การกำกับหมวดคำสำหรับข้อความภาษาไทย

โดย

บุญเสริม กิจศิริกุล

โครงการวิจัยเลขที่ 51G-COM-2540

ทุนงบประมาณแผ่นดิน

ปี 2540

สถาบันวิจัยและพัฒนาคณะวิศวกรรมศาสตร์
คณะวิศวกรรมศาสตร์
จุฬาลงกรณ์มหาวิทยาลัย
กรุงเทพฯ

พฤศจิกายน 2541

การกำกับหมวดคำสำหรับข้อความภาษาไทย

โดย

บุญเสริม กิจศิริกุล

โครงการวิจัยเลขที่51G-COM-2540

ทุนงบประมาณแผ่นดิน

ปี 2540

สถาบันวิจัยและพัฒนาคณะวิศวกรรมศาสตร์

คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

กรุงเทพฯ

พฤศจิกายน 2541

กิตติกรรมประกาศ

งานวิจัยนี้สำเร็จลุล่วงไปด้วยดีได้ เพราะได้รับการสนับสนุนทุนวิจัยจาก สำนักงานคณะกรรมการวิจัยแห่งชาติ ผู้วิจัยขอขอบพระคุณมา ณ โอกาสนี้ และขอขอบคุณฝ่ายวิจัย คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ที่อำนวยความสะดวกในเรื่องต่างๆ จนสิ้นสุดงานวิจัยนี้

บทคัดย่อ

รายงานนี้แสดงการประยุกต์ใช้วิธีการทางสถิติในการกำกับหมวดคำให้กับคำในคลังข้อความภาษาไทย เนื่องจากคำในภาษาไทยเขียนติดต่อกันโดยไม่มีเครื่องหมายแบ่งคำ ดังนั้นการศึกษาวิธีกำกับหมวดคำของคลังข้อความภาษาไทยจึงต้องศึกษาการตัดคำร่วมด้วย ในงานวิจัยนี้เราใช้แบบจำลองไตรแกรมทั้งในการตัดคำและการกำกับหมวดคำ ผลการทดลองแสดงให้เห็นว่าแบบจำลองไตรแกรมใช้ได้อย่างมีประสิทธิผลในการตัดคำและการกำกับหมวดคำด้วยเปอร์เซนต์ความถูกต้องที่สูง รายงานนี้ยังแสดงการใช้ประโยชน์ของคลังข้อความที่มีหมวดคำกำกับแล้วในงาน 2 ประเภทคือ การแก้ไขคำผิดที่เกิดจาก OCR ภาษาไทยและการระบุคำที่ไม่รู้จักในภาษาไทย ผลการทดลองในงานทั้งสองแสดงให้เห็นถึงความสำเร็จในการใช้คลังข้อความที่มีหมวดคำกำกับโดยให้ผลความถูกต้องที่สูงทั้งในการแก้ไขคำผิดของ OCR และการระบุคำที่ไม่รู้จัก

Abstract

This report shows the method of statistical technique for part-of-speech tagging of words in Thai corpus. As in Thai language, words are written consecutively without delimiters, the study of tagging of a Thai corpus has to incorporate of word segmentation. Here we approach the problems of word segmentation as well as part-of-speech tagging by using trigram model. Experimental results show that the trigram model effectively performs word segmentation and part-of-speech tagging with high accuracy. The report also demonstrates the use of the tagged corpus in two applications; i.e., Thai OCR error correction and Thai unknown word identification. The experiments on these applications show the successful use of the tagged corpus by obtaining high accuracy of OCR error correction and unknown word identification.

สารบัญ

รายการตารางประกอบ

หน้า

รายการภาพประกอบ

หน้า

1. บทนำ

การประมวลผลภาษาธรรมชาติ (Natural Language Processing : NLP) เป็นสาขาย่อยในปัญญาประดิษฐ์ ซึ่งศึกษาเกี่ยวกับการแจงส่วนประโยค (parsing sentences) การกำหนดความหมายของข้อความ (assigning semantic to sentences) และอื่นๆ โดยมีจุดมุ่งหมายที่จะทำให้คอมพิวเตอร์สามารถเข้าใจภาษา วิธีการประมวลผลภาษาธรรมชาติโดยมากจะใช้ความรู้ประเภทต่างๆเช่น ความรู้เกี่ยวกับไวยากรณ์ของภาษา (syntactic knowledge) ความรู้เกี่ยวกับความหมายของคำ (semantic knowledge) เป็นต้น ในขณะที่การรู้จำเสียง (speech recognition) ซึ่งเป็นเทคโนโลยีที่พยายามทำให้คอมพิวเตอร์รู้จำเสียงนั้น ส่วนมากจะเน้นถึงการใช้วิธีการทางสถิติเพื่อที่จะทำนายว่าเสียงที่พูดนั้นมีความน่าจะเป็นที่จะเป็นเสียงใดมากที่สุด โดยอาศัยข้อมูลที่มีอยู่ก่อน

ในปัจจุบัน งานวิจัยทาง NLP ได้ให้ความสนใจในการนำเทคนิคที่ใช้ในการรู้จำเสียงมาประยุกต์ใช้เข้ากับระบบการประมวลผลภาษาธรรมชาติ ได้มีการใช้วิธีทางสถิติเพื่อดึงข้อมูลต่างๆจากคลังข้อความ (corpus) ขนาดใหญ่ ซึ่งคลังข้อความนี้เป็นเครื่องมือสำคัญในการศึกษา NLP นักวิจัยจำนวนมากได้ศึกษาเกี่ยวกับการกำกับหมวดคำให้กับคำในคลังข้อความ เช่น ไก่ขันตอนเช้า จะถูกกำกับหมวดคำเป็น NCMN VACT ADVN (NCMN : common noun, VACT : action verb, ADVN : normal adverb) ซึ่งประโยชน์อย่างหนึ่งของคำที่ถูกกำกับหมวดคำคือช่วยให้การแจงส่วนประโยคเป็นไปได้ดีขึ้น ซึ่งแต่เดิมแล้วการแจงส่วนประโยคจะประสบปัญหาเพราะว่าคำหนึ่งๆมีได้หลายหมวดคำ ทำให้การประมวลผลเกิดความกำกวมส่งผลให้ประมวลผลไม่ถูกต้องและไม่ได้ประสิทธิภาพเท่าที่ควร เมื่อเราใช้วิธีทางสถิติเข้ามาช่วยเพื่อทำนายว่าคำนั้นๆมีความน่าจะเป็นที่จะมีหมวดคำเป็นอะไร ก็จะช่วยลดความกำกวมนี้ได้

ในต่างประเทศได้มีการรวบรวมสร้างคลังข้อความขึ้นมาเพื่อประโยชน์ในการประมวลผลภาษาธรรมชาติ เช่น Brown corpus (W.Francis & H.Kucera) (ประกอบด้วย 1 ล้านคำ) หรือ Penn TreeBank (M.P.Marcus et.al.,1993) (ประกอบด้วย 4.5 ล้านคำ) ปัจจุบันในประเทศได้มีผู้สนใจและศึกษาการประมวลผลภาษาไทยจำนวนมาก ผู้วิจัยจึงเห็นว่าควรที่จะทำคลังข้อความภาษาไทยขึ้นมาเพื่อใช้เป็นแหล่งข้อมูลในการประมวลผลภาษาไทย เท่าที่ผ่านมาได้มีการวิจัยและการตีพิมพ์เทคนิคต่างๆที่เกี่ยวกับวิธีการกำกับหมวดคำในภาษาอังกฤษ งานวิจัยที่สำคัญๆได้แก่ งานของ Church (K.W.Church, 1988) ซึ่งใช้แบบจำลองไตรแกรม (trigram model) ในการคำนวณความน่าจะเป็นของแต่ละหมวดคำเพื่อกำกับให้กับคำหนึ่งๆ หลังจากนั้น Charniak และคณะ (Charniak et.al, 1993) ได้ปรับปรุงแบบจำลองของ Church และพบว่าสามารถกำกับคำได้ถูกต้องขึ้น สำหรับงานวิจัยที่เกี่ยวข้องกับการกำกับหมวดคำในภาษาไทยนั้นมีงานวิจัยของ วิลาศ และอำไพ (V.Wuwongse & A.Pornprasertsakul, 1994; A.Kawtrakul et.al., 1995; A.Kawtrakul et.al., 1997) เนื่องจากภาษาไทยมีข้อแตกต่างจากภาษาอังกฤษ จึงไม่สามารถนำเทคนิคเหล่านั้นมาใช้

ได้โดยตรง ข้อแตกต่างอย่างหนึ่งก็คือภาษาไทยไม่มีการแสดงขอบเขตของคำและของประโยคอย่างชัดเจน ดังนั้นในการศึกษาการกำกับคำในภาษาไทยนั้นจึงจำเป็นต้องศึกษา วิธีการตัดคำโดยใช้คลังข้อความควบคู่ไปด้วย จุดมุ่งหมายของงานวิจัยนี้จึงเป็นการศึกษาเกี่ยวกับการกำกับหมวดคำและการตัดคำโดยใช้คลังข้อความภาษาไทยแบบอัตโนมัติ เพื่อจัดทำคลังข้อความภาษาไทยที่มีหมวดคำกำกับ

นอกจากนี้รายงานวิจัยนี้ยังแสดงให้เห็นถึงการนำคลังข้อความไปใช้ประโยชน์ในการประมวลผลภาษาธรรมชาติ ซึ่งได้แก่ (1) การแก้ไขคำผิดที่เกิดจาก OCR ภาษาไทย และ (2) การระบุคำที่ไม่รู้จัก เราพบว่าคลังข้อความที่มีหมวดคำกำกับแล้ว สามารถนำมาช่วยให้การแก้ไขคำผิดที่เกิดจาก OCR ทำได้ถูกต้องดียิ่งขึ้น และทำให้การระบุคำที่ไม่รู้จักทำได้อย่างมีประสิทธิภาพ

2. การตัดคำ

หลักการตัดคำในภาษาไทยสามารถแบ่งออกได้เป็น 2 หลักการใหญ่คือ หลักการตัดคำโดยใช้กฎ (สุรศักดิ์และคณะ 2528, สุรินทร์ 2526) และหลักการตัดคำโดยใช้พจนานุกรม (ยืนและวิวรรธน์ 2529, ดวงแก้ว 2533)

ในงานวิจัยนี้เราเลือกศึกษาการตัดคำโดยใช้พจนานุกรมร่วมกับการใช้ข้อมูลทางสถิติที่ได้จากคลังข้อความ เนื่องจากงานวิจัยที่ผ่านมาได้แสดงให้เห็นว่าการตัดคำโดยพจนานุกรมสามารถตัดคำได้ถูกต้องมากกว่า สำหรับหลักการตัดคำโดยใช้พจนานุกรมนี้สามารถแบ่งเป็นสองวิธีการใหญ่ๆคือ วิธีตัดคำให้ยาวที่สุด (ยืนและวิวรรธน์ 2529) และ วิธีตัดจำนวนคำให้น้อยที่สุด (วิรัช 2536)

2.1 วิธีตัดคำให้ยาวที่สุด

วิธีนี้จะสแกนประโยคจากซ้ายไปขวา เทียบกับพจนานุกรมเพื่อทำเครื่องหมายกับทุกสายอักขระที่สามารถสร้างเป็นหนึ่งคำให้เป็นจุดย้อนกลับ และเลือกสายอักขระที่ยาวที่สุดเป็นตัวเลือกสำหรับคำแรก ถ้าตัวเลือกนี้สามารถทำให้อัลกอริทึมค้นหาคำที่เหลือได้สมบูรณ์ ตัวเลือกนี้ก็จะเป็นคำแรกจริง ไม่เช่นนั้นอัลกอริทึมก็จะกลับไปยังจุดย้อยกลับที่ทำเครื่องหมายไว้เพื่อใช้เป็นตัวเลือกสำหรับคำแรกใหม่ และทำการค้นหาคำที่เหลือต่อไปเป็นเช่นนี้ไปเรื่อยๆ ถ้าเราให้ประโยค

ไม่มีผลต่ออุณหภูมิ

ผลที่ได้จะเป็น

ไม่มี ผล ต่อ อุณหภูมิ

ข้อด้อยหนึ่งของวิธีนี้คือโดยปกติอัลกอริทึมจะพยายามตัดส่วนต้นๆของประโยคให้ได้คำที่ยาวมากเกินไป และอาจทำให้ส่วนท้ายๆของประโยคประกอบด้วยสายอักขระสั้นๆจำนวนมาก ตัวอย่างเช่น

ไปหามเหสี

จะถูกตัดคำเป็น

ไป หาม เห สี

ผลที่ได้ไม่ถูกต้อง ผลที่ถูกซึ่งไม่สามารถค้นพบได้โดยอัลกอริทึมนี้คือ

ไป หา มเหสี

2.2 วิธีตัดคำให้จำนวนน้อยที่สุด

วิธีนี้จะแบ่งคำเป็นส่วนย่อยๆทั้งหมดที่เป็นไปได้แล้วเลือกการแบ่งคำที่ได้จำนวนคำน้อยที่สุด ตัวอย่างเช่น

ไปหามเหสี

จะถูกแบ่งคำเป็น

ไป หา มเหสี

แม้ว่าการค้นหาการแบ่งทุกทางที่เป็นไปได้อาจทำให้ต้องเสียค่าใช้จ่ายในการคำนวณมาก แต่ก็สามารถลดค่าใช้จ่ายได้โดยวิธีของการโปรแกรมแบบพลวัติ (dynamic programming) โดยทั่วไปวิธีนี้จะให้ความถูกต้องที่สูงกว่าวิธีตัดคำให้ยาวที่สุด ดังเช่นที่แสดงโดยตัวอย่างด้านบน แต่อย่างไรก็ตาม ก็มีประโยคจำนวนมากที่ไม่สามารถตัดได้ถูกต้องโดยวิธีนี้ เช่นประโยค

เทคโนโลยีทางการผลิต

ถูกตัดคำเป็น

เทคโนโลยี ทางการ ผลิต

2.3 การตัดคำโดยใช้แบบจำลองไตรแกรม

ในงานวิจัยนี้เราพัฒนาวิธีตัดคำโดยใช้ข้อมูลทางสถิติที่ได้จากคลังข้อความที่เตรียมไว้ ซึ่งคลังข้อความนี้เราได้ให้นักภาษาศาสตร์ตัดคำและกำกับหมวดคำที่เหมาะสม ปัญหาของการตัดคำในภาษาไทยสามารถเขียนแทนด้วยสมการดังต่อไปนี้

$$\begin{aligned}\underset{w_{1,n}}{arg\,max}\ \ P(w_{1,n}|c_{1,m}) &= \underset{w_{1,n}}{arg\,max}\ \ \frac{P(c_{1,m}|w_{1,n})\cdot p(w_{1,n})}{p(c_{1,m})} \\ &= \underset{w_{1,n}}{arg\,max}\ \ P(w_{1,n}) \\ &= \underset{w_{1,n}\ \ t_{1,n}}{argmax} \sum p(w_{1,n}, t_{1,n})\end{aligned} \qquad (1)$$

ซึ่งมีความหมายว่าจากสายอักขระของตัวอักษร $c_{1,m}$ ที่เป็นอินพุต เราต้องการแบ่งสายอักขระนี้เป็นคำ $w_1, w_2, \ldots, w_n$ หรือเขียนสั้นๆ $w_{1,n}$ และมีหมวดคำเป็น $t_1, t_2, \ldots, t_n$ หรือเขียนโดยย่อเป็น $t_{1,n}$ เราต้องการตัดคำเพื่อให้ได้ค่าของ $p(w_{1,n}|c_{1,m})$ สูงสุด ถ้าเราตั้งสมมติฐานว่า (1) ความ

น่าจะเป็นที่คำหนึ่งๆจะปรากฏ ณ ตำแหน่งใดๆในประโยคไม่ขึ้นกับสิ่งอื่นๆ และ (2) หมวดคำหนึ่งๆจะขึ้นอยู่กับหมวดคำก่อนหน้า 2 หมวดคำเท่านั้นแล้ว สมการด้านบนจะเป็น

$$= \underset{w_{1,n}}{arg\,max} \sum_{t_{1,n+1}} \prod_{i=1}^{n} p(w_i|t_i) \cdot p(t_i|t_{i-1}, t_{i-2}) \qquad (2)$$

จากคลังข้อความที่มีอยู่เราสามารถหาค่าของ $p(w_i|t_i)$ ได้โดยนับจำนวนของหมวดคำ t_i ที่เป็นหมวดคำของคำ w_i หารด้วยจำนวนของ t_i ที่เป็นหมวดคำของคำใดๆ (จำนวนของ t_i ทั้งหมด) ส่วน $p(t_i|t_{i-1}, t_{i-2})$ หาได้โดยนับจำนวนหมวดคำ t_i ที่มีหมวดคำ t_{i-1} และ t_{i-2} ตามหลังที่ปรากฏในคลังข้อความหารด้วยจำนวนของหมวดคำ t_{i-1} และ t_{i-2} ที่ปรากฏทั้งหมด จากค่าเหล่านี้ที่คำนวณได้เรานำมาใช้ในการตัดคำภาษาไทย และได้เขียนซอฟต์แวร์เพื่อทดลองผล

คลังข้อความที่ใช้ในการทดลองนำมาจากเอกสารในหลายสาขาที่ประกอบด้วย ข่าว บทความ ข้อความ พจนานุกรมและอื่นๆ นอกจากคำแล้วภาษาไทยยังไม่มีเครื่องหมายแสดงการจบประโยคอย่างชัดเจน ดังนั้นคลังข้อความจะถูกแยกเป็นประโยคก่อน จากนั้นคำจะถูกกำกับหมวดคำที่เหมาะสมโดยนักภาษาศาสตร์ คลังข้อความที่กำกับหมวดแล้วจะถูกใช้เป็นคลังข้อความสำหรับสอน หมวดคำที่ใช้ในการทดลองประกอบด้วย 47 หมวดดังที่แสดงในหัวข้อที่ 3.1 ที่จะกล่าวต่อไป

ข้อมูลที่นำมาใช้ในการทดลอง ได้นำมาจากคลังข้อความภาษาไทย (Corpus) ที่ได้มีการตัดคำและกำกับหน้าที่ของคำไว้แล้วโดยนักภาษาศาสตร์ จำนวนประมาณ 22,500 ประโยค โดยแบ่งเป็น 2 ส่วนคือ 1.เพื่อใช้ในการเก็บค่าสถิติต่างๆ จำนวนประมาณ 18,500 ประโยค 2.ได้นำคลังข้อความภาษาไทยที่เหลืออีกจำนวนประมาณ 4,000 ประโยคเพื่อใช้ในการทดสอบประสิทธิภาพของแต่ละวิธี ซึ่งผลการทดลองแต่ละวิธีได้แสดงตามตารางที่ 1

ตารางที่ 1 เปรียบเทียบเปอร์เซนต์ความถูกต้องของการตัดคำ

	การตัดคำให้ยาวที่สุด	การตัดคำให้จำนวนน้อยที่สุด	วิธีการที่เสนอ
เปอร์เซนต์ของคำที่ตัดคำถูกต้อง	94.40	94.11	96.74

ผลแสดงให้เห็นว่าความถูกต้องของวิธีการที่เสนอเป็น 96.74% เปรียบเทียบกับ 94.11% ที่ได้โดยวิธีตัดคำจำนวนน้อยที่สุด และ 94.40% ของวิธีตัดคำให้ยาวที่สุด ผลที่ได้แสดงให้เห็นว่าจากการใช้ข้อมูลทางสถิติของคลังข้อความเพื่อช่วยในการตัดคำพบว่าวิธีการที่เสนอได้ผลดีกว่าวิธี

ตัดคำน้อยที่สุดซึ่งเป็นวิธีที่ดีที่สุดวิธีหนึ่งที่มีอยู่ ประโยคซึ่งพบว่าวิธีตัดคำที่น้อยที่สุดไม่สามารถตัดได้ถูกต้องได้แก่

เราเรียกว่าการวิเคราะห์	--- ตัดเป็น --->	เรา เรียก ว่าการ วิเคราะห์
ที่ไม่จำเป็นมาใช้	--- ตัดเป็น --->	ที่ ไม่ จำ เป็นมา ใช้
เทคโนโลยีทางการผลิต	--- ตัดเป็น --->	เทคโนโลยี ทางการ ผลิต
อุปกรณ์ทางการสื่อสาร	--- ตัดเป็น --->	อุปกรณ์ ทางการ สื่อสาร

ซึ่งถ้าใช้วิธีการที่เสนอในงานวิจัยนี้จะตัดได้อย่างถูกต้อง อย่างไรก็ตามก็ยังมีประโยคบางประเภทที่ยังมีปัญหาอยู่ ได้แก่ประโยคที่ประกอบด้วยคำที่ไม่รู้จัก ซึ่งยังต้องการการปรับปรุงให้ดีขึ้น ซึ่งจะกล่าวในหัวข้อที่ 4.3 ต่อไป

3. วิธีการกำกับหมวดคำ

บทนี้กล่าวถึงการใช้แบบจำลองไตรแกรมในการกำกับหมวดคำให้กับคำในภาษาไทย ก่อนที่จะกล่าวถึงการใช้แบบจำลองไตรแกรมขอกล่าวถึงหมวดคำทั้งหมดที่ใช้ในงานวิจัยนี้ในหัวข้อต่อไป

3.1 หมวดคำในภาษาไทย

คำในประโยคภาษาไทยมีหน้าที่แตกต่างกันออกไป โดยในปัจจุบันได้มีงานวิจัยทางภาษาศาสตร์ซึ่งทำการแบ่งประเภทของคำในภาษาไทยออกเป็นประเภทต่างๆกันตามหน้าที่ของคำ 47 ประเภท (Charoenporn et.al.,1997) ดังแสดงในตารางที่ 2 ด้านล่างนี้

ตารางที่ 2 หมวดคำที่ใช้ในงานวิจัย

หมวดคำ	รายละเอียด	ตัวอย่าง
NPRP	Proper name	วินโดว์ส 95, โตโยต้า, โค้ก, พระอาทิตย์
NCNM	Cardinal number	หนึ่ง, สอง, สาม, 1, 2, 3
NONM	Ordinal number	ที่หนึ่ง, ที่สอง, ที่สาม, ที่1, ที่2, ที่3
NLBL	Label noun	1, 2, 3, 4, ก, ข, ค, ง
NCMN	Common noun	หนังสือ, อาหาร, อาคาร, คน
NTTL	Title noun	ดร., พลเอก
PPRS	Personal pronoun	คุณ, เขา, ฉัน
PDMN	Demonstrative pronoun	นี่, นั่น, ที่นั่น, ที่นี่
PNTR	Interrogative pronoun	ใคร, อะไร, อย่างไร
PREL	Relative pronoun	ที่, ซึ่ง, อัน, ผู้
VACT	Active verb	ทำงาน, ร้องเพลง, กิน
VSTA	Stative verb	เห็น, รู้, คือ
VATT	Attribute verb	อ้วน, ดี, สวย
XVBM	Pre-verb auxiliary, befor negator "ไม่"	เกิด, เกือบ, กำลัง
XVAM	Pre-verb auxiliary, after negator "ไม่"	ค่อย, น่า, ได้
XVMM	Pre-verb, befor or after negator "ไม่"	ควร, เคย, ต้อง

หมวดคำ	รายละเอียด	ตัวอย่าง
XVBB	Pre-verb auxiliary, in imperative mood	กรุณา, จง, เชิญ, อย่า, ห้าม
XVAE	Post-verb auxiliary	ไป, มา, ขึ้น
DDAN	Definite determiner, after noun without classifier in between	นี่, นั่น, โน่น, ทั้งหมด
DDAC	Definite determiner, allowing classifier in between	นี้, นั้น, โน้น, นู้น
DDBQ	Definite determiner, between noun and classifier or preceding quantitative expression	ทั้ง, อีก, เพียง
DDAQ	Definite determiner, following quantitative expression	พอดี, ถ้วน
DIAC	Indefinite determiner, following noun; allowing classifier in between	ไหน, อื่น, ต่างๆ
DIBQ	Indefinite determiner, between noun and classifier or preceding quantitative expression	บาง, ประมาณ, เกือบ
DIAQ	Indefinite determiner, following quantitative expression	กว่า, เศษ
DCNM	Determiner, cardinal number expression	หนึ่งคน, เสือ 2 ตัว
DONM	Determiner, ordinal number expression	ที่หนึ่ง, ที่สอง, ที่สุดท้าย
ADVN	Adverb with noun form	เก่ง, เร็ว, ช้า, สม่ำเสมอ
ADVI	Adverb with iterative form	เร็วๆ, เสมอๆ, ช้าๆ
ADVP	Adverb with prefixed form	โดยเร็ว
ADVS	Sentenial adverb	โดยปกติ, ธรรมดา
CNIT	Unit classifier	ตัว, คน, เล่ม
CLTV	Collective classifier	คู่, กลุ่ม, ฝูง, เชิง, ทาง, ด้าน, แบบ, รุ่น

หมวดคำ	รายละเอียด	ตัวอย่าง
CMTR	Measurement classifier	กิโลกรัม, แก้ว, ชั่วโมง
CFQC	Frequency classifier	ครั้ง, เที่ยว
CVBL	Verbal classifier	ม้วน, มัด
JCRG	Coordinating conjunction	และ, หรือ, แต่
JSBR	Comparative conjunction	กว่า, เหมือนกับ, เท่ากับ
RPRE	Preposition	จาก, ละ, ของ, ใต้, บน
INT	Interjunction	โอ้ย, โอ้, เออ, เฮ้, อ๋อ
FIXN	Nominal prefix	การทำงาน, ความสนุกสนาน
FIXV	Adverbial prefix	อย่างเร็ว
EAFF	Ending for affirmative sentence	จ๊ะ, ค่ะ, ครับ, นะ, น่า, เถอะ
EITT	Ending for interrogative sentence	หรือ, เหรอ, ไหม, มั้ย
NEG	Negator	ไม่, มิได้, ไม่ได้, มิ
PUNC	Punctuation	(,), “, „, ฯ

3.2 วิธีการกำกับหมวดคำ

การกำกับหมวดคำคือการกำหนดค่าของหมวดคำให้กับประโยคที่รับเข้ามา เช่น ไก่ขันตอนเช้า จะถูกกำกับหมวดคำเป็น NCMN VACT ADVN เป็นต้น ประโยคที่มีหมวดคำกำกับแล้วจะช่วยให้การแจงส่วนประโยคทำได้ถูกต้องและง่ายขึ้น ซึ่งแต่เดิมแล้วการแจงส่วนประโยคจะประสบปัญหาเพราะว่าคำหนึ่งๆมีได้หลายหมวดคำ ทำให้การประมวลผลเกิดความกำกวมส่งผลให้ประมวลผลไม่ถูกต้องและไม่ได้ประสิทธิภาพเท่าที่ควร วิธีการกำกับหมวดคำสามารถแบ่งเป็น 2 หลักการใหญ่ๆคือ วิธีการกำกับหมวดคำโดยใช้กฎ และโดยใช้แบบจำลองไตรแกรม (Trigram Model)

3.2.1 การกำกับหมวดคำโดยใช้กฎ

การกำกับหมวดคำโดยใช้กฎ (Barbara, et.al., 1971, Brill 1993) ทำโดยเขียนกฎทางภาษาขึ้นมาเพื่อกำกับหมวดคำให้กับคำหนึ่งๆ โดยพิจารณารูปแบบและหมวดคำของคำที่อยู่ก่อนหน้าและหลัง เช่น กฎ “คำปัจจุบันจะไม่เป็นคำกริยา ถ้าคำที่อยู่ก่อนหน้าเป็นคำบ่งชี้ (determiner)” เป็นต้น กฎเหล่านี้ในงานวิจัยแรกๆจะต้องใช้คนเขียนขึ้น ต่อมา Brill (Brill, 1993) ได้เสนอวิธีการสร้างรูปแบบ (Template) ของกฎขึ้นมาโดยไม่ระบุรายละเอียด จากนั้นให้โปรแกรมกำกับหมวดคำแก้ไขกฎให้ดีขึ้นโดยใช้ข้อมูลจากคลังข้อความที่ใช้สอน ซึ่งอาศัยการเรียนรู้จากข้อผิดพลาด

แล้วแปลงกฎให้ถูกต้องยิ่งขึ้น ข้อดีของวิธีการโดยใช้กฎนี้คือ ได้กฎที่มีขนาดเล็กเมื่อเทียบกับวิธีการโดยใช้แบบจำลองไตรแกรมที่จะกล่าวต่อไป และทำงานได้รวดเร็วกว่า ข้อด้อยคือการเขียนกฎหรือรูปแบบต้องอาศัยนักภาษาศาสตร์ และการเขียนกฎที่สมบูรณ์ก็ทำได้ยาก

3.2.2 การกำกับหมวดคำโดยใช้แบบจำลองไตรแกรม

วิธีการนี้เป็นเทคนิคที่นำฮิดเดนมาร์คอฟโมเดล (Hidden Markov Model) ซึ่งเป็นวิธีการทางการรู้จำเสียงมาประยุกต์กับการกำกับหมวดคำ โดยการรวบรวมค่าสถิติของความน่าจะเป็นของคำและหมวดคำต่างๆไว้ใช้ในการคำนวณ ปัญหาของการกำกับหมวดคำสามารถแสดงได้โดยสมการด้านล่างนี้

$$\begin{aligned} \underset{t_{1,n}}{arg\,max} \quad P(t_{1,n}|w_{1,n}) &= \underset{t_{1,n}}{arg\,max} \quad \frac{P(w_{1,n},t_{1,n})}{p(w_{1,n})} \\ &= \underset{t_{1,n}}{arg\,max} \quad P(w_{1,n},t_{1,n}) \end{aligned} \qquad (3)$$

ซึ่งแสดงว่าการกำกับหมวดคำคือการหาค่าของ $t_{1,n}$ ที่ทำให้ความน่าจะเป็นที่จะได้หมวดคำเป็น $t_{1,n}$ เมื่อรู้ $w_{1,n}$ มีค่ามากที่สุด โดยที่ $t_{1,n}$ เป็นหมวดคำตั้งแต่คำที่หนึ่งถึงคำที่ n และ $w_{1,n}$ เป็นคำที่หนึ่งถึงคำที่ n แบบจำลองไตรแกรมคือแบบจำลองที่ตั้งสมมติฐานว่า ความน่าจะเป็นที่คำหนึ่งๆจะปรากฏ ณ ตำแหน่งใดๆในประโยคไม่ขึ้นกับสิ่งอื่นๆ และหมวดคำหนึ่งๆจะขึ้นอยู่กับหมวดคำก่อนหน้า 2 หมวดคำเท่านั้นแล้ว จากสมมติฐานของแบบจำลองไตรแกรม สมการด้านบนจะเป็น

$$= \underset{t_{1,n}}{arg\,max} \quad \prod_{i=1}^{n} P(w_i|t_i)\cdot P(t_i|t_{i-1},t_{i-2}) \qquad (4)$$

ค่าความน่าจะเป็นต่างๆในสมการนี้สามารถหาได้โดยรวบรวมจากคลังข้อความที่มีอยู่ เช่น ค่าของ $p(w_i|t_i)$ ได้โดยนับจำนวนของหมวดคำ t_i ที่เป็นหมวดคำของคำ w_i หารด้วยจำนวนของ t_i ที่เป็นหมวดคำของคำใดๆ (จำนวนของ t_i ทั้งหมด) ส่วน $p(t_i|t_{i-1},t_{i-2})$ หาได้โดยนับจำนวนหมวดคำ t_i ที่มีหมวดคำ t_{i-1} และ t_{i-2} ตามหลังที่ปรากฏในคลังข้อความหารด้วยจำนวนของหมวดคำ t_{i-1} และ t_{i-2} ที่ปรากฏทั้งหมด

สมการที่ (4) ที่ได้นี้มีความคล้ายกับสมการที่ (2) ที่ใช้สำหรับตัดคำ ข้อแตกต่างระหว่างสมการทั้งสองคือสมการที่ (4) ต้องการหาค่าชุดของหมวดคำ $t_{1,n}$ ที่ทำให้ความน่าจะเป็นในการแบ่งประโยคของเป็นคำๆพร้อมทั้งกำหนดหมวดคำของคำเหล่านั้นให้ได้สูงสุด ในขณะที่สมการที่ (2) ต้องการค่าของชุดของคำ $w_{1,n}$ โดยรวมค่า $t_{1,n}$ ทุกแบบที่เป็นไปได้ที่ทำให้ความน่าจะเป็นในการตัดคำค่าสูงสุด

แบบจำลองไตรแกรมนี้ได้รับการวิจัยอย่างกว้างขวางและพบว่าสามารถกำกับหมวดคำด้วยความถูกต้องสูงโดยเฉพาะกับภาษาอังกฤษ (W.Church, 1988, E.Charniak et.al, 1993) ข้อดีของแบบจำลองไตรแกรม คือให้ผลความถูกต้องที่สูง และไม่จำเป็นต้องอาศัยนักภาษาศาสตร์เขียนกฎทางไวยกรณ์

3.2.3 ผลการทดลองใช้แบบจำลองไตรแกรมในการกำกับหมวดคำภาษาไทย

จากข้อดีของการกำกับหมวดคำโดยแบบจำลองไตรแกรม ทำให้เราเลือกแบบจำลองนี้สำหรับการกำกับหมวดคำภาษาไทย แม้ว่าข้อด้อยของแบบจำลองไตรแกรมนี้จะมีอยู่บ้างคือต้องใช้เนื้อที่หน่วยความจำมากในการเก็บตารางค่าความน่าจะเป็นต่างๆ และความเร็วในการคำนวณช้ากว่าเมื่อเปรียบเทียบกับการกำกับหมวดคำโดยใช้กฎ แต่เนื่องจากปัจจุบันหน่วยความจำมีราคาถูกลงมาก และความเร็วในการประมวลผลของคอมพิวเตอร์ได้รับการพัฒนาอย่างต่อเนื่อง จึงคิดว่าข้อด้อยเหล่านี้ไม่เป็นปัญหามากนัก

ข้อมูลที่นำมาใช้ในการทดลองการกำกับหมวดคำภาษาไทย ได้นำมาจากคลังข้อความเดียวกับในหัวข้อ 2.3 กล่าวคือใช้ประโยคจำนวนทั้งสิ้นประมาณ 22,500 ประโยค โดยแบ่งเป็น 2 ส่วน คือ ส่วนแรกจำนวน 18,500 ประโยค ใช้สำหรับเก็บค่าสถิติต่างๆ และส่วนที่สอง 4,000 ประโยคที่เหลือใช้เพื่อทดสอบประสิทธิภาพของแบบจำลองไตรแกรม ผลการทดลองแสดงตามตารางที่ 3 เพื่อให้เห็นประสิทธิภาพของการกำกับหมวดคำได้ชัดเจนขึ้น ตารางที่ 3 ได้รวมเปอร์เซนต์ความถูกต้องของการตัดคำโดยแบบจำลองไตรแกรมนี้ไว้ด้วย

ตารางที่ 3 ผลการทดลองการกำกับหมวดคำโดยแบบจำลองไตรแกรม

	แบบจำลองไตรแกรม
เปอร์เซนต์ของคำที่ตัดคำถูกต้อง	96.74
เปอร์เซนต์ของคำที่ตัดถูกต้อง และมีหมวดคำถูกต้อง	89.52

ผลการทดลองแสดงให้เห็นว่าแบบจำลองไตรแกรมสามารถกำกับหมวดคำ ได้เปอร์เซนต์ของคำที่มีหมวดคำถูกต้องถึง 89.52 % ซึ่งถือว่าค่อนข้างสูงมาก และในจำนวนของคำที่ตัดได้อย่างถูกต้องนั้นมีบางส่วนที่ได้หมวดคำไม่ถูกต้อง ถึงแม้ว่าเปอร์เซนต์ความถูกต้อง 89.52 % ที่ได้จะไม่สมบูรณ์ถึง 100 % ก็ตาม แต่ก็ช่วยให้คนสามารถกำกับหมวดคำได้อย่างสะดวกและมีประสิทธิภาพยิ่งขึ้น

4. การใช้ประโยชน์ของคลังข้อความ

บทนี้กล่าวถึงการนำคลังข้อความที่มีหมวดคำกำกับแล้ว มาช่วยในการประมวลผลภาษาธรรมชาติ ซึ่งได้แก่ (1) การแก้ไขคำผิดจาก OCR ภาษาไทย และ (2) การระบุคำที่ไม่รู้จัก ก่อนที่จะกล่าวถึงการนำคลังข้อความมาใช้ในงานทั้งสอง เราขอกล่าวถึงอัลกอริทึม Winnow ที่สามารถนำข้อมูลในคลังข้อความมาใช้ให้เกิดประโยชน์ในงานทั้งสองนี้

4.1 อัลกอริทึม Winnow

หัวข้อนี้กล่าวถึงอัลกอริทึมเรียนรู้ออนไลน์ (online learning algorithm) แบบหนึ่งที่เรียกว่า Winnow ซึ่งใช้สำหรับงานประยุกต์ในบทนี้ อัลกอริทึม Winnow เป็นอัลกอริทึมแบบค่าขีดแบ่งเชิงส้น (linear threshold algorithm) และแก้ไขน้ำหนักแบบคูณ (multiplicative weight updating algorithm) (N. Littlestone, 1988; A.R.Golding & D. Roth, 1996) อัลกอริทึมแสดงในรูปที่ 1 ด้านล่างนี้

กำหนดให้ $v_1,...,v_m$ เป็นค่าของคอนเซ็ปท์เป้าหมายที่ต้องการเรียน และ x_i เป็นคุณสมบัติ (feature) ที่ผู้เชี่ยวชาญ i ตรวจสอบ

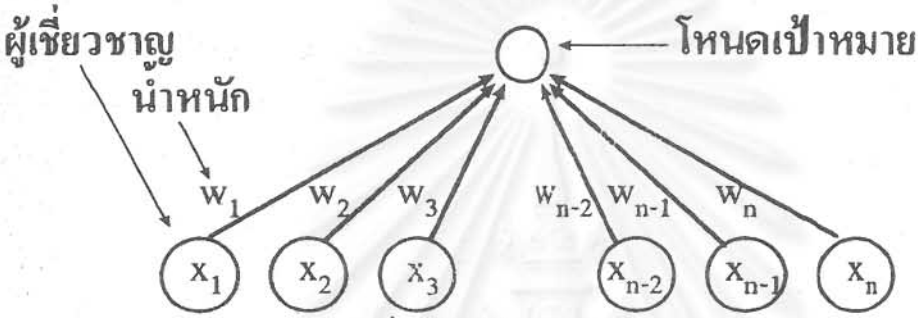
1. กำหนดค่าเริ่มต้นของ $w_1,...,w_n$ ของทุกคุณสมบัติให้เป็น 1
2. **For Each** ตัวอย่าง $x = \{x_1,...,x_n\}$. **Do**
   - 2.1 ให้ V เป็นค่าของคอนเซ็ปท์เป้าหมายของ x
   - 2.2 เอ๊าท์พุต $v'_j = \arg\max v_j \in \{v_1,...,v_m\} \sum_{i:x_i=v_j} w_i$
   - 2.3 **If** อัลกอริทึมโดยรวมพยากรณ์ผิด ($v'_j \neq V$) **then**
     - (a) for each $x_i = V$, $w_i \leftarrow w_i \bullet \alpha$
     - (b) for each $x_i = v'_j$, $w_i \leftarrow w_i \bullet \beta$
   - 2.4 **If** อัลกอริทึมโดยรวมพยากรณ์ถูก ($v'_j = V$) **then**
     - for each $x_i \neq v'_j$, $w_i \leftarrow w_i \bullet \beta$

โดยที่ θ, $\alpha > 1$ และ $\beta < 1$ คือค่าขีดแบ่ง, โปรโมชันพารามิเตอร์ และดี-โปรโมชันพารามิเตอร์ ตามลำดับ ในการทดลองค่าเหล่านี้ เป็น 1, 3/2 และ 1/2 ตามลำดับ

รูปที่ 1 อัลกอริทึม Winnow

เราสามารถมองว่า Winnow เป็นเน็ตเวิร์กของโหนดเป้าหมาย 1 โหนด ซึ่งเชื่อมต่อกับโหนดอีก n โหนด ที่เรียกว่า "ผู้เชี่ยวชาญ (specialist)" ดังแสดงในรูปที่ 2 ผู้เชี่ยวชาญแต่ละโหนด

จะตรวจสอบคุณสมบัติ x_i ที่อยู่ในตัวอย่าง แนวคิดพื้นฐานของอัลกอริทึมคือในการที่จะดึงคุณสมบัติที่มีประโยชน์นั้น อัลกอริทึมจะสอบถามความเห็นจากทุกผู้เชี่ยวชาญ ซึ่งแต่ละโหนดจะมีความเชี่ยวชาญในคุณสมบัติหนึ่งๆ จากนั้นอัลกอริทึมจะรวมความเห็นทั้งหมดเพื่อการตัดสินใจโดยรวม ในการทดลองที่จะกล่าวในหัวข้อต่อไปนั้น เราให้ผู้เชี่ยวชาญแต่ละโหนดตรวจสอบคุณสมบัติหนึ่งหรือสองอย่างของตัวอย่าง ตัวอย่างเช่น ผู้เชี่ยวชาญอาจจะคาดคะเนค่าของคอนเซ็ปท์เป้าหมายโดยการตรวจสอบเงื่อนไข (คุณสมบัติที่1=ค่าที่1) และ (คุณสมบัติที่2=ค่าที่2) คู่ลำดับเหล่านี้จะเป็นตัวเลือกของคุณสมบัติที่เราพยายามจะดึงออกมา



รูปที่ 2 เน็ตเวิร์กของ Winnow

ผู้เชี่ยวชาญจะให้คำพยากรณ์เฉพาะในกรณีที่เงื่อนไขถูกต้องเท่านั้น ("(คุณสมบัติที่1=ค่าที่1)" เป็นจริง ในกรณีของผู้เชี่ยวชาญที่ตรวจสอบคุณสมบัติหนึ่งอย่าง หรือ "(คุณสมบัติที่1=ค่าที่1) และ (คุณสมบัติที่2=ค่าที่2)" เป็นจริง ในกรณีของผู้เชี่ยวชาญที่ตรวจสอบคุณสมบัติสองอย่าง) และในกรณีเช่นนั้นผู้เชี่ยวชาญจะให้คำพยากรณ์ที่พบบ่อยที่สุดในจำนวน k ครั้งที่เคยเห็น ในกรณีที่ผู้เชี่ยวชาญนั้นไม่เคยเห็นค่าของคุณสมบัติเดียวกันนี้ในตัวอย่างที่ผ่านมา ผู้เชี่ยวชาญอาจจะเลือกที่จะอยู่เฉยแทนที่จะให้คำพยากรณ์ก็ได้

อัลกอริทึมโดยรวมจะแก้ไขน้ำหนัก w_i ของผู้เชี่ยวชาญโดยพิจารณาที่คำพยากรณ์ของผู้เชี่ยวชาญนั้น น้ำหนักของแต่ละผู้เชี่ยวชาญจะถูกกำหนดค่าเริ่มต้นให้เป็น 1 ในกรณีที่อัลกอริทึมโดยรวมพยากรณ์ไม่ถูกต้อง น้ำหนักของผู้เชี่ยวชาญที่พยากรณ์ไม่ถูกต้องจะถูกหารด้วย 2 และน้ำหนักของผู้เชี่ยวชาญที่พยากรณ์ถูกต้องจะถูกคูณด้วย 3/2 นอกจากนั้นน้ำหนักของผู้เชี่ยวชาญจะถูกหารด้วย 2 เมื่อพยากรณ์ผิดพลาด แม้ว่าอัลกอริทึมโดยรวมจะพยากรณ์ถูกต้อง วิธีการแก้ไขน้ำหนักนี้เหมือนกับที่ใช้ใน (A.Blum,1997) ข้อดีของ Winnow ที่ทำให้เราตัดสินใจใช้สำหรับงานของเราคือ (1) อัลกอริทึมเป็นแบบออนไลน์ทำให้สามารถเรียนรู้ได้อย่างต่อเนื่อง และ (2) อัลกอริทึมเรียนรู้ได้อย่างมีประสิทธิภาพแม้ว่าตัวอย่างจะประกอบด้วยคุณสมบัติที่ไม่เกี่ยวข้องจำนวนมาก (N.Littlestone, 1988)

4.2 การแก้ไขคำผิดของ OCR ภาษาไทย

ปัญหาของการแก้ไขคำผิดที่เกิดจากโปรแกรม OCR สามารถนิยามได้ดังต่อไปนี้

กำหนดให้ S เป็นสายอักขระของอักษร $S=c_1c_2...c_n$ ซึ่งถูกผลิตโดย OCR ให้หาลำดับของคำ $W=w_1w_2...w_l$ ซึ่งทำให้ค่าความน่าจะเป็น $P(W|S)$ มีค่าสูงสุด ก่อนที่จะอธิบายวิธีการที่ใช้สำหรับสร้างโมเดลของ $P(W|S)$ ขอกล่าวถึงคุณสมบัติของภาษาไทยที่ทำให้การแก้ไขคำผิดประสบความยากลำบาก

- ในภาษาไทยนั้น ไม่มีเครื่องหมายแบ่งคำ ซึ่งต่างจากภาษาอังกฤษที่มีช่องว่างคั่นแสดงให้เห็นขอบเขตของคำอย่างชัดเจน
- ภาษาไทยประกอบด้วยตัวอักษรหลายระดับ คือระดับบน ระดับกลาง และ ระดับล่าง และอักษรบางตัวยังมีตำแหน่งในการวางคาบเกี่ยวระหว่างระดับ ตัวอย่างเช่น "ฟุ้ง" ประกอบด้วยตัวอักษร 4 ตัวที่อยู่ในระดับต่างๆกัน คือ "สระอุ" อยู่ในระดับล่าง "ไม้โท" อยู่ในระดับบน "ง" อยู่ในระดับกลาง และ "ฟ" อยู่คาบเกี่ยวระหว่างระดับกลางและระดับบน

ตัวอักษรที่มีตำแหน่งในการวางคาบเกี่ยวระหว่างระดับ เช่น "ฟ" นี้มักทำให้เกิดการเชื่อมต่อหรือทับกับอักษรอื่นที่อยู่ในระดับบนหรือระดับล่าง ตัวอักษรพวกนี้ส่วนมากทำให้โปรแกรม OCR ภาษาไทยไม่สามารถรู้จำตัวอักษรได้ถูกต้อง เช่น "ฟ้" ถูกรู้จำผิดพลาดเป็น "ฟ" ซึ่งเป็นกรณีที่ตัวอักษรที่อยู่ในระดับบนหายไป หรืออาจถูกรู้จำผิดพลาดเป็น "โ" ซึ่งเป็นกรณีที่ตัวอักษรทั้ง 2 ตัว ถูกแทนที่ด้วยตัวอักษรอื่นเพียงตัวเดียว

4.2.1 แบบจำลองไตรแกรม

เราสามารถใช้แบบจำลองไตรแกรมเพื่อหาค่า W ที่ทำให้ $P(W|S)$ มีค่าสูงสุดได้ดังต่อไปนี้

$$\underset{W}{argmax}\, P(W|S) = \underset{W}{argmax}\, P(W)P(S|W)/P(S)$$

$$= \underset{W}{argmax}\, P(W)P(S|W) \qquad (5)$$

ค่าความน่าจะเป็น $P(W)$ ถูกกำหนดโดยโมเดลของภาษาและสามารถประมาณได้โดยใช้แบบจำลองไตรแกรมดังต่อไปนี้:

$$P(W) = P(W,T) = \prod_i P(t_i|t_{i-2}, t_{i-1})P(w_i|t_i) \qquad (6)$$

$P(S|W)$ เป็นลักษณะเฉพาะของ OCR หนึ่งๆ และสามารถประมาณได้จากการรวบรวมข้อมูลทางสถิติจากข้อความดั้งเดิมก่อนผ่านโปรแกรม OCR นั้นและข้อความที่ได้จาก OCR เราสมมติว่ากำหนดให้ W เป็นลำดับของคำดั้งเดิมที่ประกอบด้วยตัวอักษร $v_1v_2...v_m$ OCR จะผลิตลำดับ S $(=c_1c_2.c_n)$ โดยการใช้ตัวกระทำ 3 อย่างดังต่อไปนี้ซ้ำหลายๆครั้ง: (1) แทนที่ตัวอักษรด้วยตัวอักษรอื่น; (2) แทรกตัวอักษรและ (3) ลบตัวอักษร สมมติให้ Si เป็น i-prefix ของ S ที่สร้างโดยตัว

อักษรตัวแรกถึงตัวอักษรตัวที่ i -ของ S $(=c_1c_2...c_i)$ และในทำนองเดียวกัน ให้ Wj เป็น j-prefix ของ W $(=v_1v_2...v_j)$ โดยการใช้การโปรแกรมแบบพลวัติ เราสามารถคำนวณ $P(S|W)$ $(=P(S_n|W_m))$ โดยใช้สมการด้านล่างนี้:

$$P(S_i|W_j) = max\{ \begin{array}{l} P(S_{i-1}|W_j)*P(ins(c_i)), \\ P(S_i|W_{j-1})*P(del(v_j)), \\ P(S_{i-1}|W_{j-1})*P(c_i|v_j) \} \end{array} \quad (7)$$

โดยที่ $P(ins(c))$, $P(del(v))$ และ $P(c|v)$ เป็นความน่าจะเป็นที่ตัวอักษร c ถูกแทรก, ตัวอักษร v ถูกลบและตัวอักษร v ถูกแทนที่ด้วย c ตามลำดับ

วิธีการหนึ่งสำหรับแก้ไขคำผิดพลาดของ OCR คือการใช้สมการที่ (5) ด้านบน เพื่อสร้างสายอักขระย่อยทั้งหมดที่เป็นไปได้จากประโยคอินพุต และให้สายอักขระย่อยเหล่านั้นเป็นคำที่จะพิจารณา (M. Nagata, 96) คำที่อยู่ในพจนานุกรมที่ตรงกับสายอักขระย่อยเหล่านั้น รวมทั้งคำที่ตรงบางส่วนกับสายอักขระย่อยเหล่านั้นจะถูกค้นคืนขึ้นมา และเพื่อที่จะจัดการกับคำที่ไม่รู้จัก สายอักขระย่อยอื่นๆทุกสายก็ต้องถูกนำมาพิจารณาด้วย อัลกอริทึมจะสแกนเพื่อหา N ลำดับคำที่ดีที่สุดเพื่อให้เป็นตัวเลือก โดยทั่วไปวิธีการนี้จะใช้งานได้ดี เพียงแต่ว่าต้องใช้เวลาในการคำนวณมาก เพราะว่าต้องสร้างคำเพื่อใช้เป็นตัวเลือกจำนวนมากมาย และต้องลองรวมคำเหล่านั้นให้ได้คำตอบที่ดีที่สุด จึงทำให้ช้ามาก

4.2.2 แบบจำลองไตรแกรมแบบเลือก (Selective Trigram Model)

เพื่อลดทอนปัญหาของการคำนวณดังกล่าว เราพยายามที่จะลดจำนวนคำที่เป็นตัวเลือกให้น้อยลง โดยการสร้างเฉพาะกรณีที่จำเป็นจริงๆ หลังจากที่ได้วิเคราะห์เอ๊าท์พุตของ OCR พบว่าส่วนของประโยคโดยมากจะถูกรู้จำได้อย่างถูกต้อง ซึ่งไม่จำเป็นต้องถูกจัดการ ดังนั้น ถ้าเราจำกัดให้หาเฉพาะบริเวณที่น่าสงสัยแทนที่จะทดลองทำทั้งประโยค เราก็จะสามารถลดเวลาในการคำนวณได้อย่างมาก ด้านล่างนี้แสดงอัลกอริทึมสำหรับการแก้ไขข้อความจาก OCR โดยเลือกแก้ไขเฉพาะบริเวณ

1. หาบริเวณที่น่าสงสัย:
   หาสายอักขระย่อยทั้งหมดในประโยคอินพุตที่ตรงพอดีกับคำในพจนานุกรม แต่ละสายอักขระย่อยอาจทับกับสายอักขระอื่น ส่วนที่เหลือของประโยคที่ไม่ถูกคลุมโดยสายอักขระย่อยใดๆจะถูกพิจารณาว่าเป็นบริเวณที่น่าสงสัย
2. สร้างสมมติฐานสำหรับ คำผิดที่ไม่เป็นคำ (nonword) และ คำที่ไม่รู้จัก (unknown word)
   (a) เมื่อได้สายอักขระทุกสายที่น่าสงสัยจากขั้นตอนที่ 1. แล้ว คำข้างๆของสายอักขระเหล่านั้นจะถูกพิจารณาให้เป็นตัวเลือกเพื่อทำการแก้ไข โดยการเชื่อมต่อสายอักขระเหล่านั้นเข้ากับสายอักขระที่น่าสงสัย

ตัวอย่างเช่น "เท ดโนโล ยี" ประกอบด้วย "ดโนโล" ซึ่งเป็นสายอักขระที่ไม่รู้จัก ในกรณีนี้ คำข้างๆจะถูกเชื่อมต่อกับสายอักขระที่น่าสงสัยเพื่อสร้างเป็นสายอักขระใหม่ ในกรณีนี้จะได้ "เทดโนโลยี"

(b) เรียก *รูทีนสร้างตัวเลือก* เพื่อสร้างคำใกล้เคียงโดยให้ระยะแก้ไขอยู่ภายในค่า k (ซึ่งจะอธิบายต่อไป) สำหรับทุกสายอักขระที่ได้รับจาก 2(a)

(c) สายอักขระย่อยทุกสายยกเว้นสายที่ขัดแย้งกับกฎไวยกรณ์ภาษาไทย เช่น ขึ้นต้นด้วยตัวอักษรที่ไม่ใช่ตัวขึ้นต้นคำ จะถูกพิจารณาให้เป็น คำที่ไม่รู้จัก

3. หาลำดับคำที่ดี:
   หาลำดับคำที่ดีที่สุด n คำตามสมการที่ (5) สำหรับคำที่ไม่รู้จักนั้น $P(w_i|$คำที่ไม่รู้จัก$)$ ถูกคำนวณโดยใช้โมเดลของคำที่ไม่รู้จักใน (M. Nagata, 1996)
4. สร้างสมมติฐานสำหรับคำผิดที่เป็นคำ:
   สำหรับคำ w_i ในลำดับคำที่ดีที่สุด N คำ ที่มีความน่าจะเป็น $P(w_{i-1},w_i,w_{i+1},t_{i-1},t_i,t_{i+1})$ ต่ำกว่าค่าขีดแบ่ง สร้างตัวเลือกโดยขั้นตอนเดียวกันกับขั้นตอนที่ 2 ยกเว้นคำผิดที่ไม่เป็นคำในขั้นตอนที่ 2 ถูกแทนที่ด้วยคำ w_i โดยกำหนดให้หาลำดับคำที่มีค่าความน่าจะเป็นตามสมการ (2) มากกว่าลำดับคำเดิม
5. หาลำดับคำที่ดีที่สุด N ลำดับ:
   เลือก N ลำดับแรกที่ดีที่สุดจากลำดับคำทั้งหมดที่ได้จากขั้นตอนที่ 4

รูทีนสร้างตัวเลือกของเราปรับปรุงจากระยะทางแก้ไขมาตรฐานและใช้ error-tolerant finite-state recognition algorithm (K. Oflazer, 1996) ระยะทางแก้ไขที่ปรับปรุงนี้ทำให้เราสามารถแทรกหรือลบตัวอักษรที่อยู่ในระดับบนและระดับล่างกี่ตัวได้ แต่ไม่ยอมให้แทรกหรือลบอักษรในระดับกลาง ในระดับกลางนั้น จะยอมให้แทนที่ตัวอักษรได้เพียง k ตัวเท่านั้น ระยะทางแก้ไขปรับปรุงนี้สะท้อนถึงคุณสมบัติของ OCR ภาษาไทยซึ่ง (1) มักจะรวมตัวอักษรที่อยู่ในระดับบนหรือล่างเข้ากับระดับกลางเวลาที่มีการคาบเกี่ยวกัน และ (2) มักจะไม่มีการแทรกตัวอักษรในระดับกลาง ตัวอย่างเช่น ถ้าเราใช้รูทีนสร้างตัวเลือก โดยใช้ระยะแก้ไขเป็น 1 โดยรับอินพุตเป็นสายอักขระ "ฟุง" เราจะได้เซตของตัวเลือกเป็น {ฟุ้ง, มุง, มุ่ง, มุ้ง, พุง, พุ่ง, ลุง, ซุง, ยุง, ยุ่ง, ยุ้ง}

จากการทดลองพบว่า โมเดลไตรแกรมแบบเลือกสามารถจัดการกับคำผิดที่ไม่เป็นคำได้ค่อนข้างดี แต่อย่างไรก็ดี โมเดลนี้ยังไม่เพียงพอสำหรับแก้ไขคำผิดที่เป็นคำหรือคำที่มีหมวดคำเดียวกัน ที่เป็นเช่นนี้เนื่องจาก โมเดลไตรแกรมจะพิจารณาเฉพาะข้อมูลของหมวดคำในช่วงแคบๆที่กำหนดเท่านั้น ทำให้ข้อมูลที่มีประโยชน์บางอย่างสูญหายไป เช่น คอลโลเคชัน (collocations) การใช้โมเดลเอ็นแกรม (N-gram) ของคำอาจช่วยแก้ไขปัญหานี้ได้ แต่ต้องใช้คลังข้อความขนาดใหญ่มากเพื่อประมาณพารามิเตอร์ให้ได้อย่างถูกต้องและต้องใช้เนื้อที่มากเพื่อจัดเก็บตารางเอ็นแกรมของคำ

ในภาษาอังกฤษ วิธีการจำนวนมากถูกเสนอเพื่อจัดการกับคำผิดที่เป็นคำในปัญหาของการแก้ไขการสะกดคำ (spelling correction) (A.R.Golding, 1995; A.R.Golding & D.Roth, 1996; A.R.Golding & Y.Schabes, 1996; T.Xiang & A.E.David, 1994) ในจำนวนนี้ วิธีการแบบใช้หลักของคุณสมบัติ (feature-based method) ได้รับการแสดงให้เห็นว่าเหนือกว่าวิธีอื่น ที่เป็นเช่นนี้เพราะว่า วิธีการนี้สามารถใช้คุณสมบัติหลายๆอย่างร่วมกันเพื่อกำหนดคำที่เหมาะสมที่สุดในบริบทที่กำหนดให้ จากที่ได้กล่าวในหัวข้อที่ 4.1 จะเห็นได้ว่า Winnow เป็นอัลกอริทึมซึ่งเป็นวิธีการเรียนรู้คุณสมบัติที่มีประโยชน์ได้ หัวข้อต่อไปแสดงวิธีการสอน Winnow

4.2.3 การสร้างเซตคอนฟิวชัน (confusion set) และนิยามคุณสมบัติเพื่อสอน Winnow

ในการที่จะใช้ Winnow ในการแก้ไขคำผิดพลาดที่เกิดจาก OCR นั้น ก่อนอื่นเรานิยามเซตคอนฟิวชันระยะแก้ไข k (k-edit distance confusion set) ดังต่อไปนี้

ให้ $S=\{c,w_1,w_2,\ldots,w_n\}$ เป็นเซตคอนฟิวชันระยะแก้ไข k ซึ่งประกอบด้วย คำศูนย์กลาง c และคำ $w_1,w_2,\ldots,w_n$ ที่สร้างโดยใช้รูทีนสร้างตัวเลือกด้วยระยะแก้ไขสูงสุดเท่ากับ k วัดจากคำศูนย์กลาง ถ้าคำ c ถูกผลิตจาก OCR แล้วมันจะถูกแก้ไขเป็น $w_1,w_2,\ldots,w_n$ หรือ c ตัวมันเอง ตัวอย่างเช่นสมมติให้ c = "ฟุง" คำที่เป็นไปได้ทั้งหมดในเซตนี้คือ {ฟุง, ฟุ้ง, มุง, มุ่ง, มุ้ง, พุง, พุ่ง, ลุง, ซุง, ยุง, ยุ่ง, ยุ้ง} นอกจากนี้คำใดๆก็ตามที่มีค่าความน่าจะเป็นน้อยกว่าค่าขีดแบ่งจะถูกลบออกจากเซตนี้ เช่นถ้าความน่าจะเป็นที่ตัวอักษร "ฟ" ถูกแทนที่ด้วย "ซ" มีค่าน้อยกว่าค่าขีดแบ่งแล้ว "ซุง" จะถูกลบออกจากเซตนี้

คุณสมบัติที่เราใช้มี 2 ประเภทคือ คำบริบท (context word) และ คอลโลเคชัน (collocations) เนื่องจากได้รับการทดสอบแล้วว่าใช้งานได้ผลในการแก้ไขการสะกดคำ (A.R.Golding, 1995) คุณสมบัติของคำบริบทใช้สำหรับทดสอบคำที่สนใจว่าอยู่ภายใน +/- M คำของคำเป้าหมายหรือไม่ และคอลโลเคชันทดสอบรูปแบบของ L คำหรือ L หมวดคำที่ติดต่อกันบริเวณคำเป้าหมาย ในการทดลองกำหนดให้ M และ L เป็น 10 และ 2 ตามลำดับ ตัวอย่างของคุณสมบัติเพื่อแบ่งแยกระหว่าง "ลุง" กับ "ฟุ้ง" คือ

(1) {ฟุ้ง,ลุง} กระจาย

(2) "ป่า"

โดยที่ (1) คือ คอลโลเคชันที่มีแนวโน้มแสดงว่าควรเป็น "ฟุ้ง" ส่วน (2) คือ คำบริบทที่มีแนวโน้มแสดงถึง "ลุง" ในกรณีเช่นนี้ อัลกอริทึมควรจะดึงคุณสมบัติ ("คำภายใน +/- 10 คำของคำเป้าหมาย" = "ป่า") และ ("คำที่ติดกับคำเป้าหมาย 1 คำด้านหลัง" = "กระจาย") เป็นคุณสมบัติที่มีประโยชน์ โดยให้ค่าน้ำหนักสูง

4.2.4 ใช้เน็ตเวิร์กเพื่อให้คะแนนกับประโยค

หลังจากที่ Winnow เรียนเน็ตเวิร์กของเซตคอนฟิวชันระยะแก้ไข k แล้ว เน็ตเวิร์กจะถูกใช้สำหรับแก้ไขประโยคที่ดีที่สุด N ประโยคที่รับจากโมเดลไตรแกรม Winnow จะประเมินคำที่เป็นคำจริงทุกคำจากประโยคที่ได้รับมา โดยเทียบกับเน็ตเวิร์กที่มีคำนั้นเป็นคำศูนย์กลาง จากนั้นเน็ตเวิร์กจะเอ๊าท์พุตคำศูนย์กลางนั้นหรือคำอื่นซึ่งขึ้นอยู่กับบริบทที่พบ เมื่อได้เอ๊าท์พุต ที่เป็นคำที่ต้องการแล้ว ระดับความมั่นใจของคำนั้นจะถูกคำนวณ เนื่องจากผู้เชี่ยวชาญแต่ละโหนดมีน้ำหนักสำหรับลงคะแนนให้กับคำเป้าหมาย เราจึงสามารถพิจารณาให้น้ำหนักเป็นระดับความมั่นใจของผู้เชี่ยวชาญที่มีต่อคำคำนั้นได้ ดังนั้นระดับความมั่นใจของคำจึงสามารถนิยามให้เป็นน้ำหนักทั้งหมดที่ลงคะแนนให้กับคำคำนั้นหารด้วยน้ำหนักทั้งหมดในเน็ตเวิร์ก จากระดับความมั่นใจของคำทุกคำในประโยค ค่าเฉลี่ยของคำทั้งหมดในประโยคจึงถูกนิยามให้เป็นระดับความมั่นใจของประโยค ประโยคที่ดีที่สุด N ประโยคจะถูกเรียงตามระดับความมั่นใจของประโยคนี้

4.2.5 ผลการทดลอง

เราได้เตรียมคลังข้อความประกอบด้วยประโยคทั้งหมดประมาณ 9,000 ประโยค (140,000 คำ และ 1,300,000 ตัวอักษร) เพื่อประเมินผลวิธีการที่เสนอ คลังข้อความถูกแยกออกเป็น 2 ส่วน ส่วนแรกประมาณ 80 % ใช้เป็นตัวอย่างสอนสำหรับทั้งโมเดลไตรแกรมและ Winnow ส่วนที่เหลือใช้เป็นตัวอย่างทดสอบ ผลที่ได้แสดงในตารางที่ 4 และ ตารางที่ 5

ตารางที่ 4 เปอร์เซนต์ของคำผิดจาก OCR

ประเภท	เปอร์เซนต์
คำผิดที่ไม่เป็นคำ	18.37 %
คำผิดที่เป็นคำ	3.60 %
รวม	21.97 %

ตารางที่ 5 เปอร์เซนต์ของคำผิดที่ถูกแก้ไขได้ถูกต้องหลังจากใช้ไตรแกรมและ Winnow

ประเภท	ไตรแกรม	ไตรแกรม +Winnow
คำผิดที่ไม่เป็นคำ	82.16 %	90.27 %
คำผิดที่เป็นคำ	75.15 %	87.60 %
คำผิดที่เกิดขึ้นใหม่	1.42 %	1.56 %

ตารางที่ 4 แสดงเปอร์เซนต์ของคำผิดในข้อความทั้งหมด ตารางที่ 5 แสดงเปอร์เซนต์ของคำผิดที่ถูกแก้ไขได้ถูกต้องหลังจากใช้แบบจำลองไตรแกรมและ Winnow ผลการทดลองแสดงให้

เห็นว่าแบบจำลองไตรแกรมสามารถแก้ไขคำผิดที่ไม่เป็นคำ และคำผิดที่เป็นคำได้ แต่ก็มีการสร้างคำผิดใหม่ๆเกิดขึ้นจากการที่แก้ไขคำที่ถูกต้องแล้วเป็นคำผิด นอกจากนั้นผลแสดงว่าคำผิดที่เป็นคำแก้ไขยากกว่าคำผิดที่ไม่เป็นคำ เมื่อนำ Winnow เข้ามาช่วยทำให้ข้อผิดพลาดลดลงได้อีก โดยเฉพาะอย่างยิ่งคำผิดที่เป็นคำจะลดลงได้อย่างมาก

4.3 การระบุคำที่ไม่รู้จักในภาษาไทย (Thai Unknown word Identification)

ปัญหาของคำที่ไม่รู้จักเป็นปัญหาสำคัญอย่างหนึ่งในการประมวลผลภาษาธรรมชาติ โดยเฉพาะอย่างยิ่งสำหรับภาษาที่ไม่มีเครื่องหมายแสดงขอบเขตของคำอย่างชัดเจน เช่น ภาษาไทย ภาษาญี่ปุ่น เป็นต้น เพราะว่านอกจากคำนั้นๆจะไม่ถูกรู้จำอย่างถูกต้องแล้ว ยังทำให้คำรอบข้างผิดพลาดไปด้วย ในหัวข้อนี้จะกล่าวถึงการนำคลังข้อความที่กำกับหมวดคำแล้ว มาช่วยให้การระบุคำที่ไม่รู้จัก

4.3.1 ปัญหาของคำที่ไม่รู้จัก

คำที่ไม่รู้จักสามารถแบ่งเป็น 2 กลุ่มใหญ่ๆ คือ คำที่ไม่รู้จักแบบชัดเจน และคำที่ไม่รู้จักแบบซ่อน

(1) คำที่ไม่รู้จักแบบชัดเจน

คำที่ไม่รู้จักแบบชัดเจนหมายถึงคำที่ไม่รู้จักที่สายอักขระย่อยของมันทุกตัวไม่อยู่ในพจนานุกรมเลย ตัวอย่างของคำที่ไม่รู้จักแบบชัดเจน มีดังต่อไปนี้

กทม.

โลตัส

สุนีย์

(2) คำที่ไม่รู้จักแบบซ่อน

คำที่ไม่รู้จักแบบซ่อนสามารถแบ่งออกได้เป็น 2 กลุ่มย่อย คือ

(2.1) คำที่ไม่รู้จักแบบซ่อนบางส่วน

คำที่ไม่รู้จักแบบซ่อนหมายถึงคำที่ไม่รู้จักซึ่งสายอักขระย่อยของมันบางตัวมีอยู่ในพจนานุกรม ตัวอย่างเช่น

สุมานี ประกอบด้วย สุ มา นี

คราพงศ์ ประกอบด้วย ครา พงศ์

ไมโครซอฟต์ ประกอบด้วย ไม โค ร ซอ ฟต์

โดยที่ตัวอักษรเข้มแสดงถึงคำที่มีอยู่ในพจนานุกรม

(2.2) คำที่ไม่รู้จักแบบซ่อนทั้งหมด

คำที่ไม่รู้จักแบบซ่อนหมายถึงคำที่ไม่รู้จักซึ่งสายอักขระย่อยของมันทุกตัวมีอยู่ในพจนานุกรม ตัวอย่างเช่น

สมชาย ประกอบด้วย สม ชาย

กนกพร ประกอบด้วย กนก พร

หัวข้อต่อไปแสดงวิธีการสร้างตัวเลือกสำหรับคำที่ไม่รู้จักทั้งสองแบบ

4.3.2 การสร้างตัวเลือกสำหรับคำที่ไม่รู้จัก

เราเสนอวิธีการ 2 วิธีสำหรับสร้างตัวเลือกของคำที่ไม่รู้จักทั้งสองแบบ หลังจากที่ตัวเลือกถูกสร้างแล้ว ตัวเลือกที่ดีที่สุดจะถูกเลือกโดยอัลกอริทึม Winnow ที่อธิบายไว้แล้วในหัวข้อที่ 4.1

4.3.2.1 วิธีการจัดการกับคำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนบางส่วน

ในกรณีที่สายอักขระไม่อยู่ในพจนานุกรม ตัวเลือกของคำที่ไม่รู้จักจะถูกสร้างโดยการรวมคำที่อยู่ข้างๆคำที่ไม่รู้จักนั้นเข้ากับคำที่ไม่รู้จัก เพื่อสร้างตัวเลือก การรวมคำทุกแบบที่เป็นไปได้ภายในบริเวณ +/- k คำรอบๆคำที่ไม่รู้จักจะถูกใช้สำหรับสร้างตัวเลือก เราสามารถเขียนสมการสำหรับการสร้างตัวเลือกได้ตามรูปที่ 3

ประโยค = $w_1 w_2 \ldots w_a U w_b \ldots w_n$

โดยที่ $w_i \in$ พจนานุกรม, $U \notin$ พจนานุกรม

n = จำนวนคำในประโยค

$UNK = \{\alpha U \beta \mid \alpha \in A, \beta \in B\}$

โดยที่ UNK = เซตของตัวเลือกของคำที่ไม่รู้จัก

$A = \{w_{a-i,a}, i \in [0,K]\} \cup \{\varepsilon\}$

$B = \{w_{b,b+j}, i \in [0,K]\} \cup \{\varepsilon\}$

$w_{i,j} = w_i \ldots w_j : i<j$

ε = null, K = ค่าคงที่

รูปที่ 3 สมการสำหรับสร้างตัวเลือกของคำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนบางส่วน

4.3.2.2 วิธีการจัดการกับคำที่ไม่รู้จักแบบซ่อนทั้งหมด

ในกรณีที่คำที่ไม่รู้จักซึ่งสายอักขระย่อยทุกตัวปรากฏอยู่ในพจนานุกรม การตรวจหาคำแบบนี้จึงทำได้ยากกว่า สมมติให้ ประโยค = $w_1 w_2 \ldots w_n$ เป็นประโยคอินพุต w_i เป็นคำในประโยคนั้น และ t_i เป็นหมวดคำของคำ w_i คำที่จะถูกเลือกเป็นตัวเลือกของคำที่ไม่รู้จักคือ

- คำที่มีค่า $P(w_i|t_i)$ น้อยกว่าค่าขีดแบ่ง หรือ
- คำที่มีค่า $P(t_i|t_{i-1}t_{i-2})$ น้อยกว่าค่าขีดแบ่ง

ในกรณีที่ $P(w_i|t_i)$ น้อยกว่าค่าขีดแบ่ง w_i จะถูกพิจารณาให้เป็นคำที่ไม่รู้จัก ในกรณีที่ $P(t_i|t_{i-1}t_{i-2})$ ของ w_i น้อยกว่าค่าขีดแบ่ง เราต้องพิจารณา w_i รวมทั้ง w_{i-1} และ w_{i-2} ให้เป็นคำที่ไม่รู้จักด้วย เพราะว่าค่าความน่าจะเป็นที่มีค่าน้อยของ w_i อาจจะเป็นผลมาจากของ w_{i-1} หรือ w_{i-2} รูปที่ 4 แสดงสมการสำหรับสร้างตัวเลือกของคำที่ไม่รู้จักแบบซ่อนทั้งหมด

ประโยค = $w_1 w_2 \ldots w_a \ldots w_n$

โดยที่ $w_i \in$ พจนานุกรม

n = จำนวนคำในประโยค

w_a เป็นคำที่มีค่าความน่าจะเป็นน้อยกว่าค่าขีดแบ่ง

$UNK = \{\alpha w \beta \mid \alpha \in A, \beta \in B\}$

โดยที่ UNK = เซตของตัวเลือกของคำที่ไม่รู้จัก

$A = \{w_{a-i,a-1}, i \in [0,K]\} \cup \{\varepsilon\}$

$B = \{w_{a+1,a+i}, i \in [0,K]\} \cup \{\varepsilon\}$

$w_{ij} = w_i \ldots w_j : i<j$

$W = w_a : P(w_a \mid t_a) <$ ค่าขีดแบ่ง หรือ

$W \in \{w_{a-2}, w_{a-1}, w_a\} : P(t_a|t_{a-1},t_{a-2}) <$ ค่าขีดแบ่ง

ε = null, K = ค่าคงที่

รูปที่ 4 สมการสำหรับสร้างตัวเลือกของคำที่ไม่รู้จักแบบซ่อนทั้งหมด

4.3.3 การสร้างตัวอย่างที่ใช้สอน Winnow

ในการทดลองของเรา เราเลือกประโยคทั้งหมดที่ประกอบด้วยชื่อเฉพาะและกำหนดให้ชื่อเฉพาะเหล่านี้เป็นคำที่ไม่รู้จัก บริบทรอบๆชื่อเฉพาะถูกใช้สำหรับสร้างคุณสมบัติในการระบุว่าชื่อเฉพาะเหล่านี้คือคำที่ไม่รู้จัก ในทำนองเดียวกัน เราสร้างตัวอย่างสำหรับคำอื่นๆที่เป็นคำที่รู้จักซึ่งไม่ใช่ชื่อเฉพาะ คุณสมบัติที่เราใช้ประกอบด้วยคำบริบทและคอลโลเคชัน หลังจากที่เน็ตเวิร์กของ Winnow ถูกเรียนรู้แล้ว เน็ตเวิร์กจะถูกใช้สำหรับจัดลำดับให้กับตัวเลือก เพื่อกำหนดตัวเลือกที่มีคะแนนสูงสุดเป็นคำตอบ

อัลกอริทึมในการระบุคำที่ไม่รู้จักประกอบด้วย 4 ขั้นตอนดังต่อไปนี้

(1) ตัดคำ

จากประโยคอินพุตที่รับเข้ามา เราใช้โมเดลไตรแกรมสำหรับแยกประโยคเป็นคำๆและให้ค่าหมวดคำ จากนั้นประโยคที่แบ่งเป็นคำแล้วที่ดีที่สุด N ประโยคจะถูกเลือกให้เป็นตัวเลือก

ตัวอย่างเช่นให้ประโยค = "ฉันไปเที่ยวน้ำตกทีลอซูกับเพื่อน" ผลที่ได้จากอัลกอริทึมตัดคำเป็นดังต่อไปนี้:

I. ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ที/t_5 ลอซู/t_6 กับ/t_7 เพื่อน/t_8

II. ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำ/t_4 ตก/t_5 ที/t_6 ลอซู/t_7 กับ/t_8 เพื่อน/t_9

โดยที่ w_i/t_i คือ คำ/หมวดคำ ลอซูเป็นคำที่ไม่รู้จัก

(2) สร้างตัวเลือกของคำที่ไม่รู้จัก

จากผลที่ได้จากขั้นตอนที่ (1) เราสร้างตัวเลือกของคำที่ไม่รู้จัก โดยใช้วิธีการจัดการคำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนที่อธิบายในหัวข้อ 4.3.2.1 ด้านบน ตัวอย่างเช่น จากประโยค I ของขั้นตอนที่ (1) เราพบว่าคำที่ 7 "ลอซู" เป็นคำที่ไม่รู้จัก ดังนั้นเราจึงใช้วิธีสำหรับจัดการกับคำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนบางส่วน และได้ตัวเลือกของคำที่ไม่รู้จักคือ "ลอซู" , "ทีลอซู", "น้ำตกทีลอซู", "ลอซูกับ", "ลอซูกับเพื่อน", "ทีลอซูกับ", "ทีลอซูกับเพื่อน", "น้ำตกทีลอซูกับ" และ "น้ำตกทีลอซูกับเพื่อน" โดยที่ K, U, A และ B ในรูปที่ 3 คือ 2, "ลอซู", {ε, ที, น้ำตกที} และ {ε, กับ, กับเพื่อน} ตามลำดับ ทุกประโยคจะถูกจัดการโดยวิธีการเดียวกัน ทำให้เราได้ตัวเลือกของคำที่ไม่รู้จักทั้งหมด

(3) กำกับหมวดคำ

ประโยคใหม่จะถูกสร้างขึ้นโดยการรวมตัวเลือกจากขั้นตอนที่ (2) เข้ากับคำที่เหลือของประโยคเดิมนั้น หมวดคำจะถูกกำหนดโดยแบบจำลองไตรแกรม คำที่ไม่รู้จักจะถูกสมมติให้เป็นชื่อเฉพาะ ตัวอย่างเช่น ประโยค I ในขั้นตอนที่ (1) หลังจากที่ถูกสร้างตัวเลือกโดยขั้นตอนที่ (2) แล้ว ประโยคใหม่ที่ได้เป็นดังต่อไปนี้

III ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ที/t_5 ลอซู/NPRP กับ/t_7 เพื่อน/t_8

IV ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ทีลอซู/NPRP กับ/t_7 เพื่อน/t_8

V ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตกทีลอซู/NPRP กับ/t_7 เพื่อน/t_8

VI ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ที/t_5 ลอซูกับ/NPRP เพื่อน/t_8

VII ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ที/t_5 ลอซูกับเพื่อน/NPRP

VIII ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ทีลอซูกับ/NPRP เพื่อน/t_8

IX ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ทีลอซูกับเพื่อน/NPRP

X ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตกทีลอซูกับ/NPRP เพื่อน/t_8

X ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตกทีลอซูกับเพื่อน/NPRP

โดยที่ t_i คือหมวดคำ และ NRPR เป็นหมวดคำชื่อเฉพาะ ประโยคที่ II ก็จะถูกจัดการในทำนองเดียวกัน

(4) ใช้ Winnow เลือกประโยคที่ดีที่สุด

ประโยคที่ได้จากขั้นตอนที่ (3) จะถูกส่งให้กับ Winnow เพื่อเลือกประโยคที่ดีที่สุด ในตัวอย่างของเราประโยคที่ IV จะถูกเลือกเป็นคำตอบ:

IV ฉัน/t_1 ไป/t_2 เที่ยว/t_3 น้ำตก/t_4 ทีลอซู/NPRP กับ/t_7 เพื่อน/t_8

4.3.4 ผลการทดลอง

เราใช้ 5,000 ประโยคจากคลังข้อความ โดยแบ่งเป็น 2 ส่วน ส่วนแรกประมาณ 80 % ใช้เป็นตัวอย่างสอน และส่วนที่เหลือใช้เป็นตัวอย่างทดสอบ เราแยกคำที่ไม่รู้จักออกเป็น 2 ประเภท คือ (1) คำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนบางส่วน และ (2) คำที่ไม่รู้จักแบบซ่อนทั้งหมด ผลการทดลองแสดงในตารางที่ 6

ตารางที่ 6 ผลการทดลองของการตรวจหาและเลือกคำที่ไม่รู้จัก

	คำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนบางส่วน		คำที่ไม่รู้จักแบบซ่อนทั้งหมด	
	ตัวอย่างสอน	ตัวอย่างทดสอบ	ตัวอย่างสอน	ตัวอย่างทดสอบ
จำนวนคำ	482	145	578	230
ตรวจพบ	100.00 %	100.00 %	87.82 %	83.25%
เลือกถูกต้องโดย Winnow	95.26 %	92.75 %	80.68 %	74.26 %

ผลการทดลองแสดงให้เห็นว่าคำที่ไม่รู้จักแบบชัดเจนและแบบซ่อนบางส่วนสามารถตรวจพบได้ทั้งหมด แต่คำที่ไม่รู้จักแบบซ่อนทั้งหมดตรวจพบได้ 83.25% และ Winnow เป็นอัลกอริทึมที่มีประสิทธิภาพในการเลือกตัวเลือกได้เป็นอย่างดี

5.สรุป

งานวิจัยนี้ได้กล่าวถึงการใช้แบบจำลองไตรแกรมในการตัดคำและกำกับหมวดคำให้กับคลังข้อความภาษาไทย ผลการทดลองแสดงให้เห็นว่าแบบจำลองไตรแกรมสามารถตัดคำได้ความถูกต้องสูงกว่าวิธีการเดิมที่ใช้อยู่ และสามารถกำกับหมวดคำได้ถูกต้องด้วยเปอร์เซนต์ที่สูง ซึ่งช่วยให้การสร้างคลังข้อความที่มีหมวดคำกับกับทำได้สะดวกและมีประสิทธิภาพยิ่งขึ้น นอกจากนั้น งานวิจัยนี้ยังได้แสดงถึงการนำคลังข้อความที่มีหมวดคำกำกับมาใช้ประโยชน์ในงาน 2 อย่าง คือ (1) การแก้ไขคำผิดจาก OCR ภาษาไทย และ (2) การระบุคำที่ไม่รู้จักในประโยคภาษาไทย โดยได้นำอัลกอริทึมเรียนรู้ Winnow เข้ามาช่วยในการดึงคุณสมบัติที่มีประโยชน์ในการแก้ไขคำผิดและระบุคำที่ไม่รู้จัก ผลการทดลองในงานทั้งสองแสดงให้เห็นว่า การใช้คลังข้อความที่มีหมวดคำกำกับร่วมกับแบบจำลองไตรแกรมและอัลกอริทึม Winnow ทำให้การแก้ไขคำผิดจาก OCR ภาษาไทยได้เปอร์เซนต์ความถูกต้องสูง และระบุคำที่ไม่รู้จักได้อย่างมีประสิทธิภาพ

ผลที่ได้จากการวิจัยคือเราได้วิธีการตัดคำและวิธีการกำกับหมวดคำที่มีประสิทธิภาพและจากวิธีการกำกับหมวดคำที่ได้ ทำให้สามารถนำไปช่วยในการกำกับหมวดคำให้กับคลังข้อความที่มีอยู่ได้ นอกจากนั้นส่วนหนึ่งของงานวิจัยนี้ได้นำเสนอในการประชุมวิชาการนานาชาติ 17th International Conference on Computational Linguistics (COLING-ACL'98) (S.Meknavin, et.al., 1998) และ1998 IEEE Asia-Pacific Conference on Circuits and Systems: Microelectronic and Integration System (P.Charoenpornsawat, et.al, 1998)

บรรณานุกรม

B.G.Barbara & M.R.Gerald (1971) *Automated grammatical tagging of English*, Dept. of Linguistics, Brown University

A.Blum (1997) Empirial Support for Winnow and Weighted-Majority Algorithm: Results on a Calendar Scheduling Domain, *Machine Learning*, 26:5-23

E.Brill (1993) Automatic Grammar Induction and Parsing Free Text : A Transformation-Based Approach, *In Proc. of ACL-93*

E.Charniak, C.Hendrickson, N.Jacobson & M.Perkowitz (1993) Equations for part-of-speech Tagging, *In Proc. of the Eleventh National Conference on Artificial Intelligence*

T.Charoenporn, V.Sornlertlamvanich & H.Isahara (1997) Building A Large Thai Text Corpus - Part-Of-Speech Tagged Corpus: ORCHID -, *In Proc. of the National Language Processing Pacific Rim Symposium 1997*, pp.509-512

P.Charoenpornsawat, B.Kijsirikul & S.Meknavin (1998) Feature-base Thai Unknown Word Boundary Identification Using Winnow, *In Proc. Of 1998 IEEE Asia-Pacific Conference on Circuits and Systems: Microelectronics and Integration Systems*, to be appeared

W.Church (1988) A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text, *In Second Conference on Applied Natural Language Processing*. ACL, pp.136-143

W.Francis & H.Kucera (1982) *Frequency Analysis of English Usage*, Houghton Mifflin

A.R.Golding (1995) A Bayesian Hybrid Method for Context-sensitive Spelling Correction. *In Proc. of the Third Workshop on Very Large Corpora*

A.R.Golding & D. Roth (1996) Applying Winnow to Context-Sensitive Spelling Correction. *In Lorenza Saitta, editor, Machine Learning:Procs. Of the 13th International Conference*, Bari, Italy

A.R.Golding & Y. Schabes (1996) Combining Trigram-based and Featured-based Methods for Context-Sensitive Spelling Correction, *Technical Report TR-93-03a*, Mitsubishi Electric Research Laboratory

A.Kawtrakul, C.Thumkanon, Y.Poovorawan, P.Varasrai & M.Suktarachan (1997) Automatic Thai Unknown Word Recognition, *In Procs. Of the Natural Language Processing Pacific Rim Symposium 1997*, pp. 341-346

A.Kawtrakul, C.Thumkanon & S.Seriburi (1995) A Statistical Approach to Thai Word Filtering, *In Proc. of SNLP'95*, pp. 398-406

N.Littlestone (1998) Learning Quickly when Irrelevant Attributes abound: A New Linear-Threshold Algorithm, *Machine Learning, 2*

M.P.Marcus, B.Santorini & M.A.Marcinkiewicz (1993) Building a Large Annotated Corpus of English: The Penn Treebank, *Computational Linguistics*, Vol.19 No.2, pp.313-330

S.Meknavin, B.Kijsirikul, A.Chotimongkol & C.Nuttee (1998) Combining Trigram and Winnow in Thai OCR Error Correction, In Proc. Of 17th International Conference Computational Linguistics, pp.836-842

M.Nagata (1996) Context-Based Spelling Correction for Japanese OCR. *In Proc. of COLING-96.* pp.806-811

K.Oflazer (1996) Error-Tolerant Finite-State Recognition with Applications to Morphological Analysis and Spelling Correction, *Computational Linguistics*, Vol. 22, No. 1, pp.73-89

V.Wuwongse & A.Pornprasertsakul (1994) Ambiguity Resolution in Thai Sentence Analysis Using Least Exception Logic, *In Proc. of National Computer Symposium*, pp.100-116

T. Xiang & A. E. David (1996) A Statistical Approach to Automatic OCR Error Correction in Context, *In Proc. of the Fourth Workshop on Very Large Corpora*, E. Everhed & I. Dagan, ed., Copenhagen, Denmark, pp. 88-99

ดวงแก้ว สวามิภักดิ์ (2533) การสร้างซอฟต์แวร์วิเคราะห์ไวยากรณ์ไทยภายใต้ระบบยูนิกซ์, สถาบันไทยคดีศึกษา ธรรมศาสตร์

ยืน ภู่วรวรรณ และ วิวรรธน์ อึ่มอารมณ์ (2529) การแบ่งแยกพยางค์ไทยด้วยดิกชันนารี, *รายงานการประชุมวิชาการทางไฟฟ้า #9*

วิรัช ศรเลิศล้ำวาณิชย์ (2536) การตัดคำไทยในระบบแปลภาษา, *การแปลภาษาด้วยคอมพิวเตอร์*, หน้า 50-55

สุรศักดิ์ สงวนพงษ์, สมนึก คีรีโต และ ยืน ภู่วรวรรณ (2538) การตัดพยางค์คำไทยด้วยการใช้โครงสร้างข้อมูลแบบต้นไม้, *รายงานการประชุมวิชาการทางไฟฟ้า #8*

สุรินทร์ จรรยาพรพงษ์ (2526) *A Thai syllable separation algorithm*, master thesis AIT