เทคนิคการวิเคราะห์ข้อมูล k-NN
(k-NN Data Analysis)

โดย

นายสุรชัย งามดีวิไลศักดิ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์
สาขาวิศวกรรมคอมพิวเตอร์
สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง
ปีการศึกษา 2535

บทคัดย่อ

k-NN เป็นเทคนิคในการวิเคราะห์ข้อมูลแบบหนึ่ง อยู่บนหลักการของ Nearest Neighborhood เป็นการแทนข้อมูลด้วยเวคเตอร์ของคุณลักษณะ (feature vector) และทำการวิเคราะห์เวคเตอร์นั้น แทนการวิเคราะห์ข้อมูล โดยการวัดค่าความแตกต่าง (distance) หรือค่าความเหมือน (similarity) ของข้อมูล ซึ่งมีโมเดลทางคณิตศาสตร์ในการวัดค่าดังกล่าวอยู่หลายโมเดล

ในปริญญานิพนธ์นี้ มีจุดมุ่งหมายที่จะศึกษาหลักการของ k-NN เพื่อให้สามารถนำไปประยุกต์ใช้ในงานด้านต่างๆ ได้ โดยนำ ไปใช้ในการแยกแยะตัวอักษร (character recognition) ซึ่งก็ปรากฏว่าสามารถใช้เทคนิคดังกล่าว ทำการทดลองแยกแยะตัวเลขไทยได้อย่างถูกต้อง

เทคนิคในการวิเคราะห์ข้อมูลอีกแบบหนึ่งคือ Self-organization Maps(SOM) ซึ่งเป็นสาขาหนึ่งในเรื่อง Neural Networks ถูกนำมาใช้ในปริญญานิพนธ์นี้ด้วย ทั้งนี้เพื่อเปรียบเทียบกับ k-NN

Abstract

The k-NN technique,based on "Nearest Neighborhood" concept, is one of serveral techniques for data analysis. When represent data with their feature vectors,we can use serveral mathmatics models to determine different class of data, the value which determine the class is called "distance value" or somtime called "similarity value".

The thesis's objective is to study k-NN methology then chose "The Printed THAI numeral Characters Recognition" to be the test case. The experiment's result is very impressive.

Another technique called "Self-organization Maps(SOM)" which concerned with "Neural Networks" methology is used to compare the performance of the k-NN and it's result is also very interesting.

สารบาญ

บทที่ 1

บทนำ

ปัจจุบันโลกอยู่ในยุคของข้อมูลข่าวสาร การวิเคราะห์ข้อมูลจึงทวีความสำคัญขึ้นอย่างรวดเร็ว โดยเฉพาะอย่างยิ่ง การวิเคราะห์ข้อมูลที่มีขนาดใหญ่ และมีความซับซ้อนสูง เทคนิคต่างๆ ที่ใช้ในการวิเคราะห์ข้อมูลได้รับการศึกษาและประยุกต์ใช้อย่างกว้างขวาง ทั้งในทางธุรกิจ และในงานวิจัยต่างๆ ในปริญญานิพนธ์นี้ จะได้ทำการศึกษาถึงเทคนิคการวิเคราะห์ข้อมูลแบบหนึ่งคือ k-NN ซึ่งพัฒนามาจากหลักการพื้นฐานคือ Nearest Neighborhood อันเป็นหลักการคลาสสิกที่เป็นที่ยอมรับกันมากในยุคหนึ่ง เนื่องจากมีพื้นฐานอยู่บนหลักคณิตศาสตร์ ทำให้สามารถอธิบายและพิสูจน์ได้ นอกจากนี้ ก็ได้ศึกษาเทคนิคที่เรียกว่า Self-organization Maps (SOM) เพื่อทำการเปรียบเทียบกับเทคนิค k-NN ด้วย

ในบทที่ 2 ได้กล่าวแนะนำถึงการวิเคราะห์รูปแบบ (pattern analysis) โดยจะให้นิยามของคำศัพท์ต่างๆ และอธิบายเพื่อให้เข้าใจถึงขั้นตอนและกระบวนการในการวิเคราะห์รูปแบบอย่างกว้างๆ จากนั้นก็จะให้รายละเอียดของเทคนิคการวิเคราะห์ข้อมูลของ k-NN ในเชิงทฤษฎี ซึ่งเป็นหัวใจของปริญญานิพนธ์นี้ รวมทั้งเทคนิตของ Self-organization Maps (SOM) ซึ่งนำมาศึกษาเพื่อเปรียบเทียบกับ k-NN

บทที่ 3 จะกล่าวถึงรายละเอียดในการนำเทคนิคการวิเคราะห์ข้อมูลทั้งสอง ไปประยุกต์ใช้กับตัวอย่างงานจริง ซึ่งตัวอย่างที่เลือกใช้ในการศึกษาก็คือ การแยกแยะรูปแบบของตัวเลขไทย ทั้งนี้เนื่องจากเป็นแอพพลิเคชันที่เป็นที่ต้องการ และสามารถหาข้อมูลมาใช้ในการทดลองได้สะดวก และมีจำนวนมากพอ

บทที่ 2

การวิเคราะห์รูปแบบ
(Pattern Analysis)

2.1 คุณลักษณะ(Feature) และรูปแบบ(Patttern)

เมื่อเราพิจารณาวัตถุใดๆ เราจะมีข้อมูลเกี่ยวกับวัตถุนั้นมากมาย ตัวอย่างเช่น เมื่อเรามองส้มผลหนึ่ง เราจะมีรายละเอียดเกี่ยวกับส้มผลนั้นหลายๆ อย่าง เช่น สีสรร, ขนาด-รูปร่าง, น้ำหนัก, ลักษณะพื้นผิว ฯลฯ คุณสมบัติหรือรายละเอียดทั้งหมดนี้เองที่ประกอบกันเป็นส้มแต่ละผล ทำให้เกิดความแตกต่างระหว่างส้มผล ก. กับผล ข. คุณสมบัติของข้อมูลดังกล่าว เราเรียกว่า คุณลักษณะของข้อมูล

คุณลักษณะ(feature) ของข้อมูล คือ ค่าใดๆ ที่สามารถวัดได้จากข้อมูล ไม่ว่าจะเป็นคุณลักษณะในเชิงตัวเลข หรือเชิงสัญลักษณ์ เช่น น้ำหนัก, ขนาด (เป็นคุณลักษณะเชิงตัวเลข) หรือ สี, รูปร่าง (เป็นคุณลักษณะเชิงสัญลักษณ์)

โดยทั่วไป เราแบ่งคุณลักษณะออกเป็น 2 ประเภท ได้แก่ คุณลักษณะขั้นต้น(low level feature) หมายถึง คุณลักษณะที่สามารถวัดได้จากข้อมูลนั้นโดยตรง เช่น ความกว้าง, ความสูง, น้ำหนัก เป็นต้น และคุณลักษณะขั้นสูง (high level feature) เป็นคุณลักษณะที่ไม่สามารถวัดได้โดยตรง จะต้องมีการคำนวณจากค่าที่วัดได้อีกชั้นหนึ่ง เช่น ความหนาแน่น ซึ่งต้องคำนวณจากน้ำหนัก และปริมาตรเป็นต้น

การสรรหาค่าคุณลักษณะ(Feature Extraction) คือกระบวนการใดๆ ที่ใช้ในการวัดค่าคุณลักษณะของข้อมูล อาจเป็นการจับภาพวัตถุด้วยกล้องโทรทัศน์ แล้ววัดค่าคุณลักษณะต่างๆ จากอิมเมจของวัตถุให้คนหรือเครื่องจักรทำการวัดโดยตรง หรือวิธีการอื่นใดก็ตาม

```
0000111111111110000
0001111111111111000
0011110000000111100
0111100000000011110
1111000000000001111
1111000000000001111
1111000000000001111
1111000000000001111
1111000000000001111
1111000000000001111
0111100000000011110
0011110000000111100
0001111111111111000
0000111111111110000
```

รูป 2.1 : ตัวอย่างภาพของตัวเลขศูนย์

ตัวอย่างในรูป 2.1, เป็นภาพที่ได้จากการแปลงบิทแมพของตัวเลขศูนย์ซึ่งได้จากสแกนเนอร์ จากภาพดังกล่าว เราสามารถวัดค่าคุณลักษณะบางประการของตัวเลขศูนย์ได้ เช่น ความกว้าง, ความสูง, จำนวนจุด เป็นต้น ซึ่งอาจจะใช้คนนับโดยตรง หรือเขียนโปรแกรมขึ้นทำการนับก็ได้

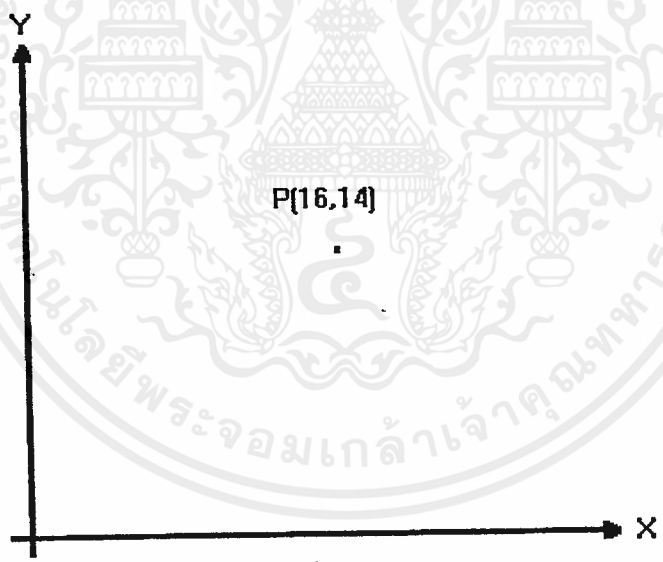
โดยปรกติ เราใช้คุณลักษณะทั้งหมดประกอบกันเป็นตัวแทนของข้อมูล เพื่อใช้ในการประมวลผลต่างๆ อย่างไรก็ตาม เพื่อให้การประมวลผลทำได้ง่าย เรามักเลือกใช้คุณลักษณะให้น้อยที่สุด เท่าที่จะใช้เป็นตัวแทนของข้อมูลได้

ตัวอย่างเช่น ในระบบที่ทำการวิเคราะห์สินค้าที่ผ่านสายการผลิต ว่าสินค้านั้นเป็นส้มหรือทุเรียน ซึ่งเราสามารถใช้เพียงน้ำหนักของสินค้า ก็สามารถจะประมวลผลได้อย่างถูกต้อง แต่ถ้าต้องการแยกพันธ์ของทุเรียน (ก้านยาว, ชะนี ฯลฯ) เราก็จะต้องใช้คุณลักษณะอื่นๆ เข้าช่วยด้วย

เวคเตอร์ของคุณลักษณะ(feature vector) หมายถึง เวคเตอร์ที่มีองค์ประกอบในแต่ละมิติ เป็นค่าของคุณลักษณะต่างๆ ที่ใช้เป็นตัวแทนของข้อมูล มิติของเวคเตอร์จะขึ้นอยู่กับจำนวนคุณลักษณะที่ใช้ในการแทนข้อมูลนั้น

สเปซของคุณลักษณะ(feature space) หมายถึง ช่วงของค่าที่เป็นไปได้ทั้งหมดของเวคเตอร์ของคุณลักษณะ ซึ่งในกรณีที่เวคเตอร์ของคุณลักษณะมีขนาดเป็น 2 มิติ เราจะสามารถแสดงสเปซของคุณลักษณะบนพิกัดคาร์ทีเซียนได้ ซึ่งก็คือบริเวณของเวคเตอร์ของคุณลักษณะที่เป็นไปได้ทั้งหมดนั่นเอง

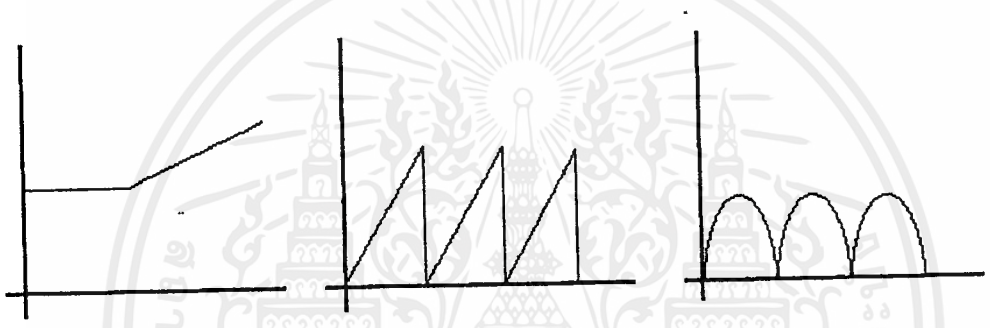
ตัวอย่างเช่น จากรูป 2.1 ถ้าใช้ความสูงและความกว้างของภาพเป็นตัวแทนของข้อมูลตัวเลขศูนย์ จะได้เวคเตอร์ของคุณลักษณะเป็นเวคเตอร์ขนาด 2 มิติ โดยมีค่าขององค์ประกอบแรกเป็น 14 หน่วย และองค์ประกอบที่สองเป็น 19 เขียนในรูปเวคเตอร์ได้เป็น (14,19) ในทำนองเดียวกัน ถ้าต้องการใช้ความสูง, ความกว้าง และจำนวนจุดในภาพเป็นตัวแทนของข้อมูลเลขศูนย์ เวคเตอร์คุณลักษณะของเลขศูนย์ในภาพจะมีค่าเป็น (14,19,128) เป็นต้น



รูป 2.2 : เวคเตอร์คุณลักษณะของข้อมูลเลขศูนย์ในรูป 2.1 พล็อตลงบนสเปซของคุณลักษณะ(feature space) (พิกัดคาร์ทีเซียน)

เมื่อพิจารณาข้อมูลจำนวนมาก เรามักพบว่าข้อมูลบางตัวหรือบางกลุ่ม ที่มีลักษณะบางอย่างใกล้เคียงกัน มีโครงสร้างเหมือนๆ กัน การที่ข้อมูลจำนวนหนึ่ง มีลักษณะใกล้เคียงกันดังกล่าว เรียกว่าข้อมูลกลุ่มนั้นมีรูปแบบ(pattern) ของข้อมูลเหมือนกัน

รูปแบบอาจจะเป็นอะไรก็ได้ แล้วแต่จะพิจารณา เช่น ลักษณะการแตกกิ่งก้านของต้นไม้, การโค้งของเส้น, ตัวอักษรต่างๆ เป็นต้น การที่ข้อมูลจำนวนหนึ่ง มีรูปแบบเดียวกัน ทำให้เกิดความรู้สึกว่า ข้อมูลกลุ่มนั้น น่าจะเป็นข้อมูลกลุ่มเดียวกัน ดังนั้นการจัดกลุ่มจึงขึ้นกับรูปแบบที่กำลังพิจารณาอยู่ด้วย



ก) ข) ค)

รูป 2.3 : ลักษณะเส้นกราฟแบบต่างๆ

ตัวอย่างเช่น การเราพิจารณาลักษณะการเปลี่ยนแปลงของเส้นกราฟ จะเห็นว่ารูป ก) และ ข) มีการเปลี่ยนแปลงเป็นเส้นตรง ส่วนแบบ ค) นั้นเป็นเส้นโค้ง ดังนั้นเราจึงจัดกราฟรูป ก) และรูป ข) เข้าเป็นกลุ่มหนึ่ง และรูป ค) เป็นอีกกลุ่มหนึ่ง

แต่ถ้าเราพิจารณาลักษณะกราฟสัมพันธ์กับเวลา เราจะพบว่า กราฟรูป ข) และ ค) ต่างก็มีค่าเป็นช่วง แต่กราฟ ก) ไม่เป็น เราก็จะจัดกราฟรูป ก) และรูป ข) เข้าเป็นกลุ่มหนึ่ง และรูป ค) เป็นอีกกลุ่มหนึ่ง

มนุษย์เรามีความสามารถในการแยกแยะ, วิเคราะห์ และจดจำรูปแบบต่างๆ ได้อย่างง่ายดาย เนื่องจากมีความคุ้นเคยกับรูปแบบต่างๆ เราสามารถจำแนกตัวอักษรได้ทันทีที่เห็น แม้ว่าตัวอักษรที่เห็น จะมีความแตกต่างหรือผิดเพี้ยนไปบ้าง ในทางคอมพิวเตอร์นั้น การวิเคราะห์รูปแบบ จะวิเคราะห์จากคุณลักษณะที่ใช้แทนข้อมูลต่างๆ การพิจารณาเลือกคุณลักษณะที่ใช้แทนข้อมูลจึงเป็นสิ่งที่มีความสำคัญมาก

2.2 การพิจารณาเลือกคุณลักษณะ

เนื่องจากเราใช้เวคเตอร์ของคุณลักษณะในการแทนข้อมูลต่างๆ เมื่อต้องการแยกแยะรูปแบบของข้อมูล เราก็จำเป็นต้องวิเคราะห์รูปแบบของข้อมูลจากเวคเตอร์ของคุณลักษณะด้วย ดังนั้น การพิจารณาเลือกคุณลักษณะที่จะใช้ จึงเป็นเรื่องที่มีความสำคัญมาก ทั้งนี้มีปัจจัยที่จะต้องคำนึงอยู่ 3 ประการ

ประการแรกคือ สามารถที่จะวัด(หรือคำนวณ) ค่าคุณลักษณะนั้นได้ และมีความเป็นไปได้ในทางปฏิบัติด้วย หากใช้เวลาหรือความสามารถในการคำนวณในการวัดคุณลักษณะนั้นมาก ก็ไม่สามารถจะนำคุณลักษณะดังกล่าวมาใช้ได้ สำหรับปัจจัยข้อนี้จะเห็นว่า การใช้คุณลักษณะขั้นต้น จะดีกว่าคุณลักษณะขั้นสูง เนื่องจากมีการคำนวณน้อยกว่า

ประการที่สองคือ จะต้องสามารถใช้คุณลักษณะนั้น ในการวิเคราะห์รูปแบบได้อย่างถูกต้อง หรือมีข้อผิดพลาดน้อยที่สุด ข้อมูลที่มีรูปแบบ(ที่เราสนใจ) คล้ายคลึงกัน ก็ควรจะมีเวคเตอร์ของคุณลักษณะที่ใกล้เคียงกันด้วย

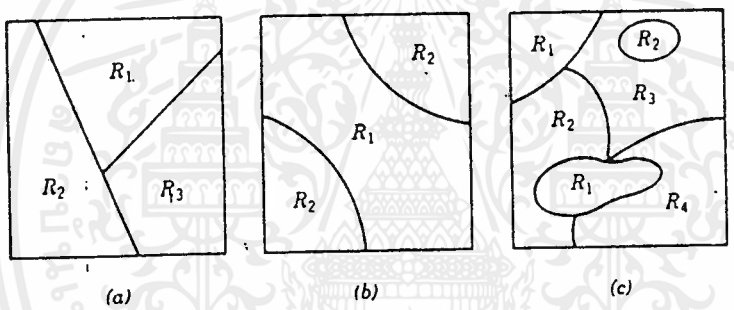
ประการสุดท้าย คือ ขนาดของเวคเตอร์ของคุณลักษณะจะต้องไม่ใหญ่จนเกินไป เนื่องจาก การที่เวคเตอร์ของคุณลักษณะมีมิติสูงๆ จะทำให้การวิเคราะห์ข้อมูลทำได้ยาก เนื่องจากมีข้อมูลต้องวิเคราะห์มาก

วิธีการในการเลือกคุณลักษณะที่นิยมใช้ มี 2 วิธี คือ การใช้เทคนิคทางคณิตศาสตร์เข้ามาประยุกต์ ซึ่งจะได้กล่าวถึงต่อไป และการทำการจำลอง(simulate) เพื่อพิจารณาถึงประสิทธิภาพของคุณลักษณะที่เลือก และทำการเปรียบเทียบเพื่อเปลี่ยนแปลงต่อไป

2.3 การแยกแยะรูปแบบโดยคุณลักษณะ

วิธีการทั่วๆ ไปในการแยกแยะรูปแบบ โดยพิจารณาจากคุณลักษณะ ก็คือการหาเกณฑ์ที่ใช้ในการแยกแยะรูปแบบ(classifier) ซึ่งสามารถจะแบ่งสเปซของคุณลักษณะ (feature space) ออกเป็นส่วนๆ ซึ่งแต่ละส่วนก็จะแสดงถึงขอบเขตของรูปแบบต่างๆ โดยมีเงื่อนไปอยู่ 2 ประการ

ประการแรก ส่วนต่างๆ ทั้งหมด เมื่อรวมกันเข้าแล้ว จะต้องครอบคลุมทั่วทั้งเสปซของคุณลักษณะ และประการที่ 2 ก็คือ ส่วนต่างๆ เหล่านั้นจะต้องไม่มีการซ้อนทับ (overlap) กัน



รูป 2.4 : ตัวอย่างของการแบ่งสเปซของคุณลักษณะเป็นส่วนๆ แบบต่างๆ

ด้วยวิธีการดังกล่าว การที่จะแยกแยะรูปแบบของเวคเตอร์ของคุณลักษณะใดๆ ก็สามารถทำได้โดยง่าย กล่าวคือเมื่อพบว่า x อยู่ในส่วนใด (หลังจากแบ่งสเปซของคุณลักษณะแล้ว) ก็จะจำแนกว่า x มีรูปแบบเดียวกันกับข้อมูลอื่นๆในส่วนนั้น วิธีดังกล่าวแม้ว่าจะเป็นวิธีที่ดูตรงไปตรงมา แต่ก็มีความยุ่งยากซ่อนอยู่เป็นอย่างมาก เนื่องจากการพิจารณาแบ่งส่วนนั้นทำได้ยาก ทั้งนี้ในการทำงานจริงนั้น มักจะไม่สามารถแบ่งสเปซ

ของคุณลักษณะเป็นส่วนๆ ได้อย่างชัดเจนนัก ทั้งนี้ทั้งนั้น ก็ขึ้นอยู่กับคุณลักษณะที่เลือกใช้

ในรูปที่ 2.4 แสดงตัวอย่างการแบ่งสเปซของคุณลักษณะ ให้มีรูปร่างแบบต่างๆ อันประกอบด้วยแบบเส้นตรง(linear), แบบควอดราติกหรือไฮเปอร์โบริก(quadratic /hyperbolic) และแบบทั่วไป(arbitary) ซึ่งเป็นรูปแบบที่มักเกิดขึ้นในงานจริง รูปร่างของของส่วนต่างๆ นั้น จะมีผลต่อรูปแบบของฟังก์ชันที่ใช้เป็นเกณฑ์ในการแยกแยะรูปแบบ

การอธิบายการแยกแยะรูปแบบดังกล่าว เป็นการอธิบายแบบทั่วๆ ไป เท่านั้น ในรายละเอียดที่สำคัญ คือ การพัฒนาฟังก์ชันที่ใช้เป็นเกณฑ์ในการแยกแยะรูปแบบ มีเทคนิคต่างๆ มากมาย ซึ่งไม่อาจจะกล่าวในรายละเอียดได้ทั้งหมด

2.4 ขั้นตอนการแยกแยะรูปแบบ

การออกแบบระบบในการแยกแยะรูปแบบ โดยทั่วไปแล้ว ผู้ออกแบบจะต้องมีข้อมูลข่าวสารต่างๆ ที่เกี่ยวข้องกับปัญหานั้นๆ เป็นส่วนประกอบในการตัดสินใจ เช่น ตัวอย่าง (เวคเตอร์ของคุณลักษณะของ)ข้อมูล ที่ทราบว่าจัดเป็นรูปแบบใด ซึ่งเรียกว่าเทรนนิงดาตา ซึ่งอาจจะเป็นความน่าจะเป็นที่ข้อมูลใดๆ จะจัดอยู่ในคลาส W_i ความเสียหายจากการแยกแยะคลาสผิด เป็นต้น การออกแบบระบบสำหรับแยกแยะรูปแบบ โดยที่ไม่มีข้อมูลเหล่านี้นั้นจะทำให้ได้ระบบที่มีประสิทธิภาพต่ำ แต่ก็ไม่ได้หมายความว่าจะกระทำไม่ได้ เมื่อพบว่าไม่สามารถหาข้อมูลข่าวสารดังกล่าวได้ ก็จำเป็นที่จะต้องออกแบบระบบขึ้นโดยปราศจากข้อมูลเหล่านั้น

ฐานข้อมูลซึ่งประกอบด้วยกลุ่มของข้อมูลที่ทราบรูปแบบของข้อมูลแล้ว เราเรียกว่าข้อมูลในการพัฒนาระบบ เป็นข้อมูลที่สำคัญในการออกแบบระบบ จะเห็นว่าจากข้อมูลในการพัฒนาระบบ เรามีทั้งข้อมูลที่เป็นอินพุทของระบบ และเอาท์พุทที่ระบบต้องการ จึงไม่ใช่เรื่องยากที่จะพิจารณาหาความสัมพันธ์ระหว่างอินพุทและเอาท์พุทได้ เพื่อพัฒนาต่อไปเป็นหลักการ หรือวิธีการที่เหมาะสมของระบบต่อไป เราเรียกขั้นตอนนี้ว่าการเทรนระบบ นอกจากนี้ เรายังสามารถใช้ข้อมูลในพัฒนาระบบ ทำให้ระบบมีความสามารถใน

การเรียนรู้ได้ด้วย

หลักการของการเรียนรู้ที่ใช้กันโดยทั่วไปก็คือ การปรับแก้จากข้อผิดพลาด (error-correction-based) ซึ่งหมายถึงการที่ระบบปรับตัวโดยดูจากข้อผิดพลาดที่ผ่านมา ด้วยวิธีการดังกล่าว ระบบจะพัฒนาไปตามประสบการณ์ หรือ จำนวนครั้งที่ระบบทำงาน เป็นผลให้ระบบมีประสิทธิภาพสูงขึ้น

การเรียนรู้ตามวิธีที่กล่าวมาแล้ว เป็นแบบ supervised learning กล่าวคือ ข้อมูลที่ใช้ในการเรียนรู้นั้น มีทั้งอินพุท และเอาท์พุทที่ต้องการ ในทางตรงข้าม หากข้อมูลที่มี เราไม่ทราบเอาท์พุทที่ต้องการ จะต้องทำการเรียนรู้ในแบบ unsupervised learning ซึ่งทำได้ยากกว่า

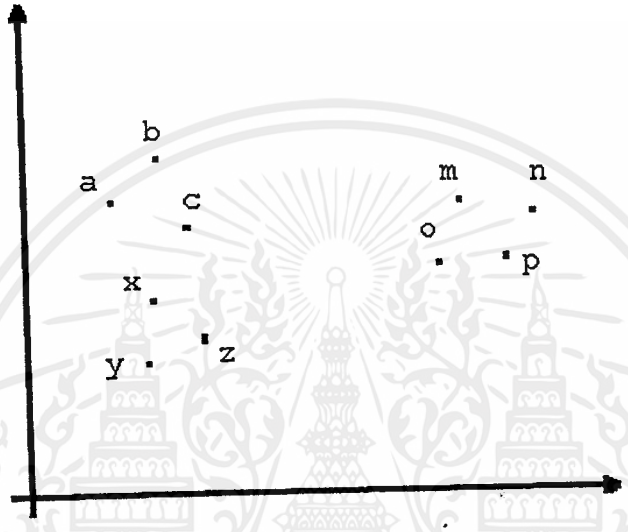
2.5 ทฤษฎีการวิเคราะห์กลุ่มข้อมูลแบบ k-NN

ในปี 2494 (ค.ศ.1951), Fix และ Hodes ได้เสนองานวิจัยออกมาชิ้นหนึ่ง เป็นการเสนอวิธีการแบบใหม่ ที่ใช้ในการแยกแยะรูปแบบ ซึ่งจัดเป็นนอนพาราเมตริก โดยวิธีการดังกล่าวจะให้ผลดีเมื่อตัวอย่างข้อมูลมีมากพอ ผลงานวิจัยชิ้นดังกล่าวได้รับความสนใจเป็นอย่างมาก และมีการวิจัยเพิ่มเติมอย่างต่อเนื่อง และพัฒนามาเป็นเทคนิคในการวิเคราะห์ข้อมูลแบบหนึ่ง เรียกว่า k-Nearest Neighborhood(k-NN)

หลักการของ k-NN ถูกนำไปประยุกต์ใช้ในหลายๆ ด้าน โดยเฉพาะอย่างยิ่งในการพิจารณาเลือกคุณลักษณะของข้อมูล, การวิเคราะห์กลุ่มข้อมูล และการแยกแยะรูปแบบ เช่น การพิจารณาจัดข้อมูลเป็นกลุ่มตามคุณลักษณะต่างๆ และการจำแนกกลุ่มให้กับข้อมูลตัวอย่าง ซึ่งใช้เป็นตัวอย่างทดลองในการศึกษาในปริญญานิพนธ์นี้

2.5.1 การวิเคราะห์กลุ่มข้อมูลด้วยเทคนิค k-NN

หลักการพื้นฐานของการวิเคราะห์กลุ่มโดยวิธีการของ k-NN นั้น ค่อนข้างจะตรงกับสามัญสำนึก คือถือว่าข้อมูลที่มีคุณลักษณะใกล้เคียงกัน จะจัดอยู่ในกลุ่มเดียวกัน ตัวอย่างในรูปที่ 2.5 ข้อมูล 10 ตัวถูกแทนด้วยคุณลักษณะ 2 ประการคือ ค่า x และค่า y และนำค่าเวคเตอร์ของคุณลักษณะที่ได้ พล็อตลงบนพิกัดคาร์ทีเซียนดังรูป



รูป 2.5 : ตัวอย่างเวคเตอร์ของคุณลักษณะของข้อมูล 10 ตัว

จากรูป หากต้องการแบ่งข้อมูลดังกล่าวออกเป็น 3 กลุ่ม ก็จะแบ่งได้เป็นกลุ่มของ ABC, กลุ่มของ xyz และกลุ่มของ mnop ในทำนองเดียวกัน หากจะแบ่งข้อมูลเป็น 2 กลุ่ม ก็จะเป็นกลุ่มของ ABCxyz กลุ่มหนึ่ง และ mnop อีกกลุ่มหนึ่ง ซึ่งการพิจารณาจัดกลุ่มทั้ง 2 กรณีนั้น พิจารณาจากค่าคุณลักษณะของข้อมูลทั้ง 10 นั่นเอง

ในทางคณิตศาสตร์ เราบอกความใกล้เคียงกันของคุณลักษณะของข้อมูล โดยการวัด ค่าความแตกต่าง (distance) ระหว่างข้อมูล ซึ่งพิจารณาได้ด้วยโมเดลทางคณิตศาสตร์ โดยจะคำนวณจากความแตกต่างของทุกๆ คุณลักษณะของข้อมูลทั้ง 2 ทั้งนี้ ก็มีบางโมเดล ที่คำนวณเป็น ค่าความเหมือน (similarity) ระหว่างข้อมูลไปแทน

โมเดลในการวัดค่าความเหมือนหรือค่าความแตกต่างดังกล่าว ได้มีการเสนอไว้หลายโมเดล ซึ่งมีวิธีการคำนวณที่แตกต่างกัน ตาราง 2.1 ได้รวบรวมโมเดลดังกล่าวไว้จำนวนหนึ่ง (ยังไม่ใช่ทั้งหมด)

Squared Euclidean distances. This is the default. This measure should be used with the centroid, median, and Ward's methods of clustering. The distance between two cases is the sum of the squared differences in values for each variable:

$$\text{Distance}(X, Y) = \sum_i (X_i - Y_i)^2$$

Euclidean distances. The distance between two cases is the square root of the sum of the squared differences in values for each variable:

$$\text{Distance}(X, Y) = \sqrt{\sum_i (X_i - Y_i)^2}$$

Cosine of vectors of variables. This is a pattern similarity measure:

$$\text{Similarity}(X, Y) = \frac{\sum_i (X_i Y_i)}{\sqrt{\sum_i (X_i^2) \sum_i (Y_i^2)}}$$

City-block or Manhattan distances. The distance between two cases is the sum of the absolute differences in values for each variable:

$$\text{Distance}(X, Y) = \sum_i |X_i - Y_i|$$

Chebychev distance metric. The distance between two cases is the maximum absolute difference in values for any variable:

$$\text{Distance}(X, Y) = MAX_i |X_i - Y_i|$$

Distances in an absolute power metric. The distance between two cases is the rth root of the sum of the absolute differences to the pth power in values on each variable.

$$\text{Distance}(X, Y) = \left[\sum_i (X_i - Y_i)^p\right]^{1/r}$$

Appropriate selection of integer parameters p and r yields Euclidean, squared Euclidean, Minkowski, city-block, minimum, maximum, and many other distance metrics.

ตาราง 2.1 : ตารางแสดงโมเดลในการวัดค่าความแตกต่าง

เมื่อพิจารณาค่าความแตกต่างระหว่างข้อมูล เราจะสามารถพิจารณาจัดข้อมูล 2 ตัว ใดๆ (ที่มีค่าความแตกต่างระหว่างกันน้อยที่สุด) เข้าด้วยกันได้ ซึ่งทำให้เกิดกลุ่มของข้อมูล (2 ตัว) ขึ้น ในการจัดกลุ่มของข้อมูลรวมเข้าเป็นกลุ่มเดียวกัน จะต้องมีการวัดค่าความแตกต่าง หรือความเหมือนระหว่างกลุ่มข้อมูล และระหว่างกลุ่มข้อมูลกับข้อมูล ซึ่งมีวิธีการต่างๆ อยู่หลายวิธี ดังที่ได้แสดงไว้ในตารางที่ 2.2 จำนวนหนึ่ง จะเห็นว่าส่วนใหญ่ก็จะมีพื้นฐานมาจากการคำนวณค่าความแตกต่างระหว่างข้อมูลนั่นเอง

Average linkage between groups (UPGMA).
Average linkage within groups.
Single linkage or nearest neighbor.
Complete linkage or furthest neighbor.
Centroid clustering (UPGMC). Squared Euclidean distances should be used with this method.
Median clustering (WPGMC). Squared Euclidean distances should be used with this method.
Ward's method. Squared Euclidean distances should be used with this method.

ตาราง 2.2 : ตารางแสดงโมเดลในการวัดค่าความแตกต่างระหว่างกลุ่มข้อมูล

จากหลักการของ k-NN ดังกล่าว เราสามารถนำไปประยุกต์ใช้ในการวิเคราะห์กลุ่มของข้อมูลได้ เมื่อกำหนดคุณลักษณะของข้อมูลที่จะใช้ในการแยกแยะกลุ่มมาให้ และเมื่อมองในมุมกลับ เราก็สามารถประยุกต์หลักการเดียวกัน ใช้ในการพิจารณาเลือกคุณลักษณะที่ควรจะใช้เป็นข้อมูล เพื่อให้สามารถวิเคราะห์กลุ่มให้ได้ผลตามที่ต้องการได้ ในบทที่ 3 จะได้กล่าวถึงรายละเอียดของการทดลองนำ k-NN ไปประยุกต์ใช้ในการพิจารณาเลือกคุณลักษณะต่อไป

2.5.2 การแยกแยะรูปแบบด้วยเทคนิค k-NN

ในกรณีที่ข้อมูลที่ใช้ในการพัฒนาระบบ ถูกแบ่งออกเป็นกลุ่มๆ ตามรูปแบบต่างๆ แล้ว เราสามารถประยุกต์ใช้หลักการของ k-NN ในการพิจารณาข้อมูลอินพุทใหม่เพื่อแยกแยะรูปแบบ โดยอาศัยข้อมูลที่ใช้ในการพัฒนาระบบได้

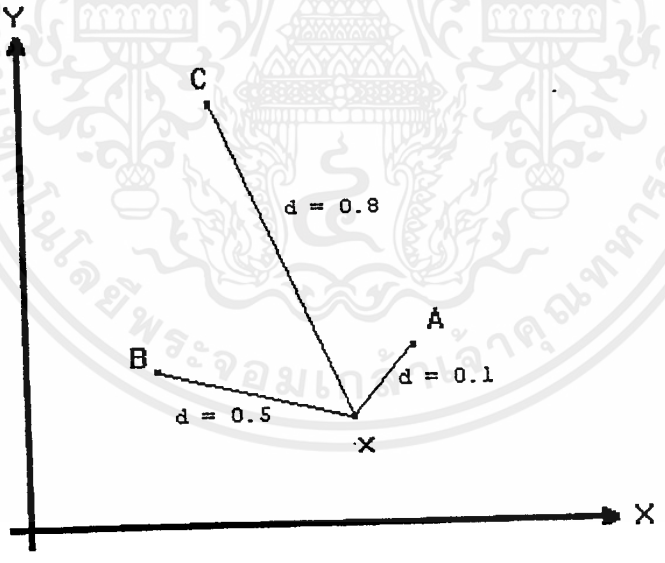
หลักการอย่างง่าย ก็คือ เมื่อต้องการทราบว่า ข้อมูลอินพุท X (น่าจะ) มีรูปแบบเดียวกับข้อมูลกลุ่มใด ก็จะทำการวัดค่าความแตกต่างระหว่างอินพุท X นั้น กับทุกๆ ตัวอย่างในข้อมูลที่ใช้ในการพัฒนาระบบที่มีอยู่ เพื่อพิจารณาว่า ตัวอย่างใดในบรรดาข้อมูลที่ใช้ในการพัฒนาระบบของเรา ที่มีความใกล้เคียงกับอินพุท X มากที่สุด หรือที่เรียกว่าเป็น ตัวอย่างใกล้เคียง(nearest neighbor) ของอินพุท X ในที่นี้สมมุติว่า ตัวอย่าง Y เป็นตัวอย่างใกล้เคียงของ X และเราทราบว่าตัวอย่าง Y อยู่ในกลุ่ม G_i เราก็จะสรุปว่า อินพุท X ก็อยู่ในกลุ่ม G_i เช่นกัน

หลักการอย่างง่ายดังกล่าว เราเรียกว่า หลักของ 1-NN (1-Nearest Neighborhood) กล่าวคือ ใช้ตัวอย่างใกล้เคียงเพียงตัวอย่างเดียว ในการพิจารณาแยกแยะรูปแบบของข้อมูล ซึ่งจากหลักการพื้นฐานนี้ ก็มีการทำวิจัยต่อออกไปในหลายๆ ด้าน ประเด็นหนึ่งก็คือ มีการเสนอว่า การใช้ตัวอย่างใกล้เคียงของอินพุท X เพียงตัวอย่างเดียวในการตัดสินใจ จะมีความถูกต้องเพียงพอหรือไม่ น่าที่จะมีการพิจารณาตัวอย่างใกล้เคียงของอินพุท X จำนวนหลายๆ ตัว เพื่อให้การแยกแยะรูปแบบ มีความน่าเชื่อถือมากขึ้น ซึ่งก็เป็นที่มาของชื่อ k-NN กล่าวคือ จะต้องใช้ตัวอย่างใกล้เคียงทั้งสิ้น k ตัวในการตัดสินใจว่า อินพุท X นั้นอยู่ในกลุ่มใด เหตุผลสำคัญที่ระบุไว้เป็น k เช่นนี้ เนื่องจาก เราไม่สามารถจะบอกได้แน่ชัดว่า ค่า k นี้ควรเป็นเท่าไร จำเป็นที่ผู้พัฒนาระบบที่ใช้แยกแยะรูปแบบ จะต้องพิจารณาค่า k ที่เหมาะสมเอง ซึ่งก็มีปัจจัยที่ต้องพิจารณามากมาย ตัวอย่างที่สำคัญๆ เช่น เวลาที่จะต้องใช้ในการคำนวณ, ขนาดของข้อมูลที่ใช้ในการพัฒนาระบบที่มี (ตามทฤษฎี ค่า k ต้องน้อยกว่าจำนวนข้อมูลที่ใช้ในการพัฒนาระบบมากๆ), ความน่าเชื่อถือของระบบ เป็นต้น

แนวที่มีการวิจัยกันมากอีกประเด็นหนึ่ง ก็คือ เมื่อต้องพิจารณาจากตัวอย่างใกล้เคียงถึง k ตัว จะมีวิธีการตัดสินอย่างไร วิธีการอย่างง่ายก็คือ การโหวต นั่นคือ

บรรดาตัวอย่างใกล้เคียงทั้ง k ตัว อยู่ในกลุ่มใดมากที่สุด ก็ถือว่า อินพุท X อยู่ในกลุ่มนั้น อย่างไรก็ตาม มีผลงานวิจัยที่เสนอวิธีการที่แตกต่างไป คือ ให้พิจารณาโดยให้คิด น้ำหนัก (Weight) ของตัวอย่างใกล้เคียงแต่ละตัวไม่เท่ากันในการโหวต เช่นให้น้ำหนักแก่ตัวอย่างใกล้เคียงในอันดับต้นๆ (มีค่าความแตกต่างน้อยกว่า) มากกว่า หรือให้น้ำหนักไปตามจำนวนตัวอย่างในข้อมูลที่ใช้ในการพัฒนาระบบ ของแต่ละกลุ่ม เป็นต้น

ตัวอย่างเช่น ในระบบสำหรับแยกแยะรูปแบบข้อมูล ใช้ข้อมูลสำหรับพัฒนาระบบทั้งสิ้น 10,000 ตัวอย่าง เป็นตัวอย่างที่อยู่ในกลุ่ม ก) 3000 ตัวอย่าง ที่เหลือเป็นตัวอย่างในกลุ่ม ข) พิจารณาด้วย 3-NN พบว่าตัวอย่างใกล้เคียงทั้ง 3 เป็นตัวอย่างในกลุ่ม ก) 2 ตัวอย่าง มีค่าความแตกต่าง 0.5 และ 0.8 ที่เหลือเป็นตัวอย่างในกลุ่ม ข) วัดค่าความแตกต่างได้ 0.1 หน่วย ดังรูป 2.6



รูป 2.6 : ตัวอย่างรูปในการแยกแยะรูปแบบด้วย 3-NN

- ตัดสินด้วย 3-NN แบบพื้นฐาน เราจะตัดสินว่า อินพุท X อยู่ในกลุ่ม ก)

- คิดน้ำหนักจากค่าความแตกต่าง จะตัดสินอินพุท X อยู่ในกลุ่ม ข) (การพิจารณาให้น้ำหนัก มีรายละเอียดแตกต่างกันไปหลายวิธี ในที่นี้พิจารณาคร่าวๆ เท่านั้น)

- คิดน้ำหนักจากจำนวนตัวอย่าง จะให้น้ำหนักในกลุ่ม ก) เป็น 0.3 และกลุ่ม ข) เป็น 0.7 ดังนั้น จะตัดสินอินพุท X อยู่ในกลุ่ม ข) (0.3*2 < 0.7*1)

จากตัวอย่างดังกล่าว จะเห็นว่า แม้จะใช้หลักการของ k-NN เช่นเดียวกัน แต่หากแตกต่างกันในรายละเอียด ก็จะให้ผลการแยกแยะที่แตกต่างกัน ทั้งนี้จะต้องพิจารณาเลือกรายละเอียดวิธีการที่จะใช้ให้เหมาะสมด้วย

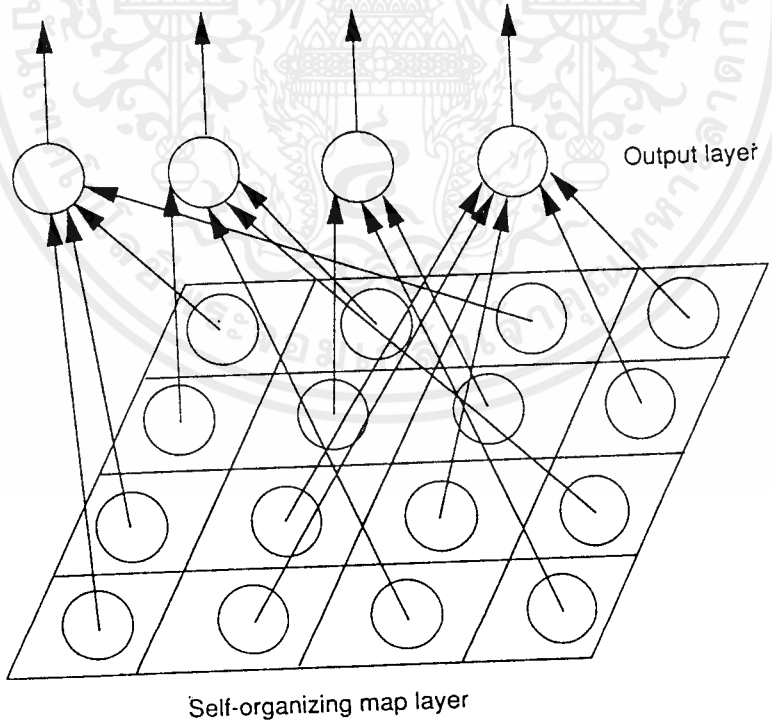
จุดอ่อนของการแยกแยะรูปแบบด้วย k-NN อยู่ที่เวลาที่ต้องใช้ในการคำนวณ เนื่องจากปริมาณของการคำนวณ เพิ่มขึ้นอย่างรวดเร็ว ตามจำนวนข้อมูลที่วิเคราะห์ โดยเฉพาะอย่างยิ่ง ข้อมูลที่ใช้พัฒนาระบบจำเป็นจะต้องมีจำนวนมาก เพื่อให้ระบบมีความถูกต้อง น่าเชื่อถือ แม้ว่าข้อด้อยดังกล่าว จะไม่ใช่ประเด็นสำคัญนัก (เนื่องจากมีการพัฒนาความเร็วของเครื่อง ให้เพิ่มขึ้นได้อย่างรวดเร็ว) แต่ก็ทำให้ต้องมีการศึกษาและวิจัยถึงวิธีการคิดคำนวณแบบต่างๆ บนพื้นฐานของ k-NN โดยพยายามลดปริมาณการคำนวณลง ซึ่งก็ทำให้การศึกษาวิจัยในแนวของ k-NN แตกแขนงออกไปอย่างกว้างขวาง และมีผลงานวิจัยออกมามายมาย ในบทที่ 3 จะได้กล่าวถึงรายละเอียดในการใช้หลักการของ k-NN ในการแยกแยะรูปแบบ โดยใช้เทคนิคที่เรียกว่า Branch and Bound ช่วยลดปริมาณการคำนวณลง

2.6 ทฤษฎีการวิเคราะห์กลุ่มข้อมูลโดยการใช้ Self-organization Maps(SOM)

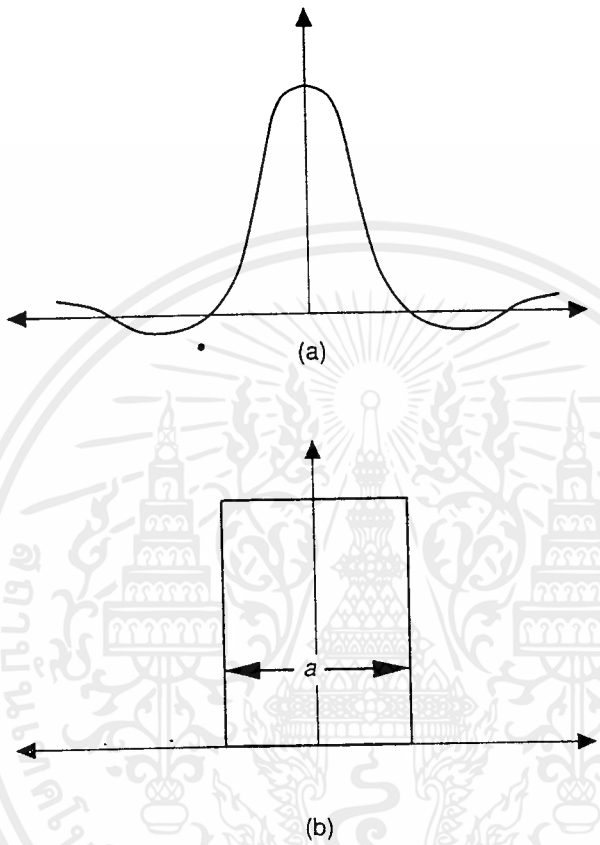
ในระบบของ Self-organization Maps จะประกอบด้วยหน่วยย่อยๆ เรียกว่า ยูนิต (Unit) โดยทฤษฎีแล้ว การทำงานของแต่ละยูนิตจะเป็นแบบขนาน (parallel processing) คือ แต่ละยูนิตจะแยกกันทำงาน และสามารถทำงานไปพร้อมๆ กันได้

ยูนิตใน SOM แบ่งออกเป็น 2 ชั้น ชั้นแรกเรียกว่า ชั้นอินพุท (Input layer) คือยูนิตที่ทำหน้าที่รับสัญญาณอินพุทต่างๆ ของระบบ และป้อนสัญญาณดังกล่าว ไปให้กับยูนิตในชั้นถัดไป เรียกว่า ชั้นเอาท์พุท (Output layer) ซึ่งจะทำหน้าที่ประมวลผลสัญญาณอินพุททุกตัวที่มันได้รับ และให้ค่าเอาท์พุทออกมา ดังแสดงในรูป 2.7

ความพิเศษของ SOM ก็คือ เรามีอัลกอริธึมที่สามารถใช้ในการเตรียมระบบด้วยการเรียนรู้แบบไม่มีข้อมูล อัลกอริธึมดังกล่าวพัฒนาโดย Kohonen ซึ่งนิยมเรียกกันทั่วไปว่า อัลกอริธึมแบบโคโฮเนน



รูป 2.7 : แสดงความสัมพันธ์ระหว่างยูนิตในชั้นอินพุท และชั้นเอาท์พุท



รูป 2.8 : แสดงตัวอย่างฟังก์ชันของค่า z_{ij} ที่ใช้ในการคำนวณเอาท์พุทของยูนิตในชั้นเอาท์พุท

2.6.1 การคำนวณการทำงานของยูนิตในชั้นเอาท์พุท

การทำงานของแต่ละยูนิต นิยามได้ด้วยเซตของสมการดังนี้

$$\dot{y}_i = -r_i(y_i) + \text{net}_i + \sum_j z_{ij} y_j \qquad (2.1)$$

$r_i(y_i)$: เป็นรูปมาตราฐานของฟังก์ชันถดถอย(loss term) ซึ่งมักใช้ $r_i(y_i) = Ay_i$ โดย A เป็นค่าคงที่

net_i : อินพุทของยูนิตในชั้นอินพุทที่ i , net_i มักคำนวณจากผลคูณของเวคเตอร์อินพุทกับเวคเตอร์น้ำหนัก(weight vector) ของหน่วยย่อยนั้น

$\sum_j z_{ij} y_j$: เป็นโมเดลของ lateral interaction ของหน่วยย่อย เป็นการหาผลรวมของทุกๆ หน่วยย่อยในระบบ หากค่า z_{ij} ได้มาจากฟังก์ชันแบบหมวกแม็กซิกัน(Maxican-hat function) ในรูปที่ 2.8a การทำงานของเน็ตเวิร์กจะมีลักษณะกระโดด ในบริเวณรอบๆ หน่วยย่อยที่มีค่า net_i สูงสุด

2.6.2 อัลกอริธึมการเรียนรู้ใน SOM

เนื่องจากค่าเอาท์พุทของเอาท์พุทยูนิตคำนวณได้จาก ผลคูณของค่าอินพุทกับค่าน้ำหนักของเอาท์พุทยูนิตนั้น การเรียนรู้ของระบบ ก็คือการปรับค่าน้ำหนักของเอาท์พุทยูนิตแต่ละยูนิต เพื่อให้ระบบให้ค่าเอาท์พุทตามที่ต้องการนั่นเอง

ในการเรียนรู้ตามอัลกอริธึมของโคโฮเนน จะนำข้อมูลสำหรับพัฒนาระบบมาป้อนเป็นอินพุทให้แก่ระบบ ในการป้อนอินพุทแต่ละครั้ง จะมีการกำหนด วินเนอร์ (winning unit) คือ ยูนิตที่มีเวคเตอร์ของน้ำหนัก ใกล้เคียงกับเวคเตอร์อินพุทมากที่สุด หรือจะนิยามด้วยสมการได้เป็น

$$\|\mathbf{x} - \mathbf{w}_c\| = \min_i\{\|\mathbf{x} - \mathbf{w}_i\|\} \qquad (2.2)$$

x : เวคเตอร์อินพุท

w_i : เวคเตอร์น้ำหนักของหน่วยย่อยที่ i

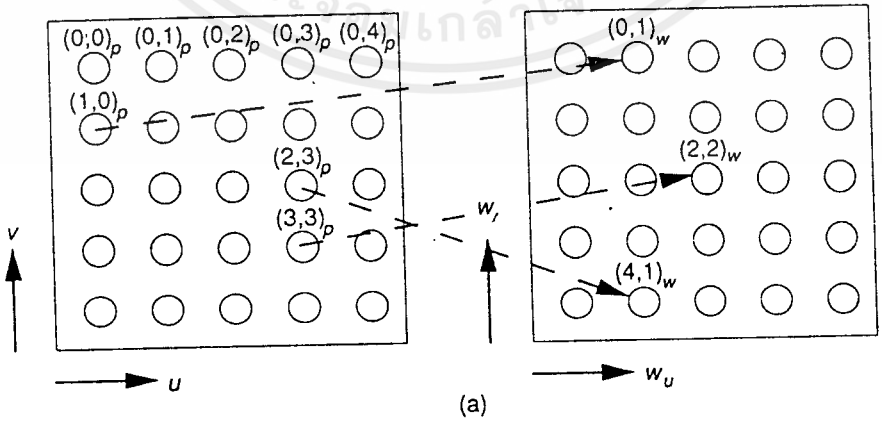
w_c : เวคเตอร์น้ำหนักของวินเนอร์

เราใช้สัญลักษณ์ c แทนวินเนอร์ และยูนิตข้างเคียงของวินเนอร์ จะแทนด้วย N_c ซึ่งในขั้นตอนการเรียนรู้แต่ละครั้ง จะมีการปรับค่าน้ำหนักของวินเนอร์และหน่วยข้างเคียงดังสมการ

$$w_i(t+1) = \begin{cases} w_i(t) + \alpha(t)(x - w_i(t)) & i \in N_c \\ 0 & \text{otherwise} \end{cases} \qquad (2.3)$$

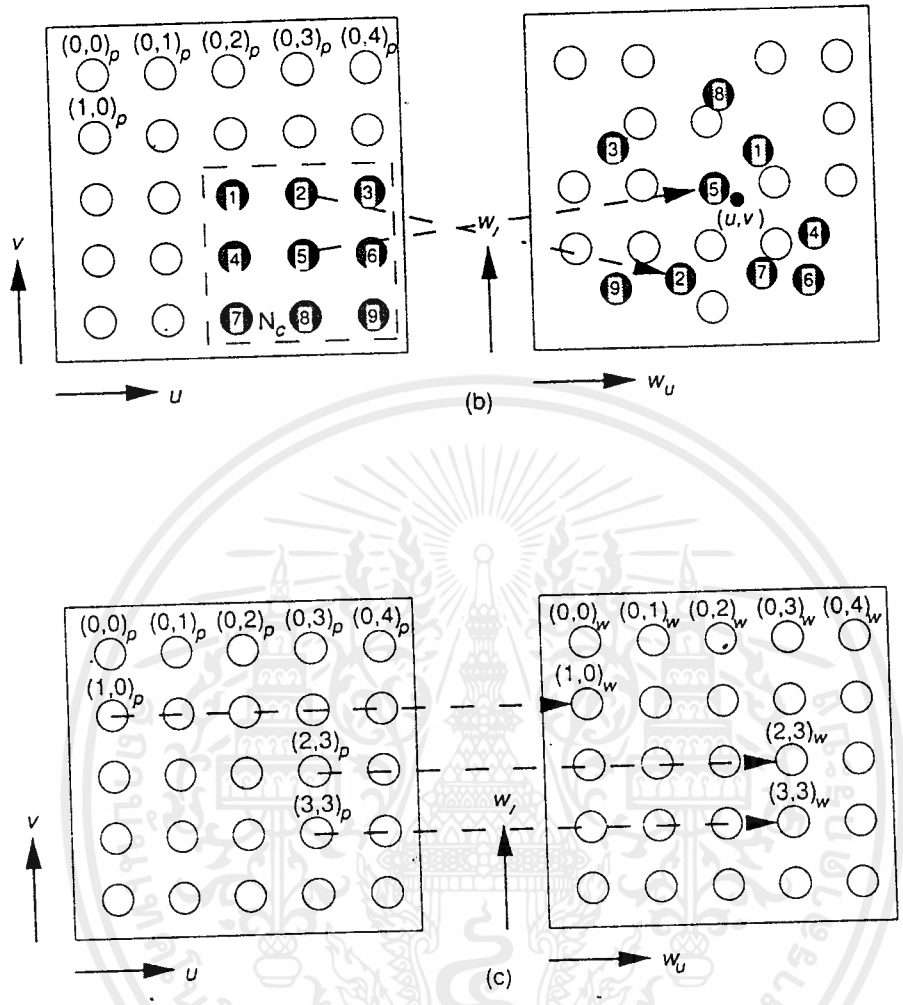
การปรับค่าดังกล่าว ทำให้เวคเตอร์น้ำหนักขยับเข้าใกล้เวคเตอร์อินพุท ดังนั้นเมื่อทำการเรียนรู้ไปถึงจุดสมบูรณ์ เวคเตอร์น้ำหนักของแต่ละยูนิต ก็จะมีค่าใกล้เคียงกับตำแหน่งของมัน

จากตัวอย่างในรูป 2.9 มีการแสดงยูนิตในชั้นเอาท์พุทใน 2 สเปซ คือในสเปซของตำแหน่งของแต่ละยูนิต คือ u,v (มีทั้งสิ้น 25 ยูนิต, เรียงตัวเป็นระนาบ) และการพล็อตตามค่าน้ำหนักของแต่ละยูนิต (ลูกศรเส้นประใช้แสดงให้เห็นว่า เอาท์พุทยูนิตตัวที่หางลูกศร มีเวคเตอร์น้ำหนักเท่าใด) กรณีนี้ ใช้อินพุท 2 ค่า จึงทำให้เวคเตอร์น้ำหนักมี 2 มิติ w_u, w_v รูป 2.8a แสดงสถานะเริ่มต้น



(a)

รูป 2.9 : รูปแสดงผลการเรียนรู้ของระบบ

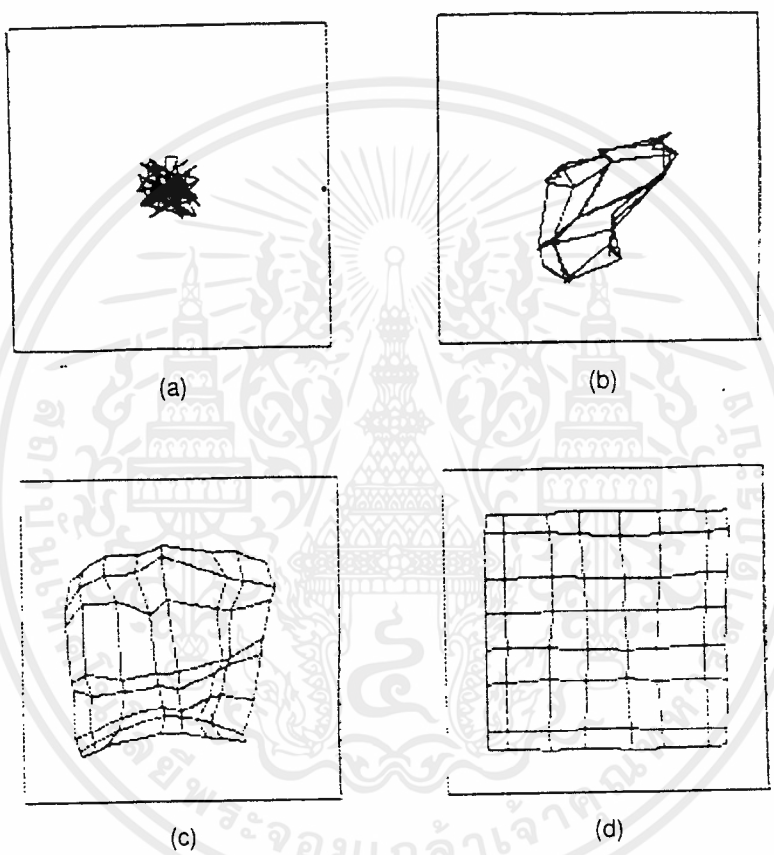


รูป 2.9 : รูปแสดงผลการเรียนรู้ของระบบ (ต่อ)

รูป 2.9b แสดงผลของการปรับค่าตามอัลกอริธึมการเรียนรู้ของโคโฮเนน เมื่ออินพุทเวคเตอร์เป็น (3,3) จะเห็นว่า ทำให้ค่าน้ำหนักของยูนิตที่ตำแหน่ง (3,3) และยูนิตรอบๆ มีการเปลี่ยนแปลงให้มีค่าเข้าใกล้ (3,3) มากขึ้น ส่วนรูป c เป็นรูปแสดงผลของระบบ เมื่อผ่านการเรียนรู้อย่างสมบูรณ์แล้ว จะเห็นว่า ค่าน้ำหนักของยูนิตต่างๆ มีค่าเท่ากับค่าตำแหน่งของมัน

ในทำนองกลับกัน ถ้าเราพล็อตยูนิตต่างๆ ตามเวคเตอร์น้ำหนักของมัน และลาก

เส้นเชื่อมระหว่างยูนิตที่อยู่ข้างเคียงกัน ในระหว่างกระบวนการเรียนรู้ เราจะเห็นจุดต่างๆ มีการเคลื่อนย้ายตำแหน่งจนกระทั่งฟอร์มตัวเป็นรูปร่างตามการกระจายของอินพุทเวคเตอร์ ดังแสดงในรูป 2.10



รูป 2.10 : แสดงการพล็อตเวคเตอร์น้ำหนักของยูนิตต่างๆ ในระหว่างกระบวนการเรียนรู้

2.6.3 การประยุกต์ SOM ในการวิเคราะห์กลุ่ม

เนื่องจากระบบของ SOM จะตอบสนองต่ออินพุทเวคเตอร์ใดๆ ด้วยวินเนอร์ยูนิตเพียงยูนิตเดียวเท่านั้น เมื่อเราให้ค่าน้ำหนักของทุกๆ ยูนิตอย่างเหมาะสมแล้ว ก็สามารถจะจัดจำแนกกลุ่มให้กับข้อมูลต่างๆ ไปตามวินเนอร์ที่ระบบตอบสนองได้

ค่าน้ำหนักที่เหมาะสมสำหรับแต่ละยูนิต จะได้มาจากกระบวนการเรียนรู้ โดยใช้เวคเตอร์คุณลักษณะของข้อมูลในการพัฒนาระบบเป็นอินพุทเวคเตอร์ ภายหลังการเรียนรู้ระบบก็จะตอบสนองต่ออินพุทเวคเตอร์ เป็นกลุ่มๆ ตามที่ต้องการ

การจัดลำดับการป้อนอินพุท ในขั้นตอนของการเรียนรู้ มีความสำคัญมาก อาจมีผลทำให้ระบบสามารถเรียนรู้ได้ภายในเวลาที่แตกต่างกัน หรือแม้แต่ไม่สามารถเรียนรู้ได้ (ผลของการจัดกลุ่ม ไม่ตรงกับที่ต้องการ) ประเด็นนี้ เป็นประเด็นที่นับได้ว่าเป็นจุดอ่อนของ SOM คือ ระบบจะเรียนรู้ได้หรือไม่นั้น ไม่สามารถคาดการณ์ล่วงหน้าได้

เราสามารถประยุกต์ใช้ SOM ในการพิจารณาเลือกคุณลักษณะของข้อมูลได้ ทั้งนี้ใช้คุณลักษณะทั้งหมด (ที่ต้องการตัดสินใจเลือก) ป้อนเป็นอินพุทเวคเตอร์ในกระบวนการเรียนรู้ เมื่อระบบเรียนรู้จนกระทั่งตอบสนองต่อข้อมูลได้อย่างถูกต้องแล้ว ค่าน้ำหนักของคุณลักษณะแต่ละตัว (พิจารณาเฉลี่ยจากทุกๆ ยูนิต) จะเป็นข้อมูลที่ช่วยในการพิจารณาเลือกคุณลักษณะของข้อมูลได้เป็นอย่างดี

นอกจากนี้ เมื่อระบบเรียนรู้จนได้ค่าน้ำหนักที่เหมาะสมแล้ว เราสามารถใช้ระบบดังกล่าว ทำการจำแนกกลุ่มให้กับข้อมูลใหม่ๆ ได้เช่นกัน ทั้งนี้การคำนวณหาวินเนอร์ของข้อมูลอินพุทตัวใหม่ จะทำให้สามารถจำแนกกลุ่มไปตามวินเนอร์ที่คำนวณได้นั้น ในบทที่ 3 จะได้กล่าวถึง การใช้ SOM ในการประยุกต์ใช้กับงานดังกล่าว โดยใช้ตัวอย่างในการทดลองเป็นตัวอย่างเดียวกับที่ใช้ k-NN เพื่อทำการเปรียบเทียบเทคนิคทั้งสองในด้านต่างๆ

บทที่ 3

การทดลองและผลการทดลอง

3.1 ตัวอย่างที่ใช้ในการทดลอง

ในการประยุกต์ใช้คอมพิวเตอร์กับงานต่างๆ นั้น เราพบว่า เรามีข้อมูลจำนวนมากที่จะต้องนำเข้าไปเก็บในคอมพิวเตอร์ รายงานการขาย, ใบสั่งซื้อสินค้า, รูปภาพแสดงสินค้าชนิดต่างๆ แผนที่ ฯลฯ วิธีในการเก็บข้อมูลนั้น พอจะแยกได้เป็น 2 วิธี

วิธีแรกคือ ออกแบบฐานข้อมูลในการจัดเก็บ แล้วใช้คนป้อนข้อมูลเข้าเครื่อง วิธีนี้ใช้ได้ในกรณีที่ข้อมูลเป็นข้อความเท่านั้น เช่น รายงาน, ใบสั่งซื้อสินค้า ที่สำคัญก็คือ วิธีนี้จะเปลืองแรงงานคนในการป้อนข้อมูลเป็นอย่างมาก

ส่วนในกรณีที่ข้อมูลมีโครงสร้างที่ซับซ้อน ไม่สามารถใช้ข้อความต่างๆ แทนเนื้อความของข้อมูลได้โดยสมบูรณ์ เช่น แผนที่ต่างๆ, รูปภาพ หรือแม้แต่เอกสารที่มีรูปภาพประกอบ กรณีนี้ เราจะใช้สแกนเนอร์(Scaner) ทำการสแกนเอกสารดังกล่าวเข้าเก็บในรูปของบิทแมพ(Bitmap) ซึ่งทำได้ค่อนข้างสะดวก รวดเร็ว อย่างไรก็ตาม การเก็บข้อมูลเป็นบิทแมพมีข้อเสียอยู่ 2 ประการ

ประการแรก ข้อมูลที่เก็บในรูปของบิทแมพจะมีขนาดใหญ่มาก เท่าให้เปลืองพื้นที่ในการจัดเก็บ ส่วนประการที่สองคือ การเก็บข้อมูลเป็นบิทแมพ ไม่สื่อความหมายใดๆ คอมพิวเตอร์ไม่สามารถเข้าใจถึงเนื้อความในบิทแมพ

อย่างไรก็ตาม ในปัจจุบัน ก็มีความนิยมใช้สแกนเนอร์ในการจัดเก็บเอกสารต่างๆ ทั้งนี้เนื่องจาก การใช้แรงงานคนในการป้อนข้อมูลนั้น เป็นเรื่องสิ้นเปลืองมาก แต่ข้อเสีย 2 ประการของการเก็บข้อมูลเป็นบิทแมพ ก็เป็นเรื่องสำคัญที่จะต้องแก้ไข จึงเกิดแนวความคิดของระบบที่เรียกว่า OCR

OCR ย่อมาจาก Optical Character Recognition เป็นระบบที่มีจุดมุ่งหมายที่จะรับการป้อนเอกสารผ่านสแกนเนอร์ และให้ผลเป็นเท็กซ์ไฟล์ ซึ่งเก็บข้อความต่างๆ ที่อยู่ในเอกสารนั้น ดังนั้น การทำงานภายในจึงต้องมีการแยกแยะตัวอักษรจากบิทแมพด้วย

การทำงานของ OCR มีโมดูลที่ซับซ้อนมากมาย เช่น การขจัดสัญญาณรบกวน (noise), การตัดตัวอักษร, การแยกแยะตัวอักษรต่างๆ ฯลฯ ในโมดูลต่างๆ เหล่านี้ การแยกแยะตัวอักษร เป็นงานที่สามารถนำเทคนิคการวิเคราะห์ข้อมูลเข้าไปประยุกต์ใช้ได้อย่างมีประสิทธิภาพ

ในปริญญานิพนธ์นี้ จะได้ทดลองนำเทคนิคการวิเคราะห์ข้อมูลที่ได้ศึกษามา นำการประยุกต์ใช้กับการแยกแยะตัวอักษรอย่างง่ายๆ ทั้งนี้เลือกใช้เรื่องของข้อมูลตัวเลขไทย (0-9) เนื่องจากเป็นข้อมูลที่สามารถหาได้ง่าย และมีจำนวนมากพอที่จะนำมาใช้ในการทดลอง(ซึ่งต้องการข้อมูลจำนวนมาก)ได้ ทั้งนี้ตัวเลขที่ใช้ในการทดลองทั้งหมดเป็นตัวพิมพ์ ทั้งสิ้น 29 ฟอนต์

```
00000000000000011110000000000000
00000000001111111111110000000000
00000000111111111111111100000000
00000111111111111111111111100000
00001111111111111111111111110000
00011111111111111111111111111000
00011111111000000000111111111100
01111111110000000000011111111100
01111111000000100000001111111110
01111110000001111100000011111110
11111100000011111111110000111111
11111100000011111111111000011111
11111100000111111001111000011111
11111100001111000111100001111111
11111100001111000111100001111111
01111110000111111111110000111110
01111111000111111111110000111110
01111111111111111111110000111110
00011111111111111111110000111110
00001111111111111111100011111100
00001111111111111111000011111100
00000000111111000000111111111000
00000000000000000001111111110000
00000000000000000011111111110000
00000000000000011111111111100000
00000000000001111111111110000000
00000000000111111111111100000000
00000000000111111111111100000000
00000000001111111110000000000000
00000000000011110000000000000000
```

รูป 3.1 : ตัวอย่างภาพข้อมูลตัวเลข 1 ซึ่งแปลงจากบิทแมพ

3.2 การเตรียมข้อมูล สำหรับการทดลอง

ข้อมูลที่ใช้ในการทดลองแยกเก็บเป็นไฟล์ ไฟล์ละ 1 ตัวอักษร โดยแปลงมาจากบิทแมพที่ได้จากสแกนเนอร์ ดังตัวอย่างในรูป 3.1 ทั้งสิ้น 29 ฟอนต์ หรือ 290 ไฟล์

ข้อมูลทั้งหมด ถูกนำไปผ่านกระบวนการสรรหาค่าคุณลักษณะ เพื่อให้ได้เวคเตอร์ของคุณลักษณะทั้ง 290 ตัว เพื่อใช้เป็นตัวแทนของตัวเลขทั้งหมด ในการวิเคราะห์ต่อไป ทั้งนี้ ได้แบ่งใช้ข้อมูล 20 ฟอนต์ไว้เป็นข้อมูลสำหรับพัฒนาระบบ ที่เหลือ 9 ฟอนต์ ใช้ในการทดสอบระบบ

3.2.1 พิจารณาเลือกคุณลักษณะที่จะใช้ในการทดลอง

ในการทดลอง จะแบ่งเป็น 2 ขั้น ก็คือทดลองเลือกคุณลักษณะที่เหมาะสม ที่จะใช้เป็นตัวแทนของตัวเลขไทย ในการวิเคราะห์ต่อไป ดังนั้น เราจึงต้องเลือกคุณลักษณะของตัวเลขไว้ใช้ในการทดลองจำนวนหนึ่ง

เริ่มจากการเลือกใช้คุณลักษณะขั้นต้นต่างๆ ทั้งหมดเท่าที่เป็นไปได้ ได้แก่ ความสูงของตัวเลข, ความกว้างของตัวเลข, จำนวนจุดดำที่ประกอบกันเป็นตัวเลข และความหนาแน่นของจุดโดยเฉลี่ยทั้งในแนวตั้งและแนวนอน ทั้งสิ้น 5 คุณลักษณะ

หลังจากนั้นจึงได้พิจารณาถึงคุณลักษณะขั้นสูงต่อไป โดยเลือกคุณลักษณะ ที่นิยมใช้ในการแยกแยะตัวอักษร และเป็นคุณลักษณะที่สามารถวัดค่าจากตัวอย่างข้อมูลได้ไม่ยากนัก จึงได้เลือกใช้คุณลักษณะในส่วนต่างของกรอบมาตราฐาน ได้แก่ ในแต่ละส่วนมีจุดดำอยู่หรือไม่ และจำนวนจุดในแต่ละส่วนดังกล่าว

ลักษณะของกรอบมาตราฐาน ก็คือการย่อ/ขยายภาพตัวอย่าง ลงในกรอบขนาด 32x32 จุด และแบ่งภาพที่ได้เป็น 9 ส่วน โดยใช้เส้นที่ 11 และ 22 เป็นเส้นแบ่ง ดังแสดงในรูป 3.1

โดยสรุป ได้เลือกใช้คุณลักษณะ ทั้งสิ้น 15 ประการ ได้แก่ความสูง, ความกว้าง, จำนวนจุดดำ, ความหนาแน่นของจุดโดยเฉลี่ยทั้งในแนวตั้ง/แนวนอน, แฟลกที่บอกว่า มีจุดอยู่ในแต่ละส่วนของกรอบมาตราฐานหรือไม่ และจำนวนจุดในแต่ละส่วน

จากตัวอย่างในรูป เป็นภาพตัวเลขหนึ่ง จะเห็นว่าภาพมีความสูงและความกว้าง

เท่ากัน คือ 32 หน่วย, มีจุดดำทั้งสิ้น 539 จุด จึงมีความหนาแน่นเฉลี่ยทั้งแนวตั้งและแนวนอนเท่ากันด้วย คือ 16.84 (539/32)

เมื่อตีกรอบแบ่งภาพเป็น 9 ส่วนที่เส้น 11 และ 22 แล้ว จะพบว่า ภาพมีจำนวนจุดดำอยู่ในแต่ละส่วนเป็น 49,61,52,65,88,69,2,59,31 ดังนั้นแฟลกของการมีจุดในแต่ละส่วนหรือไม่ จึงมีค่าเป็น 111111111 ฐาน 2 หรือ 511

นั่นคือ จากข้อมูลเลขหนึ่งในรูป 3.1 จะได้เวคเตอร์ของคุณลักษณะซึ่งมี 15 มิติเป็น (32,32,539,16.84,16.84,511,49,61,52,65,88,69,2,59,31)

3.2.2 การสร้างโปรแกรมที่ใช้วัดค่าคุณลักษณะจากข้อมูล

ดังที่ได้กล่าวแล้วว่า ข้อมูลทั้งหมดแยกเก็บเป็นไฟล์ ประกอบด้วย '0' และ '1' โดยมี '\n' ที่ท้ายของแต่ละบรรทัด อักษร '1' แทนจุดดำประกอบกันเป็นตัวเลข ในการวัดค่าคุณลักษณะทั้ง 15 อย่าง จึงมีขั้นตอนคร่าวๆ ดังนี้

1. วนทำซ้ำๆ กับไฟล์ที่เก็บตัวอย่างทั้งสิ้น 290 ไฟล์ ทำข้อ 2-6
2. อ่านข้อมูลจากไฟล์ตัวอย่าง เข้าเก็บในตัวแปรภายใน เรียกว่าอิมเมจ
3. จากอิมเมจที่ได้ ทำการวัดค่าคุณลักษณะพื้นฐานต่างๆ ตามลำดับ ได้แก่ ความสูง, ความกว้าง, จำนวนจุด และความหนาแน่นของจุด ในแนวตั้งและแนวนอน
4. ทำการเปลี่ยนขนาดของอิมเมจไปเป็นขนาดมาตราฐาน (32x32 จุด)
5. วัดค่าคุณลักษณะจากอิมเมจขนาดมาตราฐานที่ได้ ได้แก่แฟลกของการมีจุดในแต่ละส่วน และจำนวนจุดในแต่ละส่วน
6. เขียนค่าคุณลักษณะ ทั้งหมดที่วัดได้ลงในเอาท์พุทไฟล์ เพื่อนำไปใช้ต่อไป

3.3 การวิเคราะห์กลุ่มข้อมูล โดยเทคนิคแบบ k-NN

การนำเทคนิค k-NN เข้าทำการวิเคราะห์กลุ่มข้อมูล เพื่อทำการแยกแยะตัวเลขไทยทั้ง 10 ตัวนั้น ประกอบด้วย 2 ขั้นตอน

ขั้นแรก ใช้การวิเคราะห์เพื่อจัดกลุ่มข้อมูล เพื่อพิจารณาเลือกคุณลักษณะที่เหมาะสมที่จะใช้เป็นตัวแทนของตัวเลขไทย ในการแยกแยะรูปแบบต่อไป จะกล่าวถึงการทดลองในขั้นนี้ ในหัวข้อย่อย 3.3.1

ส่วนในขั้นที่สอง จะใช้เทคนิค k-NN ในการแยกแยะรูปแบบ คือ ตัวเลขไทยทั้ง 10 รูปแบบ ทั้งนี้จะมีการใช้เทคนิคในการลดปริมาณการคำนวณเข้าช่วย เพื่อให้ระบบที่ได้มีประสิทธิภาพมากขึ้น จะกล่าวถึงการทดลองในขั้นนี้ ในหัวข้อย่อย 3.3.2

3.3.1 การวิเคราะห์เพื่อจัดกลุ่มข้อมูล

จากเทคนิคการวิเคราะห์กลุ่มข้อมูลแบบ k-NN ดังที่ได้กล่าวไปแล้วนั้น เราจะทำการทดลองเพื่อพิจารณาเลือกคุณลักษณะ ที่จะใช้แทนตัวเลขไทยได้อย่างเหมาะสม จากคุณลักษณะทั้ง 15 ประการที่ได้เตรียมไว้

หลักการที่ใช้ ก็คือ ทดลองใช้คุณลักษณะต่างๆ ที่มีในการวิเคราะห์กลุ่ม และสังเกตุผลของการจัดกลุ่มที่ได้ ว่าสามารถจัดกลุ่มตัวเลขที่ใช้เป็นข้อมูลในการพัฒนาระบบทั้ง 200 ตัวอย่างได้อย่างถูกต้องหรือไม่ ถ้าสามารถจัดได้อย่างถูกต้อง ก็แสดงว่า เราสามารถใช้คุณลักษณะเหล่านั้น แทนข้อมูลตัวเลขไทยได้ ทั้งนี้ จะต้องทำการทดลองใช้คุณลักษณะทั้ง 15 ประการ ในทุกๆ รูปแบบ เพื่อหาคุณลักษณะชุดที่ดีที่สุด

3.3.1.1 โปรแกรมสำเร็จรูป SPSS

SPSS เป็นโปรแกรมสำเร็จรูปทางสถิติที่เป็นที่นิยมมากตัวหนึ่ง, มีความสามารถในการทำงานด้านสถิติมากมาย รวมทั้งการจัดกลุ่มข้อมูล ตามหลักการของ k-NN ด้วย เราจึงเลือกใช้ SPSS ในการทดลองจัดกลุ่มตามคุณลักษณะต่างๆ

คำสั่งในการจัดกลุ่มข้อมูลใน SPSS คือคำสั่ง CLUSTER ซึ่งเป็นการจัดกลุ่มข้อมูลตามหลักการของ k-NN คือใช้การวัดค่าความแตกต่างระหว่างเวคเตอร์ของคุณลักษณะ

เป็นหลักในการจัดกลุ่ม ทั้งนี้ SPSS จะทำงานเป็นขั้นๆ แต่ละขั้นจะเลือก 2 กลุ่ม(หรือตัวอย่าง) ที่มีค่าความแตกต่างระหว่างกันน้อยที่สุดเข้าด้วยกัน ดังนั้นจะต้องทำงานทั้งสิ้น n-1 ขั้นตอน ในกรณีที่มีข้อมูล n ตัวอย่าง

คำสั่ง CLUSTER นั้น สามารถเลือกวิธีการคำนวณค่าความแตกต่างได้ถึง 6 วิธี โดยระบุในคำสั่งย่อย MEASURE ฟังก์ชันที่ใช้ในการคำนวณค่าความแตกต่างทั้ง 6 แบบ แสดงในรูป 3.2

MEASURE has the following keywords:

SEUCLID *Squared Euclidean distances.* This is the default. This measure should be used with the centroid, median, and Ward's methods of clustering. The distance between two cases is the sum of the squared differences in values for each variable:

$$\text{Distance}(X, Y) = \sum_i (X_i - Y_i)^2$$

EUCLID *Euclidean distances.* The distance between two cases is the square root of the sum of the squared differences in values for each variable:

$$\text{Distance}(X, Y) = \sqrt{\sum_i (X_i - Y_i)^2}$$

COSINE *Cosine of vectors of variables.* This is a pattern similarity measure:

$$\text{Similarity}(X, Y) = \frac{\sum_i (X_i Y_i)}{\sqrt{\sum_i (X_i^2) \sum_i (Y_i^2)}}$$

BLOCK *City-block or Manhattan distances.* The distance between two cases is the sum of the absolute differences in values for each variable:

$$\text{Distance}(X, Y) = \sum_i |X_i - Y_i|$$

CHEBYCHEV *Chebychev distance metric.* The distance between two cases is the maximum absolute difference in values for any variable:

$$\text{Distance}(X, Y) = MAX_i |X_i - Y_i|$$

POWER(p,r) *Distances in an absolute power metric.* The distance between two cases is the rth root of the sum of the absolute differences to the pth power in values on each variable.

$$\text{Distance}(X, Y) = \left(\sum_i (X_i - Y_i)^r\right)^{1/r}$$

Appropriate selection of integer parameters p and r yields Euclidean, squared Euclidean, Minkowski, city-block, minimum, maximum, and many other distance metrics.

DEFAULT *Same as SEUCLID.*

รูป 3.2 : แสดงฟังก์ชันในการคำนวณค่าความแตกต่างระหว่างตัวอย่าง ในคำสั่ง CLUSTER ของโปรแกรม SPSS

นอกจากนั้น ยังสามารถระบุวิธีการวัดค่าความแตกต่างระหว่างกลุ่มตัวอย่างได้ถึง 7 วิธี โดยการระบุไว้ในคำสั่งย่อย METHOD วิธีการทั้ง 7 แสดงในรูป 3.3

BAVERAGE	*Average linkage between groups (UPGMA).* This is the default.
WAVERAGE	*Average linkage within groups.*
SINGLE	*Single linkage or nearest neighbor.*
COMPLETE	*Complete linkage or furthest neighbor.*
CENTROID	*Centroid clustering (UPGMC).* Squared Euclidean distances should be used with this method.
MEDIAN	*Median clustering (WPGMC).* Squared Euclidean distances should be used with this method.
WARD	*Ward's method.* Squared Euclidean distances should be used with this method.
DEFAULT	*Same as BAVERAGE.*

รูป 3.3 : แสดงฟังก์ชันในการคำนวณค่าความแตกต่างระหว่างกลุ่มตัวอย่าง ในคำสั่ง CLUSTER ของโปรแกรม SPSS

เพื่อให้เกิดความเข้าใจในวิธีการจัดกลุ่มของ SPSS, ในรูป 3.4 ได้สมมุติตัวอย่างข้อมูลทั้งสิ้น 8 ตัวอย่าง ทำการคำนวณค่าความแตกต่างระหว่างตัวอย่างทั้ง 8 ดังรูป และแสดงขั้นตอนการจัดกลุ่มของ SPSS เมื่อใช้วิธีการวัดค่าความแตกต่างระหว่างกลุ่มแบบ Complete linkage

ขั้นต้น เป็นการรวมตัวอย่างที่ 2 กับตัวอย่างที่ 6 เข้าด้วยกัน เนื่องจากมีค่าความแตกต่างระหว่างกัน ต่ำที่สุด คือ 0.0022 หน่วย

ในขั้นที่สอง เราพบว่าค่าความแตกต่างระหว่างตัวอย่างที่ 1 กับกลุ่มของตัวอย่าง 2 กับ 6 มีค่าต่ำสุด คือ มีค่า 0.0695 หน่วย (ซึ่งได้จากค่าความแตกต่างสูงสุดระหว่างกลุ่มตัวอย่าง 1 กับกลุ่มตัวอย่าง 2-6) จึงรวมทั้ง 3 ตัวอย่างเข้าเป็นกลุ่มเดียวกัน และทำในขั้นตอนต่อๆ ไป ด้วยหลักการเดียวกัน ดังรูป

ตารางค่าความแตกต่างระหว่างตัวอย่าง

ตัวอย่าง	1	2	3	4	5	6	7
2	0.0695						
3	3.8568	4.7547					
4	7.3979	8.5901	0.5758				
5	7.5713	9.0275	1.2599	1.0346			
6	0.0470	0.0022	4.5917	8.3776	8.7632		
7	2.8999	3.6478	0.0849	1.0427	1.9009	3.5111	
8	0.5125	0.2956	7.1750	11.8018	11.6938	0.3214	5.8506

ขั้นตอนการจัดกลุ่มข้อมูล

Step	I	II	ค่าความแตกต่าง
1.	2	6	0.0022
2.	1	2-6	0.0695
3.	3	7	0.0849
4.	2-6-1	8	0.5125
5.	4	5	1.0346
6.	3-7	4-5	1.9009
7.	2-6-1-8	3-7-4-5	11.8018

รูป 3.4 : แสดงค่าความแตกต่างระหว่าวตัวอย่างสมมุติ 8 ตัวอย่าง และขั้นตอนในการจัดกลุ่มโดยใช้ METHOD = COMPLETE

ผลของคำสั่ง CLUSTIER ของ SPSS นั้น สามารถแสดงให้ดูได้ในหลายๆ รูปแบบ โดยเฉพาะอย่างยิ่ง การติดตามขั้นตอนของการจัดกลุ่มข้อมูล ในทำนองเดียวกับรูป 3.4 ทั้งนี้ จะขอกล่าวถึง วิธีการแสดงผลของ SPSS ไว้คร่าวๆ ดังนี้

- SCHEDULE : จะแสดงลำดับในการจัดกลุ่มเข้าด้วยกัน เริ่มตั้งแต่จัด 2 ตัวอย่างแรกที่ใกล้เคียงกันที่สุด จนสุดท้ายจัด 2 กลุ่มสุดท้ายเข้าด้วยกัน โดยจะแสดงหมายเลขของ 2 กลุ่มนั้น และความแตกต่างระหว่างมัน รวมทั้งมีอินเด็กซ์ไปยังการจัดกลุ่มครั้งถัดไปที่ 2 กลุ่มนั้นๆ (ซึ่งรวมกันแล้ว) ไปรวมเข้ากับกลุ่มอื่นอีกต่อไป
- CLUSTER(min,max) : แสดงหมายเลขกลุ่มที่ตัวอย่างใด ๆ ถูกจัดไว้ โดยจะแสดงในทุกๆ กรณีที่จัดเป็น m กลุ่มเมื่อ min < m < max นอกจากนี้ยังระบุพารามิเตอร์เพียงตัวเดียวได้ เช่น CLUSTER(10) หมายถึงให้แสดงเฉพาะหมายเลขกลุ่มเมื่อจัดตัวอย่างทั้งหมดเป็น 10 กลุ่มนั่นเอง
- DISTANCE : แสดงค่าความแตกต่างระหว่างตัวอย่างใดๆ ซึ่งจะแสดงในลักษณะของตาราง
- VICICLE : แสดงลำดับการจัดกลุ่มคล้ายกับ SHEDULE แต่จะแสดงในรูปกึ่งๆ กราฟ
- HECICLE : มีลักษณะเช่นเดียวกับ VICICLE เพียงแต่แสดงในแนวนอนแทนที่จะเป็นแนวตั้งเหมือนกับ VICICLE เพื่อให้เลือกใช้ได้ตามความเหมาะสม

  การใช้ VICICLE และ HECICLE สามารถกำหนดให้แสดงในบางขั้นตอนเท่านั้น ได้โดยกำหนดพารามิเตอร์ในรูปของ (min,max,step) ซึ่งจะแสดงตั้งแต่ขั้นที่ min ไปจนถึงขั้นที่ max โดยกำหนด step ได้ด้วย
- DENDROGRAM : เป็นแผนผังการจัดกลุ่มของข้อมูล แสดงในลักษณะคล้าย tree ซึ่งแสดงให้เห็นถึงการจัดตัวอย่างใดๆ เข้าด้วยกัน มีข้อเด่นคือ จะแสดงผลในโดยคำนวณสเกล ซึ่งทำให้ในกรณีที่ตัวอย่างที่ทำ CLUSTER

มากๆ ก็สามารถดูผลได้ ในขณะที่การแสดงผลแบบอื่นๆ นั้น เนื่องจากต้องแสดงข้อมูลมาก ทำให้เห็นผลได้ไม่ชัดเจน

3.3.1.2 วิธีการทดลองเพื่อพิจารณาเลือกคุณลักษณะ

ในระบบที่ใช้ในการทดลอง มีคุณลักษณะของตัวเลขไทยทั้งสิ้น 15 ประการ ประกอบกับโปรแกรม SPSS มีวิธีการวัดค่าความแตกต่างถึง 42 วิธี ดังนั้นจะมีรูปแบบต่างๆ ในการทดลองที่เป็นไปได้ถึง 1,376,214 การทดลอง ซึ่งหากไม่มีข้อมูลอื่นๆ ช่วยในการพิจารณา ก็จะต้องทำการทดลองหมดทุกๆ แบบ

สำหรับการทดลองจริง ไม่สามารถจะทำการทดลองในทุกๆ รูปแบบได้ จึงได้เลือกทำการทดลองโดยเปลี่ยนคุณลักษณะที่ใช้ในการวิเคราะห์กลุ่ม ต่างๆ กัน 11 รูปแบบ และเปลี่ยนวิธีการวัดค่าความแตกต่าง 14 วิธี รวมทั้งสิ้น 154 การทดลอง ใช้เวลาในการทดลองทั้งสิ้นประมาณ 80 ชั่วโมง (บนเครื่องพีซี 386-33)

จากการวิเคราะห์ ผลการทดลองทั้งหมด ซึ่งมีขนาดกว่า 70 เมกกาไบต์ ก็สามารถสรุปได้ว่า คุณลักษณะที่สามารถจะใช้เป็นตัวแทนของตัวเลขไทย เพื่อใช้ในการวิเคราะห์และแยกแยะรูปแบบได้อย่างถูกต้อง และมีจำนวนคุณลักษณะน้อยที่สุด ก็คือ จำนวนจุดดำในส่วนต่างๆ ของภาพทั้ง 9 ส่วน หลังจากที่ได้ทำการปรับขนาดเป็น 32x32 แล้ว

3.3.1.3 บทสรุปของการวิเคราะห์เพื่อจัดกลุ่มข้อมูล

จุดประสงค์ของการทดลองนี้ ก็คือการทดลองใช้เทคนิคการวิเคราะห์ข้อมูล k-NN เข้าประยุกต์ในการพิจารณาเลือกคุณลักษณะของข้อมูล ทั้งนี้ก็มีการใช้โปรแกรมสำเร็จรูป คือ SPSS เข้ามาช่วยในการทดลอง ผลของการทดลองเป็นที่น่าพอใจ คือเราสามารถใช้ k-NN ในการพิจารณาเลือกคุณลักษณะของข้อมูลได้ตามที่ต้องการ ซึ่งคุณลักษณะที่เลือกไว้นี้ ก็จะถูกนำไปใช้ในการทดลองขั้นต่อๆ ไปด้วย

3.3.2 การแยกแยะรูปแบบของตัวเลขไทย

การทดลองในหัวข้อนี้ เป็นการใช้หลักการของ k-NN ในการแยกแยะรูปแบบซึ่งจากตัวอย่างที่ใช้ ก็คือการทำการแยกแยะตัวอักษร ทั้งนี้ จะเลือกใช้ค่า k=1 โดยใช้คุณลักษณะที่ได้เลือกไว้จากการทดลองในหัวข้อ 3.3.1

ดังที่ได้กล่าวในบทที่ 2 แล้วว่า ข้อด้อยของ k-NN อยู่ที่เวลาที่ต้องใช้ในการคำนวณ เนื่องจากเมื่อต้องการพิจารณาแยกแยะรูปแบบของข้อมูลใดๆ ก็จะต้องทำการคำนวณค่าความแตกต่างระหว่างข้อมูลอินพุทนั้น กับข้อมูลในการพัฒนาระบบทุกๆ ตัว (ซึ่งมักมีจำนวนมาก เพื่อความถูกต้องของระบบ)

มีการเสนอวิธีการต่างๆ ที่ช่วยลดปริมาณการคำนวณ ในเทคนิค k-NN ซึ่งในปริญญานิพนธ์นี้ ได้นำเทคนิคที่เรียกว่า Branch and Bound ซึ่งทำการแบ่งข้อมูลในการพัฒนาระบบออกเป็นกลุ่มๆ เพื่อจัดโครงสร้างให้เป็นทรี (tree) ในการพิจารณาตัวอย่างใกล้เคียงเพื่อการแยกแยะรูปแบบ ก็อาศัยการค้นหาในทรีเข้ามาช่วย ซึ่งทำให้สามารถลดปริมาณการคำนวณลงได้อย่างมีนัยสำคัญ

3.3.2.1 หลักการของ Branch and Bound

สมมุติมีข้อมูลในการพัฒนาระบบทั้งสิ้น N ตัว $\{x_1,\ldots,x_n\}$ เมื่อต้องการแยกแยะรูปแบบของข้อมูลตัวอย่าง X โดยใช้หลักการของ k-NN ทั้งนี้จะแสดงการทำแบบ 1-NN เท่านั้น ส่วนการประยุกต์ให้ใช้ค่า k อื่นๆ นั้น จะขอละไว้

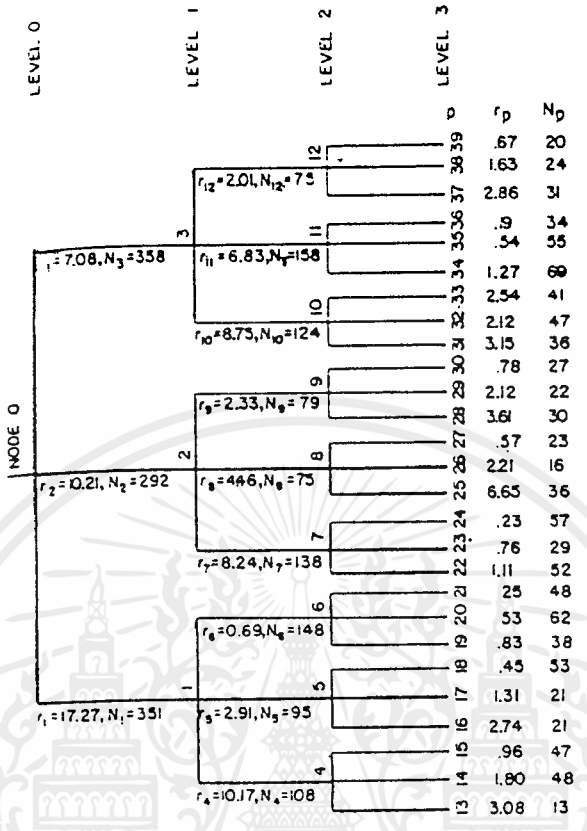
ก่อนที่จะทำการแยกแยะรูปแบบจะต้องมีการเตรียมข้อมูลเสียก่อน ในขั้นแรก จะต้องทำการแบ่งข้อมูลในการพัฒนาระบบออกเป็น 1 กลุ่ม และแต่ละกลุ่มก็แบ่งออกเป็น 1 กลุ่มต่อไปเรื่อยๆ เพื่อจัดโครงสร้างเป็นทรี ดังแสดงตัวอย่างในรูป 3.5 ซึ่งใช้ 1=3 และจัดแบ่งได้เป็น 4 ระดับ

ขั้นตอนถัดไป จะต้องคำนวณค่าพารามิเตอร์ของแต่ละกลุ่มที่แบ่งไว้ 3 ค่าด้วยกัน คือ

N_p : จำนวนข้อมูลที่อยู่ในกลุ่ม p

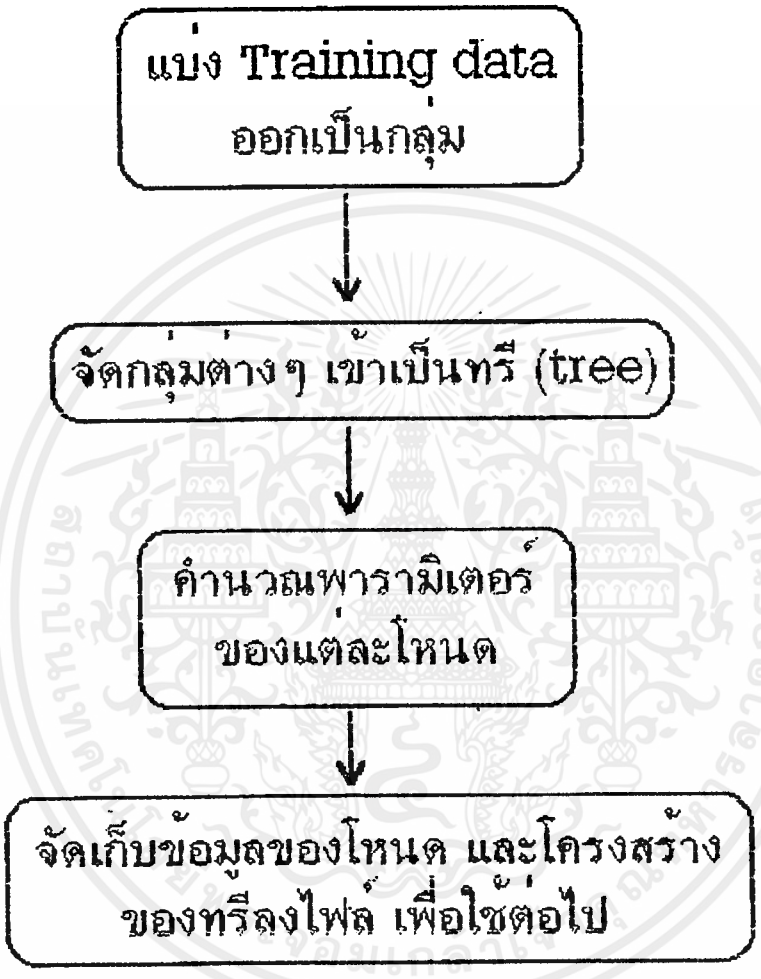
M_p : เวคเตอร์ของคุณลักษณะเฉลี่ยของข้อมูลในกลุ่ม p

r_p : ค่าความแตกต่างสูงสุดจาก M_p ไปยังข้อมูลอื่นๆ ในกลุ่ม p



รูป 3.5 : แสดงการจัดแบ่งกลุ่มข้อมูลโดยใช้ l=3

โดยหลักการของ Branch and Bound แล้ว การจัดแบ่งกลุ่มของข้อมูลในการพัฒนาระบบนั้น จะจัดแบ่งอย่างไรก็ได้ ไม่จำเป็นที่จะต้องจัดให้ข้อมูลที่มีรูปแบบเดียวกันอยู่ในกลุ่มเดียวกัน ส่วนค่า l ที่ใช้ จะใช้ค่าใดก็ได้ อย่างไรก็ตามยิ่งใช้ค่า l สูงๆ ก็จะทำให้ทรีมีความสูงน้อยลง ซึ่งจะมีผลให้ สามารถพิจารณาหาตัวอย่างใกล้เคียงเพื่อทำการแยกแยะรูปแบบได้รวดเร็วยิ่งขึ้น



รูป 3.6 : ขั้นตอนในการเตรียมข้อมูล

สำหรับขั้นตอนการแยกแยะรูปแบบนั้น เราใช้กฎ 2 ข้อ ในการพิจารณาว่า ตัวอย่างใกล้เคียงของข้อมูลอินพุท X มีโอกาสจะอยู่ในกลุ่ม p หรือไม่ และทำการค้นหาไปตามทรี กฎทั้ง 2 แสดงในรูป 3.7

1. ตัวอย่างใกล้เคียงของ X จะไม่อยู่ในกลุ่ม p ถ้า

$$B + r_p < d(X,M_p)$$

B เป็นค่าความแตกต่างระหว่าง X กับตัวอย่างใกล้เคียง ในขณะนั้น ซึ่งจะต้องเซตค่าเริ่มต้นให้เป็น

2. x_i (x_i เป็นสมาชิกในกลุ่ม p) จะไม่ใช่ตัวอย่างใกล้เคียงของ X ถ้า

$$B + d(x_i,M_p) < d(X,M_p)$$

รูป 3.7 : กฎทั้งสองข้อในการพิจารณาหาตัวอย่างใกล้เคียง

3.3.2.2 อัลกอริธึมในการแยกแยะรูปแบบของ Branch and Bound

รูป 3.8 แสดงขั้นตอนในการแยกแยะรูปแบบตามวิธีการของ Branch and Bound ซึ่งก็คือการค้นหาตัวอย่างใกล้เคียงในทรี ซึ่งอธิบายในรูปของอัลกอริทึมได้ดังนี้

1. /* ตั้งค่าเริ่มต้น */

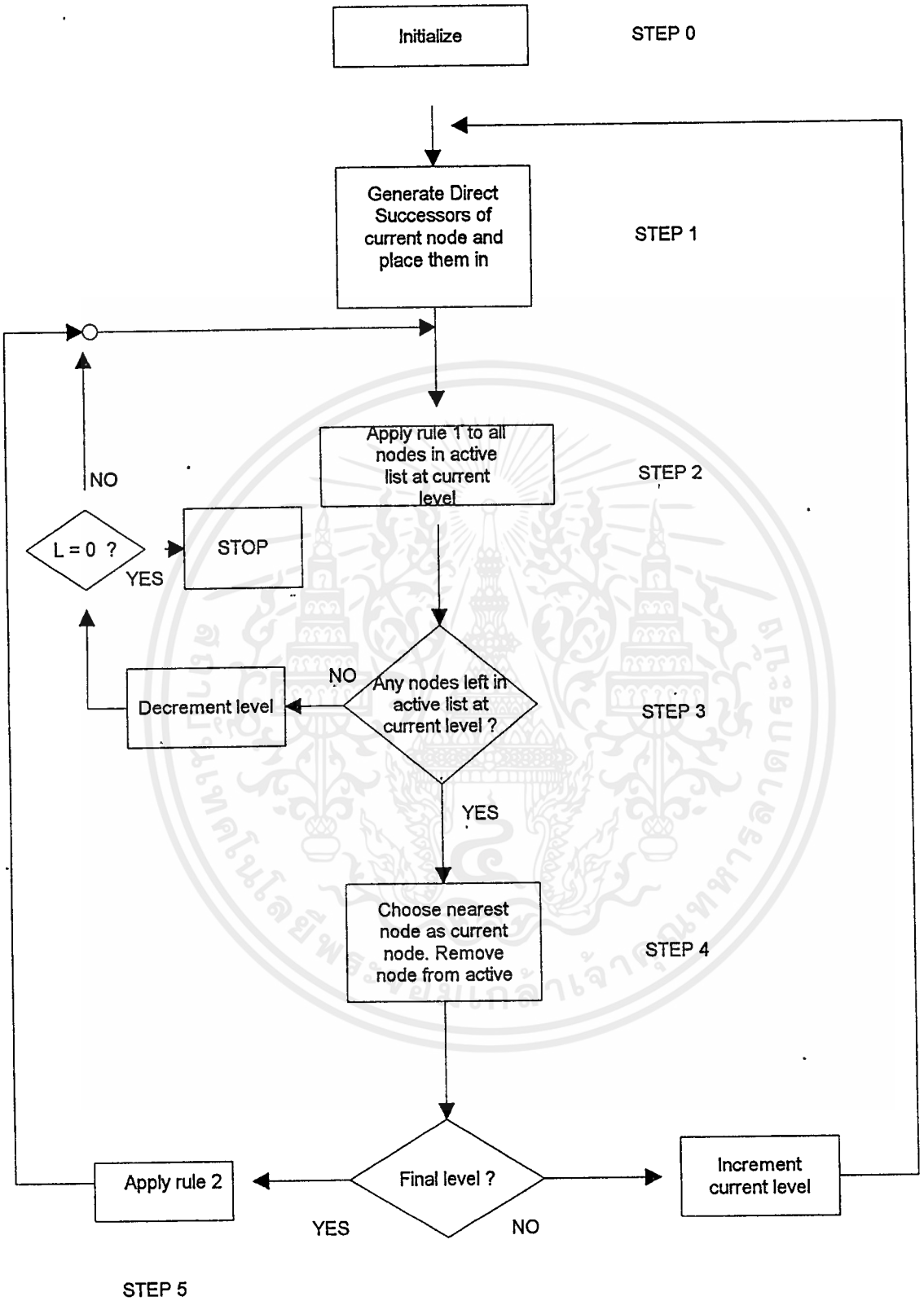
   B = , L = 0 (Current Level), N = 0 (Current Node)

2. /* พิจารณาโหนดลูกของ N */

   พิจารณาโหนดที่เป็นลูกของ N ทั้งหมดไว้ในลิส และทำการคำนวณค่า $d(X,M_p)$ ของโหนดลูกดังกล่าว

3. /* ทดสอบกลุ่มที่ N ด้วยกฎข้อ 1 */

   เช็คกฎข้อ 1 กับทุกๆ โหนดในลิส ถ้าโหนดใดไม่เป็นไปตามเงื่อนไขให้นำออกจากลิส



รูป 3.8 : ขั้นตอนในการแยกแยะรูปแบบตามวิธีการของ Branch and Bound

4. /* คำนวณย้อนกลับ */

ถ้าไม่มีโหนดใดอยู่ในลิส ให้ย้อนกับไปทำการคำนวณในระดับก่อนหน้า โดยการลดค่า L ลง 1 (หาก L เป็น 0 จบการทำงาน) แล้วไปทำข้อ 3 แต่หากยังมีโหนดอยู่ในลิส, ไปทำข้อ 5

5. /* เลือกโหนดที่เหมาะสมเพื่อเลื่อนไปยังระดับถัดไป */

เลือกโหนดในลิสที่มีค่า $d(X,M_p)$ ต่ำที่สุด ใช้เป็น N แทนค่าเดิม และนำโหนดนั้นออกจากลิส

ถ้า L เป็นระดับสุดท้าย ไปทำข้อ 6 แต่ถ้ายังไม่ใช่ให้เพิ่มค่า L ขึ้น 1 และไปทำข้อ 2

6. /* ทดสอบข้อมูลในกลุ่ม N ด้วยกฎข้อ 2 */

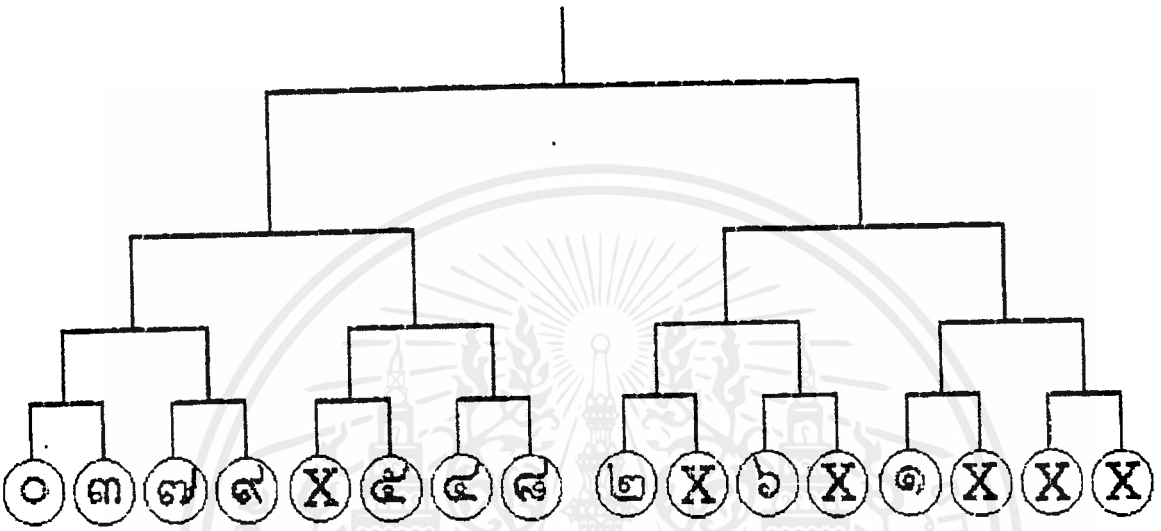
ใช้กฎข้อสองเพื่อคัดข้อมูลในกลุ่ม N ที่มีโอกาสจะเป็นตัวอย่างใกล้เคียงของของ X จากนั้นจึงคำนวณ $d(X,x_i)$ เฉพาะ x_i ที่ผ่านเงื่อนไขในกฎ เพื่อพิจารณาตัวอย่างใกล้เคียงตัวใหม่ และเปลี่ยนค่า B ไปเป็นค่า $d(X,x_i)$ ที่คำนวณได้

จากนั้น ไปทำข้อ 3 เพื่อทดสอบโหนดอื่นๆ ต่อไป

3.3.2.3 วิธีการทดลองแยกแยะรูปแบบของตัวเลขไทย

ในขั้นแรก คือการเตรียมข้อมูลก่อนทำการแยกแยะตามวิธีการ Branch and Bound โดยทำการแบ่งข้อมูลที่เตรียมไว้สำหรับพัฒนาระบบจำนวน 200 ตัวอย่าง ออกเป็น 10 กลุ่มตามรูปแบบของข้อมูล ทั้งนี้เพื่อความสะดวกในการแยกแยะรูปแบบ โดยใช้ค่า 1 = 2 ซึ่งทำให้ต้องมีโหนดเทียม (dummy node) อยู่ 6 โหนด

การใช้ 1=2 ทำให้ทรีที่ได้ มีความสูง 5 ระดับ ซึ่งในการจัดโครงสร้างของทรีดังกล่าว พยายามจัดลำดับในการจัดกลุ่มทั้ง 10 ให้ตรงกับลำดับที่ได้จากทดลองในหัวข้อ 3.3.2 ด้วย



รูป 3.9 : ทรีที่ใช้ในการแยกแยะรูปแบบของตัวเลขไทย

การจัดกลุ่มดังกล่าว รวมทั้งการคำนวณค่าพารามิเตอร์ของแต่ละกลุ่มนั้น ได้เขียนโปรแกรมขึ้นเพื่อจัดการ โดยอ่านอินพุทคือ ค่าคุณลักษณะ (ทั้ง 9 ประการ) ของ ข้อมูลในการพัฒนาระบบทั้ง 200 ตัวอย่างจากไฟล์ชื่อ "TRAIN.DAT" หลังจากทำการคำนวณแล้วก็ทำการเขียนข้อมูลที่จำเป็นต้องใช้ ลงในไฟล์ชื่อ "TRAINING.DAT" เพื่อเก็บไว้ใช้ในขั้นตอนการแยกแยะต่อไป โดยไม่ต้องทำการจัดกลุ่ม และคำนวณอีก ซอร์สโปรแกรมของการจัดกลุ่มและคำนวณดังกล่าว ได้แสดงไว้ด้วยแล้ว(PREPARE.C)

ส่วนโปรแกรมซึ่งทำการแยกแยะรูปแบบนั้น ในส่วนของการค้นหาตัวอย่างใกล้เคียงจะใช้ทั้งวิธีการพื้นฐาน และวิธีการ Branch and Bound ทั้งนี้ก็เพื่อเปรียบเทียบให้เห็นถึงความแตกต่าง โปรแกรมจะรับอินพุทจากอินพุทไฟล์ ซึ่งจะป้อนอินพุทได้ 2 แบบ คือ ป้อนค่าคุณสมบัติ (ทั้ง 9) โดยตรง และป้อนเป็นชื่อไฟล์ที่เก็บภาพของตัวเลขที่ต้องการพิจารณาแยกแยะรูปแบบ โดยโปรแกรมจะทำการวัดค่าคุณลักษณะจากไฟล์ภาพดังกล่าว และทำการพิจารณารูปแบบให้ ทั้งนี้ในอินพุทไฟล์จะต้องมีฟอร์แมตที่แน่นอนคือ จำนวนอินพุทแบบระบุค่าคุณลักษณะ ตามด้วยอินพุทเหล่านั้น จากนั้นจึงเป็นจำนวนอินพุทที่ระบุเป็นชื่อไฟล์ ตามด้วยชื่อไฟล์เหล่านั้น ทั้งนี้ อาจระบุจำนวนเป็น 0 ได้ทั้ง 2 กรณี ถ้าต้องการ โปรแกรมจะทำการคำนวณ และเก็บผลที่ได้ไว้ในเอาท์พุทไฟล์

การใช้งานโปรแกรมจะต้องระบุชื่ออินพุทไฟล์ และเอาท์พุทไฟล์เป็นพารามิเตอร์ เอาท์พุทที่ได้นอกจาก รูปแบบที่พิจารณาได้แล้ว จะระบุค่าความแตกต่างระหว่างอินพุทกับตัวอย่างใกล้เคียงที่คำนวณได้ รวมทั้งเวลาที่ใช้ในการหาตัวอย่างใกล้เคียง เพื่อใช้เปรียบเทียบเวลาที่ใช้ ระหว่างวิธีการพื้นฐาน กับวิธี Branch and Bound

โปรแกรมดังกล่าว แยกเป็น 2 ไฟล์คือ BR&BO.C และ FEATURE.C ซึ่งซอร์สโปรแกรมของทั้งสองไฟล์ ได้แสดงไว้ในรายงานด้วย

3.3.2.4 บทสรุปของการแยกแยะรูปแบบด้วย k-NN

จุดอ่อนของ k-NN คือเวลาที่ใช้ในการคำนวณ โดยเฉพาะอย่างยิ่ง เมื่อใช้ k-NN ในการพิจารณาแยกแยะรูปแบบ ซึ่งมักต้องการการตอบสนองที่รวดเร็ว สาเหตุที่ทำให้ k-NN ใช้เวลาในการคำนวณสูงมากก็คือ กระบวนการค้นหาตัวอย่างใกล้เคียงนั้นใช้เวลามากเนื่องจาก จะต้องคำนวณค่าความแตกต่างระหว่างข้อมูลที่มีอยู่ทุกๆตัว กับข้อมูลที่ต้องการแยกแยะรูปแบบ วิธีการ Branch and Bound ช่วยให้สามารถค้นหาตัวอย่างใกล้เคียงได้ โดยสามารถลดจำนวนครั้งในการคำนวณค่าความแตกต่างได้เป็นอย่างมาก ซึ่งมีผลทำให้เวลาที่ใช้ในการแยกแยะรูปแบบลดลงด้วย

จากตัวอย่างที่ใช้ในการทดลอง เมื่อเปรียบเทียบเวลาที่ใช้ในการแยกแยะรูปแบบ

เมื่อใช้ Branch and Bound และใช้วิธีพื้นฐาน จะเห็นได้ชัดเจนว่า เวลาในวิธีพื้นฐานนั้นค่อนข้างจะคงที่ (เนื่องจากปริมาณการคำนวณเท่าก้นทุกครั้ง) และเวลาในกรณีที่ใช้ Brancd and Bound นั้นจะน้อยกว่าอย่างน้อยครึ่งหนึ่ง ซึ่งจะเห็นว่ามากน้อยกว่ากันไม่มากนัก (เฉลี่ยประมาณ 25 วินาที) ทั้งนี้ก็เนื่องจากขนาดของข้อมูลในการพัฒนาระบบที่มี มีขนาดเพียง 200 ตัวอย่างเท่านั้น ในระบบที่จะนำไปใช้งานจริงนั้น เพื่อให้มีความเชื่อถือได้สูง จะต้องใช้ข้อมูลดังกล่าว จำนวนมากกว่านี้มาก ความเร็วที่เพิ่มขึ้นของ Branch and Bound นั้น จึงมีความสำคัญมาก

เมื่อสร้างระบบดังกล่าวได้แล้ว ก็ได้นำข้อมูลที่เตรียมไว้สำหรับทดสอบระบบ ทั้ง 90 ตัวอย่าง มาทำการทดลองแยกแยะรูปแบบ เพื่อทดสอบการทำงานของระบบ ผลการทดลองพบว่า ระบบที่สร้างสามารถแยกแยะรูปแบบของข้อมูลที่ใช้ทดสอบได้อย่างถูกต้องทั้งหมด อย่างไรก็ตาม ไม่ได้หมายความว่า ระบบนี้จะสามารถนำไปใช้เป็นในการแยกแยะตัวเลขไทยได้อย่างสมบูรณ์ ทั้งนี้เนื่องจากมีปัจจัยหลายๆ อย่างที่ได้ทำการควบคุมไว้ในการทดลอง เช่น ข้อมูลตัวเลขทั้งหมดเป็นตัวพิมพ์ และไม่ใช่ฟอนต์อักษรประดิษฐ์ เป็นต้น การจะสร้างระบบสำหรับแยกแยะตัวเลขไทยให้ใช้งานได้จริงนั้น จะต้องมีการปรับปรุงอีกมากในหลายๆ ด้าน ซึ่งไม่ได้อยู่ในขอบเขตของปริญญานิพนธ์นี้

3.4 การวิเคราะห์กลุ่มข้อมูล โดย Self-organization Maps

ในปริญญานิพนธ์นี้ ได้นำ SOM มาทำการทดลองเรื่องการวิเคราะห์กลุ่มข้อมูล โดยมีจุดประสงค์เพียงเพื่อเปรียบเทียบวิธีการ, ความสะดวก และประสิทธิภาพของ SOM กับ k-NN เท่านั้น จึงไม่ได้ทำการศึกษาวิธีการของ SOM โดยละเอียด เน้นเพียงแต่ให้สามารถนำมาใช้งานได้เท่านั้น ทั้งนี้ โปรแกรมที่นำมาใช้งาน ก็มิได้พัฒนาขึ้นเอง แต่ได้ใช้ซอร์สโปรแกรมจากตำราของ Self-organization Maps โปรแกรมที่นำมาใช้เป็นระบบที่มีความยืดหยุ่นสูง สามารถประยุกต์ใช้ได้กับหลายๆ งาน ซอร์สโปรแกรมดังกล่าว ได้นำมาแสดงไว้ในรายงานด้วย

อย่างไรก็ตาม ก่อนจะกล่าวถึงการทดลอง ก็จะกล่าวอธิบายในหลักการอย่างง่ายๆ

3.4.1 หลักการคร่าวๆ ของ SOM

ในระบบของ SOM จะแบ่งเป็นยูนิต ทำงาน(การคำนวณ)ขนานกัน ยูนิตเหล่านี้จะรับเวคเตอร์อินพุทซึ่งมีมิติเป็น d นำมาประมวลผลร่วมกับเวคเตอร์น้ำหนักของแต่ละยูนิต (ซึ่งไม่เท่ากัน) ซึ่งมีมิติเป็น d เช่นกันตามสมการ

$$\text{ผลลัพท์ของยูนิตที่ n} \quad = \quad \sum_{i} I_i \cdot W_{ni}$$

แม้ว่าทุกๆ ยูนิตจะได้รับอินพุทชุดเดียวกันก็ตาม แต่เนื่องจากเวคเตอร์น้ำหนักของแต่ละยูนิตมีค่าไม่เท่ากัน จึงทำให้ผลลัพท์ของยูนิตต่างๆ ไม่เท่ากันไปด้วย เราเรียกยูนิตที่ให้ผลลัพท์มีค่าสูงที่สุดว่า วินเนอร์

SOM มีอัลกอริธึมในการเรียนรู้ ซึ่งพัฒนาโดย Teuvo Kohonen โดยมีหลักการอยู่ที่การพยายามปรับค่าเวคเตอร์น้ำหนักของแต่ละยูนิต เพื่อให้ระบบให้ตอบสนองต่อเวคเตอร์อินพุท โดยมีความสัมพันธ์เป็นสัดส่วนกับค่าของเวคเตอร์นั้น กล่าวคือ เมื่อผ่านเวคเตอร์อินพุทที่มีค่าใกล้เคียงกัน ระบบก็จะตอบสนอง โดยให้วินเนอร์เป็นยูนิตเดียวกัน (หรืออยู่ในกลุ่มเดียวกัน)

อัลกอริธึมของโคโฮเนน จะทำปรับค่าเวคเตอร์น้ำหนักเป็นสัดส่วนตามผลต่างเวคเตอร์อินพุทกับเวคเตอร์น้ำหนักเดิม เขียนเป็นสมการก็คือ

$$W_{ni}(t+1) \quad = \quad W_{ni}(t) \quad + \quad eta * (W_{ni}(t) - I_i)$$

ค่า eta เรียกว่าสัมประสิทธิ์การเรียนรู้ (learning ratio) ซึ่งจะมีค่าน้อยกว่า 1 เสมอ โดยในระหว่างกระบวนการเรียนรู้ จะมีการลดค่าของ eta ลงอย่างสม่ำเสมอ จะเห็นว่า กระบวนการเรียนรู้จะต้องใช้เวคเตอร์อินพุทในกระบวนการด้วย ซึ่งจะต้องเตรียมข้อมูลในการพัฒนาระบบไว้อย่างเพียงพอ กระบวนการดังกล่าว จะต้องทำซ้ำๆ จนกระทั่งระบบสามารถทำงานได้อย่างถูกต้อง (โดยปรกติเป็นหลักแสนหรือหลักล้านรอบ)

3.4.2 วิธีการทดลองวิเคราะห์กลุ่มข้อมูลด้วย SOM

หัวใจของระบบ อยู่ในขั้นตอนการเรียนรู้ เพื่อให้ระบบสามารถวิเคราะห์กลุ่มข้อมูลได้อย่างถูกต้อง โดยจะต้องเรียนรู้จากข้อมูลอินพุทจำนวนหนึ่ง ทั้งนี้ ในการทดลองเราได้ใช้ข้อมูลชุดที่เตรียมไว้เป็นข้อมูลในการพัฒนาระบบทั้ง 200 ตัวอย่าง ป้อนให้กับระบบ ซึ่งในกรณีนี้จะมีขนาดของอินพุทเป็น 15 มิติ และกำหนดให้ระบบมีเอาท์พุทยูนิตจำนวน 64 ยูนิต และจำนวนรอบในการเรียนรู้ 30,000 รอบ (นับว่าน้อยมาก)

จากการทดลอง พบว่า ระบบใช้เวลาในการเรียนรู้ประมาณ 30 ชั่วโมง บนเครื่อง 486-33 (คอมไพล์โปรแกรมด้วย BORLAND C++, 3.1)

ในการทดลองให้ระบบเรียนรู้หลายๆ ครั้ง พบว่า บางครั้งระบบก็สามารถเรียนรู้จนตอบสนองต่อข้อมูลได้ตามที่ต้องการ แต่บางครั้งก็ไม่ ตรงจุดนี้ เข้าใจว่าขึ้นกับค่าเวคเตอร์น้ำหนักตอนเริ่มต้นโปรแกรม ซึ่งใช้การแรนดอม

นอกจากนี้ยังพบว่า ลำดับของการป้อนข้อมูลในขั้นตอนการเรียนรู้ มีผลต่อการเรียนรู้ด้วย การจัดลำดับอย่างถูกต้อง จะทำให้ระบบสามารถเรียนรู้ได้เร็ว แต่ก็ยังไม่สามารถสรุปได้ว่า จัดลำดับอย่างไร จึงจะให้ผลดีที่สุด อย่างไรก็ตาม ลำดับของการป้อนข้อมูลที่สังเกตุว่าให้ผลดีคือ จัดเรียงข้อมูลที่มีรูปแบบเหมือนๆ กัน ป้อนติดต่อกันไป

3.4.3 ผลการทดลอง

หลังจากผ่านกระบวนการเรียนรู้แล้ว ก็ทำการทดสอบระบบในการแยกแยะรูปแบบโดยใช้ข้อมูลที่เตรียมไว้สำหรับทดสอบระบบทั้ง 90 ตัวอย่าง ผลการทดลองพบว่าระบบสามารถแยกแยะรูปแบบของตัวอย่างทั้ง 90 ตัวอย่างได้อย่างถูกต้อง

เนื่องจากระบบที่ใช้เป็นระบบทั่วๆ ไป สามารถนำไปประยุกต์ใช้กับเรื่องใดๆ ก็ได้ การแสดงผลต่างๆ จึงเป็นกลางๆ การตีความผลของระบบจะต้องมีความเข้าใจในระบบพอสมควร จึงได้ทำการพัฒนาโปรแกรมขึ้นมาใหม่ เพื่อทำการแยกแยะรูปแบบในเรื่องตัวเลขไทย โดยนำผลที่ได้จากการเรียนรู้ในขั้นแรกมาใช้ เพื่อให้มีการแสดงผลในการจำแนกกลุ่มได้ชัดเจน และเข้าใจง่ายขึ้น แต่ก็ยังคงใช้หลักการของระบบเดิมนั่นเอง โปรแกรมดังกล่าว ได้แสดงซอร์สโปรแกรมไว้เช่นกัน

3.5 บทสรุป k-NN กับ Self-organization Maps

ในปริญญานิพนธ์นี้ จะทำการเปรียบเทียบเทคนิคการวิเคราะห์ข้อมูลทั้งสอง เฉพาะในแง่มุมของการประยุกต์ใช้ในปริญญานิพนธ์นี้เท่านั้น ซึ่งจะเห็นได้ว่า ทั้งสองเทคนิคสามารถจะนำมาใช้สร้างระบบสำหรับแยกแยะรูปแบบตัวเลขไทยได้อย่างถูกต้อง (เฉพาะกับข้อมูลที่ใช้ในการทดลอง)

ในขั้นตอนของการวิเคราะห์เพื่อจัดกลุ่มข้อมูล, k-NN จะต้องทดลองทำการจัดกลุ่มโดยใช้คุณลักษณะต่างๆ กัน ซึ่งมีหลายๆ รูปแบบ และนำผลการจัดกลุ่มที่ได้มาวิเคราะห์เพื่อหาวิธีการที่เหมาะสมต่อไป ส่วน SOM นั้น เราสามารถปล่อยให้ระบบเรียนรู้ได้เอง ซึ่งมีความสะดวกกว่ากันมาก อย่างไรก็ตาม ในการเรียนรู้ได้เองของ SOM นั้น เป็นที่ทราบกันดีว่า อาจจะไม่ได้ผลตามที่ต้องการทุกครั้ง จากการทดลอง เราพบว่า บางครั้งระบบจะเรียนรู้ได้เร็ว บางครั้งก็ช้า และมีหลายครั้งที่การจัดกลุ่มผิดไปจากที่ควรจะเป็น ซึ่งในเรื่องนี้ เป็นที่ข้อด้อยของ SOM

สำหรับการแยกแยะรูปแบบ, k-NN จะต้องทำการคำนวณค่าความแตกต่างระหว่างเวคเตอร์อินพุท กับข้อมูลในการพัฒนาระบบทุกๆ ตัว จึงจะสามารถแยกแยะรูปแบบได้ ทำให้ระบบมี เวลาในการตอบสนอง (response time) สูง ส่วนใน SOM การคำนวณของระบบมีเพียงการคำนวณผลลัพท์ของเวคเตอร์อินพุทของยูนิตต่างๆ เพื่อพิจารณาวินเนอร์ ซึ่งมีการคำนวณไม่มากนัก จึงทำให้ SOM มีเวลาในการตอบสนองที่ดีกว่า อย่างไรก็ตาม ก็มีงานวิจัยต่างๆ สนับสนุน k-NN อยู่หลายชิ้น ที่พยายามหาวิธีการ ที่จะลดปริมาณการคำนวณลง ซึ่งก็ได้นำหนึ่งในวิธีการเหล่านั้น มาใช้ในการทดลองด้วย คือวิธีการที่เรียกว่า Branch and Bound

จากการทดลอง ถ้าจะเปรียบเทียบกันในเรื่องประสิทธิภาพ โดยจะขอนิยามประสิทธิภาพด้วยความถูกต้องและ เวลาในการตอบสนอง จะเห็นได้ว่า เรื่องของความถูกต้องของระบบนั้น ทั้ง 2 เทคนิคมีความสามารถเท่าเทียมกัน เนื่องจากต่างก็สามารถแยกแยะรูปแบบของตัวเลขไทยได้ถูกต้อง ส่วนในเรื่องเวลาในการตอบสนองนั้น SOM จะเหนือกว่า k-NN อยู่มาก (ตามเทคโนโลยีในปัจจุบัน)

กิตติกรรมประกาศ

ปริญญานิพนธ์นี้ สำเร็จลงได้ด้วยความกรุณาของหลายๆ ท่าน ข้าพเจ้าขอกราบขอบพระคุณอาจารย์วัชระ ฉัตรวิริยะ อาจารย์ที่ปรึกษา ซึ่งได้ให้ความช่วยเหลือ ตลอดจนให้คำแนะนำในหลายๆ ด้าน

ขอขอบพระคุณบุคลากรของภาควิชาวิศวกรรมคอมพิวเตอร์ ซึ่งได้อำนวยความสะดวกในการทำงาน ตลอดจนการเครื่องคอมพิวเตอร์ของภาควิชา

ขอขอบคุณคุณสุวิชา มุสิกจรัล ซึ่งได้ให้ความช่วยเหลือเป็นอย่างมาก ในการจัดหาข้อมูลที่ใช้ในการทดลอง

ขอบคุณเพื่อนๆ และน้องๆ นิสิตคณะวิศวกรรมศาสตร์ ทั้งภาควิชาคอมพิวเตอร์ และภาควิชาอื่นๆ ที่คอยให้กำลังใจ คำแนะนำ ตลอดจนความช่วยเหลือต่างๆ ในการทำปริญญานิพนธ์ฉบับนี้จนสำเร็จลุล่วงด้วยดี

ขอกราบขอบพระคุณ พ่อ แม่ ที่ออกทั้งกำลังกาย กำลังใจ และกำลังทรัพย์ ในการสนับสนุนการทำงาน

และสุดท้ายนี้ ขอขอบพระคุณสถาบันเทคโนโลยี พระจอมเกล้า เจ้าคุณทหารลาดกระบัง ที่ให้การศึกษาทั้งในด้านทฤษฎีและปฏิบัติ

หนังสืออ้างอิง

1. ผศ. อรุณ จริวัฒน์กุล (บรรณาธิการ), "ชีวสถิติ", ภาควิชาชีวสถิติและประชากรศาสตร์, คณะสาธารณสุขศาสตร์ มหาวิทยาลัยขอนแก่น, 2531
2. Eric Davalo and Patrick Naim, "Neural Networks", University of Manchester, 1991
3. James A. Freeman, David M. Skapura, "Neural Networks, Algorithms, Applications and Programming Techniqus"
4. Norusis, Marija J., "SPSS/PC+ V 2.0 base manual", SPSS Inc. 1988
5. Norusis, Marija J., "SPSS/PC+ advanced statistics V 2.0", SPSS Inc. 1988
6. Robert J. Schalkoff, Clemson University, 1992, "Pattern Recognition, Statistical, Structural and Neural Approaches".
7. Russell C. Eberhart, Roy W. Dobbins, "Neural Network PC Tools, A Practical Guide"
8. Tzay Y. Young, University of Miami, King-Sun Fu, Purdue University, "Handbook of Pattern Recognition and Image Processing", 1986
9. Keinosuke, Fukunaga and Patrenahalli M. Narendra, "Branch and Bound Algorithm", IEEE Transaction on computer, 1974