

บทที่ 3

วิธีการดำเนินการวิจัย

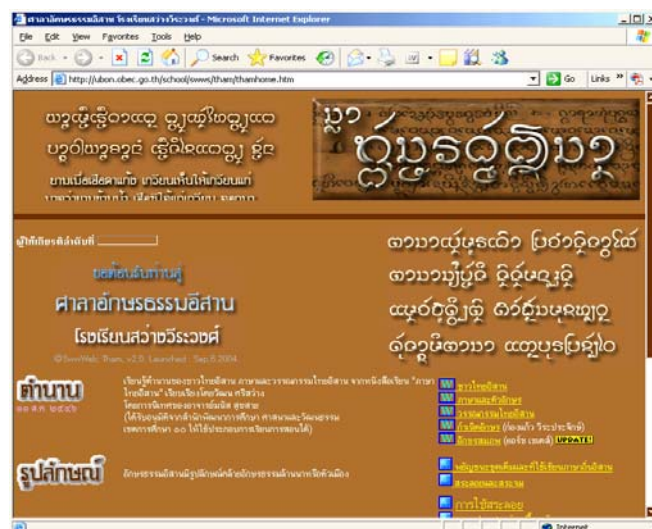
งานวิจัยนี้ เป็นการศึกษาถึงวิธีการเลือกสรรเอาคุณลักษณะเฉพาะของอักขระตัวอักษรธรรมอีสาน ที่คอมพิวเตอร์จะสามารถรู้จำได้อย่างแม่นยำและมีประสิทธิภาพ โดยใช้ตัวแบบฮิดเดนมาร์คอฟเป็นกลไกหลักในการรู้จำ สำหรับบทนี้จะกล่าวถึงข้อมูลรูปแบบตัวอักษรธรรมอีสานสำหรับงานพิมพ์บนเครื่องคอมพิวเตอร์ที่ได้มีผู้เชี่ยวชาญจัดทำสำเร็จไว้แล้ว และสามารถดาวน์โหลดมาใช้งานบนอินเทอร์เน็ตโดยไม่เสียค่าใช้จ่าย ที่นำมาใช้ในการวิจัย จากนั้นนำแบบตัวอักษรแต่ละแบบมาจัดทำเป็นรูปภาพตัวอักษรเพื่อนำไปเป็นข้อมูลนำเข้าสำหรับการประมวลผลเบื้องต้น กระบวนการเลือกสรรลักษณะเฉพาะของตัวอักขระ รวมไปถึงกระบวนการสร้างตัวแบบสำหรับตัวอักขระ

3.1 ข้อมูลแบบตัวอักษรธรรมอีสานที่ใช้ในการวิจัย

ข้อมูลแบบตัวอักษรธรรมอีสานที่ใช้ในการการวิจัยนั้น เป็นแบบตัวอักษรธรรมที่ได้จากโรงเรียนสว่างวีระวงศ์ มหาวิทยาลัยราชภัฏอุบลราชธานี มหาวิทยาลัยมหาสารคาม และมหาวิทยาลัยมหาจุฬาลงกรณราชวิทยาลัย วิทยาเขตขอนแก่น รวม 6 แบบ โดยแยกข้อมูลออกเป็น 1 ตัวอักษร ต่อ 1 ข้อมูล ซึ่งข้อมูลหนึ่งๆ จะมีลักษณะเป็นภาพใบนารีบีตแมปที่มีขนาด 150x200 pixel และในแบบตัวอักษรหนึ่งแบบจะมี 91 ข้อมูล รวมทั้งสิ้น 546 ข้อมูล

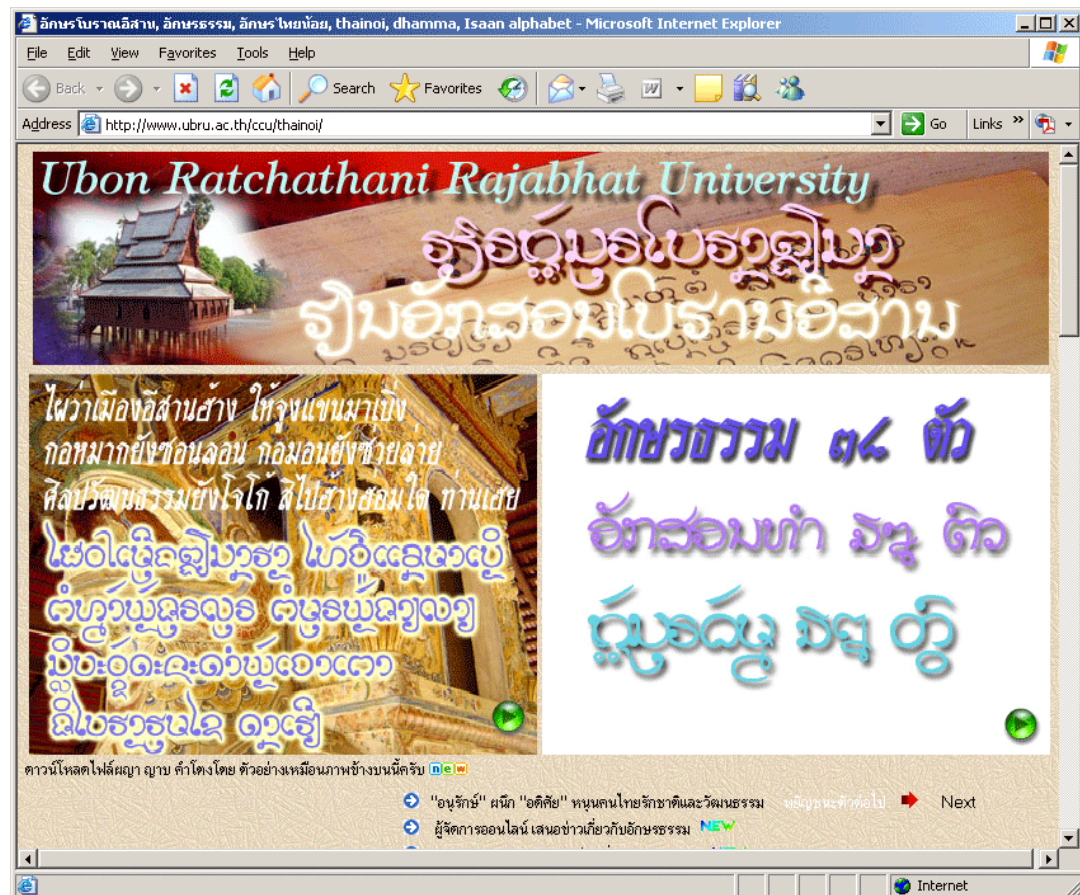
3.1.1 แบบอักษรธรรมจากโรงเรียนสว่างวีระวงศ์

โรงเรียนสว่างวีระวงศ์ ตั้งอยู่ที่ ตำบลสว่าง อำเภอสว่างวีระวงศ์ จังหวัดอุบลราชธานี มุ่งเน้นการเผยแพร่ความรู้ด้านอักษรธรรมอีสาน, อักษรไทน้อย และภาษาลาว ดังตัวอย่างหน้าเว็บในภาพที่ 3.1



ภาพที่ 3.1 แสดงตัวอย่างเว็บไซต์โรงเรียนสว่างวีระวงศ์ในส่วนของการเรียนอักษรธรรมอีสาน

UbWThawat (ดังภาพที่ 3.5) นั้นก็ได้ตั้งชื่อตาม ศ.วรัช ปุณโณทก เพื่อเป็นเกียรติแก่ท่านในฐานะผู้เชี่ยวชาญอักษรโบราณ

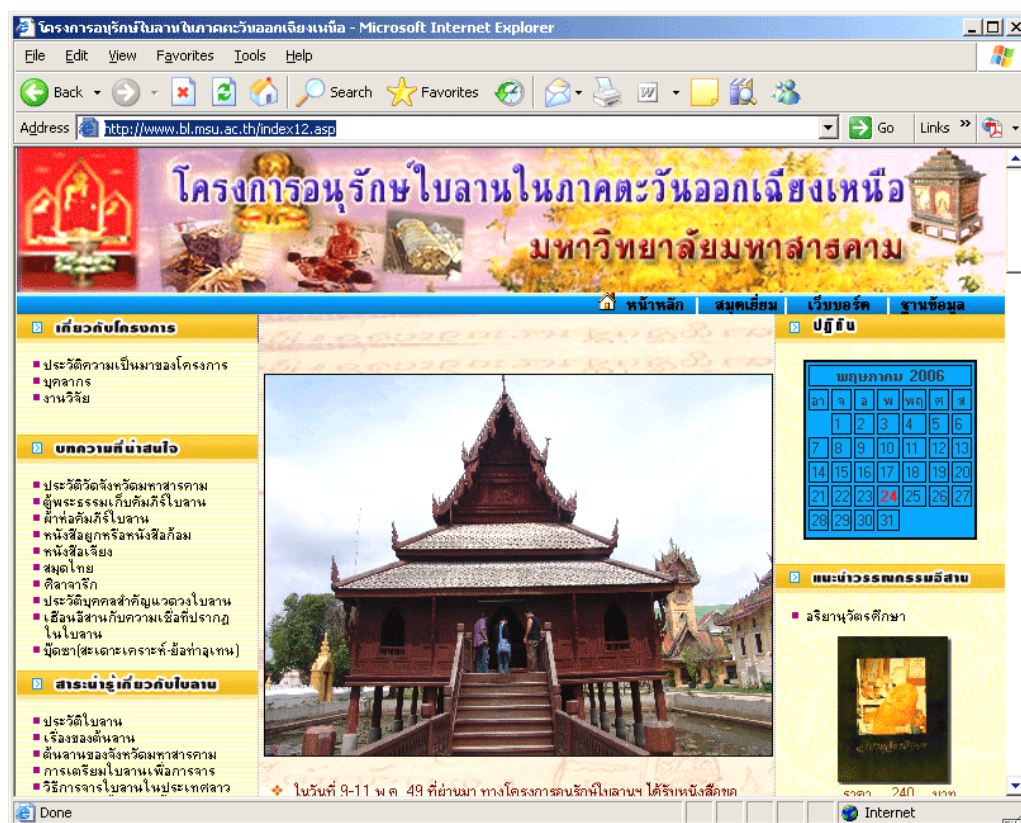


ภาพที่ 3.6 แสดงตัวอย่างเว็บไซต์เรียนอักษรโบราณอีสานของมหาวิทยาลัยราชภัฏอุบลราชธานี

3.1.3 แบบอักษรธรรมจากศูนย์อนุรักษ์โบราณในภาคตะวันออกเฉียงเหนือ มหาวิทยาลัยมหาสารคาม

ศูนย์อนุรักษ์โบราณในภาคตะวันออกเฉียงเหนือ เป็นศูนย์ที่ รองศาสตราจารย์ ดร.ภาวิช ทองโรจน์ อธิการบดีมหาวิทยาลัยมหาสารคามในขณะนั้น ให้จัดตั้งขึ้น ด้วยเล็งเห็นว่าควรมีการดำเนินการด้านการสำรวจ อนุรักษ์ และพัฒนาด้านเอกสารโบราณอย่างจริงจังก่อนที่จะข้อมูลทั้งหลายที่บันทึกไว้ในเอกสารโบราณจะเสื่อมสูญไปตามกาลเวลา เพราะเอกสารโบราณเป็นเอกสารชั้นต้นที่มีความสำคัญต่อวงวิชาการเป็นอย่างมาก เนื่องจากในอดีต การบันทึกสรรพวิชาการ คติความเชื่อ ขนบธรรมเนียม ประเพณี วิถีชีวิต และการสร้างสรรค์ของผู้คนในท้องถิ่น จะบันทึกไว้ในโบราณโดยใช้ตัวอักษรโบราณที่ใช้ในชุมชนนั้นเขียนบันทึกไว้ ซึ่งบรรดาข้อมูลที่อยู่ในเอกสารโบราณเหล่านี้สะท้อนให้เห็นถึงภูมิปัญญา วิถีชีวิตชุมชน ความศรัทธาความเชื่อของชุมชนท้องถิ่นซึ่งมีคุณค่าต่อการศึกษาย่างยิ่ง จึงเห็นว่าการจัดตั้งเป็นโครงการเฉพาะเพื่อดำเนินการด้านนี้อย่างเจาะลึก จึงได้ตั้งคณะกรรมการ โครงการขึ้น เมื่อวันที่ 28

พฤษภาคม 2546 โดยมีผู้ช่วยศาสตราจารย์วิภา วิสเพ็ญ เป็นประธานโครงการ และได้รับความร่วมมือเป็นเบื้องต้นจากอาจารย์ภาควิชาภาษาไทยและภาษาตะวันออก คณะมนุษยศาสตร์และสังคมศาสตร์ เป็นผู้ร่วมดำเนินงานในเบื้องต้น¹



ภาพที่ 3.7 แสดงตัวอย่างหน้าเว็บโครงการอนุรักษ์เอกสารโบราณในภาคตะวันออกเฉียงเหนือ

ในภาพที่ 3.7 แสดงหน้าเว็บของโครงการฯ (<http://www.bl.msu.ac.th/bailan/bailan.html>) ซึ่งเนื้อหาส่วนใหญ่ภายในเว็บจะเป็นการเก็บรวบรวมโบราณในภาคตะวันออกเฉียงเหนือ และได้มีการปริวรรต(แปลจากอักษรธรรมมาเป็นอักษรภาษาไทย)เนื้อหาที่อยู่บนโบราณที่ได้รวบรวมมาไว้ซึ่งได้แบ่งไว้เป็นหมวดหมู่ เช่น พุทธศาสนา, ลัทธิพิธีกรรม, วรรณกรรมและนิทานพื้นบ้าน, ฃญาสุภาษิต, โหราศาสตร์ และยาสมุนไพร นอกจากนี้จะมีส่วนให้ความรู้เกี่ยวกับอักษรธรรม และให้ผู้เข้าชมสามารถดาวน์โหลดแบบอักษรธรรมอีสาน ชื่อว่า Agsorntham.Tang โดยแบบอักษรธรรมอีสานนี้พัฒนาโดย นายสัญญา วุฒิสารนักวิชาการศึกษา ประจำโครงการ ฯ ดังตัวอย่างในภาพที่ 3.8

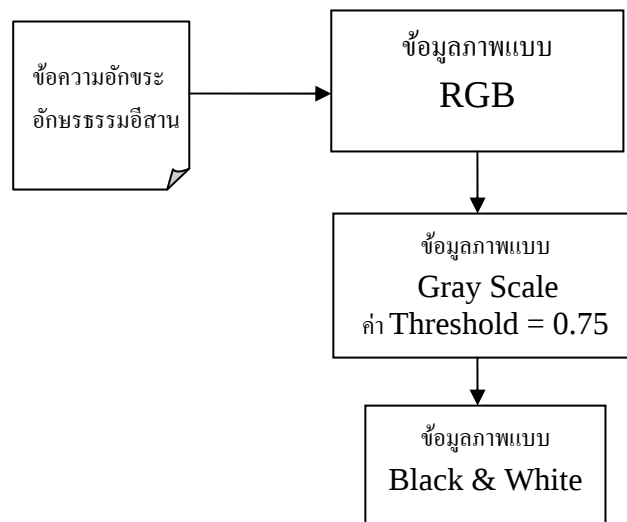
¹ <http://www.bl.msu.ac.th/pag2.asp>



ภาพที่ 3.8 แสดงตัวอย่างแบบอักษรธรรมอีสานชื่อ Agsorntham.Tang

3.1.4 แบบอักษรธรรมอีสานที่ผู้วิจัยพัฒนาขึ้นมาเอง

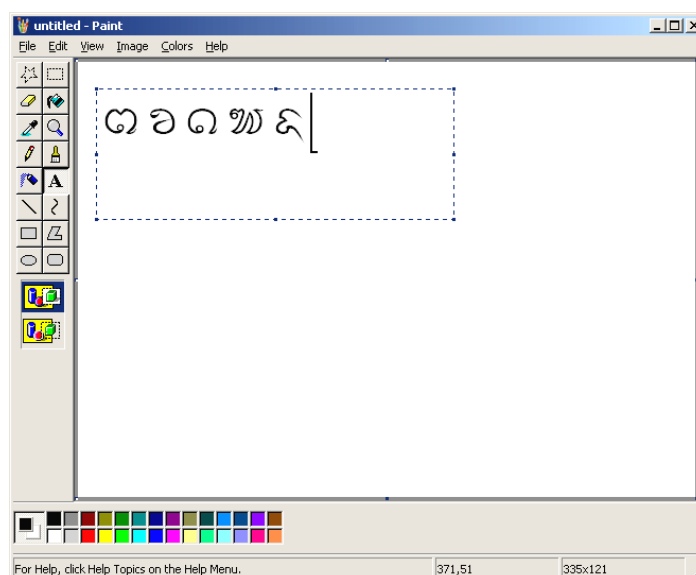
แบบอักษรธรรมอีสานชุดนี้ผู้วิจัยได้พัฒนาขึ้นมาเมื่อปลายปี พ.ศ. 2546 โดยได้รับคำแนะนำถึงรูปแบบการเขียนจาก พระพุทธิสารมณี (สุนันท์ สุภาจโร) รองเจ้าคณะจังหวัดขอนแก่น อดีตรองอธิการบดีวิทยาเขตขอนแก่น และเป็นที่ปรึกษาอธิการบดีวิทยาเขตขอนแก่นรูปปัจจุบัน ซึ่งแบบอักษรชุดนี้มีชื่อว่า AksornDharm2 ดังตัวอย่างในภาพที่ 3.9



ภาพที่ 3.10 แสดงขั้นตอนการแปลงข้อมูล

3.2.1 แปลงข้อมูล Text เป็นข้อมูล Image

เนื่องจากงานวิจัยนี้ต้องการศึกษาถึงกระบวนการรู้จำแบบอักษรธรรมอีสานของเครื่องคอมพิวเตอร์ โดยแบบอักษรธรรมอีสานจะอยู่ในรูปแบบของรูปภาพ แล้วส่งให้คอมพิวเตอร์อ่านและจดจำ ดังนั้น ข้อมูลแบบอักษรต่างๆ ที่รวบรวมมาได้นั้น จะต้องถูกแปลงข้อมูลจากข้อความ(Text) มาเป็นรูปภาพ(Image) โดยวิธีการแปลงนั้นจะใช้โปรแกรม Paint ซึ่งเป็นโปรแกรมสำหรับวาดภาพ ดังตัวอย่างในภาพที่ 3.11 พิมพ์อักขระอักษรธรรมอีสานลงไปแล้วทำการบันทึกเป็นแฟ้มข้อมูล Image ที่มีนามสกุลเป็น .bmp

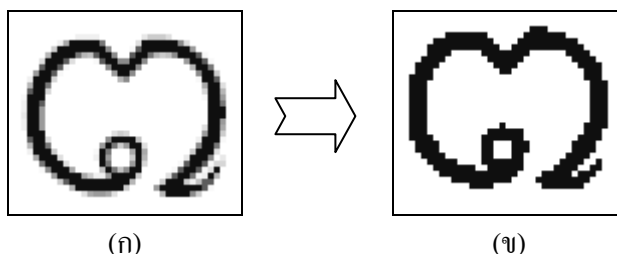


ภาพที่ 3.11 แสดงตัวอย่างการพิมพ์อักขระด้วยโปรแกรม Paint

ในภาพที่ 3.11 เป็นการพิมพ์ข้อความอักขระของอักษรธรรมอีสาน แล้วบันทึกเป็นแฟ้มรูปภาพ โดยแฟ้มรูปภาพที่ได้นี้จะมีความละเอียดเป็น Gray Scale ที่มีความละเอียดของสีเป็น 256 ระดับ ซึ่งถ้าหากจะนำไปเข้าสู่กระบวนการลดความหนาของตัวอักษรต้องเปลี่ยนเป็นแฟ้มรูปภาพแบบ ขาวและดำ (Black and White) ดังนั้น เมื่อแปลงข้อมูลข้อความเป็นข้อมูลรูปภาพเสร็จแล้ว ขั้นตอนต่อไปจะนำเอาข้อมูลรูปภาพเหล่านั้นแปลงไปเป็นข้อมูลภาพขาวและดำ

3.2.2 แปลงข้อมูลรูปภาพเป็นภาพขาวดำ

เมื่อได้ข้อมูลรูปภาพอักขระอักษรธรรมอีสานแล้ว ก่อนนำเข้าสู่กระบวนการลดความหนาของตัวอักษร จะต้องแปลงข้อมูลรูปภาพให้เป็นข้อมูลภาพแบบขาวดำเสียก่อนด้วยโปรแกรม MatLab² โดยกำหนดค่า Threshold เท่ากับ 0.75 จะได้ผลลัพธ์ดังภาพที่ 3.11



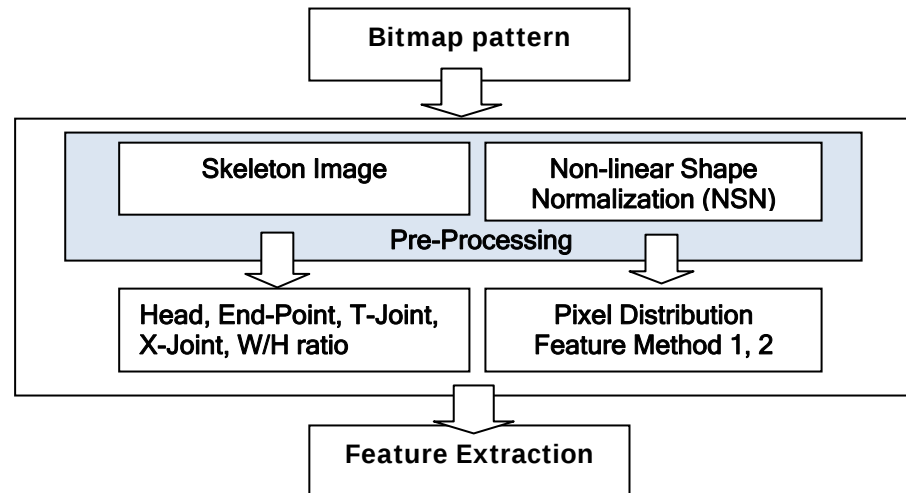
ภาพที่ 3.12 แสดงภาพอักขระอักษรธรรมอีสาน (ก) ก่อนแปลง และ (ข) หลังแปลง

จากภาพที่ 3.12 จะเห็นความแตกต่างระหว่างข้อมูลภาพอักขระอักษรธรรมอีสานแบบ Gray Scale (ภาพ ก) และภาพอักขระอักษรธรรมอีสานแบบ Black & White (ภาพ ข) ซึ่งขอบของภาพนั้นจะมีความชัดเจนต่างกัน จากนั้นทำการบันทึกข้อมูลภาพนั้นเป็น Bitmap ที่ความละเอียด 16 bit เพื่อเข้าสู่กระบวนการลดความหนาของตัวอักษรต่อไป

3.3 การประมวลผลเบื้องต้น (Pre-processing)

ก่อนที่จะทำการสกัดลักษณะเฉพาะ (Feature Extraction) ของตัวอักขระอักษรธรรมอีสาน แต่ละตัว จะต้องนำตัวอักษรธรรมอีสานมาผ่านกระบวนการประมวลผลเบื้องต้นก่อน เพื่อให้ข้อมูลมีความเหมาะสมกับกระบวนการในการสกัดลักษณะเฉพาะ ซึ่งมี 2 กลุ่มใหญ่ ๆ คือ กลุ่มที่ต้องทำการลดความหนาของเส้น และกลุ่มที่ต้องผ่านการปรับมาตรฐานรูปร่างแบบไม่เชิงเส้นโดยวิธีปรับการกระจายความหนาแน่นของจุด (ดูภาพที่ 3.13) ซึ่งการประมวลผลเบื้องต้น มีดังนี้

1 <http://www.mathworks.com/>



ภาพที่ 3.13 ขั้นตอนกระบวนการเตรียมข้อมูล

3.3.1 การลดความหนาของตัวอักษร

การลดความหนาของตัวอักษรอักษรธรรมอีสานที่ใช้ในงานวิจัยนี้ ได้พัฒนามาจากการใช้กฎและตัวแบบเปรียบเทียบ (Template) โดยอาศัยเทคนิคการค้นหาขอบภาพ (Edge detection) เพื่อค้นหาและกำจัดจุดที่ไม่ต้องการออกไป ดังนี้

(1) กฎการตรวจหาขอบภาพ (Edge detection rules)

ในการตรวจหาขอบภาพ จะใช้กฎทั้งหมด 14 กฎ โดยกฎที่ 1-8 จะมีลักษณะดังรูปที่ 3.14 ใช้ในการตรวจหาขอบของภาพ และกฎที่ 9-14 จะมีลักษณะดังภาพที่ 3.15 ใช้ในการกำจัดส่วนเกินที่ไม่ต้องการที่ระบุไว้ตามข้อกำหนด ในการประมวลผลจะใช้ภาพในการพิจารณา 2 ชุดนั่นคือภาพต้นฉบับ (Original image) และภาพที่กำลังประมวลผล (Current image) ซึ่งภาพที่กำลังประมวลผลจะเป็นภาพที่มีการอัปเดตเมื่อส่วนของ neighborhood สามารถเปรียบเทียบได้ตรงกับกฎในแต่ละรอบของการพิจารณา โดยจุดแต่ละจุดที่ตรงกับกฎจะเป็นจุดที่เป็นขอบของอักขระ (Edge Point) หรือเป็นจุดที่ถูกพิจารณาว่าเป็นส่วนเกินซึ่งจะต้องถูกกำจัดออกไป

0	0	X
0	1	1
X	1	X
กฎที่ 1		
X	0	0
1	1	0
X	1	X
กฎที่ 2		
X	1	X
1	1	0
X	0	0
กฎที่ 3		
X	1	X
0	0	0
X	1	X
กฎที่ 4		
0	0	0
X	1	X
1	1	0
X	X	0
กฎที่ 5		
0	0	0
0	1	1
0	0	0
กฎที่ 6		
X	1	1
X	1	X
0	0	0
กฎที่ 7		
0	X	X
0	1	1
0	X	1
กฎที่ 8		

ภาพที่ 3.14 แสดงกฎการตรวจหาขอบภาพ

ในแต่ละกฎจะมีข้อมูลอยู่ 4 ชนิด คือ

‘0’ หมายถึงจุดที่เป็นสีขาว

‘1’ หมายถึงจุดที่เป็นสีดำ

‘X’ หมายถึงจุดที่เป็นสีขาวหรือสีดำ (Don’t care)

‘E’ หมายถึงจุดที่ถูกพิจารณาว่าเป็นขอบภาพ (Edge Point) และเป็นจุดที่จะต้องถูกลบออกใน รอบถัดไป

1	1	E
E	1	0
0	0	0
กฎที่ 9		
1	1	E
E	1	E
0	0	0
กฎที่ 10		
E	E	0
1	1	0
1	E	0
กฎที่ 11		
1	1	0
E	1	0
0	0	0
กฎที่ 12		
0	E	1
0	1	E
0	0	0
กฎที่ 13		
0	E	1
0	1	1
0	0	0
กฎที่ 14		

ภาพที่ 3.15 แสดงกฎการจัดส่วนเกิน

(2) ขั้นตอนวิธีการประมวลผล

การประมวลผลจะกระทำเป็นรอบ (Iteration) โดยแต่ละรอบจะทำการเปรียบเทียบ neighborhood ของจุดที่เป็น สีดำกับกฎ 14 กฎ โดยกระทำกับทุก ๆ จุดของภาพ จากนั้นจึงจะทำการลบ

ขอบภาพเฉพาะเมื่อประมวลผลที่รอบคู่เท่านั้น จนกระทั่งไม่มี neighborhood ของจุดใดตรงกับกฎทั้งหมดที่กำหนดได้ไว้ ซึ่งมีกระบวนการในการประมวลผลดังนี้

Algorithms EdgeDetection (BinaryImage)

Input : An $m \times n$ element array of binary image

Output : An $m \times n$ element array of skeletons image

```

for all black pixels
  if neighborhood is according to 14 rules
  then mark pixel is 'E' (edge)
  end
  if loop mod 2 = 0 then
    for all 'E' pixel
      remove 'E' pixel
    end
  end
end
return BinaryImage
  
```

(3) คำอธิบายกฎ

พิจารณาภาพที่ 3.16 ซึ่งใช้เป็นทิศทางและหลักการในการกำหนดกฎการตรวจหาขอบภาพ โดยที่ทุกกฎจะพิจารณาว่าจุดที่เป็นขอบภาพจะต้องไม่ใช่ส่วนที่มีความหนาเพียง 1 จุด จากนั้นจึงพิจารณาทิศทางในการตรวจหาขอบดังนี้

กฎที่ 1 ใช้ในการตรวจหาขอบที่ 135 องศา ดังภาพที่ 3.17 (ก)

กฎที่ 2 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 45 องศา ดังภาพที่ 3.17 (ข)

กฎที่ 3 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 315 องศา ดังภาพที่ 3.17 (ค)

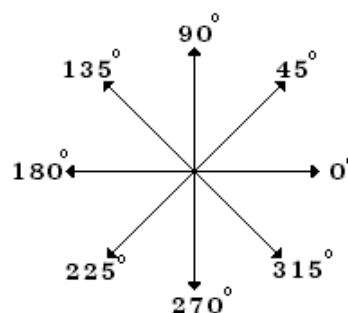
กฎที่ 4 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 225 องศา ดังภาพที่ 3.17 (ง)

กฎที่ 5 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 90 องศา ดังภาพที่ 3.17 (จ)

กฎที่ 6 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 0 องศา ดังภาพที่ 3.17 (ฉ)

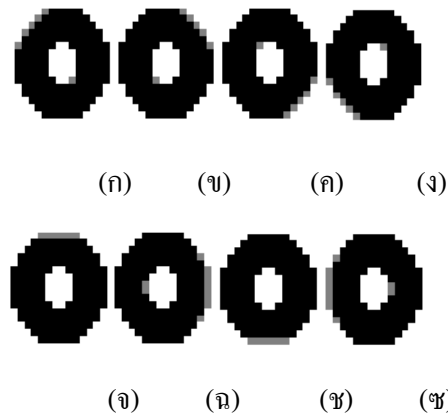
กฎที่ 7 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 270 องศา ดังภาพที่ 3.17 (ซ)

กฎที่ 8 ใช้ในการตรวจหาขอบที่ตำแหน่งทิศทาง 180 องศา ดังภาพที่ 3.17 (ช)



ภาพที่ 3.16 แสดงทิศทางของการพิจารณากำหนดกฎ

โดยในภาพที่ 3.17 และภาพที่ใช้แสดงผลจากการใช้กฎอื่นๆ จะแสดงตัวอย่างในลักษณะของภาพ Gray Scale ทั้งนี้ เนื่องจากการแสดงให้เห็นว่ากฎแต่ละกฎมีผลกับภาพต้นฉบับอย่างไร เพราะถ้าใช้ลักษณะของ Binary image จะไม่สามารถสังเกตได้เนื่องจากส่วนที่ตรงกับกฎจะกลายเป็นจุดสีขาว



ภาพที่ 3.17 ผลลัพธ์จากกฎการตรวจหาขอบภาพ

นอกจากนี้ ยังมีกฎที่ใช้ในการกำจัดส่วนเกินในบริเวณที่กฎการตรวจหาขอบไม่สามารถตรวจหาได้ ซึ่งจะต้องถูกใช้หลังจากใช้กฎการตรวจหาขอบแล้วเท่านั้น โดยกฎการกำจัดส่วนเกินมีทั้งหมด 6 กฎดังภาพที่ 3.18 แต่ละกฎจะใช้ในการกำจัดจุดที่จะทำให้เกิดส่วนเกินในทิศทางดังนี้

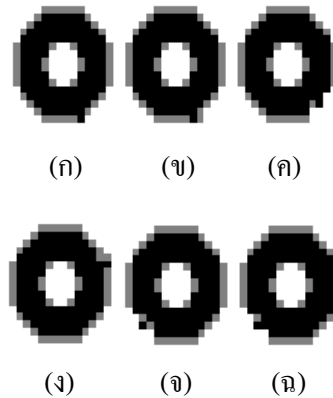
กฎที่ 9, 10 และ 11 ใช้กำจัดจุดที่จะทำให้เกิดเส้นส่วนเกินที่ตำแหน่งทิศทาง 270 องศา ดังภาพที่ 3.18 (ก-ค)

กฎที่ 12 ใช้กำจัดจุดที่จะทำให้เกิดเส้นส่วนเกินที่ตำแหน่งทิศทาง 0 องศา ดังภาพที่ 3.18 (ง)

กฎที่ 13 ใช้กำจัดจุดที่จะทำให้เกิดเส้นส่วนเกินที่ตำแหน่งทิศทาง 225 องศา ดังภาพที่ 3.18 (จ)

กฎที่ 14 ใช้กำจัดจุดที่จะทำให้เกิดเส้นส่วนเกินที่ตำแหน่งทิศทาง 180 องศา ดังภาพที่ 3.18 (ฉ)

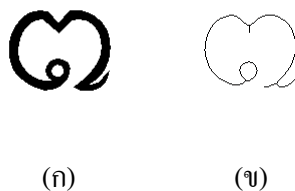
กฎที่ 9, 10 และ 11 เป็นบริเวณจุดที่จะทำให้เกิดส่วนเกินซึ่งอยู่ระหว่างจุดที่ถูกกำหนดว่าเป็นขอบ และกฎที่ 12, 13 และ 14 เป็นบริเวณจุดที่จะทำให้เกิดส่วนเกินซึ่งอยู่ระหว่างจุดที่ถูกกำหนดว่าเป็นขอบกับพื้นหลังของภาพ



ภาพที่ 3.18 แสดงส่วนเกินที่ต้องถูกกำจัด

ในขั้นตอนการลดความหนาของตัวอักษรโดยใช้กฎนี้ มีจุดเด่นของขั้นตอนวิธีที่นำเสนอ คือ
- ผลลัพธ์เป็นเส้นที่ต่อเนื่องกันและอยู่กึ่งกลางเส้นของอักขระ โดยมีความหนาของเส้นเท่ากับ 1 จุด เท่านั้น

- สามารถกำจัดส่วนเกินและเส้นที่ไม่ต้องการได้อย่างมีประสิทธิภาพ
- สามารถคงสัดส่วนโครงสร้างเดิมของอักขระตัวเขียนไว้ได้ ดังตัวอย่างในภาพที่ 3.19



ภาพที่ 3.19 แสดงภาพต้นฉบับ(ก) และผลลัพธ์ที่ได้ (ข)

เมื่อผ่านกระบวนการลดความหนา

(4) ผลการลดความหนาของอักขระแบบอักษรธรรมอีสาน

4.1) แบบอักษรธรรมจากโรงเรียนสว่างวีระวงศ์

กลุ่มตัวอย่างแรกที่ได้นำศึกษาวิจัยนี้ได้แก่ แบบอักษรธรรมจากโรงเรียนสว่างวีระวงศ์ ซึ่งมี 2 แบบ คือ แบบอักษร Tham2004 กับ Thamawatra ซึ่งผลที่ได้จากการนำเข้าสู่กระบวนการลดความหนาสามารถแสดงได้ดังตัวอย่างในภาพที่ 3.20 และภาพที่ 3.21

4.2) แบบอักษรธรรมจากเว็บเรียนอักษรโบราณอีสานของมหาวิทยาลัยราชภัฏอุบลราชธานี

กลุ่มตัวอย่างชุดที่สองนี้ เป็นแบบอักษรธรรมจากมหาวิทยาลัยราชภัฏอุบลราชธานี ซึ่งมี 2 แบบ เช่นเดียวกันคือ แบบอักษร UbWManut กับ UbWThawat และผลที่ได้จากการนำเข้าสู่กระบวนการลดความหนาสามารถแสดงได้ดังตัวอย่างในภาพที่ 3.22 และภาพที่ 3.23



ภาพที่ 3.22 แสดงแสดงผลการลดความหนาของอักขระแบบอักษรธรรม UbWManut

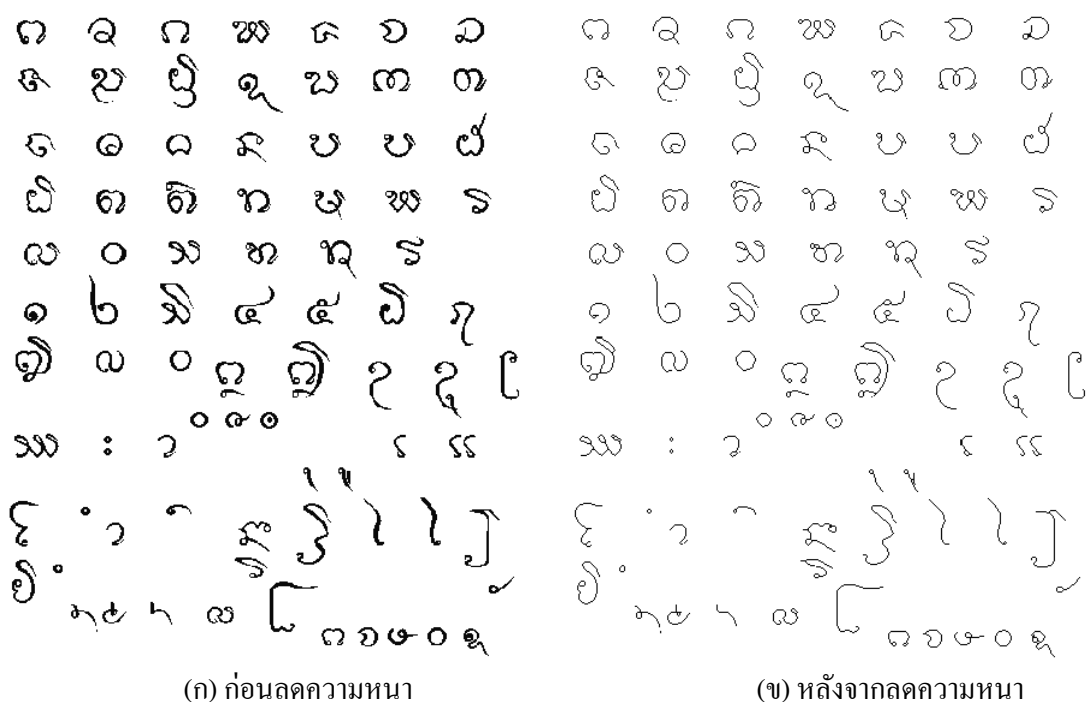


ภาพที่ 3.23 แสดงแสดงผลการลดความหนาของอักขระแบบอักษรธรรม UbWThawat

4.3) แบบอักษรธรรมจากโครงการอนุรักษ์โบราณในภาคตะวันออกเฉียงเหนือ มหาวิทยาลัย

มหาสารคาม

กลุ่มตัวอย่างชุดที่สองนี้ เป็นแบบอักษรธรรมจากมหาวิทยาลัยมหาสารคาม มี 1 แบบ คือ แบบอักษร Agsorntham.Tang และผลที่ได้จากการนำเข้าสู่กระบวนการลดความหนาสามารถแสดงได้ดังตัวอย่างในภาพที่ 3.24



ภาพที่ 3.24 แสดงแสดงผลการลดความหนาของอักขระแบบอักษรธรรม Agsorntham.Tang

4.3) แบบอักษรธรรมจากมหาวิทยาลัยมหาจุฬาลงกรณราชวิทยาลัย วิทยาเขตขอนแก่น

กลุ่มตัวอย่างชุดที่สองนี้ เป็นแบบอักษรธรรมที่ผู้วิจัยพัฒนาขึ้นมาเอง ฉะนั้น จึงขอกล่าวว่าเป็นแบบอักษรธรรมจากมหาวิทยาลัยมหาจุฬาลงกรณราชวิทยาลัย วิทยาเขตขอนแก่น มี 1 แบบ คือ แบบอักษร AksornDharm2 และผลที่ได้จากการนำเข้าสู่กระบวนการลดความหนาสามารถแสดงได้ดังตัวอย่างในภาพที่ 3.25



ภาพที่ 3.25 แสดงแสดงผลการลดความหนาของอักขระแบบอักษรธรรม AksornDharm2

3.3.2 การปรับมาตรฐานรูปร่างแบบไม่เชิงเส้น (Non-linear Shape Normalization : NSN)

ในกระบวนการเตรียมข้อมูลกลุ่มที่สองนี้ เป็นวิธีที่ใช้ในการปรับมาตรฐานของขนาดภาพ โดยจะใช้ลักษณะเด่นที่เป็น Pixel Distribution Features หากไม่มีกระบวนการในการเตรียมข้อมูลที่ดีแล้วจะทำให้ข้อมูลลักษณะเด่นของภาพที่ได้ในแต่ละโซนไม่มีคุณภาพ ซึ่งจะส่งผลกับประสิทธิภาพของการรู้จำด้วย ดังนั้นในขั้นตอนนี้จะใช้เทคนิคของการปรับการกระจายความหนาแน่นของจุด (Dot density equalization) (Lee et al., 1993) โดยขั้นตอนดังนี้

(1) **Feature Projection** : กำหนดให้ $f(i, j)$ เป็นภาพไบนารีบิตแมป ที่ $i = 1, 2, \dots, I$ และ $j = 1, 2, \dots, J$ โดย $H(i)$ และ $V(j)$ เป็น Projection ของจุดสีดำในแนวนอนและแนวตั้งตามลำดับ

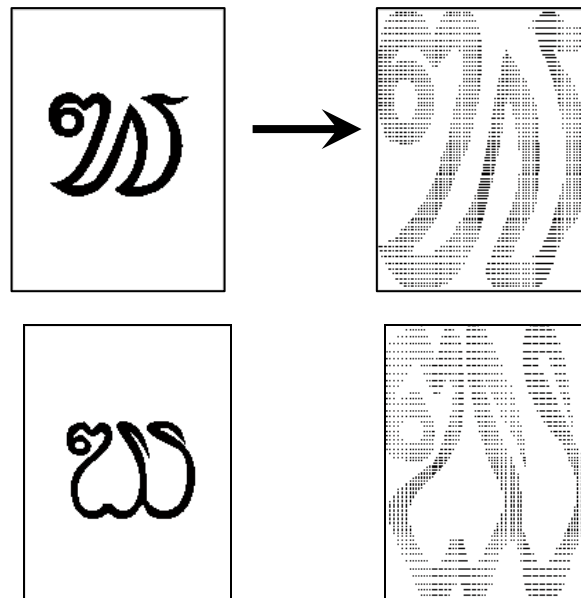
$$H(i) = \sum_{j'=1}^J f(i', j) \quad (12)$$

$$V(j) = \sum_{i'=1}^I f(i, j') \quad (13)$$

(2) **Feature density equalization** กำหนดให้ $G(m, n)$ เป็นภาพที่ถูกปรับขนาดแล้ว ที่ $m = 1, 2, \dots, M$ และ $n = 1, 2, \dots, N$ โดย m และ n เป็นตำแหน่งทางแนวนอนและแนวตั้งตามลำดับ ซึ่งตัวอย่างในการปรับขนาดแสดงดังภาพที่ 3.26

$$m = \left\lceil \sum_{k=1}^i H(k) \times \frac{M}{\sum_{k=1}^I H(k)} \right\rceil \quad (14)$$

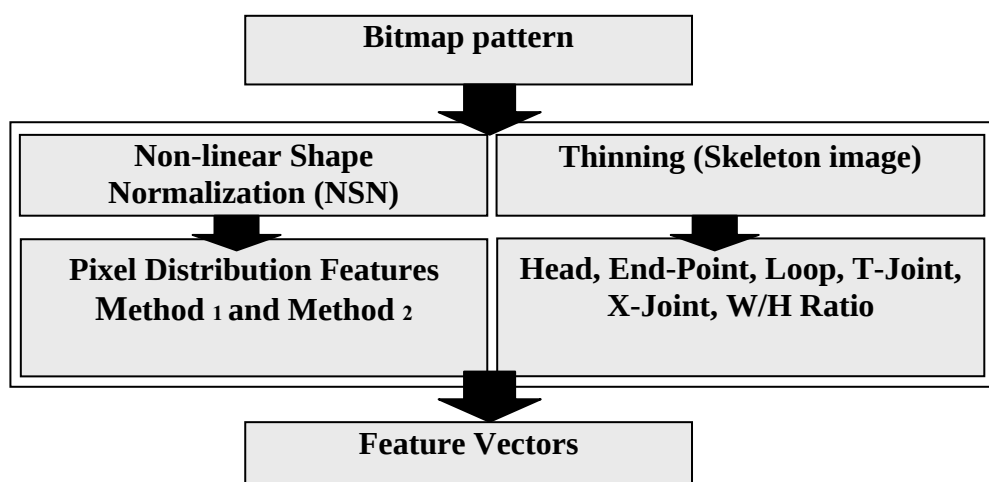
$$n = \left\lceil \sum_{l=1}^j V(l) \times \frac{N}{\sum_{l=1}^J V(l)} \right\rceil \quad (15)$$



ภาพที่ 3.26 ตัวอย่างการปรับการกระจายความหนาแน่นของจุดของอักขระ 'ฉ'

3.4 การสกัดลักษณะเฉพาะของอักขรธรรมอีสาน

การสกัดเอาลักษณะเฉพาะของแต่ละอักขระอินพุตออกมาเป็นเวกเตอร์เพื่อนำไปใช้เป็นอินพุตในการฝึกฝนและทดสอบระบบมีเทคนิคและขั้นตอนในการสกัดลักษณะเฉพาะดังภาพที่ 3.26 โดยแบ่งออกเป็น 2 ส่วนใหญ่ ๆ คือ ส่วนที่ใช้วิธีการปรับการกระจายความหนาแน่นของจุดและส่วนที่ต้องทำการลดความหนาของเส้น



ภาพที่ 3.26 ขั้นตอนการสกัดลักษณะเฉพาะ

ส่วนที่ 1 เป็นการสกัดลักษณะเฉพาะที่ใช้กับข้อมูลที่ผ่านการปรับมาตรฐานรูปร่างแบบไม่เชิงเส้น โดยวิธีปรับการกระจายความหนาแน่นของจุด ซึ่งประกอบด้วยวิธีการดังต่อไปนี้

3.4.1 Pixel Distribution Features วิธีที่ 1 โดยการแบ่งพื้นที่ของตัวอักษรออกเป็น 4x4 ส่วน เท่าๆ กันดังภาพที่ 3.27 แล้วหาจำนวนของจุดที่เป็นสีดำจากพื้นที่ย่อยนั้น ๆ และคำนวณดังนี้

$$\begin{aligned} \text{กำหนดให้ } z(n) &= \text{จำนวน black pixel ที่โซน } Z_n \\ fz(n) &= \text{Pixel Distribution Features ที่โซน } Z_n \\ &= z(n)/z_{\max} \\ z_{\max} &= \max[z(1), \dots, z(16)] \end{aligned}$$

Z1	Z2	Z3	Z4
Z5	Z6	Z7	Z8
Z9	Z10	Z11	Z12
Z13	Z14	Z15	Z16

ภาพที่ 3.27 การแบ่งโซนของภาพ

จากผลลัพธ์ที่ได้จากการคำนวณหาค่า $fz(n)$ เมื่อนำมาแบ่งส่วนค่าของผลลัพธ์ออกเป็น 40 ค่า และทำการกำหนดค่าลักษณะเฉพาะในแต่ละส่วนจะได้ดังนี้

```

if fz(n) <= 0.025   feature=32;
else if fz(n) <= 0.05   feature=33;
else if fz(n) <= 0.075   feature=34;
else if fz(n) <= 0.1   feature=35;
else if fz(n) <= 0.125   feature=36;
else if fz(n) <= 0.15   feature=37;
else if fz(n) <= 0.175   feature=38;
else if fz(n) <= 0.2   feature=39;
else if fz(n) <= 0.225   feature=40;
else if fz(n) <= 0.25   feature=41;
else if fz(n) <= 0.275   feature=42;
else if fz(n) <= 0.3   feature=43;
else if fz(n) <= 0.325   feature=44;
else if fz(n) <= 0.35   feature=45;

```

```

else if fz(n)<=0.375 feature=46;
else if fz(n)<=0.4 feature=47;
else if fz(n)<=0.425 feature=48;
else if fz(n)<=0.45 feature=49;
else if fz(n)<=0.475 feature=50;
else if fz(n)<=0.5 feature=51;
else if fz(n)<=0.525 feature=52;
else if fz(n)<=0.55 feature=53;
else if fz(n)<=0.575 feature=54;
else if fz(n)<=0.6 feature=55;
else if fz(n)<=0.625 feature=56;
else if fz(n)<=0.65 feature=57;
else if fz(n)<=0.675 feature=58;
else if fz(n)<=0.7 feature=59;
else if fz(n)<=0.725 feature=60;
else if fz(n)<=0.75 feature=61;
else if fz(n)<=0.775 feature=62;
else if fz(n)<=0.8 feature=63;
else if fz(n)<=0.825 feature=64;
else if fz(n)<=0.85 feature=65;
else if fz(n)<=0.875 feature=66;
else if fz(n)<=0.9 feature=67;
else if fz(n)<=0.925 feature=68;
else if fz(n)<=0.95 feature=69;
else if fz(n)<=0.975 feature=70;
else feature=71; end

```

3.4.2 Pixel Distribution Features วิธีที่ 2

กำหนดให้ $z(n)$ = จำนวน black pixel ที่โซน Zn

$$f1' = z(1) + z(2) + z(5) + z(6)$$

$$f2' = z(3) + z(4) + z(7) + z(8)$$

$$f3' = z(9) + z(10) + z(13) + z(14)$$

$$f4' = z(11) + z(12) + z(15) + z(16)$$

$f_{\max} = \max(f_1', f_2', f_3', f_4')$

$f_1 = f_1'/f_{\max}, f_2 = f_2'/f_{\max}$

$f_3 = f_3'/f_{\max}, f_4 = f_4'/f_{\max}$

ซึ่งจากผลลัพธ์ที่ได้จากการคำนวณได้ทำการกำหนดค่าลักษณะเฉพาะดังนี้

if $f_n \leq 0.25$ then $feature = 26$;

else if $f_n \leq 0.5$ then $feature = 27$;

else if $f_n \leq 0.75$ then $feature = 28$;

else $feature = 29$;

end

ส่วนที่ 2 คือ การสกัดลักษณะเฉพาะที่ใช้กับข้อมูลที่ผ่านการลดความหนาของเส้นเพื่อให้มีความหนาเพียง 1 จุดแล้ว ประกอบด้วยวิธีการดังต่อไปนี้

3.4.3 หัวของอักขระ คือ ส่วนที่มีลักษณะเป็นรูปปิด ซึ่งพิจารณาจากจำนวนของหัวอักขระที่ทำการตรวจ สอบได้โดยการประมวลผลในลักษณะการเดินท่องตามเส้นของอักขระ (Line-Tracking) โดยในขั้นตอนแรกจะต้องทำการค้นหาจุดเริ่มต้นซึ่งเป็นจุดที่เป็นสีดำ (โดยไม่จำเป็นต้องเป็นจุดที่ผู้เขียนเริ่มเขียนจริง) จากนั้นจึงทำการเดินทางไปยังจุดต่อไปซึ่งเป็นจุดสีดำไปเรื่อย ๆ จนกระทั่งเจอทางแยกหรือหมดความยาวของเส้นของอักขระ เมื่อเจอทางแยกให้เก็บตำแหน่งต่าง ๆ ของทางแยกเอาไว้เพื่อตรวจสอบว่าเป็นจุดร่วมของหัวอักขระหรือไม่และใช้เป็นเส้นทางในการเดินทางต่อไป และถ้าการเดินทางจากจุดใดจุดหนึ่งหรือจุดที่เป็นทางแยกมาบรรจบกับจุดที่ได้เก็บค่าไว้นั้นหมายความว่า ณ จุดนั้นสามารถพิจารณาว่าเป็นรูปปิดหรือหัวของอักขระได้ ดังกระบวนการวิธีต่อไปนี้

Algorithms CharacterHeadDetection (BinaryImage)

Input : An $m \times n$ element array of binary image

Output : Number of character head

for all pixels

if pixel = black pixel **then** startPoint = position(pixel); break; **end**

end

for all black pixels

countPath = countBlackPixel(neighbor(startPoint))

keepPath = getAllBlackPosition(neighbor(startPoint))

if countPath != 0 **then** nextPoint = getPath(keepPath) **end**

if checkIdentical(nextPoint, keepPath) **then** numHead = numHead+1 **end**

startPoint = nextPoint

end

return *numHead*

ซึ่งจากผลลัพธ์ที่ได้จากการตรวจหาจำนวนหัวของอักขระได้ทำการกำหนดค่าลักษณะเฉพาะ
ดังนี้

```
if numHead = 0 then feature = 6;
  else if numHead = 1 then feature = 7;
  else if numHead = 2 then feature = 8;
  else feature = 9;
end
```

3.4.4 หัวอักขระในตำแหน่งโซนของอักขระ โดยพิจารณาว่าอยู่ในตำแหน่งใดในกลุ่มโซนต่าง ๆ
ดังภาพที่

3.27 ดังนี้

[Z1,Z2,Z3,Z4], [Z5,Z6,Z7,Z8], [Z9,Z10,Z11,Z12], [Z13,Z14,Z15,Z16],
[Z1,Z5,Z9,Z13], [Z2,Z6,Z10,Z14], [Z3,Z7,Z11,Z15], [Z4,Z8,Z12,Z16]

และจากผลลัพธ์ที่ได้จากการตรวจหาหัวอักขระในตำแหน่งโซนต่าง ๆ ได้ทำการกำหนดค่า
ลักษณะเฉพาะดังนี้

```
if head-Position = 0 then feature = 30;
  else feature = 31;
end
```

3.4.5 T-Joints พิจารณาจาก 3x3 neighborhoods ดังภาพที่ 3.28 โดย T-Joint คือจุดที่มี neighbor
เป็นสีดำจำนวน 3 จุด [11] ซึ่งจะพิจารณาจาก $\text{sum}(p[0], p[1], p[2], p[3], p[4], p[5], p[6], p[7]) = 3$ เมื่อ p
= 1

$$T\text{-Joints} = \begin{cases} 0 & ; 0 \text{ T-Joints} \\ 1 & ; 1 \text{ T-Joints} \\ 2 & ; 2 \text{ T-Joints} \\ 3 & ; \geq 3 \text{ T-Joints} \end{cases}$$

p[3]	p[2]	p[1]
p[4]	p	p[0]
p[5]	p[6]	p[7]

ภาพที่ 3.28 3x3 neighborhoods

ซึ่งจากผลลัพธ์ที่ได้จากการตรวจหาจำนวนของ T-Joints ได้ทำการกำหนดค่าลักษณะเฉพาะ
ดังนี้

```
if  $T\text{-Joints} = 0$  then  $feature = 18$ ;
  else if  $T\text{-Joints} = 1$  then  $feature = 19$ ;
  else if  $T\text{-Joints} = 2$  then  $feature = 20$ ;
  else  $feature = 21$ ;
end
```

3.4.6 X-Joints โดยพิจารณาจาก 3x3 neighborhoods ดังภาพที่ 3.28 โดย X-Joint คือจุดที่มี neighbor เป็นสีดำจำนวน 4 จุด [11] ซึ่งจะพิจารณาจาก $\text{sum}(p[0], p[1], p[2], p[3], p[4], p[5], p[6], p[7]) = 4$ เมื่อ $p = 1$

$$X\text{-Joints} = \begin{cases} 0 & ; 0 \text{ X-Joints} \\ 1 & ; 1 \text{ X-Joints} \\ 2 & ; 2 \text{ X-Joints} \\ 3 & ; \geq 3 \text{ X-Joints} \end{cases}$$

ซึ่งจากผลลัพธ์ที่ได้จากการตรวจหาจำนวนของ X-Joints ได้ทำการกำหนดค่าลักษณะเฉพาะ
ดังนี้

```
if  $X\text{-Joints} = 0$  then  $feature = 22$ ;
  else if  $X\text{-Joints} = 1$  then  $feature = 23$ ;
  else if  $X\text{-Joints} = 2$  then  $feature = 24$ ;
  else  $feature = 25$ ;
end
```

3.4.7 End-Points พิจารณาจาก 3x3 neighborhoods ดังภาพที่ 3.28 โดย End-Point คือ จุดที่มี neighbor เป็นสีดำจำนวน 1 จุด ซึ่งจะพิจารณาจาก $\text{sum}(p[0], p[1], p[2], p[3], p[4], p[5], p[6], p[7]) = 1$ เมื่อ $p = 1$

$$\text{End-Points} = \begin{cases} 0 & ; 0 \text{ End-Points} \\ 1 & ; 1 \text{ End-Points} \\ 2 & ; 2 \text{ End-Points} \\ 3 & ; 3 \text{ End-Points} \\ 4 & ; \geq 4 \text{ End-Points} \end{cases}$$

ซึ่งจากผลลัพธ์ที่ได้จากการตรวจหาจำนวนของ End-Points ได้ทำการกำหนดค่า
ลักษณะเฉพาะดังนี้

```

if End-Points = 0 then feature = 10;
  else if End-Points = 1 then feature = 11;
  else if End-Points = 2 then feature = 12;
  else if End-Points = 3 then feature = 13;
  else feature = 14;
end

```

3.4.8 ตำแหน่งของ End-Points โดยพิจารณาว่าอยู่ในตำแหน่งใดในกลุ่มโซนต่าง ๆ ดังภาพที่ 3.27 ดังนี้

[Z1,Z2,Z3,Z4], [Z5,Z6,Z7,Z8], [Z9,Z10,Z11,Z12], [Z13,Z14,Z15,Z16],
 [Z1,Z5,Z9,Z13], [Z2,Z6,Z10,Z14], [Z3,Z7,Z11,Z15], [Z4,Z8,Z12,Z16]

ซึ่งจากผลลัพธ์ที่ได้จากการตรวจหาตำแหน่งของ End-Points ได้ทำการกำหนดค่าลักษณะเฉพาะดังนี้

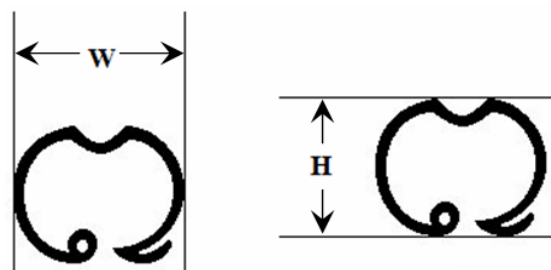
```

if End-Points-Position = 1 then feature = 1;
  else if End-Points-Position = 2 then feature = 2;
  else if End-Points-Position = 3 then feature = 3;
  else if End-Points-Position = 4 then feature = 4;
  else feature = 5;
end

```

3.4.9 สัดส่วนระหว่างความยาว (Width : W) และความสูง (Height : H) โดยพิจารณาจากขอบของตัวอักขระในแนวตั้งและแนวนอนดังภาพที่ 3.29 เพื่อเปรียบเทียบสัดส่วนดังเงื่อนไขต่อไปนี้

$$\text{Feature W/H Ratio} = \begin{cases} 0 & ; \text{W/H Ratio} < 1 \\ 1 & ; \text{W/H Ratio} = 1 \\ 2 & ; \text{W/H Ratio} > 1 \end{cases}$$



ภาพที่ 3.29 ลักษณะการวัดความสูงและความยาวของอักขระ

ซึ่งจากผลลัพธ์ที่ได้จากการคำนวณหาสัดส่วนระหว่างความยาวและความสูง ได้ทำการกำหนดค่าลักษณะเฉพาะดังนี้

```
if WHRatio < 1 then feature = 15;
else if WHRatio = 1 then feature = 16;
else feature = 17;
end
```

3.5. การสร้างตัวแบบ

ในการนำตัวแบบฮิดเดนมาร์คอฟไปใช้ในการรู้จำนั้นต้องทำการสอนตัวแบบเพื่อให้ตัวแบบรู้จักและสามารถจดจำตัวอักษรนั้นๆ ได้ และเนื่องจากตัวอักษรแต่ละตัวจะมีลักษณะเฉพาะไม่เหมือนกัน ดังนั้นจึงต้องสร้างตัวแบบฮิดเดนมาร์คอฟให้กับตัวอักษรแต่ละตัว โดยมีขั้นตอนดังต่อไปนี้

3.5.1 กำหนดค่าพารามิเตอร์เริ่มต้นให้กับตัวแบบฮิดเดนมาร์คอฟ

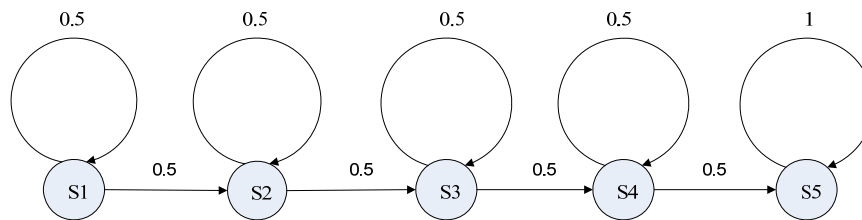
การกำหนดค่าเริ่มต้นต่าง ๆ ให้กับตัวแบบฮิดเดนมาร์คอฟ จะต้องกำหนดให้ตรงกับชนิดของตัวแบบที่ใช้ ในงานวิจัยนี้ได้เลือกใช้ตัวแบบฮิดเดนมาร์คอฟชนิด Left-Right เนื่องจากเป็นตัวแบบมีประสิทธิภาพสูงในการนำมาใช้กับกระบวนการรู้จำ ซึ่งวิธีการในการกำหนดค่าพารามิเตอร์เริ่มต้นให้กับตัวแบบ ทำได้ดังนี้

3.5.1.1 การกำหนดค่าความน่าจะเป็นของสถานะเริ่มต้น ($\pi = \{\pi_i\}$)

เนื่องจากงานวิจัยนี้ใช้ตัวแบบฮิดเดนมาร์คอฟชนิด Left-Right ดังนั้นต้องกำหนดให้เริ่มต้นที่สถานะแรกเสมอ โดยให้ $\pi_1 = 1$ ส่วนสถานะที่เหลือเป็น 0

3.5.1.2 การกำหนดค่าความน่าจะเป็นของการเปลี่ยนสถานะ ($A = \{a_{ij}\}$)

ค่าความน่าจะเป็นของการเปลี่ยนสถานะประกอบด้วย 3 ส่วน คือ ค่าความน่าจะเป็นของการอยู่ที่สถานะเดิม, ค่าความน่าจะเป็นของการเปลี่ยนสถานะไปยังสถานะถัดไป และค่าความน่าจะเป็นของการเปลี่ยนสถานะย้อนกลับ โดยในงานวิจัยนี้จะกำหนดค่าเริ่มต้นของความน่าจะเป็นของการเปลี่ยนสถานะให้มีค่าเฉลี่ยเท่ากันทั้งหมดแต่อยู่ภายใต้เงื่อนไขของผลรวมในแนวนอนต้องมีค่าเท่ากับ 1 เสมอตามทฤษฎีพื้นฐานของความน่าจะเป็นและตัวแบบฮิดเดนมาร์คอฟ โดย แสดงดังภาพที่ 3.30 และตัวอย่างในการกำหนดค่าเริ่มต้นของความน่าจะเป็นของการเปลี่ยนสถานะของตัวแบบฮิดเดนมาร์คอฟชนิด Left-Right ที่มีจำนวนสถานะเท่ากับ 5 สถานะ แสดงดังตารางที่ 3.1



ภาพที่ 3.30 ลักษณะของตัวแบบชนิด Left-Right

ตารางที่ 3.1 ค่าความน่าจะเป็นของการเปลี่ยนสถานะของตัวแบบชนิด Left-Right

สถานะ	1	2	3	4	5
1	0.5	0.5	0	0	0
2	0	0.5	0.5	0	0
3	0	0	0.5	0.5	0
4	0	0	0	0.5	0.5
5	0	0	0	0	1

3.5.1.3 การกำหนดค่าความน่าจะเป็นของสัญลักษณ์ (Symbol) ($B = \{b_j(k)\}$)

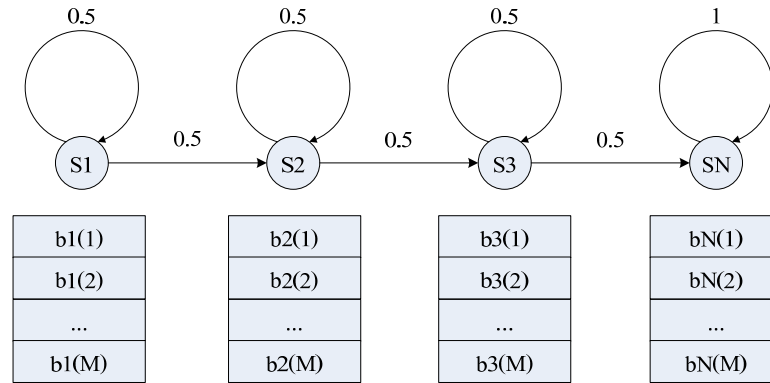
โดยโครงสร้างพื้นฐานของตัวแบบฮิดเดนมาร์คอฟ ในแต่ละสถานะจะต้องมีจำนวนของสัญลักษณ์เท่ากันทั้งหมดดังภาพที่ 3.31 โดยจำนวนสัญลักษณ์จะต้องสอดคล้องกับค่าของเวกเตอร์ลักษณะเฉพาะที่สกัดได้ในขั้นตอนของการสกัดลักษณะเฉพาะ ซึ่งในการวิจัยนี้ใช้จำนวนสัญลักษณ์ทั้งหมดเท่ากับ 71 สัญลักษณ์ สำหรับแต่ละสถานะของตัวแบบ โดยกำหนดค่าเริ่มต้นความน่าจะเป็นของสัญลักษณ์ให้มีค่ากระจายเท่ากันทั้งหมดที่ทุกสถานะ โดยคำนึงถึงผลรวมค่าความน่าจะเป็นของแต่ละสัญลักษณ์ในทุกสถานะซึ่งต้องมีผลรวมเท่ากับ 1 เสมอ ซึ่งคำนวณได้โดยสมการ

$$b = \frac{1}{N} \quad (16)$$

เมื่อ b = ค่าความน่าจะเป็นของสัญลักษณ์แต่ละสถานะ

N = จำนวนของสถานะทั้งหมด

ตัวอย่างเช่นในตัวแบบที่มีจำนวนของสถานะเท่ากับ 4 สถานะ เมื่อแทนค่าในสมการจะได้เท่ากับ $1/71$ ดังนั้นจะได้ว่าค่าเริ่มต้นความน่าจะเป็นของสัญลักษณ์ในแต่ละสถานะมีค่าเท่ากับ 0.014085



ภาพที่ 3.31 ลักษณะของสัญลักษณ์ในแต่ละสถานะของตัวแบบ

3.5.2 ค้นหาความน่าจะเป็นของลำดับการสังเกต(O) ต่อตัวแบบ

สามารถคำนวณได้อย่างมีประสิทธิภาพโดยใช้หลักการ Forward Algorithm หรือ Backward Algorithm ดังนี้

3.5.2.1 Forward Algorithm

กำหนดให้

$\alpha_t(i)$ เป็นตัวแปร Forward ซึ่งแสดงค่าความน่าจะเป็น ณ เวลา t ที่สถานะ i

$$\alpha_t(i) = P(O_1, O_2, \dots, O_t, q_t = S_i | \lambda)$$

π_i เป็นค่าความน่าจะเป็น ณ เวลา $t = 1$ ที่สถานะ i

$b_j(O_t)$ เป็นค่าความน่าจะเป็นของข้อมูล O ณ เวลา t ที่สถานะ j

โดยมีขั้นตอนในการคำนวณดังต่อไปนี้

Initialization

$$\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N \quad (17)$$

Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1}) \quad (18)$$

$$1 \leq t \leq T-1, \quad 1 \leq j \leq N$$

Termination

$$P(O|\lambda) = \sum_{i=1}^N \alpha_T(i) \quad (19)$$

3.5.2.2 Backward Algorithm

กำหนดให้

$\beta_t(i)$ เป็นตัวแปร Backward ซึ่งแสดงค่าความน่าจะเป็น ณ เวลา t ที่สถานะ i

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \dots, O_T | q_t = S_i, \lambda)$$

π_i เป็นค่าความน่าจะเป็น ณ เวลา $t = 1$ ที่สถานะ i

$b_j(O_t)$ เป็นค่าความน่าจะเป็นของข้อมูล O ณ เวลา t ที่สถานะ j

โดยมีขั้นตอนในการคำนวณดังต่อไปนี้

Initialization

$$\beta_T(i) = 1, \quad 1 \leq i \leq N \quad (20)$$

Induction

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \quad (21)$$

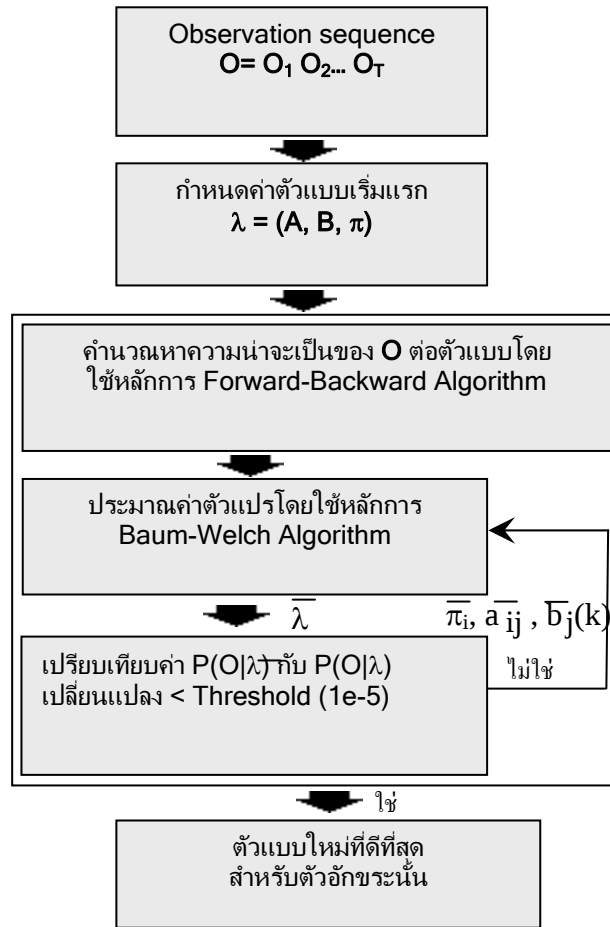
$$t = T-1, T-2, \dots, 1, \quad 1 \leq i \leq N$$

Termination

$$P(O|\lambda) = \sum_{i=1}^N \pi_i b_i(O_1) \beta_1(i) \quad (22)$$

3.5.3 การสอนตัวแบบเพื่อการเรียนรู้จำตัวอักษร

เนื่องจากตัวอักษรแต่ละตัวจะมีลักษณะเฉพาะไม่เหมือนกัน ดังนั้นในการรู้จำจึงต้องสร้างตัวแบบฮิดเดนมาร์คอฟให้กับตัวอักษรแต่ละตัว ซึ่งต้องกำหนดค่าพารามิเตอร์เริ่มต้นให้กับตัวแบบแต่ละตัวแบบ แล้วคำนวณหาความน่าจะเป็นของลำดับการสังเกต(O) ต่อตัวแบบ หลังจากนั้นจึงทำการประมาณค่าตัวแปรเพื่อหาพารามิเตอร์ใหม่ให้กับตัวแบบ ซึ่งจะเป็นการปรับค่าต่าง ๆ ของตัวแบบนั่นเอง โดยในการสอนตัวแบบจะทำการสอนตัวแบบจนกระทั่งได้ค่าความน่าจะเป็นคู่เข้าค่าใดค่าหนึ่ง



ภาพที่ 3.32 ขั้นตอนการสอนและประมาณค่าตัวแปรใหม่ให้กับตัวแบบ

ในการประมาณค่าตัวแปรเพื่อหาค่าพารามิเตอร์ใหม่ให้กับตัวแบบใช้หลักการของ Baum-Welch (Rabiner, 1989) ซึ่งจะนำผลลัพธ์ที่ได้จาก Forward-Backward Algorithm มาใช้ในการคำนวณ โดยมีกระบวนการวิธีในการประมาณค่าตัวแปรดังภาพที่ 3.32 และมีรายละเอียดของการคำนวณดังนี้

3.5.3.1 กำหนดให้ $\xi_t(i, j)$ คือ ความน่าจะเป็นของการอยู่ในสถานะ i ที่เวลา t และสถานะ j ที่เวลา $t+1$

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \quad (23)$$

3.5.3.2 กำหนดให้ $\gamma_t(i)$ เป็นความน่าจะเป็นของการอยู่ในสถานะ i ที่เวลา t ซึ่งเกิดจากลำดับการสังเกตและตัวแบบ

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j) \quad (24)$$

3.5.3.3 การประมาณค่าตัวแปรใหม่ (Reestimation) ของ A, B และ π คือ

$$\overline{\pi}_i = \gamma_1(i) \quad (25)$$

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad (26)$$

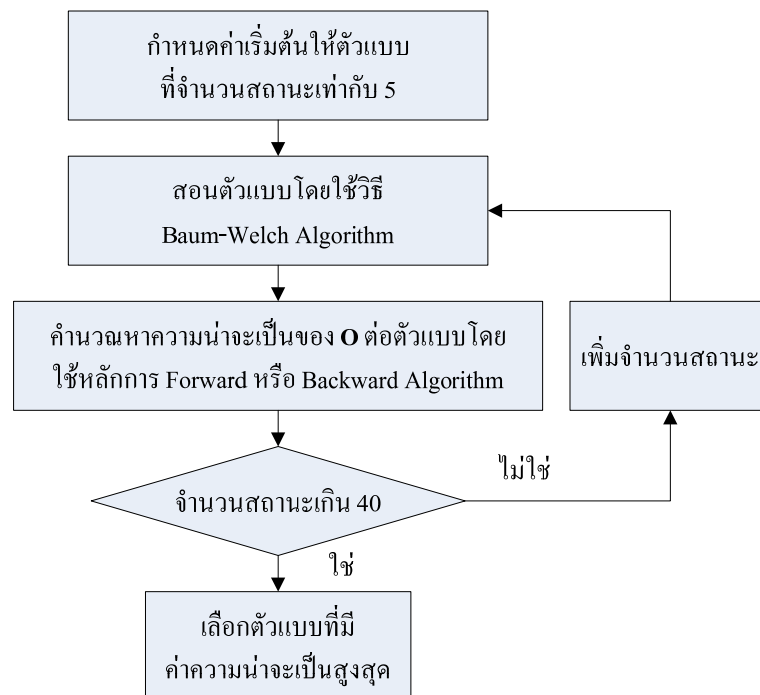
$$\overline{b}_j(k) = \frac{\sum_{t=1}^T \gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)} \quad (27)$$

3.6 การทดลองและผลการทดลอง

การทดสอบการรู้จำอักษรธรรมอีสานโดยใช้ตัวแบบฮิดเดนมาร์คอฟในงานวิจัยนี้ จะใช้ตัวแบบฮิดเดนมาร์คอฟชนิด Left-Right ซึ่งวิธีการสร้างตัวแบบในขั้นตอนการสอนตัวแบบจะไม่กำหนดจำนวนของสถานะที่แน่นอน ซึ่งมีขั้นตอนการทดลองดังต่อไปนี้

3.6.1 การรู้จำอักษรธรรมอีสานโดยใช้ตัวแบบฮิดเดนมาร์คอฟชนิด Left-Right ในการสร้างตัวแบบจะไม่กำหนดจำนวนสถานะที่แน่นอน ซึ่งจะใช้วิธีการค้นหาตัวแบบที่มีจำนวนสถานะที่เหมาะสมที่สุดสำหรับตัวอักขระแต่ละตัว โดยในขั้นตอนของการสอนตัวแบบจะกำหนดช่วงของจำนวนสถานะเท่ากับ 5 – 40 สถานะ เพื่อทดสอบและเปรียบเทียบค่าความน่าจะเป็นที่เกิดจากลำดับการสังเกต (O) กับตัวแบบนั้น ๆ ว่าตัวแบบที่มีจำนวนสถานะเท่าใดที่มีค่าความน่าจะเป็นสูงที่สุด และจะเลือกตัวแบบนั้นเป็นตัวแบบที่ดีที่สุดสำหรับตัวอักขระนั้น โดยมีขั้นตอนในการสร้างตัวแบบของทุกตัวอักขระดังภาพที่

3.33



ภาพที่ 3.33 ขั้นตอนสร้างตัวแบบของการทดลองที่ 2