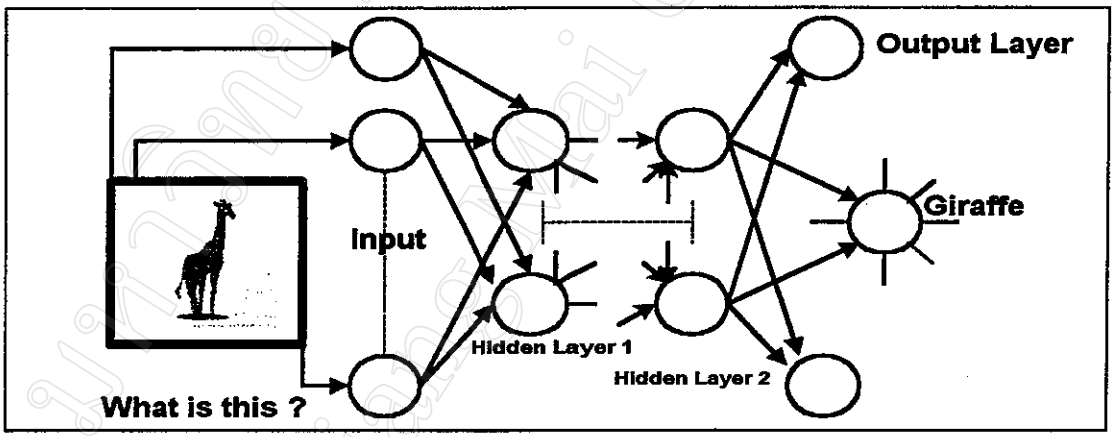
# บทที่ 3
# กระบวนการเรียนรู้แบบแพร่กลับ

## 3.1 บทนำ

ในการนำโครงข่ายประสาทเทียม ไปใช้เพื่อการวิเคราะห์การรู้จำภาพนั้น จะอยู่ในลักษณะที่เรียกว่า รูปแบบของการรู้จำ (Pattern Recognition) เพราะเป็นการสอนให้คอมพิวเตอร์สามารถรู้จำรูปแบบ (Pattern) ตามที่กำหนดและให้มีความสามารถในการระลึกจำ (Recognition) นอกจากนี้ต้องสามารถแยกแยะตัดสินใจได้ [16]




รูป 3.1 แบบจำลองโครงข่ายประสาทเทียมในการรู้จำภาพยีราฟ

รูป 3.1 เป็นแบบจำลองโครงข่ายประสาทเทียมในการรู้จำและบ่งชี้ (Identity) ภาพ (Image) ที่กำหนดได้ โดยชั้นอินพุต (Input Layer) เปรียบเป็นหน่วยรับภาพจากจอ ซึ่งนิวรอนที่ชั้นอินพุตนี้ จะตอบสนองต่อข้อมูล (Data) ที่ได้รับจากหลายๆส่วนของภาพ และส่งต่อไปให้ที่ชั้นซ่อน 1 , ที่นิวรอนแต่ละตัว , เป็นเช่นนี้ไป จนที่ชั้นเอาต์พุตจะรวบรวมสมมติฐาน (Hypothesis) เพื่อที่จะบอกว่าภาพนั้นเป็นรูปแบบเดียวกับที่รับเข้ามาทางอินพุต[4]

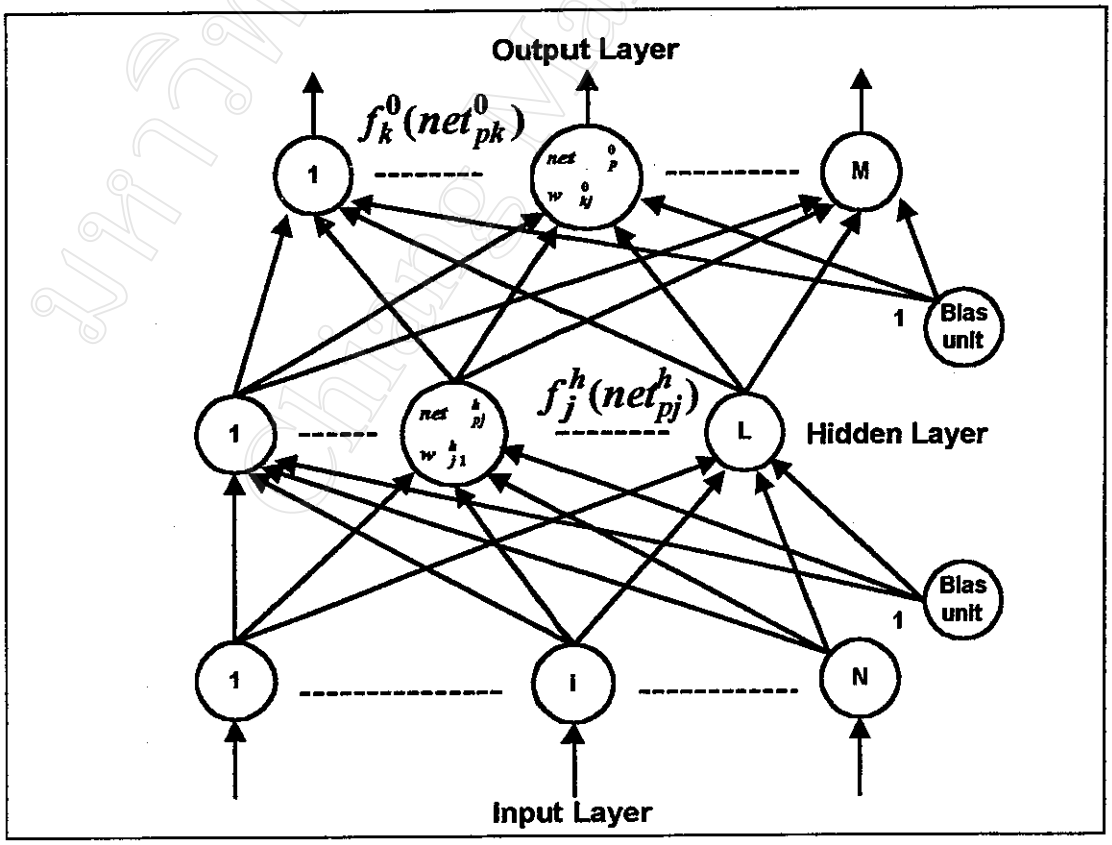
จะเห็นได้ว่า ในการเรียนรู้เพื่อรู้จำภาพด้วยโครงข่ายประสาทเทียมนั้นจะเป็นโครงข่ายแบบหลายชั้น ที่มีความซับซ้อน ซึ่งกระบวนการเรียนรู้ที่มีความเหมาะสมในการนำมาใช้คือ การเรียนรู้แบบแพร่กลับ (Backpropagation Learning)

3.2 การเรียนรู้แบบแพร่กลับ (Backpropagation Learning)

วิธีการนำค่าผิดพลาด (Error) ที่หน่วยเอาต์พุตให้ย้อนกลับ (Backpropagation) มายังชั้นซ่อน หรือที่เรียกว่า Backpropagation of Errors นั้นได้ค้นพบมาก่อนหน้านี้แล้ว โดย Werbos , David Paker และ Lecum ตั้งแต่ปี ค.ศ.1970 แต่ยังไม่เป็นที่นิยมแพร่หลายมากนัก ต่อมาในช่วง ค.ศ.1980 นั้นได้มีการพัฒนาขึ้นตามลำดับ ด้วยเหตุผลที่ว่าเพอร์เซ็ปตรอนแบบชั้นเดียว (Single Layer Perceptrons) ยังไม่สามารถแก้ปัญหาง่ายๆ อย่างเช่น Xor Function ได้ [9]และยังไม่มีวิธีการเรียนรู้สำหรับโครงข่ายแบบหลายชั้นที่เหมาะสม

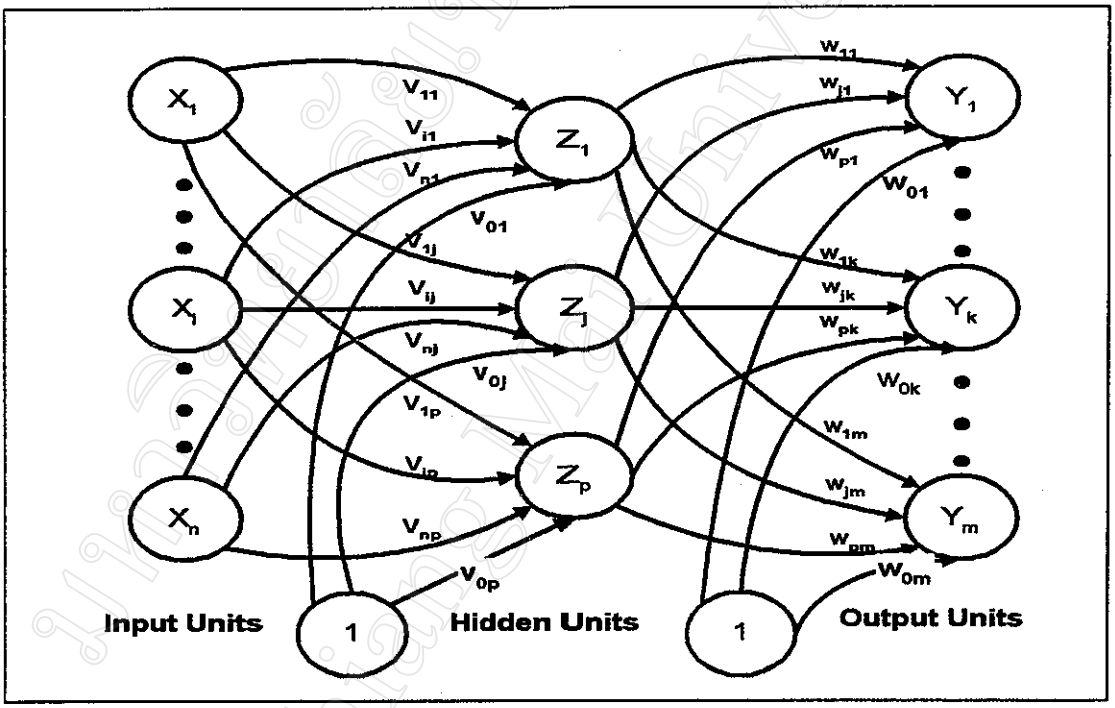
กระบวนการสำคัญของการเรียนรู้แบบแพร่กลับนี้มี 3 ขั้นตอน คือ การ Feedforward ของรูปแบบอินพุต (Input Pattern) , การคำนวณและส่งค่าผิดพลาดกลับคืน (Backpropagation of Errors) และการปรับค่าน้ำหนักให้เหมาะสม

3.3 สถาปัตยกรรมโครงข่ายของกระบวนการเรียนรู้แบบแพร่กลับ




รูป 3.2 โครงข่ายของกระบวนการเรียนรู้แบบแพร่กลับ [7]

สถาปัตยกรรมโครงข่ายของกระบวนการเรียนรู้แบบแพร่กลับ แสดงดังรูป 3.2 ประกอบไปด้วยเซลล์นิวรอนที่เรียงกันอย่างน้อย สาม ชั้น เพื่อทำให้โครงข่ายสามารถจำแนกแพทเทอร์นแบบไม่เป็นเชิงเส้น (Non Linear Separable) โดยใช้เทคนิคการเรียนรู้แบบแพร่กลับ ชั้นแรก เป็นชั้นอินพุต ชั้นถัดมา จะเป็นชั้นซ่อน ซึ่งชั้นนี้สามารถมีได้หลายๆ ชั้น และ ชั้นสุดท้าย เป็นชั้นเอาต์พุต ซึ่งในแต่ละชั้นจะมีการเชื่อมโยงถึงกันหมด และเซลล์นิวรอนจะเชื่อมโยงกับทุกๆ หน่วยก่อนหน้านี้และถัดไป แต่ไม่ได้เชื่อมโยงกันระหว่างเซลล์นิวรอนในชั้นเดียว ส่วน Bias Unit มีค่าเป็น +1 เสมอ โดยจะเชื่อมโยงกับทุกๆ โหนดในแต่ละชั้น




รูป 3.3 โครงข่ายประสาทเทียมแบบแพร่กลับที่มีจำนวนชั้นซ่อน 1 ชั้น [ 14 ]

## 3.4 หลักการพื้นฐานโดยทั่วไป

รูปที่ 3.3 แสดงตัวอย่างง่ายๆ ของโครงข่ายประสาทเทียมแบบแพร่กลับที่มีโครงสร้างแบบหลายชั้น และมีจำนวนชั้นซ่อน $Z$ จำนวน 1 ชั้น อยู่ระหว่างหน่วยเอาต์พุต $Y$ และหน่วยเอาต์พุต $X$

ค่าไบแอสที่ส่งให้หน่วยเอาต์พุต $Y_k$ แทนด้วย $w_{ok}$ และค่าไบแอสที่ส่งให้หน่วยชั้นซ่อน $Z_j$ แทนด้วย $v_{oj}$ ทิศทางการไหลของสัญญาณ จะไหลไปข้างหน้า เมื่ออยู่ในช่วงของการทำงาน แต่เมื่ออยู่ในช่วงแพร่กลับ (Backpropagation) สัญญาณจะไหลย้อนทิศทางเดิม (Reverse Direction)

ในระหว่างทิศทางการไหลของสัญญาณ ไหลไปข้างหน้า หน่วยอินพุตแต่ละตัว ( $X_i$ ) ได้รับสัญญาณและกระจายสัญญาณไปให้หน่วยชั้นซ่อนแต่ละตัว $Z_1$ ,..., $Z_p$ เพื่อทำการคำนวณ หาค่าระดับการกระตุ้น ( $z_j$ ) แล้วส่งให้หน่วยเอาต์พุตต่อไป จากนั้นหน่วยเอาต์พุตแต่ละตัว ($Y_k$ ) ก็จะคำนวณหาค่าระดับการกระตุ้น ( $y_k$ ) เพื่อจัดเป็นผลตอบสนองของโครงข่ายที่มีต่อรูปแบบอินพุตที่กำหนดให้

ในระหว่างการเรียนรู้ หน่วยเอาต์พุตแต่ละตัว จะทำการเปรียบเทียบค่าระดับการกระตุ้น ($y_k$ ) ที่คำนวณได้กับค่าเป้าหมาย (Target Value) $t_k$ เพื่อนำไปหาค่าผิดพลาดที่หน่วยนั้นๆ ค่าผิดพลาดนี้ใช้สัญลักษณ์เป็น $\delta_k$ ($k=1,...,m$) ค่า $\delta_k$ ที่เป็นค่าผิดพลาดของหน่วยเอาต์พุต $Y_k$ นี้ จะถูกส่งกลับไปให้ทุกหน่วยในชั้นก่อนหน้า หรือก็คือหน่วยชั้นซ่อน ที่เชื่อมต่อกับหน่วยเอาต์พุต $Y_k$ จากนั้นจึงนำไปปรับปรุงค่าน้ำหนัก ระหว่างชั้นเอาต์พุต กับชั้นซ่อน ต่อไป

ในทำนองเดียวกันค่าผิดพลาด $\delta_j$ ($j=1,...,p$) จะถูกคำนวณที่แต่ละหน่วยชั้นซ่อน $z_k$ แต่ไม่จำเป็นต้องส่งค่ากลับไปยังชั้นอินพุต เพียงแต่นำไปปรับปรุงค่าน้ำหนักระหว่างชั้นซ่อน และชั้นอินพุต เท่านั้น

เมื่อได้ค่าผิดพลาด $\delta$ แล้ว การปรับปรุงค่าน้ำหนักสำหรับทุกชั้น จะเกิดขึ้นอย่างทันที โดยการปรับปรุงค่าน้ำหนัก $w_{jk}$ จากหน่วยชั้นซ่อน $Z_k$ ไปยังชั้นเอาต์พุต $Y_k$ ภายใต้เงื่อนไขของค่าผิดพลาด $\delta_k$ และค่าระดับการกระตุ้น $z_j$ ของหน่วยชั้นซ่อน $Z_j$ ส่วนการปรับปรุงค่าน้ำหนัก $v_{ij}$ จากหน่วยอินพุต $X_1$ ไปยังหน่วยชั้นซ่อน $Z_k$ ทำภายใต้เงื่อนไขของค่าผิดพลาด $\delta_j$ และค่าระดับการกระตุ้น $x_1$ ของหน่วยอินพุต

ค่าตัวแปรต่างๆ ที่ใช้แทนในสมการ

| | |
|---|---|
| $x$ | เวคเตอร์อินพุตเทรนนิ่ง (Input Training Vector) $x=(x_1,...x_i,...,x_n)$ ...(3.1) |
| $t$ | เวคเตอร์เอาต์พุตเป้าหมาย $t=(t_1,...,t_k,...,t_m)$ ...(3.2) |
| $\delta_k$ | ค่าผิดพลาดเนื่องจากหน่วยเอาต์พุต $Y_k$ ที่แพร่กลับมายังหน่วยชั้นซ่อน เพื่อนำไปปรับปรุงค่าน้ำหนัก $w_{jk}$ |
| $\delta_j$ | ค่าผิดพลาดเนื่องจากหน่วยชั้นซ่อน $Z_j$ เพื่อนำไปปรับปรุงค่าน้ำหนัก $v_{ij}$ |
| $\alpha$ | อัตราการเรียนรู้ |

$X_i$ หน่วยอินพุตที่ $i$

สำหรับหน่วยอินพุต สัญญาณอินพุตและสัญญาณเอาต์พุตใช้สัญลักษณ์ $x_i$ เหมือนกัน

$v_{0j}$ ค่าไบแอสบนหน่วยชั้นซ่อนที่ $j$

$Z_j$ หน่วยชั้นซ่อนที่ $j$

อินพุตลัพธ์ที่ส่งไปให้ $Z_j$ ใช้สัญลักษณ์ว่า $z\_in_j$ โดย

$$z\_in_j = v_{oj} + \sum_i x_i v_{ij} \qquad \text{...(3.3)}$$

สัญญาณเอาต์พุต หรือค่าระดับการกระตุ้นของ $Z_j$ ใช้สัญลักษณ์ $z_j$

$$z_j = f(z\_in_j) \qquad \text{...(3.4)}$$

$w_{ok}$ ค่าไบแอสบนหน่วยเอาต์พุตที่ $k$

$Y_k$ หน่วยเอาต์พุตที่ $k$

อินพุตลัพธ์ที่ส่งไปให้ $Y_k$ ใช้สัญลักษณ์ว่า $y\_in_k$ โดย

$$y\_in_k = w_{0k} + \sum_j x_j w_{jk} \qquad \text{... (3.5)}$$

สัญญาณเอาต์พุตหรือค่าระดับการกระตุ้นของ $Y_k$ ใช้สัญลักษณ์ $y_k$

$$y_k = f(y\_in_k) \qquad \text{...(3.6)}$$

## 3.5 ฟังก์ชันการกระตุ้น (Activation Function)

ฟังก์ชันการกระตุ้นที่ใช้ในโครงข่ายประสาทเทียมแบบแพร่กลับนี้ มีคุณลักษณะที่สำคัญมากเช่น ควรจะเป็นฟังก์ชันที่มีความต่อเนื่องและไม่เป็นเชิงเส้นสามารถหาค่าอนุพันธ์ได้ และง่ายต่อการคำนวณซึ่งค่าอนุพันธ์สามารถเขียนในรูปเทอมของฟังก์ชันนั้น [10]

ฟังก์ชันที่นิยมใช้ทั่วไป คือฟังก์ชันไบนารีซิกมอยด์ (Binary Sigmoid Function) โดยมีค่าอยู่ในระหว่าง 0 ถึง 1 ตามนิยาม

$$f_1(x) = \frac{1}{1 + \exp(-x)} \qquad \text{...(3.7)}$$

ค่าอนุพันธ์เขียนในรูปเทอมของฟังก์ชัน

$$f_1'(x) = f_1(x)[1 - f_1(x)] \qquad \text{...(3.8)}$$

ฟังก์ชันการกระตุ้นอีกแบบ ที่นิยมใช้คือ ไบโพล่าร์ซิกมอยด์ (Bipolar Sigmoid) โดยมีค่าอยู่ในช่วง –1 ถึง 1 ตามนิยาม

$$f_2(x) = \frac{2}{1+\exp(-x)} - 1 \qquad \ldots(3.9)$$

และค่าอนุพันธ์เป็น

$$f_2' = \tfrac{1}{2}[1 + f_2(x)][1 - f_2(x)] \qquad \ldots(3.10)$$

ซึ่งจะมีความสัมพันธ์กับฟังก์ชัน

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \qquad \ldots(3.11)$$

3.6 <u>ขั้นตอนในกระบวนการเรียนรู้ (Training Algorithm)</u>

*ขั้นที่ 0* การกำหนดค่าน้ำหนักเริ่มต้นให้ตั้งค่าสุ่มที่น้อยมาก (Small Random Values)

*ขั้นที่ 1* ทดสอบเงื่อนไขของการสิ้นสุดการเรียนรู้ถ้ายังไม่เป็นไปตามเงื่อนไขที่กำหนดไว้ (ทดสอบแล้วเป็น เท็จ) ให้ทำตามขั้นตอนที่ 2 ถึง ขั้นตอนที่ 9

*ขั้นที่ 2* สำหรับแต่ละคู่ของการเรียนรู้ (Training Pair) ให้ทำตามขั้นตอนที่ 3 ถึงขั้นตอนที่ 8

ช่วง *feedforward*

*ขั้นที่ 3* แต่ละหน่วยอินพุต $(X_i, i=1,\ldots, n)$ ได้รับสัญญาณอินพุต $X_i$ และกระจายสัญญาณไปให้หน่วยชั้นซ่อน

*ขั้นที่ 4* แต่ละหน่วยชั้นซ่อน $(Z_j, j=1,\ldots, p)$ จะรวมผลคูณของค่าน้ำหนักกับสัญญาณอินพุต ดังนี้

$$z\_in_j = v_{oj} + \sum_{i=1}^{n} x_i v_{ij} \qquad \ldots(3.12)$$

จากนั้นนำฟังก์ชันการกระตุ้น ไปคำนวณ ค่าสัญญาณเอาต์พุต เพื่อส่งสัญญาณนี้ให้ทุกหน่วย ในชั้นถัดไป (หน่วยเอาต์พุต) ตามสมการ

$$z_j = f(z\_in_j) \qquad \ldots(3.13)$$

*ขั้นที่ 5* แต่ละหน่วยเอาต์พุต $(Y_k, k = 1,...,m)$ รวมค่าผลคูณของค่าน้ำหนักกับสัญญาณอินพุต จะได้เป็น

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk} \qquad ...(3.14)$$

จากนั้น นำฟังก์ชันการกระตุ้น ไปคำนวณค่าระดับการกระตุ้น เพื่อเป็นสัญญาณเอาต์พุตต่อไป

ช่วง *backpropagation of error*

*ขั้นที่ 6* แต่ละหน่วยเอาต์พุต $(Y_k, k = 1,...,m)$ ได้รับรูปแบบเป้าหมาย (Target Pattern) ที่มีความสัมพันธ์กับ รูปแบบอินพุต (Input Training Pattern) แล้วนำไปคำนวณค่าผิดพลาด (Error Information Terms : $\delta_k$) โดยค่า $\delta_k$ นี้จะถูกส่งไปยังทุกหน่วยของชั้นที่ต่ำกว่า

$$\delta_k = (t_k - y_k) f'(y\_in_k) \qquad ...(3.15)$$

คำนวณค่าน้ำหนักที่ถูกต้อง ( Weight Correction Term : $\Delta w_{ij}$ ) รวมถึงค่าไบแอสที่ถูกต้อง (Bias correction Term : $\Delta w_{ok}$ ) เพื่อนำไปปรับปรุงค่าน้ำหนักต่อไป

$$\Delta w_{jk} = \alpha \delta_k z_j \qquad ...(3.16)$$

$$\Delta w_{ok} = \alpha \delta_k \qquad ...(3.17)$$

*ขั้นที่ 7* แต่ละหน่วยชั้นซ่อน $(Z_j, j = 1,..., p)$ จะรวมค่าเดลต้าอินพุต (Delta input) ของหน่วยนั้นๆ ที่ได้รับจากชั้น ในชั้นที่สูงกว่าตามความสัมพันธ์

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \qquad ...(3.18)$$

จากนั้น นำไปคำนวณค่าผิดพลาด ของแต่ละหน่วยจากสมการ

$$\delta_j = \delta\_in_j f'(z\_in_j) \qquad ...(3.19)$$

แล้วนำไปคำนวณหาค่าน้ำหนักและค่าไบแอสที่ถูกต้องเพื่อนำไปปรับปรุงค่าน้ำหนักต่อไป

$$\Delta v_{ij} = \alpha \delta_j x_i \qquad \text{...(3.20)}$$

$$\Delta v_{oj} = \alpha \delta_j \qquad \text{...(3.21)}$$

ช่วง *Update Weights and Biases*

*ขั้นที่ 8* แต่ละหน่วยเอาต์พุต $(Y_k, k = 1,...,m)$ จะทำการปรับค่าไบแอสและค่าน้ำหนัก $(j = 0,...,p)$ ดังนี้

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \qquad \text{...(3.22)}$$

แต่ละหน่วยชั้นซ่อน $(Z_j, j = 1,...,p)$ ก็ทำการปรับค่าไบแอสและค่าน้ำหนัก $(i = 0,...,n)$ ดังนี้

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij} \qquad \text{...(3.23)}$$

*ขั้นที่ 9* สภาวะหยุดการทดสอบ

## 3.7 การเลือกค่าน้ำหนัก และค่าไบแอส เริ่มต้น

การเลือกค่าน้ำหนักเริ่มต้นจะมีอิทธิพลต่อการเกิดค่าผิดพลาดที่น้อยที่สุดของโครงข่ายได้ ปกติแล้วการปรับปรุงค่าน้ำหนักระหว่างสองหน่วยจะขึ้นอยู่กับ ค่าอนุพันธ์ของฟังก์ชันการกระตุ้นของหน่วยที่สูงกว่า กับหน่วยที่ต่ำกว่า ดังนั้นจึงเป็นเหตุผลสำคัญ ที่ต้องหลีกเลี่ยงค่าน้ำหนัก ที่จะทำค่าระดับการกระตุ้น และค่าอนุพันธ์ของค่าระดับการกระตุ้น เป็นศูนย์

ค่าน้ำหนักเริ่มต้น ไม่ควรเป็นค่าที่ใหญ่มากนัก จะเป็นผลทำให้ค่าอนุพันธ์ของฟังก์ชันการกระตุ้น แบบฟังก์ชันซิกมอยด์มีค่าเล็กมาก เรียกว่าอยู่ในย่านของการอิ่มตัว (Saturation Region) แต่ถ้าค่าน้ำหนักเริ่มต้นมีค่าเล็กเกินไป ค่าอินพุตลัพธ์ที่จะส่งไปยังหน่วยชั้นซ่อน หรือหน่วยเอาต์พุต จะมีค่าเข้าใกล้ศูนย์ ซึ่งเป็นสาเหตุให้การเรียนรู้ทำได้ช้า

โดยทั่วไปค่าน้ำหนักเริ่มต้น และค่าไบแอสเริ่มต้นจะสุ่มค่าระหว่าง –0.5 ถึง 0.5 (หรือระหว่าง –1 ถึง 1 ตามความเหมาะสม) สำหรับการสุ่มค่าเริ่มต้นที่นิยม จะใช้วิธีที่เรียกว่า NguyenWidrow Intialization [11] ซึ่งเป็นวิธีที่สามารถทำให้โครงข่ายเกิดการเรียนรู้ที่รวดเร็วขึ้นได้ โดยมีนิยามดังนี้

$$\beta = 0.7(p)^{1/n} = 0.7\sqrt[n]{p} \qquad \text{...(3.24)}$$

เมื่อ $n$ คือ จำนวนของหน่วยอินพุต

$p$ คือ จำนวนของหน่วยชั้นซ่อน

$\beta$ คือ ค่าสเกลแฟคเตอร์ (Scale Factor)

กำหนดเวคเตอร์ค่าน้ำหนักจากหน่วยอินพุตไปยังแต่ละหน่วยชั้นซ่อน $(j = 1,\ldots,p)$ เป็นค่าสุ่มระหว่าง –0.5 ถึง 0.5 หรือกำหนดเป็นระหว่างค่า $-\gamma$ ถึง $\gamma$ ดังนี้

$$v_{ij}(old) = \text{Random Number Between } -\gamma \text{ and } \gamma \qquad \ldots(3.25)$$

คำนวณขนาดของเวคเตอร์ค่าน้ำหนัก

$$\| v_{ij}(old) \| \qquad \ldots(3.26)$$

คำนวณค่าน้ำหนักเริ่มต้นได้จาก

$$v_{ij} = \frac{\beta v_{ij}(old)}{\| v_{ij}(old) \|} \qquad \ldots(3.27)$$

กำหนดค่าไบแอสเริ่มต้นเป็น

$$v_{oj} = \text{Random Number Between } -\beta \text{ and } \beta \qquad \ldots(3.28)$$

3.8 <u>ระยะเวลาที่ใช้ในการเรียนรู้</u>

ในการเรียนรู้โครงข่าย สิ่งที่เราต้องการคือความสมดุล (Balance) ระหว่างการตอบสนองที่ถูกต้องต่อรูปแบบการเรียนรู้ (Training Pattern) และการตอบสนองที่ดีต่อรูปแบบอินพุต (Input Pattern) ชุดใหม่ หรือเป็นความสมดุลระหว่างความสามารถด้าน Memorization และความสามารถด้าน Generalization ดังนั้นจึงไม่จำเป็นว่าจะต้องทำการสอน จนกระทั่งค่าผลรวมของค่ากำลังสองของค่าผิดพลาด (Total Square Error) มีค่าน้อยที่สุด แล้วจึงสิ้นสุดการสอน

ข้อแนะนำประการหนึ่งคือให้ใช้ข้อมูล 2 ชุดในระหว่างการเรียนรู้ ชุดหนึ่งเป็นรูปแบบการเรียนรู้ ส่วนอีกชุดหนึ่งเป็นรูปแบบทดสอบการเรียนรู้ (Training Testing Pattern) โดยในการปรับปรุงค่าน้ำหนักให้ใช้ชุดข้อมูลรูปแบบการเรียนรู้ ส่วนการคำนวณค่าผิดพลาดจะใช้รูปแบบทดสอบการเรียนรู้

เงื่อนไขของระยะเวลาในการเรียนรู้จะกำหนดว่า ถ้าค่าผิดพลาดของรูปแบบทดสอบการเรียนรู้มีค่าลดลง ให้ทำการสอนต่อไป แต่เมื่อค่าผิดพลาดนี้เริ่มต้นที่จะมีค่าเพิ่มขึ้น ให้สิ้นสุดการสอนทันที หรือกล่าวว่าโครงข่ายเริ่มมีความสามารถในด้าน Memorization มากขึ้นและเริ่มจะสูญเสียความสามารถในด้าน Generalization ของตัวเองไป

3.9 ค่าโมเมนตัม (Momentum)

หลักการของค่าโมเมนตัม ($\mu$) อาศัยเหตุผลที่ว่าในกรณีที่ค่าข้อมูลในการเรียนรู้บางค่า มีความแตกต่างมากเมื่อเทียบกับค่าข้อมูลส่วนใหญ่ เพื่อหลีกเลี่ยงการกระจายตัวของข้อมูลเช่นนี้จึงต้องใช้ค่าอัตราการเรียนรู้ ($\alpha$) ที่มีค่าน้อยๆ แต่การทำเช่นนี้จะส่งผลให้ใช้ระยะเวลายาวนาน ในการเรียนรู้ เพราะค่าน้ำหนักที่ถูกต้อง (Weight Correction Term) มีความละเอียดมาก โครงข่ายจึงต้องมีการปรับปรุงค่าน้ำหนักหลายครั้ง ดังนั้นเพื่อให้กระบวนการเรียนรู้ เกิดการลู่เข้า(Convergence) ได้เร็วขึ้น จึงต้องมีการเพิ่มค่าโมเมนตัมเพิ่มเข้าไปในสมการของการปรับปรุงค่าน้ำหนัก

ในการนำค่าโมเมนตัมมาใช้ในการเรียนรู้นั้น ต้องมีการเก็บข้อมูล (Save) ของค่าน้ำหนักที่ปรับปรุงแล้ว สำหรับรูปแบบการเรียนรู้ก่อนหน้านี้ไว้ด้วย หรือ ค่าน้ำหนักค่าใหม่ที่ได้จากขั้นตอนที่ $t+1$ จะได้มาจากการเรียนรู้ในขั้นตอนที่ $t$ และ $t-1$ ดังนั้นจะได้สมการสำหรับการปรับปรุงค่าน้ำหนักเมื่อมีการเพิ่มค่าโมเมนตัมเข้ามาเป็น

$$w_{jk}(t+1) = w_{jk}(t) + \alpha\delta_k z_j + \mu[w_{jk}(t) - w_{jk}(t-1)] \quad \ldots(3.29)$$

หรือ

$$\Delta w_{jk}(t+1) = \alpha\delta_k z_j + \mu\Delta w_{jk}(t) \quad \ldots(3.30)$$

และ

$$v_{ij}(t+1) = v_{ij}(t) + \alpha\delta_j x_i + \mu[v_{ij}(t) - v_{ij}(t-1)] \quad \ldots(3.31)$$

หรือ

$$\Delta v_{ij}(t+1) = \alpha\delta_j x_i + \mu\Delta v_{ij}(t) \quad \ldots(3.32)$$

เมื่อ $\mu$ คือพารามิเตอร์ค่าโมเมนตัม ที่มีค่าจำกัดอยู่ระหว่างค่า 0 ถึง 1

3.10 การปรับค่าอัตราการเรียนรู้ (Adaptive Learning Rate)

อีกวิธีหนึ่งที่มีการนำมาใช้เพื่อให้การเรียนรู้ เกิดความรวดเร็วมากขึ้น คือการเปลี่ยนแปลงค่าอัตราการเรียนรู้ ($\alpha$) ในระหว่างการเรียนรู้ เรียกว่ากฎ Delta-Bar-Delta โดยที่จะให้ค่าน้ำหนักแต่ละตัวมีค่าอัตราการเรียนรู้ของตัวเอง ถ้าค่าน้ำหนักเปลี่ยนแปลงในทิศทางเดียวกันติดต่อกันหลายครั้ง อาจจะเพิ่มขึ้นเหมือนกันหรือลดลงเหมือนกัน ค่าอัตราการเรียนรู้สำหรับค่าน้ำหนักนั้น ควรจะเพิ่มขึ้นด้วย แต่ถ้าทิศทางของการเปลี่ยนแปลงค่าน้ำหนักเปลี่ยนไป คือเพิ่มขึ้นหรือลดลงสลับกัน ค่าอัตราการเรียนรู้ควรจะมีค่าลดลง

กฎ Delta – Bar – Delta นี้เป็นการรวมกฎของการปรับปรุงค่าน้ำหนัก (Weight Update Rule) กับกฎของการปรับปรุงค่าอัตราการเรียนรู้ (Learning Rate Update Rule) เข้าด้วยกัน [11] โดยมีการกำหนดให้

$w_{ij}(t)$ เป็นค่าน้ำหนักที่เวลา $t$

$\alpha_{ij}(t)$ เป็นค่าอัตราการเรียนรู้ สำหรับค่าน้ำหนัก $w_{ij}(t)$ ที่เวลา $t$

$E$ เป็นค่าผิดพลาดกำลังสอง (Square Error) สำหรับรูปแบบนั้นที่เวลา $t$

กฎ Delta-Bar-Delta จะให้การเปลี่ยนแปลงค่าน้ำหนักเป็นไปตามสมการ

$$w_{jk}(t+1) = w_{jk}(t) - \alpha_{jk}(t+1)\frac{\partial E}{\partial w_{jk}} \qquad \ldots(3.33)$$

$$= w_{jk}(t) + \alpha_{jk}(t+1)\delta_k z_j \qquad \ldots(3.34)$$

การเปลี่ยนแปลงค่าน้ำหนักเช่นนี้เป็นมาตรฐานในการเรียนรู้แบบแพร่กลับ ด้วยการนำค่าอนุพันธ์ย่อย ของค่าผิดพลาดเทียบกับค่าน้ำหนักมาใช้ในการคำนวณด้วย

สำหรับแต่ละหน่วยเอาต์พุต จะมีนิยาม "Delta"

$$\Delta_{jk} = \frac{\partial E}{\partial w_{jk}} = -\delta_k z_j \qquad \ldots(3.35)$$

และสำหรับแต่ละหน่วยชั้นซ่อน

$$\Delta_{jk} = \frac{\partial E}{\partial w_{jk}} = -\delta_j x_i \qquad \ldots(3.36)$$

นิยามค่า "Delta-Bar" สำหรับแต่ละหน่วยเอาต์พุต

$$\bar{\Delta}_{jk} = (1-\beta)\Delta_{jk}(t) + \beta\bar{\Delta}_{jk}(t-1) \qquad \ldots(3.37)$$

และสำหรับแต่ละหน่วยชั้นซ่อน

$$\bar{\Delta}_{ij} = (1-\beta)\Delta_{ij}(t) + \beta\bar{\Delta}_{ij}(t-1) \qquad \ldots(3.38)$$

ค่าของพารามิเตอร์นี้จะกำหนดโดยผู้ใช้ ปกติมีค่าอยู่ในช่วง $0 < \beta < 1$

ค่าอัตราการเรียนรู้ใหม่จะถูกกำหนดให้เป็น

$$\alpha_{jk}(t+1)\begin{cases} \alpha_{jk}(t)+K & if\ \bar{\Delta}_{jk}(t-)\Delta_{jk} > 0 \\ (1-\gamma)\alpha_{jk}(t) & if\ \bar{\Delta}_{jk}(t-)\Delta_{jk} > 0 \\ \alpha_{jk}(t) & otherwise \end{cases} \quad \text{...(3.39)}$$

เช่นเดียวกับค่าพารามิเตอร์ $K$ และ $\gamma$ จะกำหนดโดยผู้ใช้งาน