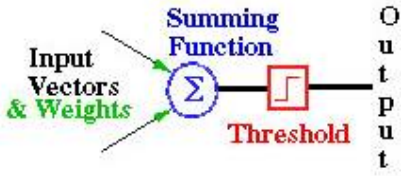
บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 บทนำ

การจัดกลุ่มข้อมูล คือกระบวนการค้นหาเซตของโมเดล (model) ซึ่งโมเดล (ตัวแทนของข้อมูลที่ใช้ในการเรียนรู้) ที่หามาได้จะนำไปใช้ในการทำนายกลุ่มของข้อมูลที่เรายังไม่ทราบว่าข้อมูลนั้นอยู่ในกลุ่มใด ซึ่งการจัดกลุ่มข้อมูลนี้มีกระบวนการเรียนรู้แบบมีผู้สอน (supervised learning) ในการจัดกลุ่มข้อมูลนั้น ข้อมูลทั้งหมดจะถูกแบ่งออกเป็น 2 ส่วนคือส่วนที่ใช้สำหรับการเรียนรู้ (training set) และส่วนที่ใช้สำหรับการทดสอบ (testing set)

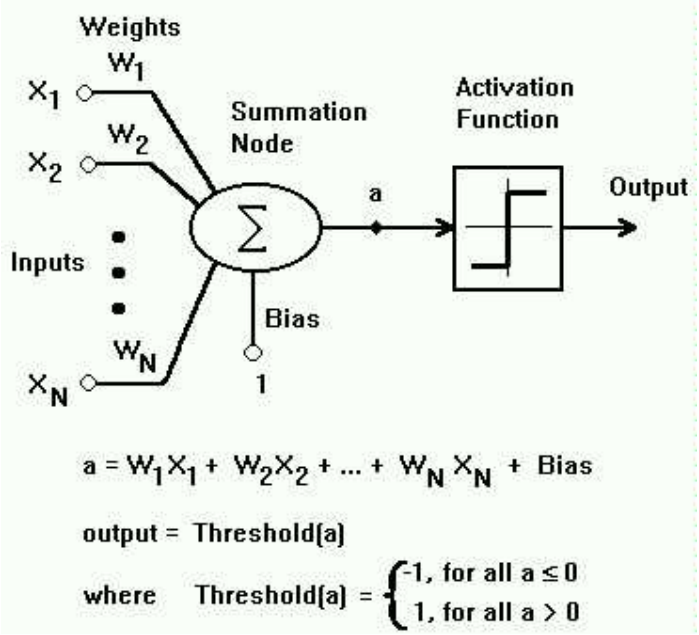
การจัดกลุ่มข้อมูล (classification) สามารถแบ่งออกได้เป็น 3 ประเภท [2] คือ การจัดกลุ่มโดยอาศัยการเรียนรู้เชิงสัญลักษณ์ (symbolic learning) การจัดกลุ่มข้อมูลเชิงสถิติ (statistic) และการจัดกลุ่มข้อมูลโดยใช้โครงข่ายประสาทเทียม (neural network) โดยในที่นี้จะกล่าวถึงเฉพาะ neural network เท่านั้น

โครงข่ายประสาทเทียมเป็นการจำลองกระบวนการการทำงานของเซลล์ประสาทสมองของมนุษย์ โครงข่ายนี้พัฒนาขึ้นมาเพื่อใช้ในการพยากรณ์ข้อมูล และการเรียนรู้ข้อมูลจากข้อมูลที่เตรียมไว้แล้ว โครงข่ายประสาทเทียมมีส่วนประกอบพื้นฐานที่เรียกว่า perceptron โดย perceptron ประกอบด้วยส่วนประกอบย่อยอีก 5 ส่วนคือ input vector, weight, ฟังก์ชันผลรวม (summing function), ค่า threshold และค่าเอาท์พุท (output) ดังแสดงในรูปที่ 2.1 ค่า threshold เป็นค่าที่ผู้ใช้เป็นผู้กำหนดเอง โดยค่านี้มีไว้เพื่อเป็นเกณฑ์ในการกำหนดค่าให้กับเอาท์พุท ซึ่งจะพิจารณาจากผลรวมของ input vector หากค่าผลรวมของ input vector มีค่ามากกว่าค่า threshold ค่าเอาท์พุทที่ได้จะมีค่าเท่ากับ +1 แต่หากค่าผลรวมของ input vector มีค่าน้อยกว่าค่า threshold ค่าเอาท์พุทที่ได้จะมีค่าเท่ากับ -1



รูปที่ 2.1 แสดงรายละเอียดของ perceptron

แนวคิดของ perceptron ได้รับการพัฒนาสู่ single neuron layer โดย input vector X ที่มีขนาด N ถูกส่งเข้าสู่โครงข่ายผ่านการเชื่อมต่อของ weight W ที่มีขนาด N เช่นกัน จากนั้นทำการหาผลรวมของผลคูณระหว่าง input X กับ weight W แล้วค่อยบวกกับค่า Bias เมื่อได้ผลรวมทั้งหมดนำผลรวมทำการเปรียบเทียบกับค่า threshold หากค่าผลรวมที่ได้มีค่ามากกว่าค่า threshold แสดงว่า perceptron นี้จะทำงาน แต่หากค่าผลรวมที่ได้มีค่าน้อยกว่าค่า threshold perceptron นี้จะไม่ทำงาน รูปที่ 2.2 แสดงตัวอย่าง single neuron perceptron



รูปที่ 2.2 แสดงตัวอย่าง single neuron perceptron

ในปัจจุบันนี้ neuron network สามารถประยุกต์ใช้ได้กับงานหลากหลายประเภท เช่น การรู้จำรูปแบบ (pattern recognition) , optical character recognition หรือการจำแนกกลุ่มของปัญหา (problem classification) เป็นต้น

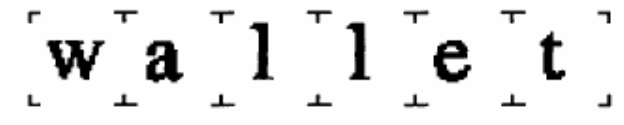
2.2 บทวิจัยที่เกี่ยวข้อง

2.2.1 An Approach to Word Image Matching Based on Weighted Hausdorff Distance [3]

บทความนี้ได้นำเสนอวิธีการเปรียบเทียบรูปภาพตัวอักษรจากเอกสารโดยใช้ Weighted Hausdorff Distance (WHD) โดยคุณภาพของภาพตัวอักษรที่ได้นั้นจะขึ้นอยู่กับการตั้งค่าความละเอียดในการสแกนเอกสาร หรืออาจจะเกิดจากคุณภาพการพิมพ์เอกสารนั้นก็ได้

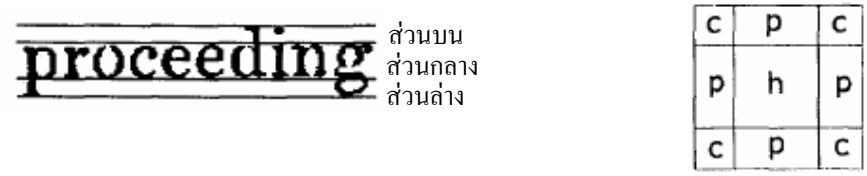
ในขั้นตอนการเตรียมภาพตัวอักษรเพื่อใช้ในการเรียนรู้นั้น จะทำการตัดช่องว่างระหว่างตัวอักษร โดยต้องการให้ตัวอักษรมีระยะห่างที่ชิดกันมากที่สุดเท่าที่จะสามารถทำได้ รูปที่ 2.3 แสดงภาพตัวอักษรต้นฉบับ และรูปที่ 2.4 แสดงภาพตัวอักษรที่ทำการตัดช่องว่างระหว่างตัวอักษรแล้ว

weight Hausdorff distance (WHD) สำหรับตัวอักษรภาษาอังกฤษจะทำการแบ่งภาพตัวอักษรออกเป็น 3 ส่วน คือ ส่วนบน ส่วนกลาง และส่วนล่าง ซึ่งจะมี weight สำหรับแต่ละส่วนแยกกัน คือ weight สำหรับส่วนบน weight สำหรับส่วนกลาง และ weight สำหรับส่วนล่าง จากนั้นจึงเข้าสู่กระบวนการหาคำเหมือน (word matching) สำหรับตัวอักษรภาษาจีน (ภาษาจีนสิงคโปร์) การแบ่งจะแตกต่างไปจากตัวอักษรภาษาอังกฤษ โดยจะทำการแบ่งตรงส่วนมุมของตัวอักษร และตรงกลางของตัวอักษรในการพิจารณา ดังแสดงในรูปที่ 2.5



รูปที่ 2.3 ภาพตัวอักษรต้นฉบับ

รูปที่ 2.4 ภาพตัวอักษรที่ทำการตัดช่องว่างระหว่างตัวอักษรแล้ว



(a) (b)

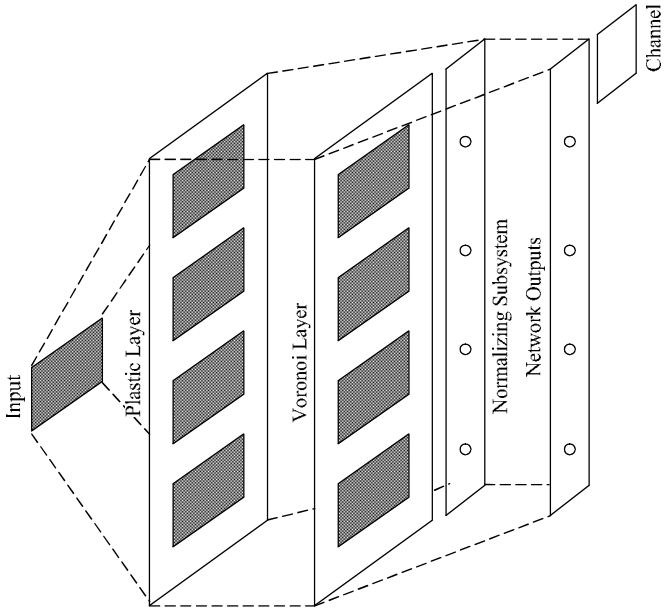
รูปที่ 2.5 (a) การแบ่ง weight สำหรับตัวอักษรภาษาอังกฤษ (b) การแบ่ง weight สำหรับตัวอักษรภาษาจีน

ผลการทดลองของ weight Hausdorff distance ทำการเปรียบเทียบกับ modified Hausdorff distance (MHD) โดยนำเอาระยะทางที่หาได้จากภาพคำต้นแบบ เปรียบเทียบกับภาพคำอื่นๆ มาใช้ในการเปรียบเทียบ ซึ่งผลที่ได้คือ weight Hausdorff distance ให้ค่าความเหมือนที่ดีกว่า และให้ค่าความต่างที่มากกว่า modified Hausdorff distance

2.2.2 Unconstrained Handwritten Numeral Recognition Using Hausdorff Distance and Multi-Layer Neural Network Classifier [4]

บทความนี้มุ่งเน้นที่จะศึกษาเกี่ยวกับลายมือเขียนของมนุษย์ โดยเจาะจงไปที่ตัวเขียนตัวเลขในรหัสไปรษณีย์ ซึ่งอินพุท (input) ที่ใช้ในการเรียนรู้ และทดสอบเป็นรูปภาพขาว-ดำเท่านั้น

HAVNET คือ โครงข่ายประสาทเทียม ที่ถูกออกแบบมาเพื่อการรู้จำภาพ 2 มิติโดยเฉพาะ HAVNET มีรูปแบบโครงข่ายเป็น multi-channel ซึ่งในแต่ละ channel จะมี plastic layer weight ซึ่งใช้เป็นตัวแทนสำหรับแต่ละ pattern ที่ได้ทำการเรียนรู้โดยโครงข่าย สถาปัตยกรรมของ HAVNET แสดงดังรูปที่ 2.6



รูปที่ 2.6 สถาปัตยกรรมของโครงข่าย HAVNET

ในบทความนี้ยังได้นำเสนอ กฎการเรียนรู้ใหม่สำหรับ HAVNET ซึ่งมีอยู่ 3 ส่วนด้วยกัน คือ (1) supervised learning (2) competitive learning (3) self-organization

supervised learning ในส่วนนี้เป็นการกำหนดชนิดของกลุ่ม (class) ให้กับ channel ที่ได้ทำการเรียนรู้ โดยชนิดของกลุ่มของ channel จะเป็นกลุ่มเดียวกับอินพุทที่ได้ทำการเรียนรู้ไปแล้ว

competitive learning ในส่วนนี้จะเป็นการกระตุ้นให้ทุกๆ output node ทำการคำนวณ แล้วจึงเลือกโหนด (node) ที่ให้ค่ามากที่สุดออกมา หากคำตอบของโหนดนั้นถูกต้อง ค่า plastic weight

ที่มีความสัมพันธ์กับโหนดที่ตอบผิดจะมีค่าน้อยลง และค่า plastic weight ที่มีความสัมพันธ์กับโหนดที่ตอบถูกจะมีค่าเพิ่มขึ้น หากเป็นค่าอื่น ค่า plastic weight จะไม่มีการเปลี่ยนแปลง

self-organization สำหรับวิธีการนี้ (Unconstrained Handwritten numeral recognition) ใช้ตัวแสดงคลาสเพียง 10 คลาส (0-9) ไม่เพียงพอ ดังนั้นจึงมีการสร้างตัวแทนขึ้นในแต่ละอัน ซึ่ง self-organization จะมีอยู่เฉพาะในช่วงของการเรียนรู้เท่านั้น กระบวนการนี้มีความคล้ายกับ Adaptive Resonance Theory (ART) จำนวนของคลาสย่อยที่จะถูกสร้างนั้น จะสร้างโดยอัตโนมัติในช่วงระหว่างการเรียนรู้ของ self-organization

บทความนี้เลือกใช้ Hausdorff distance เป็นส่วนหนึ่งในของ HAVNET เนื่องจากให้ค่าความเหมือนที่ใกล้เคียงกับมนุษย์มากกว่าการวัดค่าความเหมือนแบบอื่นๆ (Euclidean Distance, Dot Product, Hamming Distance เป็นวิธีการวัดค่าที่ถูกนำมาเปรียบเทียบกับ Hausdorff distance)

การทดลองจะทำการแบ่งข้อมูลทั้งหมดออกเป็น 2 ส่วนคือ ส่วนที่ใช้สำหรับการเรียนรู้และส่วนที่ใช้สำหรับการทดสอบ ข้อมูลที่ใช้ในการเรียนรู้มีจำนวน 5000 ตัวอย่าง และข้อมูลที่ใช้ในการทดสอบมี 5000 ตัวอย่าง ซึ่งผลการทดลองอยู่ที่ความถูกต้อง 97%

2.2.3 Training Algorithms for Robust Face Recognition using a Template-matching Approach [5]

บทความนี้นำเสนอระบบการรู้จำใบหน้า เป็นการเปรียบเทียบภาพใดๆ กับแม่แบบที่เก็บไว้ และนำอัลกอริทึมสำหรับการเรียนรู้มาประยุกต์ใช้ เพื่อปรับปรุงประสิทธิภาพการทำงานของระบบ

ปัญหาที่ต้องการแก้ไข คือ (1) correct classification experiments คือค่าความถูกต้องในการรู้จำ และบอกได้ถูกต้องว่าภาพนี้อยู่ในฐานข้อมูล และ (2) false positive experiments คือค่าความถูกต้องที่สามารถระบุได้ว่าภาพนี้ไม่ได้อยู่ในฐานข้อมูล

การที่จะบอกว่าภาพใดอยู่ในฐานข้อมูลรูปภาพหรือไม่นั้น จะมีตัวกำหนดค่า (threshold) อยู่ด้วยกันสองตัวคือ L_{m^*} และ U_{m^*} โดยที่ค่า L_{m^*} (lower threshold) คือ ค่าที่กำหนดไว้เพื่อบอกว่าอินพุท (input) กับ กลุ่มตัวอย่างนั้น มีความเหมือนกันเพียงพอ ส่วน U_{m^*} (upper threshold) คือ ค่าที่กำหนดเพื่อบอกว่าอินพุทกับ กลุ่มตัวอย่างนั้น มีความต่างกันเพียงพอ ที่จะทำให้เราสามารถทิ้งอินพุทตัวนี้ออกไปได้

ค่า L_m กับ U_m นั่นสามารถกำหนดได้โดย การแบ่งข้อมูลทั้งหมดออกเป็น 2 กลุ่มคือ classification training set X และ false positive training set Y โดยที่ classification training set ประกอบด้วยตัวอย่างใบหน้าของแต่ละคนในฐานข้อมูล และชุดข้อมูลนี้ใช้เพื่อการปรับปรุงความสามารถในการพิสูจน์บุคคล และความสามารถในการจำแนกบุคคลของระบบ ส่วน false positive training ประกอบด้วยตัวอย่างใบหน้าของแต่ละคน ที่ไม่ได้อยู่ในฐานข้อมูล และชุดข้อมูล

นี้จะถูกใช้เพื่อปรับปรุงความสามารถในการตัดใบหน้าของบุคคล ที่ไม่ได้อยู่ในฐานข้อมูลของระบบอีกด้วย

บทความนี้แบ่งข้อมูลใน classification training set ออกเป็น 2 ชุดข้อมูล (ชุดที่ 1 คือ X_1 และชุดที่ 2 คือ X_2) จากนั้นคำนวณหาระยะทางของข้อมูลแต่ละตัวของ x ของคนที่ m ใน X_1 เทียบกับทุกๆ ข้อมูลใน X_2 จะทำการเปรียบเทียบจนครบทุกๆ ข้อมูลของคนที่ m ใน X_1 จากนั้นจะใช้ค่าเฉลี่ยของระยะทางที่หาได้กำหนดเป็นค่า lower threshold (L_m) สำหรับภาพของคนที่ m นั้นๆ จากนั้นเราจะทำการคำนวณจนครบทุกๆ คนใน X_1

การหาค่า upper threshold (U_m) นั้นเราจะทำการคำนวณหาระยะทางระหว่างข้อมูล x ของคนที่ m เปรียบเทียบกับทุกๆ ภาพใน เซต Y จากนั้นเลือกค่าระยะทางที่น้อยที่สุดเป็นค่า U_m สำหรับคนที่ m นั้นๆ

ภาพที่ใช้ในการทดลองนั้นเป็นภาพมาจากคนทั้งหมด 68 คน ใช้ภาพใบหน้าคนละ 10 ภาพ (รวมเป็น 680 ภาพ) ซึ่งจะมีทั้งภาพผู้ชาย และผู้หญิง และมีอายุที่แตกต่างกัน (อายุอยู่ในช่วง 15 – 50 ปี) ในการเรียนรู้นั้นเราจะแบ่งในการเรียนรู้ของแต่ละคนออกเป็น 2 กลุ่ม กลุ่มที่ 1 สำหรับการเรียนรู้ของระบบ จำนวน 8 ภาพ และกลุ่มที่ 2 สำหรับการทดสอบอีก 2 ภาพ ซึ่งขนาดภาพที่ใช้ คือ ขนาด 72 x 72 พิกเซล

การทดลองใช้รูปภาพทั้งหมดจำนวน 680 ภาพ โดยผลการจำแนกกลุ่มมีค่าความถูกต้องมากกว่า 99% และค่าความผิดพลาดที่ไม่สามารถระบุได้ว่าภาพนี้ไม่ได้อยู่ในฐานข้อมูลนั้นน้อยกว่า 1.5%

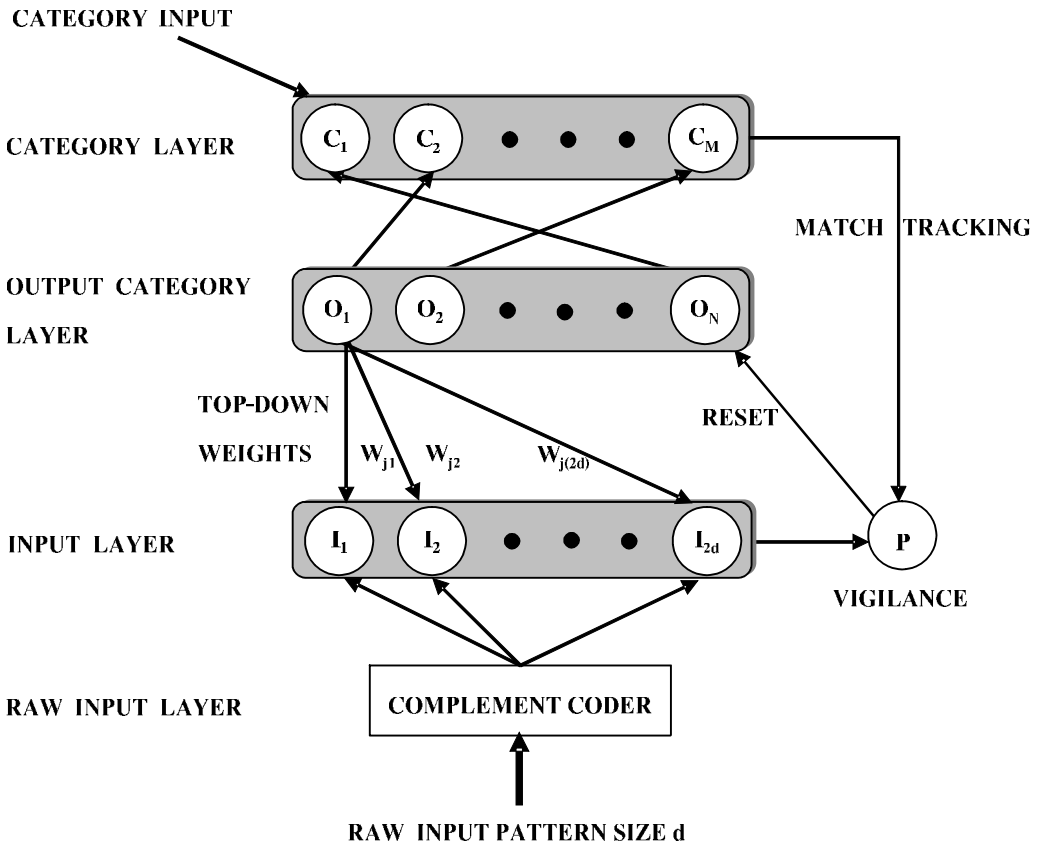
2.3 Simplified Fuzzy ARTMAP (SFAM)

โครงสร้างสถาปัตยกรรมของ Simplified Fuzzy ARTMAP ดังรูปที่ 2.1 อินพุทจะถูกส่งเข้าไปยังโครงข่ายโดยผ่านขั้นตอน complement coding ซึ่งจะทำการเพิ่มขนาดของอินพุทให้เป็นสองเท่า เมื่ออินพุทที่ได้ทำการเพิ่มขนาดแล้ว (I) จะถูกส่งไปยัง input layer โดยโหนดใน input layer จะทำการเชื่อมต่อกับโหนดใน output category layer ผ่านทางค่า weight (w) แบบบนลงล่าง (Top-down weight) ถัดไปคือ category layer เป็นชั้นที่บอกถึงประเภทของกลุ่มข้อมูลซึ่งโครงข่ายต้องเรียนรู้ทั้งหมด M ชนิด โดยที่หนึ่ง output node จะถูกจัดให้อยู่ได้ประเภทเดียวเท่านั้น ดังนั้น output node จะถูกชี้ไปยังตำแหน่งใน category layer ได้เพียงตำแหน่งเดียวเท่านั้น ส่วน category input สามารถเพิ่มเข้าไปยัง category layer ได้เฉพาะระหว่างการเรียนรู้ของโครงข่ายเท่านั้น ซึ่งการเรียนรู้ของโครงข่ายจะเป็นการเรียนรู้แบบมีผู้สอน (supervised learning)

สิ่งสำคัญอีกส่วนหนึ่งของสถาปัตยกรรมแบบนี้ คือ ρ (rho) เรียกว่า vigilance โดยใช้เป็นค่าตั้งต้นพื้นฐาน (base line) ซึ่งมีค่าระหว่าง 0 ถึง 1 เท่านั้น ซึ่งค่า vigilance เป็นพารามิเตอร์ตัวเดียว

เท่านั้นที่ผู้ใช้สามารถกำหนดค่าได้ ค่า vigilance จะเป็นตัวควบคุมการจัดกลุ่มของ output node ว่า input pattern จะต้องมีความคล้ายกันมากน้อยเพียงใดจึงจะสามารถจัดให้อยู่ในคลาสของเอาท์พุตเดียวกันได้ ยิ่งค่า vigilance สูง input pattern ยิ่งต้องมีความเหมือนกันมากจึงจะจัดให้อยู่ในคลาสของเอาท์พุตเดียวกันได้ และหากค่า vigilance มีค่าน้อย input pattern ที่มีความเหมือนกันน้อยก็สามารถจัดให้อยู่ในคลาสเดียวกันได้ ทั้งนี้ค่าของ vigilance ยิ่งสูงก็จะทำให้จำนวนโหนดของ output category ยิ่งมากขึ้นไปด้วย

ขั้นตอนการทำ match tracking นั้น จะเป็นส่วนที่โครงข่ายทำการปรับค่า vigilance ระหว่างการเรียนรู้ โดยจะทำการเพิ่มค่าขึ้นเล็กน้อย ขั้นตอนนี้จะทำเมื่อมีความผิดพลาดในการจำแนกชนิดข้อมูลในระหว่างช่วงของการเรียนรู้ของโครงข่าย



รูปที่ 2.7 สถาปัตยกรรมของซิมพลีไฟด์ฟัซซีอาร์ทแมพ

กระบวนการทำงานของ Simplified Fuzzy ARTMAP (SFAM)

Input ที่เข้ามาจะทำการปรับโดยทำ complement coding ถ้า input pattern vector a มีฟีเจอร์ทั้งหมด d ฟีเจอร์ เวคเตอร์ที่ทำคอมพลีเมนท์ $\bar{a}$ แล้ว สามารถเขียนให้อยู่ในรูปสมการได้ดังนี้

$$\bar{a} = 1 - a \tag{2.1}$$

เนื่องจากโครงข่ายนี้มีการดำเนินการที่ใช้ Fuzzy Logic ดังนั้น input ทั้งหมดใน SFAM จะมีค่าระหว่าง 0 ถึง 1 และ input vector ที่ผ่านการทำคอมพลีเมนท์แล้วจะมีขนาดเป็น 2 เท่า คือ 2d และสามารถแทนค่าของ input ได้ดังนี้

$$I = (a, \bar{a}) = (a_1, ..., a_d, \bar{a}_1, ..., \bar{a}_d) \tag{2.2}$$

โดยที่

$$|I| = |(a, \bar{a})| = \sum_{i=1}^{d} a_i + \left(d - \sum_{i=1}^{d} a_i \right) = d \tag{2.3}$$

เมื่อค่า $|\,.\,|$ สามารถคำนวณได้ดังนี้

$$|p| = \sum_{i=1}^{d} p_i \tag{2.4}$$

โครงข่ายนี้มีการกระตุ้นโหนดใน output category layer ให้มีการตอบสนองต่ออินพุทที่ถูกส่งเข้ามาในระบบ เรียกกระบวนการนี้ว่าฟังก์ชันกระตุ้น (activation function) สามารถคำนวณได้ดังนี้

$$T_j(I) = \frac{|I \wedge W_j|}{\alpha + |W_j|} \tag{2.5}$$

โดย T_j คือ ค่าการกระตุ้นของโหนดใน output category later เมื่อ j คือ ลำดับของ output node I คืออินพุทที่ผ่านการทำ complement-coding และค่า α เป็นค่าที่กำหนดไว้ให้มีค่าน้อยมากๆ ที่เข้าใกล้ 0 โดยปกติจะให้มีค่าประมาณ 0.0000001 ระบบจะทำการเลือก output node ที่มีค่า T_j สูงที่สุดให้เป็นโหนดที่ชนะ

$$\text{Winner} = \max(T_j) \tag{2.6}$$

หากมีโหนดที่มีค่า T_j สูงที่สุดเกินกว่า 1 โหนด ระบบจะทำการเลือก output node ที่ถูกสร้างในระบบก่อน output node อื่นๆ ที่มีค่า T_j เท่ากันให้เป็นโหนดที่ชนะ โดยชนิดของรูปแบบที่สัมพันธ์กับ output node ที่ชนะ คือ วิธีการจำแนกประเภทของ input pattern ที่โครงข่ายนี้ใช้นั่นเอง

Match function คือ การนำเอา input feature ที่ทำ complement-coding แล้ว เปรียบเทียบกับค่าน้ำหนักโดยจะนำมาพิจารณาร่วมกับค่า vigilance เพื่อดูว่าอินพุทมีความเหมือนกับ output node ที่เกี่ยวข้องเพียงพอหรือไม่ หรือควรเพิ่ม output node ใหม่เพื่อเป็นตัวแทนของ input pattern นี้ หากค่า match function มีค่ามากกว่าค่า vigilance แล้ว แสดงว่าเกิดสภาวะที่เรียกว่า resonance ซึ่งสามารถแสดงได้ดังนี้

$$\frac{\left|I \wedge W_j\right|}{d} \geq p \tag{2.7}$$

การเกิด resonance หมายถึง output node j ดีเพียงพอที่จะเป็นตัวแทนของ input I ได้ และ output node j นี้จะต้องเป็น output node กลุ่มเดียวกันกับอินพุท I ด้วย จากนั้นค่าน้ำหนักของ output node j จะทำการปรับค่าเพื่อทำการเรียนรู้ input pattern ตัวใหม่นี้ แต่ถ้าหากกลุ่มของ output node j กับกลุ่มของอินพุท I ต่างกัน ค่าน้ำหนักจะไม่ทำการเปลี่ยนแปลงใดๆ โดยสังเกตเห็นได้ว่า จะมีเพียง output node เพียงโหนดเดียวเท่านั้นที่จะสามารถปรับค่าได้ เมื่ออินพุทหนึ่งตัวเข้ามาทำการเรียนรู้ในระบบ

เมื่อ output node j เป็น node ที่ชนะและผ่านเงื่อนไข (2.7) พร้อมทั้งเกิด resonance แล้ว ค่าน้ำหนักจะได้รับการปรับค่าตามสมการต่อไปนี้

$$w_j^{new} = \beta(I \wedge w_j^{old}) + (1-\beta)w_j^{old} \tag{2.8}$$

ค่า β คือ ค่า learning rate จะมีค่าอยู่ระหว่าง 0 ถึง 1 ซึ่ง β มีค่ามากการเรียนรู้ก็จะรวดเร็วมากขึ้น ในการปรับค่าน้ำหนักสามารถทำให้ง่ายขึ้นได้โดยให้ learning rate มีค่าเท่ากับ 1 (fast learning) ดังนั้นสมการสำหรับปรับค่าน้ำหนักสามารถเขียนได้เป็น

$$w_j^{new} = (I \wedge w_j^{old}) \tag{2.9}$$

ในกรณีเงื่อนไข (2.7) เป็นเท็จ จะทำการเพิ่มค่า vigilance ขึ้นเล็กน้อย และพิจารณาค่า T ของโหนดที่มีค่ามากที่สุดถัดไป หากพิจารณาจนครบแล้วไม่มีค่า T ใดๆ ที่ผ่านเงื่อนไขนี้เลย หรือกลุ่มของ output node j กับกลุ่มของอินพุท I เป็นคนละกลุ่มกัน จะทำการเพิ่ม output node ใหม่ โดย output node ที่เพิ่มมานั้นเป็นกลุ่มเดียวกับอินพุท I และมีค่าน้ำหนักเท่ากับค่าอินพุท I ที่ทำ complement-coding แล้ว และในกรณีที่เป็นอินพุทแรกหรืออินพุทนั้นยังไม่ได้รับการเรียนรู้มาก่อนเลย ก็จะทำการสร้าง output node ใหม่ที่มีกลุ่มเป็นกลุ่มเดียวกับอินพุทนั้น และมีค่าน้ำหนักเท่ากับค่าอินพุทที่ทำ complement-coding แล้ว

2.4 Hausdorff Distance

Hausdorff Distance ได้รับการยอมรับว่าเป็นตัววัดค่าความเหมือน (Similarity Metric) ที่มีประสิทธิภาพ และมีหลักการทำงานใกล้เคียงกับวิธีการของมนุษย์มากกว่าตัววัดค่าความเหมือนอื่นๆ Hausdorff Distance เป็นการวัดระยะทางของแต่ละจุดในเซตของอินพุทว่าอยู่ใกล้กับจุดใดในโมเดล เป็นระยะทางเท่าใด โดยถ้ากำหนดให้ $A = \{a_1,\ldots,a_p\}$ และ $B = \{b_1,\ldots b_q\}$ ซึ่งทั้งสองเซตเป็นเซตของจุดที่มีจำนวนสมาชิกจำกัด (finite point sets) Hausdorff Distance สามารถนิยามได้ดังนี้

$$H(A,B) = \max\{h(A,B), h(B,A)\} \tag{2.10}$$

เมื่อฟังก์ชั่น $h(A,B)$ คือ directed Hausdorff Distance จาก A ไปยัง B สามารถคำนวณได้ดังนี้

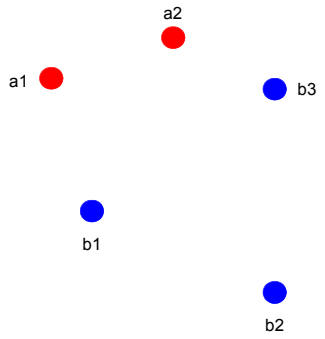
$$h(A,B) = \max_{a\in A}\{\min_{b\in B}\{\|a-b\|\}\} \tag{2.11}$$

เมื่อ $\|a-b\|$ ในที่นี้คือค่า Euclidean distance ระหว่างจุด a และ b

Hausdorff Distance คือ การหาจุดใน A ซึ่งอยู่ห่างจากจุดใน B ที่ใกล้จุดนั้นที่สุดเป็นระยะทางมากที่สุด แล้ววัดระยะทางนั้น

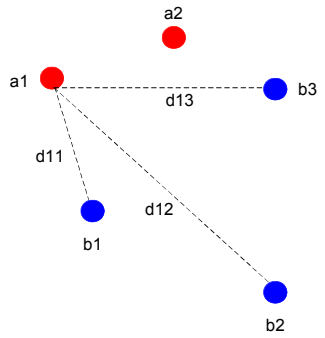
ตัวอย่างการหาค่า Hausdorff Distance $H(A,B)$

1. กำหนดเซต A และเซต B ดังรูปที่ 2.8



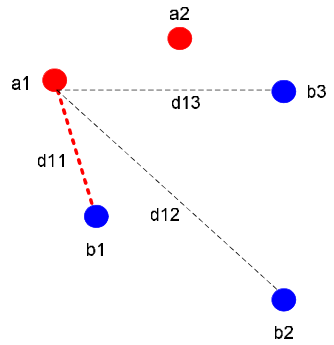
รูปที่ 2.8 แสดงเซต A และเซต B

2. เริ่มคำนวณหา $h(A,B)$ โดยเริ่มจากคำนวณหาระยะทางจาก a1 ไปยังทุกๆ จุดในเซต B



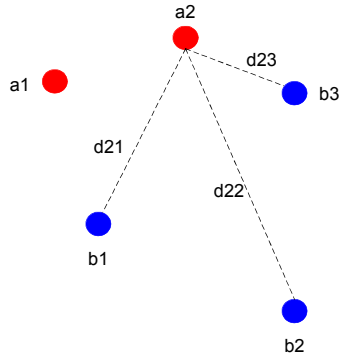
รูปที่ 2.9 แสดงการหาระยะทางระหว่างจุด a1 ไปยังทุกๆ จุดในเซต B

3. ทำการเลือกระยะทางที่สั้นที่สุด คือจุด d11



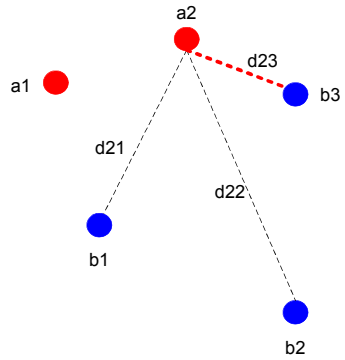
รูปที่ 2.10 แสดงการเลือกระยะ d11 ซึ่งเป็นระยะทางที่สั้นที่สุด

4. คำนวณหาระยะทางจาก a2 ไปยังทุกๆ จุดในเซต B



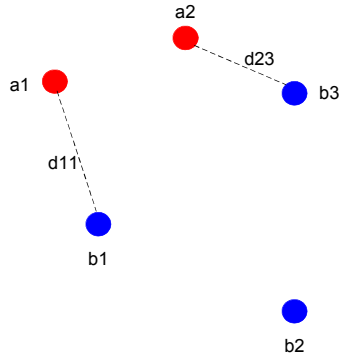
รูปที่ 2.11 แสดงการหาระยะทางระหว่างจุด a2 ไปยังทุกๆ จุดในเซต B

5. ทำการเลือกระยะทางที่สั้นที่สุด คือจุด d23

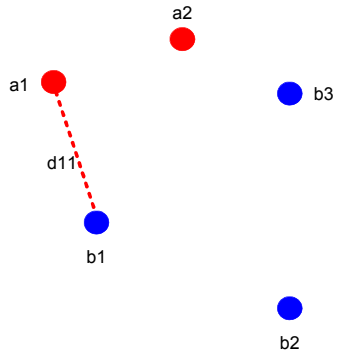


รูปที่ 2.12 แสดงการเลือกระยะ d23 ซึ่งเป็นระยะทางที่สั้นที่สุด

6. เมื่อทำครบทุกจุดในเซต A ให้นำเอาระยะทางที่หาได้นำมาเปรียบเทียบกัน ดังรูปที่ 2.13 จากนั้นเลือกระยะทางที่มากที่สุดเป็นคำตอบ ของ directed Hausdorff Distance จาก A ไปยัง B ดังรูปที่ 2.14

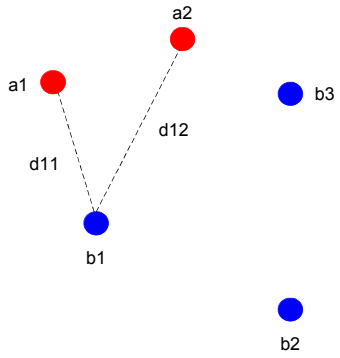


รูปที่ 2.13 แสดงการเปรียบเทียบหาระยะทางที่หาได้



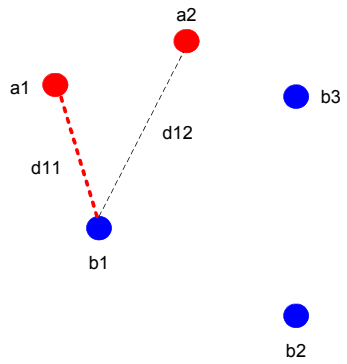
รูปที่ 2.14 แสดงระยะทางที่เลือก

7. เริ่มคำนวณหา $h(B, A)$ โดยเริ่มจากคำนวณหาระยะทางจาก b1 ไปยังทุกๆ จุดในเซต A



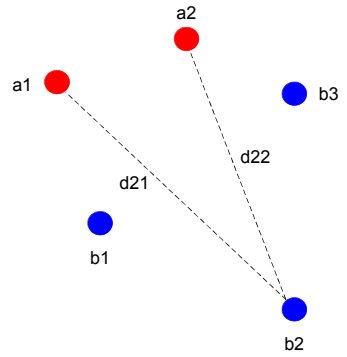
รูปที่ 2.15 แสดงการหาระยะทางระหว่างจุด b1 ไปยังทุกๆ จุดในเซต A

8. ทำการเลือกระยะทางที่สั้นที่สุด คือจุด d11



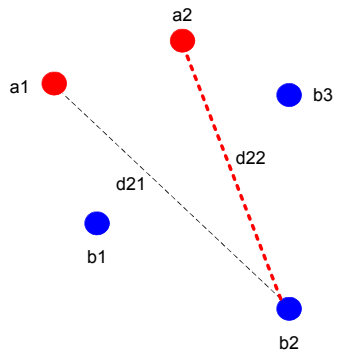
รูปที่ 2.16 แสดงการเลือกระยะ d11 ซึ่งเป็นระยะทางที่สั้นที่สุด

9. คำนวณหาระยะทางจาก b2 ไปยังทุกๆ จุดในเซต A



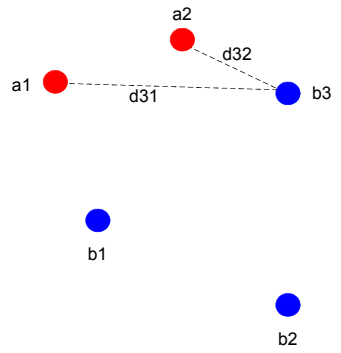
รูปที่ 2.17 แสดงการหาระยะทางระหว่างจุด b2 ไปยังทุกๆ จุดในเซต A

10. ทำการเลือกระยะทางที่สั้นที่สุด คือจุด d22



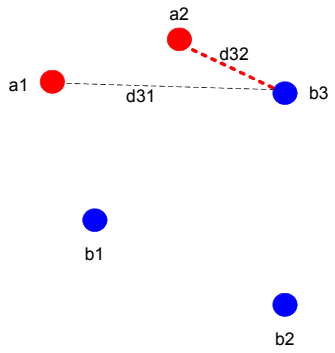
รูปที่ 2.18 แสดงการเลือกระยะ d22 ซึ่งเป็นระยะทางที่สั้นที่สุด

11. คำนวณหาระยะทางจาก b3 ไปยังทุกๆ จุดในเซต A



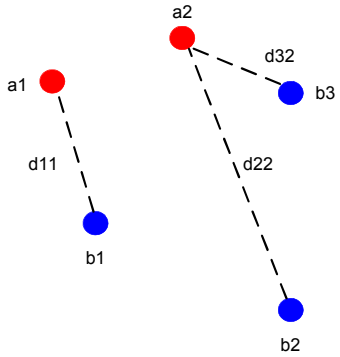
รูปที่ 2.19 แสดงการหาระยะทางระหว่างจุด b3 ไปยังทุกๆ จุดในเซต A

12. ทำการเลือกระยะทางที่สั้นที่สุด คือจุด d32

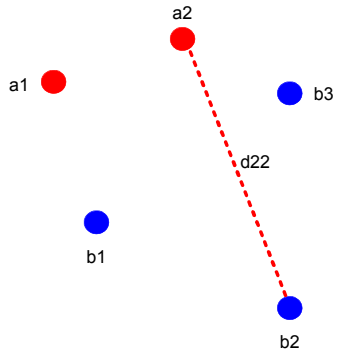


รูปที่ 2.20 แสดงการเลือกระยะ d32 ซึ่งเป็นระยะทางที่สั้นที่สุด

13. เมื่อทำครบทุกจุดในเซต B ให้นำเอาระยะทางที่หาได้นำมาเปรียบเทียบกัน ดังรูปที่ 2.21 จากนั้นเลือกระยะทางที่มากที่สุดเป็นคำตอบ ของ directed Hausdorff Distance จาก B ไปยัง A ดังรูปที่ 2.22

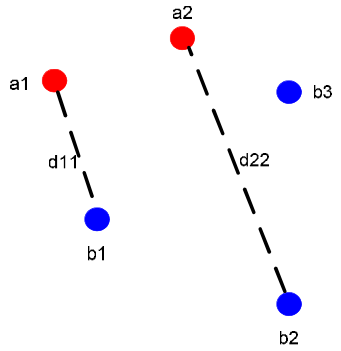


รูปที่ 2.21 แสดงการเปรียบเทียบหาระยะทางที่หาได้



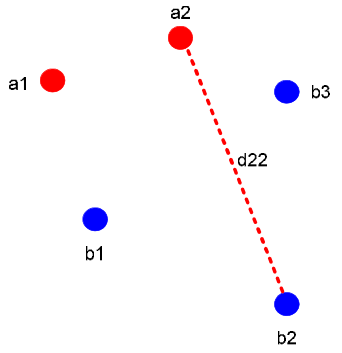
รูปที่ 2.22 แสดงระยะทางที่เลือก

14. หา $H(A,B)$ จากรูปที่ 2.14 และรูปที่ 2.22



รูปที่ 2.23 แสดงระยะทางที่ถูกเลือก

15. เลือกระยะทางที่มากที่สุดจากรูปที่ 2.23 เพื่อเป็นคำตอบของ $H(A,B)$ ดังรูปที่ 2.24



รูปที่ 2.24 แสดงระยะทางที่เป็นคำตอบของ $H(A,B)$

ในการนำ Hausdorff Distance มาใช้กับโครงข่ายประสาทเทียมจำเป็นที่จะต้องมีการปรับวิธีการคำนวณหา directed Hausdorff Distance ในสมการ (2.11) ใหม่ เนื่องจากการคำนวณค่า directed Hausdorff Distance ต้องคำนวณจากทุกๆ จุด โดยคำนวณได้ดังนี้

$$d(a,B) = \min_{b \in B} \{\|a - b\|\} \tag{2.12}$$

เมื่อ $d(a,B)$ คือ Hausdorff Distance แบบจุด (Pointwise Hausdorff Distance) ของจุดใดๆ ใน a และ Hausdorff Distance แบบจุดของ A คำนวณได้จากสมการต่อไปนี้

$$h(A,B) = \frac{\sum_{a \in A} d(a,B)}{|A|} \tag{2.13}$$

โดย $|A|$ คือจำนวนจุดในเซต A