก

3.6.5. ปัจจัยที่มีผลต่อการปรับสอน [25,26]

การปรับสอนให้แก่เครือข่ายโดยใช้กฎการเรียนรู้ที่กล่าวข้างต้นนั้นเป็นการปรับค่าถ่วงน้ำหนักให้ลู่เข้า การลู่เข้าจะเร็วหรือช้าขึ้นอยู่กับว่ากฎการเรียนรู้นั้นจะปรับค่าถ่วงน้ำหนักในทิศทางที่ถูกต้องหรือไม่ ปัจจัยอื่นนอกเหนือจากกฎการเรียนรู้ที่จะทำให้การลู่เข้าเร็วขึ้นคือ

(1) จำนวนชั้นซ่อน

(2) จำนวนยูนิตของชั้นซ่อน

(3) จำนวนข้อมูลที่ใช้ในการปรับสอน

(4) ค่าถ่วงน้ำหนักเริ่มต้น

ซึ่งมีรายละเอียดดังต่อไปนี้

(1) จำนวนชั้นของชั้นซ่อน ขึ้นอยู่กับความซับซ้อนของเครือข่ายว่ามีความซับซ้อนมากน้อยเพียงใด ถ้ามีความซับซ้อนมากจะต้องใช้จำนวนชั้นซ่อนหลายชั้น แต่สำหรับงานโดยทั่วไปจำนวนชั้นซ่อน 1 ชั้นก็เพียงพอ

(2) จำนวนยูนิตของชั้นซ่อน ยังไม่มีวิธีใดแน่นอนที่จะบอกว่าจะต้องใช้กี่ยูนิตจึงจะเหมาะสมที่สุด ซึ่งในการทำงานจริงๆ จะทดลองกำหนดค่ายูนิตของชั้นซ่อน แล้วพิจารณาอัตราการลู่เข้า จำนวนยูนิตซ่อนที่เหมาะสมควรเป็นค่าที่น้อยที่สุดโดยที่ยังคงอัตราการลู่เข้าที่ดี เพราะถ้าใช้จำนวนยูนิตซ่อนมากจะทำให้เวลาในการคำนวณระหว่างการปรับสอนช้าเพิ่มขึ้น และจำนวนยูนิตชั้นซ่อนที่ใช้ยังขึ้นอยู่กับชนิดของงานที่จะนำเครือข่ายประสาทไปใช้อีกด้วย

(3) จำนวนข้อมูลที่ใช้ในการปรับสอน ยิ่งมีข้อมูลมากการเรียนรู้จะทำให้คำตอบถูกต้องยิ่งขึ้นแต่จะเสียเวลาในการปรับสอนเพิ่มขึ้นด้วย แต่ถ้าใช้จำนวนน้อยเกินไปจะทำให้ผลลัพธ์มีความคลาดเคลื่อนสูง ในงานทั่วไปจำนวนรูปแบบที่เพียงพอในการปรับสอน (p) สามารถคำนวณได้จากสมการที่ (3.67)

$$p = \frac{W}{e} \tag{3.67}$$

โดยที่ W คือจำนวนค่าถ่วงน้ำหนักที่ใช้ในเครือข่ายทั้งหมด

e คือค่าความผิดพลาดที่ต้องการ

(4) การสุ่มค่าเริ่มต้นให้กับค่าถ่วงน้ำหนัก หากใช้ค่าเริ่มต้นที่ดีจะทำให้ใช้จำนวนรอบในการปรับสอนลดลงหรือการปรับสอนเร็วขึ้นในทางปฏิบัติจะใช้การสุ่มค่าถ่วงน้ำหนักที่มีค่าระหว่าง -0.5 ถึง 0.5