บทที่ 2

แนวคิดและทฤษฎี

1. ตัวอักษรภาษาไทย

ตัวอักษรภาษาไทยจะประกอบด้วยตัวอักษรพยัญชนะ, ตัวอักษรสระ และตัววรรณยุกต์ ซึ่งสามารถแยกออกได้เป็นดังนี้

1.1. อักษรพยัญชนะ มีทั้งหมด 44 ตัวอักษร แต่ที่ใช้งานในปัจจุบันมี 42 ตัวอักษร และไม่ได้ใช้งาน 2 ตัวอักษรคือ ฃ, ฅ

1.2. อักษรสระ สามารถแบ่งแยกเป็น สระระดับบน, สระระดับล่าง และสระระดับพยัญชนะ

1.3. อักษรวรรณยุกต์ มีทั้งหมด 4 ตัว

1.4. อักษรเลขไทย มีทั้งหมด 10 ตัว

1.5. เครื่องหมายต่างๆ

ส่วนหัวของอักษรภาษาไทยนั้น ส่วนมากขึ้นต้นด้วยส่วนหัวที่เป็นวงกลมที่มีลักษณะแตกต่างกันซึ่งมี วงกลมหัวเข้า, วงกลมหัวออก หรือวงกลมหัวหยัก เช่น ผ ป ฃ อีกทั้งตำแหน่งของหัวก็มีทั้งด้านบนซ้าย ล่างซ้าย ขวาบน และกึ่งกลางตัวอักษร เช่น บ ถ ง ด และในบางครั้งตัวอักษรที่มีหัวเหล่านี้ก็สามารถเขียนแบบไม่มีหัว โดยที่ความหมายไม่เปลี่ยนแปลง เช่น ท, ด, ง กับ ท, ด, ง ระดับของตัวอักษรภาษาไทยสามารถแบ่งระดับการเขียนออกเป็น 4 ระดับคือ

1. ระดับอักษรวรรณยุกต์ ได้แก่ ่ ้ ๊ ๋
2. ระดับอักษรสระบน คือสระที่มีตำแหน่งเหนือพยัญชนะ เช่น ิ ี ึ ื
3. ระดับอักษรพยัญชนะ, ตัวเลข, สัญลักษณ์พิเศษ หรือสระที่มีระดับเดียวกับพยัญชนะ โดยสามารถแยกย่อยได้ดังนี้

- พยัญชนะทั่วไป เช่น ก, ต, พ
- พยัญชนะที่เลยขึ้นไปในระดับสระบน เช่น ป, ฟ, ฝ
- พยัญชนะที่เลยลงมาในระดับสระล่าง เช่น ฎ, ฏ
- พยัญชนะที่แยกออกเป็น 2 ส่วน เช่น ญ, ฐ

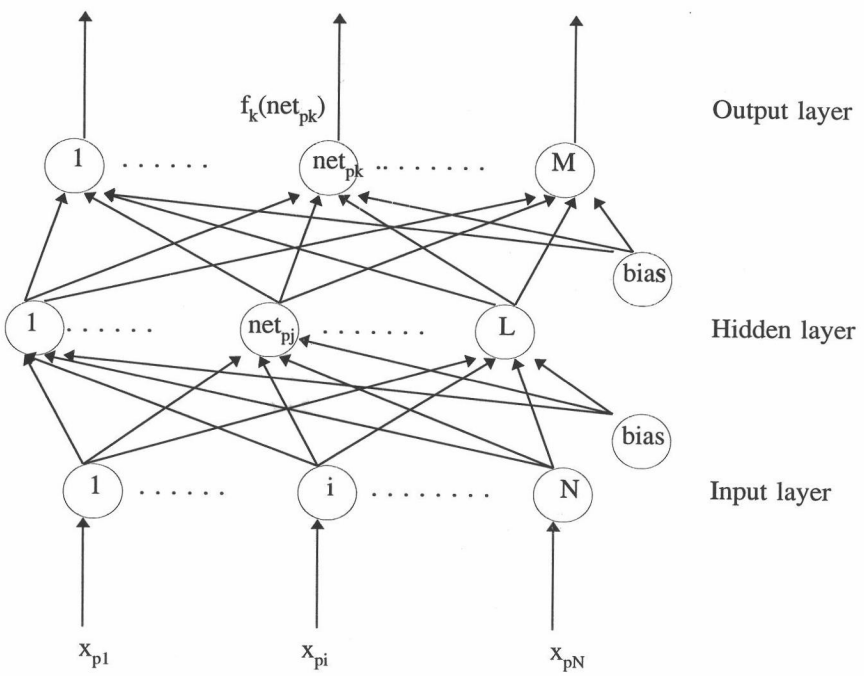
4. ระดับอักษรสระล่าง คือสระที่มีตำแหน่งต่ำกว่าพยัญชนะ เช่น ุ ู

จากการแบ่งระดับของตำแหน่งของตัวอักษร จะทำให้การจำแนกตัวอักษรที่มีลักษณะคล้ายกันแต่อยู่ต่างระดับกันทำได้ง่ายขึ้น เช่น อักษร ข, สระอู และ ไม้โท ดังนั้น ถ้าสามารถกำหนดเส้นฐาน (baseline) ของตัวอักษรก็สามารถจำแนกตัวอักษรตามระดับได้ ซึ่งจะทำให้การรู้จำตัวอักษรถูกต้องมากขึ้น พิพัฒน์ และ มนลดา ได้แบ่งกลุ่มของอักษรพยัญชนะภาษาไทยที่มีลักษณะรูปร่างคล้ายคลึงกันไว้ดังนี้

กลุ่มที่ 1	ได้แก่	ก, ถ, ภ, ฏ, ฎ, ฤ
กลุ่มที่ 2	ได้แก่	ค, ด, ศ, ต
กลุ่มที่ 3	ได้แก่	ม, ฆ
กลุ่มที่ 4	ได้แก่	บ, ป, ษ, ข, ช, ซ
กลุ่มที่ 5	ได้แก่	ร, ธ
กลุ่มที่ 6	ได้แก่	ณ, ฒ, ญ
กลุ่มที่ 7	ได้แก่	ฉ, น
กลุ่มที่ 8	ได้แก่	พ, ฟ, ฬ, ผ, ฝ
กลุ่มที่ 9	ได้แก่	ท, ห
กลุ่มที่ 10	ได้แก่	ล, ส, จ
กลุ่มที่ 11	ได้แก่	อ, ฮ, ว

2. นิวรอลเน็ตเวิร์ก (Neural Networks)

ระบบนิวรอลเน็ตเวิร์ก เป็นระบบที่พยายามเลียนแบบโครงสร้างเซลล์สมองของมนุษย์ โดยเป็นการประยุกต์ใช้ความรู้และเทคโนโลยีบางส่วนของวิชาชีววิทยา และชีวฟิสิกส์ที่เกี่ยวข้องกับเซลล์สมองของมนุษย์ (Neuro-physiology) รูปแบบของระบบนิวรอลเน็ตเวิร์กมีอยู่ด้วยกันหลายชนิด เช่น ADALINE, MADALINE, Preceptron, Hopfield Network, Backpropagation, Counterpropagation, Self-Organizing Map, Adaptive Resonance Theory, Neocognitron ฯลฯ ซึ่งสามารถแบ่งแยกได้เป็น 2 ชนิด ตามลักษณะการเรียนรู้ของระบบ คือ Supervised Learning และ Unsupervised Learning แต่ในงานวิจัยนี้จะใช้วิธีการของ Backpropagation เป็นหลัก เนื่องจากเป็นรูปแบบของระบบนิวรอลเน็ตเวิร์กที่มีความเหมาะสมกับปัญหาด้านการรู้จำตัวอักษร หรือ complex pattern-matching (Freeman and Skapura, 1992) ดังรูปที่ 2.1แสดงโครงสร้างของระบบนิวรอลเน็ตเวิร์กแบบ Backpropagation



รูปที่ 2.1 แสดง Three-layer Backpropagation Network (BPN)

การทำงานของนิวรอลเน็ตเวร์กแบบ Backpropagation จะเริ่มจากการฝึกระบบ (Network training) โดยใช้คู่ตัวอย่างของข้อมูลฝึกอินพุตและเอาต์พุต (input-output example pairs) ในการฝึกระบบ ในขั้นตอนแรกจะป้อนตัวอย่างข้อมูลฝึกอินพุตให้กับระดับข้อมูลเข้า (Input layer) จากนั้นข้อมูลฝึกอินพุตจะแพร่กระจายไปยังระดับซ่อนตัว (Hidden layer) จนถึงระดับแสดงผล (Output layer) ผลที่ได้ (actual output) จะถูกเปรียบเทียบกับตัวอย่างของผลที่ต้องการ (desired output) เพื่อคำนวณหาค่าความผิดพลาด (error signal) ของแต่ละหน่วยแสดงผล (Output unit) ค่าความผิดพลาดจะถูกแพร่กระจายย้อนจากระดับแสดงผล (Output layer) ลงไปเรื่อยๆจนถึงระดับรับข้อมูลเข้า (Input layer) เพื่อทำการปรับน้ำหนักการเชื่อมต่อ (connection weight) ระหว่างโหนดของแต่ละระดับ (layer) จากนั้นก็จะเริ่มทำซ้ำโดยป้อนตัวอย่างข้อมูลฝึกอินพุตให้กับระดับข้อมูลเข้า (Input layer) เปรียบเทียบผลที่ได้ (actual output) กับตัวอย่างข้อมูลผลที่ต้องการ (desired output) และปรับปรุงน้ำหนักการเชื่อมต่อ (connection weight) ของโหนดแต่ละระดับ (layer) กระบวนการฝึกดังกล่าวนี้จะถูกทำซ้ำไปเรื่อยๆ จนกว่าการฝึกจนสิ้นสุด (ค่าความผิดพลาดอยู่ในระดับที่ยอมรับได้)

เมื่อการฝึกสิ้นสุดลงน้ำหนักการเชื่อมต่อ (connection weight) จะเปรียบเสมือนกับความรู้ของระบบที่ได้รับการฝึกจากคู่ตัวอย่างของข้อมูลฝึกอินพุตและเอาต์พุต (input-output example pairs) ในกรณีที่ข้อมูลอินพุตที่ระบบไม่เคยฝึกมาก่อน (arbitrary input pattern) ถูกป้อนให้กับระดับข้อมูลเข้า (Input layer) ข้อมูลอินพุตดังกล่าวจะแพร่ไปในแต่ละระดับ (layer) เพื่อคำนวณผลในระดับแสดงผล (Output layer) โดยผลที่ได้จะขึ้นอยู่กับคู่ตัวอย่างของข้อมูลฝึกอินพุตและเอาต์พุต (input-output example pairs) ที่ระบบได้รับการฝึก กล่าวคือผลที่ได้จากข้อมูลอินพุตที่ระบบไม่ได้รับการฝึกมาก่อน (arbitrary input pattern) จะได้ข้อมูลเอาต์พุตที่คล้ายคลึงกับข้อมูลฝึกเอาต์พุตของคู่ฝึกตัวอย่าง ที่มีข้อมูลอินพุตที่ระบบไม่เคยฝึกมาก่อน (arbitrary input pattern) เหมือนกับข้อมูลฝึกอินพุตของคู่ฝึกตัวอย่าง การทำงานของนิวรอลเน็ตเวิร์กชนิดนี้ จะมีลักษณะเหมือนการสร้างความสัมพันธ์ระหว่างข้อมูลอินพุตกับข้อมูลเอาต์พุต (mapping network)

จากรูปที่ 2.1 แสดง Three-layer Backpropagation Network (BPN) 3 layer หลักคือ

- ระดับข้อมูลเข้า (Input layer)
- ระดับซ่อนตัว (Hidden layer)
- ระดับแสดงผล (Output layer)

กำหนด $X_P = (x_{P1}, x_{P2}, \ldots, x_{PN})^t$ เป็นข้อมูล (Input vector) ที่ถูกป้อนให้กับระดับข้อมูลเข้า (Input layer) ดังนั้นจะได้สมการ net input ที่ j th ของระดับซ่อนตัว (Hidden layer) เป็น

$$\text{net}_{pj}^{h} = \sum_{i=1}^{N} w_{ji}^{h} x_{pi} + \theta_{j}^{h}$$

โดยที่ w_{ji}^{h} เป็นน้ำหนัก (weight) ของการเชื่อมต่อ (connection) จาก i th (input unit), θ_{j}^{h} เป็นค่าเอนเอียง (bias) ที่ป้อนให้แก่ระบบนิวรอลเน็ตเวิร์ก, h เป็นจำนวนของระดับซ่อนตัว (Hidden layer) โดยที่รายละเอียดของค่าพารามิเตอร์เหล่านี้ จะกล่าวถึงต่อไปภายหลัง ดังนั้นจะได้ผล (Output) ของระดับซ่อนตัว (Hidden layer) เป็น

$$i_{pj} = f_{j}^{h}(\text{net}_{pj}^{h})$$

และจะได้สมการของระดับแสดงผล (Output layer) ดังนี้

$$\text{net}_{pk}^{o} = \sum_{j=1}^{L} w_{kj}^{o} i_{pj} + \theta_{k}^{o}$$

$$o_{pk} = f_{k}^{o}(\text{net}_{pk}^{o})$$

โดยที่ o จะหมายถึงจำนวนของระดับแสดงผล (Output layer)

2.1 การปรับน้ำหนักการเชื่อมต่อของระดับแสดงผล (Update of Output-layer Weight)

กำหนดให้ y_{pk} เป็นผลที่ต้องการ (desired output) และ o_{pk} เป็นผลที่แท้จริง (actual output) ดังนั้นจะได้ค่าผิดพลาด (δ_{pk}) ที่หน่วยแสดงผลที่จุดใดจุดหนึ่ง (single output unit) เป็น

$$_{pk} = (y_{pk} - o_{pk})$$

โดยที่ p หมายถึง pth ของ องค์ประกอบของแบบฝึก (training vector) และ k หมายถึง kth ของหน่วยแสดงผล (output unit)

เนื่องจากค่าความผิดพลาด (error) จะถูกทำให้มีค่าน้อยที่สุดโดยวิธีของ generalized data rule (GDR) (Freeman and Skapura, 1992) ซึ่งค่าเฉลี่ยยกกำลังสองของค่าผิดพลาด (mean square error) หรือค่าผิดพลาดที่คาดหวังในหน่วยแสดงผล (Output unit) ทั้งหมดจะเป็น

$$E_p = \frac{1}{2}\sum_{k=1}^{M}\delta_{pk}^{2}$$

การกำหนดทิศทางในการปรับเปลี่ยนน้ำหนักการเชื่อมต่อนั้น สามารถคำนวณได้จาก negative of gradient ของ E_P (∇E_P) เทียบกับน้ำหนัก (weight) w_{kj}

$$\frac{\partial E_p}{\partial w_{kj}^{o}} = -(y_{pk} - o_{pk})\frac{\partial f_k^{o}}{\partial(\text{net}_{pk}^{o})}\frac{\partial(\text{net}_{pk}^{o})}{\partial w_{kj}^{o}}$$

$$\frac{\partial(\text{net}_{pk}^{o})}{\partial w_{kj}^{o}} = \left(\frac{\partial}{\partial w_{kj}^{o}}\sum_{j=1}^{L} w_{kj}^{o} i_{pj} + \theta_k^{o}\right) = i_{pj}$$

จากสมการข้างบนจะได้ negative gradient เป็น

$$-\frac{\partial E_p}{\partial w_{kj}^{o}} = (y_{pk} - o_{pk}) f_k^{o}{}'(\text{net}_{pk}^{o}) i_{pj}$$

การปรับปรุงน้ำหนักการเชื่อมต่อที่ระดับแสดงผล (output layer) จะเป็นสัดส่วนกับ negative gradient ดังนั้นจะได้ค่าน้ำหนักการเชื่อมต่อใหม่ดังนี้

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \Delta_p w_{kj}^{o}(t)$$

โดยที่

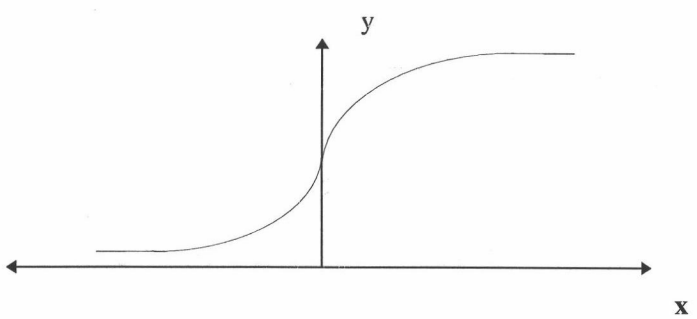
$$\Delta_p w_{kj}^{o} = \eta(y_{pk} - o_{pk}) f_k^{o}{}'(\text{net}_{pk}^{o}) i_{pj}$$

η คือ learning-rate parameter ซึ่งจะมีค่าเป็นบวกและมีค่าน้อยกว่า 1

ฟังก์ชัน f_k^{o}' จะต้องเป็นฟังก์ชันที่สามารถหาค่าอนุพันธ์ได้ (differentiable) โดยที่เอาต์พุตฟังก์ชันนั้น จะมีใช้งานกันอย่างแพร่หลายอยู่ 2 ชนิด คือ

- $f_k^o(\text{net}_{jk}^o) = \text{net}_{jk}^o$
- $f_k^o(\text{net}_{jk}^o) = (1+e^{-\text{net}_{jk}^o})^{-1}$

ฟังก์ชันแรกจะเป็นลักษณะการกำหนดเอาต์พุตแบบเชิงเส้น (linear output unit) ฟังก์ชันที่สองเรียกว่า sigmoid หรือ logistic function ดังแสดงในรูปที่ 2.2 การเลือกใช้เอาต์พุตฟังก์ชันนั้นจะขึ้นอยู่กับผล (output data) ที่ต้องการ เช่น ถ้าต้องการผล (output unit) เป็นสัญญาณเชิงทวิภาค (binary) จะใช้ sigmoid output function เนื่องจากเป็นฟังก์ชันที่มีลักษณะ output-limiting หรือ quasi-bistable แต่สามารถหาค่าอนุพันธ์ได้



รูปที่ 2.2 แสดง sigmoid function

กำหนดให้

$$\delta_{pk}^{o} = (y_{pk} - o_{pk}) f_k^{o}{}'(\text{net}_{pk}^o)$$

$$= \delta_{pk} f_k^{o}{}'(\text{net}_{pk}^o)$$

ดังนั้นจะได้สมการของการปรับปรุงน้ำหนักการเชื่อมต่อ (weight-update equation) เป็น

$$w_{kj}^o(t+1) = w_{kj}^o(t) + \eta \delta_{pk}^{o} i_{pj}$$

2.2 การปรับน้ำหนักการเชื่อมต่อของระดับซ่อนตัว (Update of Hidden-layer Weight)

การปรับปรุงน้ำหนักการเชื่อมต่อในระดับซ่อนตัว (Hidden layer) นี้จะมีลักษณะคล้ายคลึงกับการปรับปรุงน้ำหนักการเชื่อมต่อในระดับแสดงผล (Output layer) โดยจะเริ่มคำนวณจากค่าความผิดพลาดทั้งหมด (total square error) E_P

$$E_p = \frac{1}{2}\sum_k (y_{pk} - o_{pk})^2$$

$$= \frac{1}{2}\sum_k (y_{pk} - f_k^o(\text{net}_{pk}^o))^2$$

$$= \frac{1}{2}\sum_k (y_{pk} - f_k^o(\sum_j w_{kj}^o i_{pj} + \theta_k^o))^2$$

คำนวณ gradient ของ E_P เทียบกับน้ำหนัก (weight) ของระดับซ่อนตัว (Hidden layer) จะได้

$$\frac{\partial E_p}{\partial w_{ji}^h} = \frac{1}{2}\sum_k \frac{\partial}{\partial w_{ji}^h}(y_{pk} - o_{pk})^2$$

$$= -\sum_k (y_{pk} - o_{pk}) \frac{\partial o_{pk}}{\partial (\text{net}_{pk}^o)} \frac{\partial (\text{net}_{pk}^o)}{\partial i_{pj}} \frac{\partial i_{pj}}{\partial (\text{net}_{pk}^o)} \frac{\partial (\text{net}_{pj}^h)}{\partial w_{ji}^h}$$

$$\frac{\partial E_p}{\partial w_{ji}^h} = -\sum_k (y_{pk} - o_{pk}) f_k^{o\prime}(\text{net}_{pk}^o) w_{kj}^o f_j^{h\prime}(\text{net}_{pj}^h) x_{pi}$$

การปรับปรุงน้ำหนัก (weight) ในระดับซ่อนตัว (Hidden layer) จะเป็นไปตามสัดส่วนของ negative gradient ดังนั้นจะได้

$$\Delta_p w_{ji}^h = \eta f_j^{h\prime}(\text{net}_{pj}^h) x_{pi} \sum_k \delta_{pk}^o w_{kj}^o$$

$$\delta_{pj}^{h} = f_j^{h}{}'(\text{net}_{pj}^{h}) \sum_k \delta_{pk}^{o} w_{kj}^{o}$$

โดยที่ η เป็น learning rate และ δ_{pj}^{h} เป็นค่าความผิดพลาดของระดับซ่อนตัว (Hidden-layer error) ดังนั้นจะได้สมการในการปรับปรุงน้ำหนัก (weight) ในระดับซ่อนตัว (Hidden layer) เป็น

$$w_{ji}^{h}(t+1) = w_{ji}^{h}(t) + \eta \delta_{pj}^{h} x_i$$

ค่าเอนเอียง (bias) ที่ป้อนให้แก่ระบบจะมีค่าเท่ากับ 1 ถ้าไม่มีค่าเอนเอียงป้อนให้แก่ระบบแล้วจะไม่สามารถให้ผลที่เป็น 1 ได้ถ้าข้อมูลเข้าเป็น 0 และในการกำหนดน้ำหนักการเชื่อมต่อ (connection weight) ครั้งแรกของระบบจะได้จากการสุ่ม (random) โดยจะมีค่าระหว่าง -0.5 ถึง 0.5 ในกรณีที่น้ำหนักการเชื่อมทั้งหมดมีค่าเริ่มต้นเท่ากัน จะทำให้เกิดการสมมาตรของน้ำหนัก (symmetry of weights) ในระบบ ซึ่งอาจจะทำให้ไม่สามารถหาน้ำหนักการเชื่อมต่อของระบบที่สามารถให้ผลที่ถูกต้องทั้งหมดได้ (Schalkoff, 1992) การกำหนดจำนวนโหนด (node) ในระดับข้อมูลเข้า (Input layer), ระดับซ่อนตัว (Hidden layer), ระดับแสดงผล (Ouput layer) หรือจำนวนชั้นของระดับซ่อนตัว (Hidden layer) นั้นขึ้นอยู่กับปัญหาที่ต้องการประยุกต์ใช้งาน ซึ่งจะกล่าวถึงรายละเอียดในบทต่อไป