บทที่ 4

การออกแบบวิธีการแก้คำผิดอัตโนมัติสำหรับหุ่นยนต์สนทนา

เนื่องจากลักษณะการเขียนในภาษาไทยนั้นไม่มีการใช้สัญลักษณ์หรือตัวอักษรมาคั่นระหว่างคำ ทว่างานในด้านการประมวลผลภาษาธรรมชาตินั้นจำเป็นต้องทราบขอบเขตของคำก่อน จึงจะสามารถนำไปประมวลผลต่อไปได้ ทำให้การค้นหาความผิดพลาดและการแก้ไขความผิดพลาดของคำไม่สามารถทำได้ถ้าหากไม่มีการตัดคำเสียก่อน

ในงานวิจัยนี้ จะนำระบบตัดคำที่มีใช้กันอยู่แล้วมาใช้งาน โดยจะใช้อัลกอริทึมของ SWATH [5] ในการทำการตัดคำ หลังจากตัดคำแล้วจะนำอัลกอริทึมของ Wordcut [8] ในการรวมคำหากได้ผลของการตัดคำออกมาเป็นอักขระเดี่ยว มารวมกับอัลกอริทึมที่พัฒนาขึ้นเอง [18] เพื่อแก้ปัญหาในเรื่องของการแก้คำผิด เพราะในการสื่อสารโดยการพิมพ์ข้อความนั้น อินพุตที่รับเข้ามาอาจมีความผิดพลาด ซึ่งเป็นความผิดพลาดจากการพิมพ์ เช่น พิมพ์เกิน พิมพ์ตก พิมพ์ผิด และพิมพ์สลับ ซึ่งตัว AIML Engine นั้น ไม่สามารถแก้ข้อผิดพลาดนี้ได้

ดังนั้นในงานวิจัยนี้ มีส่วนที่ต้องออกแบบพัฒนาขึ้นเองคือ ส่วนการแก้ไขคำผิดแบบไม่ตั้งใจโดยอัตโนมัติ *ซึ่งจะทำเฉพาะ 3 กรณีที่มีระยะแก้ไขเป็น 1 คือ พิมพ์แทนที่ พิมพ์เกิน และพิมพ์สลับ* (จะไม่ทำกรณีพิมพ์ตก เนื่องจากต้องใช้โครงสร้างข้อมูลทรัยที่ต้องการหน่วยความจำในการทดลองสูง) โดย 3 กรณีนี้คิดเป็น 76.11 เปอร์เซ็นต์ของคำผิดทั้งหมด หรือคิดเป็น 81.37 เปอร์เซ็นต์ของคำผิดแบบไม่ตั้งใจ ซึ่งน่าจะเพียงพอต่อการนำไปใช้แก้คำผิดกับหุ่นยนต์สนทนาได้

หลังจากนั้นจะทำการรวมโปรแกรมตัดคำและส่วนแก้ไขคำผิด เข้ากับส่วนตัวแปลภาษา (Interpreter) โดยที่ส่วนตัวแปลภาษานั้นจะนำมาจากซอร์สโค้ดที่เผยแพร่ใน [2] แต่จะทำการปรับปรุงเล็กน้อยเพื่อให้ทำงานกับเอกสารเอไอเอ็มแอลภาษาไทยได้

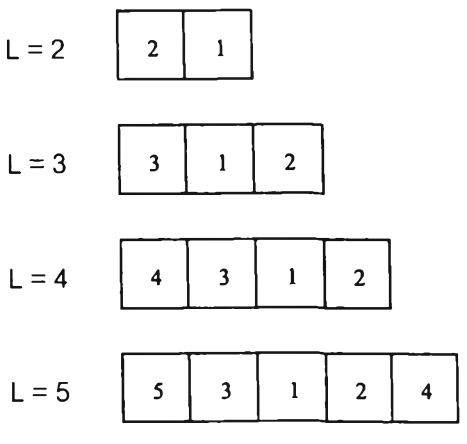
รายละเอียดของแนวคิดสำหรับพัฒนาโปรแกรมย่อยในแต่ละส่วน ดังต่อไปนี้

4.1 วิธีการแก้ไขคำผิด

ในงานวิจัยนี้ จะนำระบบตัดคำที่มีใช้กันอยู่แล้วมาใช้งาน โดยจะใช้อัลกอริทึมของ SWATH [5] ในการทำการตัดคำ ถึงแม้ว่าโปรแกรมตัดคำ SWATH จะมีสมมติฐานว่าประโยคที่จะนำมาตัดคำต้องถูกต้อง แต่จากการทดลองกับประโยคที่มีการพิมพ์ผิดแบบไม่ตั้งใจ พบได้ว่าส่วนใหญ่แล้วจะได้ผลลัพธ์ในการตัดคำของคำที่ผิดนั้นออกมาเป็นอักขระเดี่ยวๆ เช่น "แบร" จะตัดได้เป็น "แบ-ร" ซึ่งหากได้คำที่ตัดออกมาแบบนี้เราก็สามารถแก้ไขได้โดยการนำอักขระเดี่ยวที่ไม่เป็นคำนี้ไปรวมกับคำข้างเคียงในภายหลังจากตัดคำ (คล้ายดังเช่นการรวมส่วนที่ไม่เป็นคำของอัลกอริทึม [8]) ทำให้คาดว่าแม้สมมติฐานจะต่างกัน แต่ก็น่าจะสามารถทำงานร่วมกันได้

หลังจากตัดคำด้วยโปรแกรม SWATH แล้ว จะนำวิธีการของ [8], [18] มาประยุกต์ใช้ร่วมกันในการแก้คำผิด โดยเมื่อรวมอักขระเดี่ยวกับคำข้างเคียงและนำไปทดลองแก้คำผิดแล้วยังไม่ได้ผลลัพธ์ซึ่งเป็นคำที่มีความหมายในพจนานุกรมออ กมา ก็จะไม่ทำการรวมอักขระเดี่ยวนั้นกับคำข้างเคียงให้ปล่อยไว้เป็นอักขระเดี่ยวเช่นเดิม

จากผลการทดลองในข้อ 3.2 จะเห็นได้ว่าตำแหน่งของอักขระที่เป็นสาเหตุของผิดนั้นจะอยู่ที่ตำแหน่งกึ่งกลางของคำแต่ค่อนไปทางด้านหลังเล็กน้อย หรือกล่าวได้ว่าอยู่ที่ตำแหน่งเฉลี่ย 58.36 เปอร์เซ็นต์ของความยาวคำ โดยนำค่าที่ได้นี้มาทำการออกแบบอัลกอริทึมในการแก้ไขคำผิด ดังนั้นในงานวิจัยนี้จึงออกแบบให้เริ่มแก้ไขคำผิดที่ตำแหน่งกึ่งกลางของคำก่อนแล้วเลื่อนไปทางขวาแล้วค่อยเลื่อนกลับมาทางซ้าย ในลักษณะของการแพร่ออกจากตรงกลาง ดังตัวอย่างในรูป 4.1 (โดยตัวเลขที่อยู่ในแต่ละช่องสี่เหลี่ยม คือ ลำดับความสำคัญในการแก้ไขคำผิด)



รูปที่ 4.1 ตัวอย่างลำดับตำแหน่งในการแก้ไขคำผิด

จากรูปที่ 4.1 จะเห็นได้ว่าจะมีการทำงานในลักษณะของการแพร่กระจายออกจากตำแหน่งตรงกลาง (การแพร่ คือการเริ่มกระจายตัวจากตรงกลางออกไปด้านข้างทั้งขวาและซ้ายพร้อมๆ กัน เหมือนกับลักษณะการกระจายของคลื่นน้ำเมื่อเราหย่อนก้อนหินลงไปในน้ำ) ซึ่งในงานวิจัยนี้จะใช้การเริ่มแก้ไขคำผิดจากตรงกลางก่อนกับการแก้ไขคำผิดทั้ง 3 กรณี คือ การพิมพ์แทนที่ เกิน และสลับ โดยจะกล่าวรายละเอียดของการแก้ไขคำผิดแยกเป็นแต่ละกรณีในภายหลัง

จากงานวิจัยของ Grudin [11] ได้ศึกษารูปแบบของการพิมพ์ผิดพบว่า ความผิดพลาดจากการพิมพ์แทนที่นั้นมักเกิดจากคีย์ที่อยู่ติดกัน และคำผิดที่เกิดจากลักษณะการแทนที่ 58 เปอร์เซ็นต์เป็นผลมาจากแป้นประชิดบนแผงแป้นอักขระ

ในงานวิจัยนี้จะนำเสนอ องค์ประกอบที่จะช่วยแก้ไขคำผิดแบบแทนที่ โดยใช้ตัวอักษรประชิดบนแผงแป้นอักขระมาลองแทนที่อักขระในตำแหน่งที่คาดว่าจะเป็นตัวที่เป็นสาเหตุให้คำนั้นผิด ตามแนวคิดของ Grudin [11] ซึ่งลองทำการทดลองได้ผลเป็นบทความ [18] แต่ว่าในบทความ

นั้นทำการทดสอบกับคำที่พิมพ์ผิดในตำแหน่งสุดท้ายของคำเท่านั้น โดยในบทความมีสมมติฐานว่าตำแหน่งที่ทำให้คำนั้นเกิดความผิดพลาด คือ ตำแหน่งสุดท้ายของคำ แต่เมื่อได้มาทำการหารูปแบบตำแหน่งอักขระของคำผิดในภาษาไทย (ดังในหัวข้อที่ 3.2) แล้วพบว่า ในภาษาไทยมักผิดที่ตำแหน่งกึ่งกลางของคำแต่ค่อนไปทางท้ายเล็กน้อย

ดังนั้นจึงควรแก้คำผิดในลักษณะที่เริ่มแพร่ออกจากกึ่งกลางไปด้านข้างจะเหมาะสมกว่า โดยจะทำเช่นนี้กับการแก้ไขคำผิดทุกกรณี เนื่องจากทุกกรณีก็ได้ตำแหน่งออกมาว่ามักผิดตรงกลางเช่นเดียวกัน และ*วิธีการที่ออกแบบในงานวิจัยนี้จะเน้นทำการแก้ไขเฉพาะคำผิดแบบไม่ตั้งใจที่มีระยะแก้ไขเป็น 1 เท่านั้น* ซึ่งสามารถนำมาออกแบบเป็นอัลกอริทึมในการแก้ไขคำผิดอัตโนมัติสำหรับหุ่นยนต์สนทนา ได้ดังรูป 4.2

ถ้าผลการตัดคำพบว่าในประโยคมีคำที่ไม่มีความหมายในพจนานุกรม จะตั้งสมมติฐานว่าคำนั้นเป็นคำผิด แล้วหากเจอคำที่มีอักขระเพียงตัวเดียวให้เริ่มทำจากคำนั้นก่อน โดยคำผิดสามารถแยกได้เป็น 2 กรณี คือ

1. **คำผิดมี 1 ตัวอักษร** จะมีวิธีการในการแก้คำผิดดังนี้

1.1) ให้นำอักขระเดี่ยวไปรวมกับคำข้างเคียงที่อยู่ข้างหน้าก่อน (หากไม่มีคำข้างหน้าจึงจะนำไปรวมกับคำที่อยู่ข้างหลัง)

1.2) ทำการแก้ไขคำผิดโดยเริ่มจากตำแหน่งกลางของคำแล้วเลื่อนมาทางขวาแล้วไปทางซ้ายในลักษณะของการแพร่ออกจากตรงกลางไปด้านข้าง ถ้าได้ผลลัพธ์เป็นคำถูก (คำที่มีความหมายในพจนานุกรม) ให้ *เก็บไว้ทุกคำตามลำดับที่ได้* ซึ่งมีลำดับในการลองแก้ไขคำผิด (ตามสถิติที่ได้มาจากการทดลองในตารางที่ 3.2) ดังนี้

- ทำการลอง ***แทนที่*** ตัวอักษรในคำผิด โดยอาศัยคุณสมบัติของตัวอักษรประชิดบนแผงแป้นอักขระ [18]
- ทำการลอง ***ลบ*** ตัวอักษรในคำผิด (โดยเริ่มจากตรงกลางของคำแพร่ออกไปด้านข้าง เช่นกัน)
- ทำการลอง ***สลับที่*** ตัวอักษรที่อยู่ติดกัน (โดยเริ่มจากตรงกลางของคำแพร่ออกไปด้านข้าง เช่นกัน)

1.3) ถ้าทำการลองแก้คำผิดดังในข้อ 1.2 แล้วยังไม่ได้คำถูก ให้แยกอักขระนั้นออกมาเดี่ยวๆเหมือนเดิมก่อน แล้วจึงนำอักขระเดี่ยวนี้ไปลองรวมกับคำข้างเคียงที่อยู่ด้านหลังแล้วลองแก้คำผิดเช่นเดิมดังข้อ 1.2

- ถ้ายังไม่ได้อีก ให้รวมทั้ง 3 ส่วน (คำข้างเคียงทางซ้าย อักขระเดี่ยว และคำข้างเคียงทางขวา)ให้เป็นคำเดียวกัน แล้วลองแก้คำผิดเช่นเดิมดังข้อ 1.2
- ถ้าลองรวมอักขระเดี่ยวกับคำข้างเคียงที่อยู่ติดกันจนครบทุกกรณีแล้ว แต่ยังไม่ได้คำถูกออกมาเลย ให้แยกอักขระนั้นไว้เดี่ยวๆ ดังเดิม (ซึ่งแสดงว่าไม่สามารถแก้คำผิดได้)

2. **คำผิดมี 2 ตัวอักษร หรือมากกว่า** จะมีวิธีการในการแก้คำผิด เหมือนดังข้อ 1.2 ทุกประการ

เมื่อแก้ไขคำผิดแล้ว ให้นำประโยคนั้นมารวมกันใหม่แล้วค่อยส่งไปตัดคำอีกครั้งหนึ่ง

รูปที่ 4.2 อัลกอริทึมในการแก้ไขคำผิด

4.1.1 **การแทนที่**

ในงานวิทยานิพนธ์นี้ นำเสนอวิธีการแทนที่ตัว อักขระในคำผิดโดยอาศัยคุณสมบัติของตัวอักษรประชิดบนแผงแป้นอักขระ มาลองแทนที่ตำแหน่งที่คาดว่าจะเป็นต้นเหตุให้คำๆ นั้นผิดพลาด มีขั้นตอนย่อย 2 ขั้นตอนหลัก ดังนี้

4.1.1.1 การหาส่วนที่ไม่เป็นคำ

นำคำทุกคำที่ได้จากขั้นตอนการตัดคำ มาเทียบกับคำในพจนานุกรม โดยในงานนี้ได้ใช้พจนานุกรมของราชบัณฑิตยสถานที่ได้มาจาก NECTEC [17] ซึ่งพจนานุกรมได้สร้างขึ้นเมื่อปีคริสตศักราช 1997 และมีคำศัพท์ในพจนานุกรม 32,895 คำ

ถ้าหากคำไหนที่เทียบกับพจนานุกรมแล้วไม่พบจะคาดว่าคำนั้นเป็นคำผิด แล้วให้ทำเครื่องหมายไว้เพื่อรอการแก้ไขคำผิดในภายหลัง

4.1.1.2 การสร้างรายการคำใกล้เคียงคำผิด

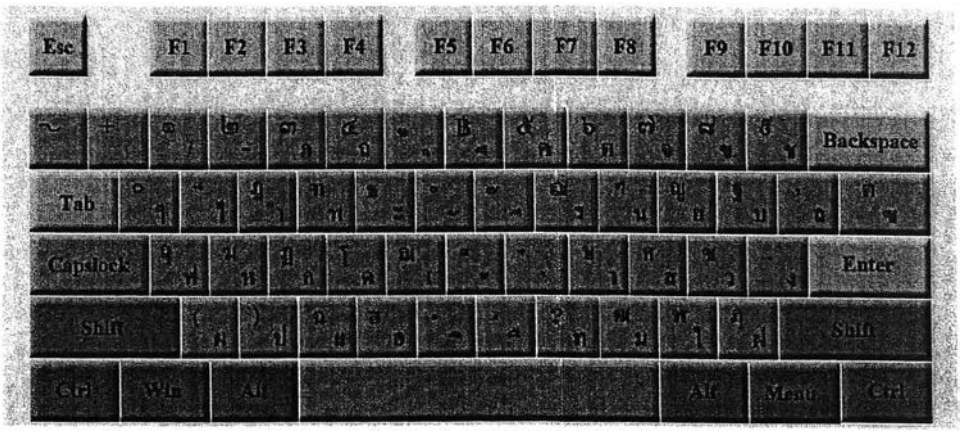
ในขั้นตอนนี้จะนำเสนอ การสร้างรายการคำผิดโดยใช้ตัวอักษรประชิดบนแผงแป้นอักขระมาลองแทนที่อักขระในตำแหน่งที่คาดว่าจะเป็นตัวที่เป็นสาเหตุให้คำนั้นผิด ซึ่งวิธีการจะจัดอยู่ในกลุ่มการแก้คำผิดโดยไม่คำนึงถึงสภาพรอบข้าง ไม่ใช้คำที่อยู่ใกล้เคียงมาช่วยในการพิจารณาเลย สามารถแบ่งขั้นตอนย่อยในการสร้างรายการ ได้ดังนี้

1) สร้างเซตอักขระใกล้เคียงของแต่ละตัวอักษร

รูปที่ 4.3 ผังอักขระที่อยู่ประชิดตัวอักษร ย

จากรูปที่ 4.3 จะเห็นได้ว่าตัวอักษร ย มีเซตของอักขระที่อยู่ประชิดทั้งหมด 13 ตัวอักษร คือ {จ, ข, น, บ, ส, ว, ๗, ๘, ฯ, ญ, ฐ, ศ, ช}

เซตของอักขระที่อยู่ประชิดแป้นของตัวอักษรตัวหนึ่งๆ นั้นจะสามารถมีสมาชิกได้ไม่เกิน 13 ตัว และถ้าหากแป้นข้างเคียงนั้นเป็นแป้นที่ไม่ใช่อักขระภาษาไทย เช่น แป้นป้อนเข้า (Enter Key) และแป้นเปลี่ยนชุดอักษร (Shift Key) ไม่จำเป็นต้องเก็บเป็นสมาชิกในเซตนั้น



รูปที่ 4.4 การวางผังแผงแป้นอักขระไทยแบบเกษมณี

ใช้หลักการนี้สร้างเซตอักขระใกล้เคียงของอักขระภาษาไทยทุกตัวบนแป้นพิมพ์ดังในรูปที่ 4.4 ซึ่งมีทั้งหมด 92 รูปอักขระด้วยกัน โดยให้สร้างเก็บเตรียมไว้ก่อน

2) **สร้างเซตรายการแทนที่คำผิดโดยใช้เซตอักขระที่อยู่ข้างเคียง**

จากผลการวิจัย Pollock and Zamora [15] พบว่าตำแหน่งของอักขระที่เป็นสาเหตุของคำผิดมักไม่ใช่อักขระแรก ซึ่งได้ผลทดสอบจากคำที่มีความยาว 3-4 ตัวอักษร และตำแหน่งที่ผิดมากน่าจะเป็นตำแหน่งอื่นๆ ที่ไม่ใช่ตำแหน่งแรกของคำ แต่ในบทที่ 3 ได้ทำการหาตำแหน่งของอักขระที่ผิดในภาษาไทย พบว่ามักผิดที่ตำแหน่งกึ่งกลางของคำและค่อนไปทางหลังเล็กน้อย

ในงานวิจัยนี้มีสมมติฐานว่าตำแหน่งที่ทำให้คำนั้นเกิดความผิดพลาด คือ ตำแหน่งกลางของคำ ซึ่งอักขระที่เป็นต้นเหตุของคำผิดอาจจะเป็นพยัญชนะ สระ หรือวรรณยุกต์ ก็ได้

3) **สร้างเซตรายการใกล้เคียงคำผิด**

นำเซตรายการแทนที่คำผิดโดยใช้ที่อักขระข้างเคียงที่ได้จากขั้นตอนในข้อที่ 2) ไปเปรียบเทียบกับคำในพจนานุกรม หากคำไหนมีในพจนานุกรมให้ทำการเก็บเป็นผลลัพธ์ที่จะนำไปใช้จริงในการแก้ไขคำผิดอัตโนมัติ หลังจากนั้นจึงทำการเลื่อนไปลองแก้ไขคำผิดในตำแหน่งถัดไป โดยสร้างเซตรายการใกล้เคียงคำผิดเช่นเดียวกับในขั้นตอนที่ 2)

<u>**ตัวอย่างที่** 4.1</u> แบน พิมพ์ผิดเป็น แบร

คำผิดนี้มีความยาว 3 อักขระ จึงทำการลองแทนที่ทั้ง 3 ตำแหน่ง โดยเริ่มจากกลาง แล้วเลื่อนไปขวา และเลื่อนกลับมาทางซ้าย ดังนี้

1. เริ่มแก้ไขจากตำแหน่งกึ่งกลางคำ ซึ่งในตัวอย่างนี้ คือ ตัวอักษร บ

- จะสามารถสร้างเซตการแทนที่คำผิดได้เป็น

{ แขร, แชร, แยร, แลร, แวร, แงร, แฐร, แ๘ร, แ๙ร, แญร, แ,ร, แซร, แ.ร }

- แต่ไม่พบในพจนานุกรม จึงไม่มีการเก็บลงในรายการคำใกล้เคียง

2. เลื่อนมาแก้ไขที่ตำแหน่งถัดไป คือ ตัวอักษร ร

- จะสามารถสร้างเซตการแทนที่คำผิดได้เป็น

{ แบค, แบต, แบี, แบน, แป, แบา, แบณ, แบ๕, แบ๖, แบ็, แบฯ, แบ่, แบษ }

- ให้เซตรายการใกล้เคียงคำผิดออกมาเป็น { แบน } เพียงตัวเดียวเท่านั้น เพราะคำว่า แบน เป็นคำเดียวจากเซตที่พบในพจนานุกรม จึงเก็บลงในรายการคำใกล้เคียงนี้ไว้

3. เลื่อนมาแก้ไขที่ตำแหน่งถัดไป คือ แ

- จะสามารถสร้างเซตการแทนที่คำผิดได้เป็น

{ กบร, ดบร, ปบร, อบร, ฉบร, ฎบร, โบร,)บร, ฮบร }

- แต่ไม่พบในพจนานุกรม จึงไม่มีการเก็บลงในรายการคำใกล้เคียงผิด

ดังนั้น จึงได้เซตรายการใกล้เคียงคำผิดจากการลองแก้ไขแบบแทนที่ คือ { แบน }

แต่วิธีการที่ผู้วิจัยนำเสนอนี้ ก็ยังมีโอกาสที่จะได้ผลลัพธ์ของรายการใกล้เคียงคำผิดได้มากกว่า 1 รายการได้ ซึ่งถ้าเกิดกรณีเช่นนี้จะนำ***ผลลัพธ์ที่ได้จากการแก้ไขคำผิดทั้งหมดทุกคำ*** ไปใช้ในการจับคู่กับแพทเทิร์นในฐานความรู้เอไอเอ็มแอลอีกทีหนึ่ง โดยจะนำไปลองจับคู่กับแพทเทิร์นในกฎทีละผลลัพธ์ (เรียงตามลำดับผลลัพธ์ที่ได้จากการลองแก้คำผิดแบบพิมพ์แทนที่ พิมพ์เกิน และพิมพ์สลับ)

4.1.2 การเกิน

การพิมพ์เกินเป็นความผิดพลาดที่สูงอันดับสองของการพิมพ์ผิดแบบไม่ตั้งใจ รองจากการพิมพ์ผิดแบบแทนที่ เมื่อคาดว่าเป็นการพิมพ์เกินก็ควรแก้ไขโดยลองลบอักขระออกทีละตัว ซึ่งจะเริ่มลบจากตำแหน่งกึ่งกลางออกไปด้านข้างเช่นเดิม

ตัวอย่างที่ 4.2 แบน พิมพ์ผิดเป็น แบร (คำผิดเดียวกับที่ใช้ในตัวอย่างที่ 4.1)

คำผิดนี้มีความยาว 3 อักขระ จึงทำการลองลบอักขระออกทีละตำแหน่ง โดยเริ่มจากกลาง แล้วเลื่อนไปขวา และเลื่อนกลับมาทางซ้าย ดังนี้

1. เริ่มลบจากตำแหน่งกึ่งกลางคำ

- ได้ออกมาเป็น { แร } ซึ่งมีในพจนานุกรม จึงเก็บไว้

2. เลื่อนมาลบที่ตำแหน่งถัดไปทางขวา

- ได้ออกมาเป็น { แบ } ซึ่งมีในพจนานุกรม จึงเก็บไว้ต่อท้ายจากคำว่า แร

3. เลื่อนมาลบที่ตำแหน่งทางซ้าย

- ได้ออกมาเป็น { บร } ซึ่งไม่พบในพจนานุกรม จึงไม่เก็บเพิ่มลงในรายการใกล้เคียงคำผิด

ดังนั้น จึงได้เซตรายการใกล้เคียงคำผิดจากการลองแก้ไขแบบลบ คือ { แร, แบ }

4.1.3 การสลับ

การพิมพ์สลับเป็นความผิดพลาดที่สูงอันดับสามของการพิมพ์ผิดแบบไม่ตั้งใจ รองจากการพิมพ์ผิดแบบแทนที่และแบบเกิน เมื่อคาดว่าเป็นการพิมพ์สลับก็ควรแก้ไขโดยลองสลับที่คู่อักขระที่อยู่ติดกันทีละคู่ ซึ่งจะเริ่มสลับจากคู่ตำแหน่งกึ่งกลางออกไปด้านข้างเช่นเดิม

ตัวอย่างที่ 4.3 แบน พิมพ์ผิดเป็น แบร (คำผิดเดียวกับที่ใช้ในตัวอย่างที่ 4.1)

คำผิดนี้มีความยาว 3 อักขระ จึงทำการลองสลับอักขระที่อยู่ติดกันทีละคู่ โดยเริ่มจากคู่กลาง แล้วเลื่อนไปขวา และเลื่อนกลับมาทางซ้าย ดังนี้

1. เริ่มสลับจากคู่ตำแหน่งกึ่งกลางคำ

- ได้ออกมาเป็น { แรบ } ซึ่งไม่พบในพจนานุกรม จึงไม่เก็บเพิ่มลงในรายการใกล้เคียงคำผิด

2. เลื่อนมาสลับที่คู่ตำแหน่งถัดไปทางซ้าย (เนื่องจากไม่มีคู่ทางขวาให้สลับแล้ว)

- ได้ออกมาเป็น { บแร } ซึ่งไม่พบในพจนานุกรม จึงไม่เก็บเพิ่มลงในรายการใกล้เคียงคำผิด

ดังนั้น เซตรายการใกล้เคียงคำผิดจากการลองแก้ไขแบบสลับ คือ { } ซึ่งเป็นเซตว่าง

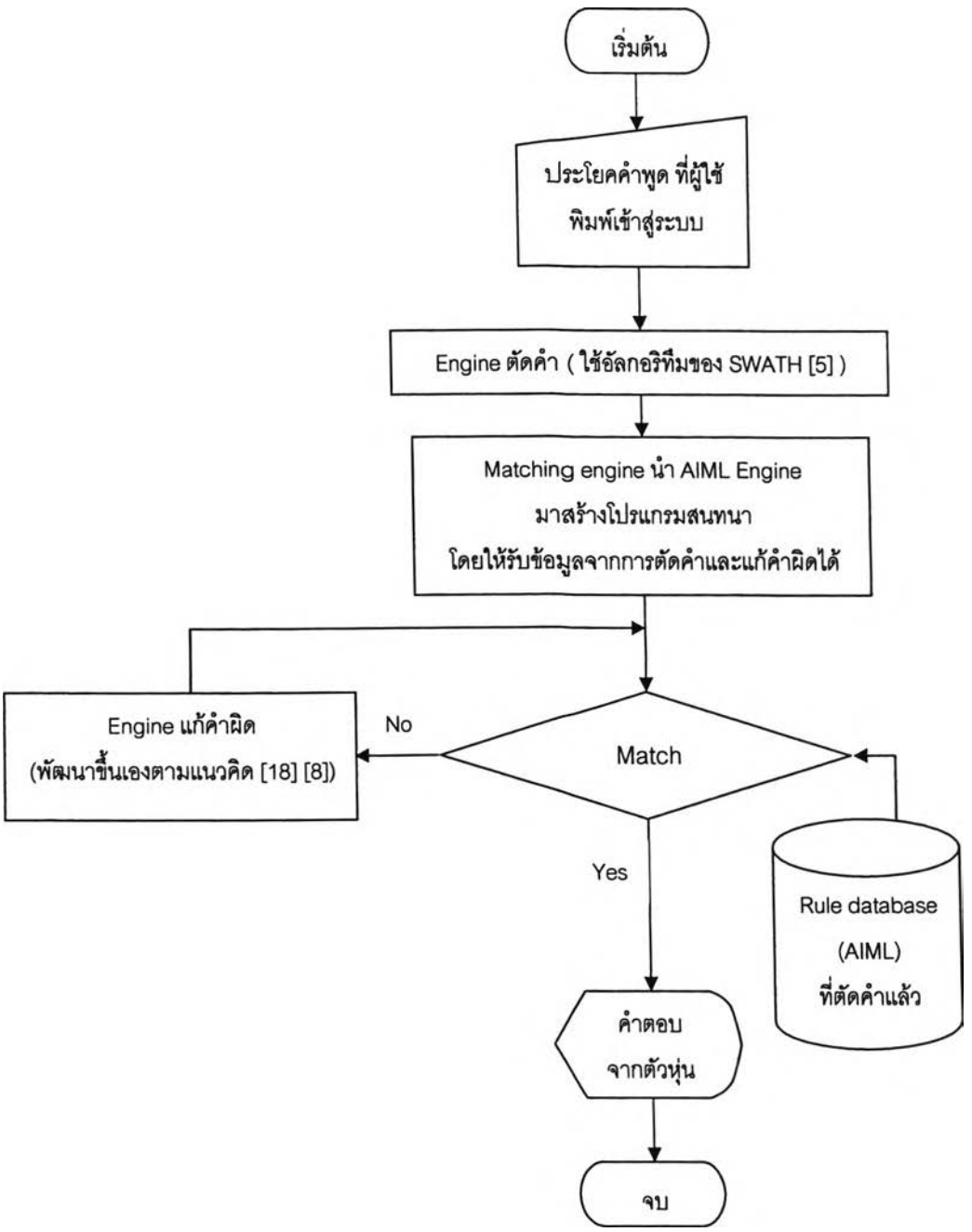
จากการลองแก้ไขคำผิดโดยใช้อัลกอริทึมซึ่งงานวิจัยนี้ได้ออกแบบกับตัวอย่างที่ 4.1-4.3 จะได้รายการใกล้เคียงคำผิดทั้งหมดของการลองแก้ไขทั้ง 3 กรณีแล้วออกมาเป็น { แบน, แร, แบ } โดยจะนำผลลัพธ์ที่ได้ทุกคำไปใช้ในการจับคู่กับแพทเทิร์นในฐานความรู้เอไอเอ็มแอลอีกทีหนึ่ง โดยจะนำไปลองจับคู่กับแพทเทิร์นในกฎทีละผลลัพธ์ แต่จะให้ลำดับความสำคัญ (Priority) ความสำคัญกับผลลัพธ์รายการแรกมากที่สุดเรียงมาตามลำดับ ในที่นี้ก็คือ จะให้ความสำคัญคำว่า "แบน" มากที่สุด ตามมาด้วยคำว่า "แร" และ "แบ" ตามลำดับ ถ้าหากจับคู่ได้แล้วก็จะไม่ทำเพิ่มอีก (เช่น ประโยคที่แทนคำว่า "แบน" ลงไปนั้นสามารถจับคู่ได้แล้ว ก็จะไม่นำประโยคอื่นไปทำการลองจับคู่อีก)

4.2 ส่วนตัวแปลภาษาสำหรับเอกสารเอไอเอ็มแอล

สำหรับหลักการทำงานของโปรแกรม จะต้องมีการสร้างฐานความรู้เอไอเอ็มแอลให้กับหุ่นเสียก่อน โดยการเขียนแพทเทิร์นของแต่ละแคตากอรีนั้นต้องมีการเว้นวรรคระหว่างคำเองเหมือนรูปแบบที่ได้จากการตัดคำด้วยโปรแกรม SWATH หลังจากนั้นหากต้องการรันเพื่อทดสอบการสนทนา จะมีขั้นตอนดังนี้

ระบบจะเริ่มจากการรับอินพุตเป็นประโยคข้อความภาษาไทยจากแป้นพิมพ์คราวละ 1 ประโยคในการสั่งงาน โดยอินพุตจะต้องเป็นประโยคที่มีใจความเดียวเท่านั้น (ในงานวิทยานิพนธ์นี้จะไม่รองรับประโยคความรวมหรือประโยคความซ้อน) หลังจากนั้นจะนำประโยคที่ได้รับมาทำการแบ่งแยกหรือตัดคำ ก่อนที่จะนำไปผ่านตัวแปลคำสั่ง (Interpreter) สำหรับนำไปประมวลผลหาคำตอบที่หุ่นจะตอบสนอง โดยส่วนของการหาคำตอบใช้หลักการของการจับคู่รูปแบบ (Pattern

Matching) กับข้อความในเอกสารเอไอเอ็มแอลที่ได้สร้างเตรียมไว้เป็นความรู้ให้กับหุ่นยนต์ หากไม่สามารถจับคู่ได้ก็จะเรียกส่วนของการแก้ไขคำผิดเข้ามาช่วย แล้วลองนำไปจับคู่เพื่อหาคำตอบอีกครั้ง ภาพรวมของการใช้งานในขณะสนทนาเป็นดังรูปที่ 4.5



รูปที่ 4.5 การทำงานของระบบสนทนาภาษาไทยอัตโนมัติ

4.3 การแก้คำผิดในกรณีที่มีคำผิดหลายที่ในประโยค

กรณีที่มีคำผิดหลายที่ในประโยคจะทำการหาคำใกล้เคียงของแต่ละคำที่ผิดเก็บเอาไว้ หลังจากนั้นนำมาสร้างประโยคผลลัพธ์แบบผลคูณคาร์ทีเซียน (Cartesian Product) เพื่อสร้างประโยคผลลัพธ์ที่เป็นไปได้ในทุกกรณี เพื่อไปใช้ในการลองจับคู่แพทเทิร์น โดยนำไปลองจับคู่ทีละประโยคตามลำดับ ซึ่งสามารถดูตัวอย่างการทำงานของวิธีการสร้างประโยคผลลัพธ์แบบผลคูณคาร์ทีเซียนได้ดังในตัวอย่างที่ 4.4

ตัวอย่างที่ 4.4 เราก้เพิ่งเป็นเมื่อกลางวัส

ประโยคตัวอย่างนี้มีคำผิด 2 ที่ คือ คำว่า "ก้" และคำว่า "วัส"

- คำว่า "ก้" มีรายการคำใกล้เคียงเป็น { กี, ก็ }
- คำว่า "วัส" มีรายการคำใกล้เคียงเป็น { วัน, วัย, วัว }

สามารถนำมาสร้างประโยคผลลัพธ์แบบผลคูณคาร์ทีเซียนได้เป็น 6 ประโยค ดังนี้

- เรากีเพิ่งเป็นเมื่อกลางวัน
- เรากีเพิ่งเป็นเมื่อกลางวัย
- เรากีเพิ่งเป็นเมื่อกลางวัว
- เราก็เพิ่งเป็นเมื่อกลางวัน
- เราก็เพิ่งเป็นเมื่อกลางวัย
- เราก็เพิ่งเป็นเมื่อกลางวัว

4.4 การทดสอบประสิทธิภาพอัลกอริทึมแก้คำผิดกับงานอื่นในภาษาไทย

การทดสอบประสิทธิภาพด้านอัลกอริทึมแก้คำผิดของงานวิจัยนี้ จะเปรียบเทียบกับงานด้านการแก้ไขคำผิดในภาษาไทยที่มีลักษณะใกล้เคียงกับวัตถุประสงค์ของงานวิจัยเป็นจำนวน 2 งาน คือ โปรแกรมประมวลผลคำ (Word Processing) ไมโครซอฟท์เวิร์ดเอ็กซ์พี และโปรแกรมตรวจสอบตัวสะกดภาษาไทยที่เป็นงานวิจัยของทิวา [19] ซึ่งจะทำการเปรียบเทียบประสิทธิภาพใน 2 ด้านคือ ด้านการจัดลำดับ และจำนวนรายการคำใกล้เคียง โดยในแต่ละงานมีข้อสรุปดังนี้

4.4.1 โปรแกรมไมโครซอฟท์เวิร์ดเอ็กซ์พี

งานนี้ไม่ได้เปิดเผยอัลกอริทึมที่ใช้ในการค้นหารายการใกล้เคียงคำผิด แต่จากการสังเกตผลลัพธ์การทำงานของโปรแกรมพบว่าใช้องค์ประกอบเรื่องระยะแก้ไข (Edit Distance) มาเป็นพื้นฐานก่อน แล้วจึงนำองค์ประกอบอื่นมาช่วยจัดลำดับในกรณีที่ได้ระยะแก้ไขเท่ากันอีกทีหนึ่ง

การทดลองเปรียบเทียบประสิทธิภาพการตัดคำเริ่มจากนำประโยคที่มีคำผิดจากภาคผนวก ข จำนวน 120 ประโยค มาตัดให้เหลือเฉพาะ คำที่ผิดแล้วตัดคำที่ซ้ำกันออกไป ท้ายสุดแล้วจะได้คำผิดเพื่อนำมาทดสอบเป็นจำนวน 99 คำ

ข้อจำกัดของการทดลองนี้ คือ จะพิจารณาเปรียบเทียบเฉพาะรายการใกล้เคียงคำผิดที่มีระยะแก้ไขเป็น 1 เท่านั้น และจะตัดผลลัพธ์ที่ได้จากการพิมพ์ตกออกไป (เนื่องจากงานวิจัยในวิทยานิพนธ์นี้ไม่สามารถแก้ไขกรณีที่เกิดจากระยะแก้ไขมากกว่าหนึ่งและกรณีที่เกิดจากการพิมพ์ตก)

จากนั้นให้นำคำที่ใช้ทดสอบจำนวน 99 คำ ไปหารายการใกล้เคียงคำผิดทั้งหมดของแต่ละงานออกมา (โดยตัดผลลัพธ์ที่เป็นข้อจำกัดทิ้งไป) แล้วพิจารณาผลการจัดเรียงลำดับที่ได้ โดยดูว่าคำเดิมที่ถูกต้องนั้นจัดอยู่ในลำดับที่เท่าใด ถ้าหากเป็นลำดับที่ 1 จะถือว่าเป็นการจัดลำดับที่ดีที่สุด และหากงานใดไม่พบคำเดิมจะบันทึกผลไว้ต่างหาก

การวิเคราะห์ผลการทดลองเริ่มจากพิจารณาว่าหากวิธีการใดวิธีการหนึ่งไม่พบคำเดิมให้ตัดผลลัพธ์นั้นออกไป ซึ่งผลลัพธ์ที่ไม่พบทั้งหมดนั้นมีจำนวน 30 ชุด ทำให้เหลือผลลัพธ์ที่จะพิจารณาเปรียบเทียบได้เพียง 69 ชุด ซึ่งสรุปได้ดังตารางที่ 4.1

ตารางที่ 4.1 เปรียบเทียบผลการทดลองกับไมโครซอฟท์เวิร์ดเอ็กซ์พี

วิธีการ	ค่าเฉลี่ยลำดับในการจัดเรียง	ไม่พบคำเดิม
ไมโครซอฟท์เวิร์ดเอ็กซ์พี	1.7	27
งานวิทยานิพนธ์นี้	1.3	3

จากตารางที่ 4.1 สังเกตได้ว่า ในกรณีที่พบคำเดิมทั้งสองวิธีการซึ่งมีจำนวน 69 ชุด จากข้อมูลทดสอบทั้งหมด 99 ชุด พบว่างานวิจัยนี้สามารถจัดลำดับรายการใกล้เคียงคำผิดได้ใกล้เคียงกับคำเดิมมากกว่าไมโครซอฟท์เวิร์ด และยังสามารถค้นหาคำเดิมได้ดีกว่ามาก ซึ่งสังเกตได้ด้วยจำนวนกรณีที่ไม่พบคำเดิมของแต่ละวิธีการ

นอกจากนี้ หากพิจารณาจำนวนรายการคำใกล้เคียง จะพบว่ากรณีทั้งสองวิธีการให้จำนวนรายการคำใกล้เคียงได้เท่ากันมีจำนวน 31 ชุด กรณีที่ไมโครซอฟท์เวิร์ดให้จำนวนรายการคำใกล้เคียงได้น้อยกว่าวิธีในวิทยานิพนธ์เป็นจำนวน 8 ชุด และวิธีการที่ใช้ในวิทยานิพนธ์นี้ให้จำนวนรายการคำใกล้เคียงได้น้อยกว่าเป็นจำนวน 29 ชุด ซึ่งการที่ได้รายการใกล้เคียงที่ลดลงนี้จะมีประโยชน์ในกรณีที่เกิดคำผิดหลายคำในประโยค เพราะจะทำให้ลดผลลัพธ์ในการทำผลคูณคาร์ทีเซียนลงไปได้

4.4.2 โปรแกรมตรวจสอบตัวสะกดภาษาไทย

งานนี้เป็นของทิวา เจริญสวัสดิพงศ์ [19] โดยเป็นบทความในงานประชุมประจำปี สวทช. ในปี 2548 มีวัตถุประสงค์ที่จะระบุขอบเขตของคำผิดให้ดีที่สุด โดยใช้หลักการตัดคำแบบเดียวกับวีร์ สัตยมาศ [8] แต่ได้เพิ่มเติมเรื่องค่าต้นทุนในการตัดคำลงไป ถ้าหากระบุขอบเขตของคำผิดได้ดีตั้งแต่ต้นแล้วก็จะส่งผลให้การหารายการคำใกล้เคียงคำผิดได้ผลใกล้เคียงกับความต้องการของผู้ใช้มากขึ้น และงานนี้มีขอบเขตคือจะค้นหารายการคำใกล้เคียงที่มีระยะแก้ไขเท่ากับ 1 มาแสดงเท่านั้น

บทความนี้มุ่งเน้นแค่การระบุขอบเขตของคำผิดเท่านั้น มิได้หาองค์ประกอบอื่นมาช่วยในการจัดลำดับ (Ranking) รายการใกล้เคียงคำผิด จึงได้สมมติค่าความสำคัญในการจัดเรียงออกมาตามการสังเกตพฤติกรรมการพิมพ์ในภาษาอังกฤษ คือ ให้ลำดับความสำคัญ (Priority) กับลักษณะการพิมพ์ผิดแบบต่างๆ เรียงจากมากไปน้อยได้เป็น พิมพ์ตก พิมพ์เกิน พิมพ์แทนที่ และพิมพ์สลับ ตามลำดับ

การทดลองเปรียบเทียบประสิทธิภาพจะมีการเตรียมข้อมูลทดสอบและวิธีการทดสอบแบบเดียวกับในหัวข้อ 4.4.1 รวมไปถึงข้อจำกัดต่างๆ และวิธีการวิเคราะห์ผลการทดลองก็ทำเช่นเดียวกัน

คำที่ใช้ทดสอบมีจำนวน 99 คำ ให้นำไปหารายการใกล้เคียงคำผิดทั้งหมดของแต่ละงานออกมา (โดยตัดผลลัพธ์ที่เป็นข้อจำกัดทิ้งไป) แล้วพิจารณาผลการจัดเรียงลำดับที่ได้ โดยดูว่าคำเดิมที่ถูกต้องนั้นจัดอยู่ในลำดับที่เท่าไหร่ ถ้าหากเป็นลำดับที่ 1 จะถือว่าเป็นการจัดลำดับที่ดีที่สุด และหากงานใดไม่พบคำเดิมจะมีการบันทึกผลไว้ต่างหาก

การวิเคราะห์ผลการทดลองเริ่มจากพิจารณาว่าหากวิธีการใดวิธีการหนึ่งไม่พบคำเดิมให้ตัดผลลัพธ์นั้นออกไป ซึ่งมีจำนวน 6 ชุด ทำให้เหลือผลลัพธ์ที่จะพิจารณาเปรียบเทียบได้เพียง 93 ชุด ซึ่งสรุปได้ดังตารางที่ 4.2

ตารางที่ 4.2 เปรียบเทียบผลการกับทดลองกับงานวิจัยของทิวา

[illegible]	[illegible]	[illegible]
งานวิจัยของทิวา	1.5	3
งานวิทยานิพนธ์นี้	1.4	3

จากตารางที่ 4.2 สังเกตได้ว่า ในกรณีที่พบคำเดิมทั้งสองวิธีการซึ่งมีจำนวน 93 ชุด จากข้อมูลทดสอบทั้งหมด 99 ชุด พบว่างานวิจัยนี้สามารถจัดลำดับรายการใกล้เคียงคำผิดได้

ใกล้เคียงกับคำเดิมมากกว่างานของทิวา นอกจากนี้ หากพิจารณาจำนวนรายการคำใกล้เคียง จะพบว่ากรณีทั้งสองวิธีการให้จำนวนรายการคำใกล้เคียงได้เท่ากันมีจำนวน 27 ชุด กรณีที่งานของทิวาให้จำนวนรายการคำใกล้เคียงได้น้อยกว่าเป็นจำนวน 8 ชุด และวิธีการที่ใช้ในวิทยานิพนธ์นี้ให้จำนวนรายการคำใกล้เคียงได้น้อยกว่าเป็นจำนวน 58 ชุด

ผลการจัดเรียงลำดับของวิธีการนี้แม้จะดีขึ้นเพียงเล็กน้อยเมื่อเทียบกับงานของทิวา แต่หากพิจารณาจำนวนคำใกล้เคียงจะพบว่าให้จำนวนรายการคำใกล้เคียงในปริมาณที่น้อยกว่ามาก ซึ่งจำนวนคำใกล้เคียงนี้จะมีผลในกรณีที่ต้องทำผลคูณคาร์ทีเซียน หากมีคำผิดเกิดขึ้นหลายที่ในประโยค (ในการนำไปใช้จับคู่แพทเทิร์นของหุ่นยนต์สนทนา)