รหัสโครงการ 11P34C686

ซีพีเอสเคคัท

หัวข้อพิเศษ BEST2009 - การแบ่งคำไทย (BEST - Thai Word Segmentation)

รายงานฉบับสมบูรณ์

เสนอต่อ

ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ

สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ

กระทรวงวิทยาศาสตร์และเทคโนโลยี

และ

สำนักงานส่งเสริมอุตสาหกรรมซอฟต์แวร์แห่งชาติ

ได้รับทุนอุดหนุนโครงการวิจัย พัฒนาและวิศวกรรม

โครงการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย ครั้งที่ 11

ประจำปีงบประมาณ 2551

โดย

นายกฤตธี ศิริสิทธิ์

นายณัฏฐ์ ปิยะปราโมทย์

อาจารย์ชัยพร ใจแก้ว

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเกษตรศาสตร์

กิตติกรรมประกาศ

ผู้พัฒนาขอขอบคุณ อาจารย์ชัยพร ใจแก้ว และภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ ขอขอบคุณพี่มุกดาที่คอยให้คำแนะนำในการทำโครงการ

โครงการซีพีเอสเคคัทได้รับทุนอุดหนุนโครงการการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทยครั้งที่ 11 จากศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ และสำนักงานส่งเสริมอุตสาหกรรมซอฟต์แวร์แห่งชาติ

ผู้พัฒนา

บทคัดย่อ

การแบ่งคำภาษาไทย (Word Segmentation) เป็นขั้นตอนสำคัญที่มีผลต่อความถูกต้องของขั้นตอนการประมวลผลภาษาไทยอื่นๆ เนื่องจากคำในประโยคภาษาไทย จะถูกเขียนติดกันโดยไม่เว้นช่องว่างระหว่างคำ ถ้าการแบ่งคำผิดพลาดจะส่งผลกระทบต่อขั้นตอนอื่นๆ ด้วย วิธีการแบ่งคำที่ถูกนำเสนอในงานวิจัยที่มีมาก่อนสามารถแบ่งออกได้ เป็น 2 กลุ่มใหญ่ คือ กลุ่มที่ใช้พจนานุกรม (Dictionary based) และกลุ่มที่ใช้หลักการอิงสถิติ (Statistically based) ปัญหาที่เกิดขึ้นกับวิธีการแบ่งคำทั้งสองวิธีคือ ปัญหาความกำกวมของการแบ่งคำ และการที่ไม่พบคำดังกล่าวในพจนานุกรม

ผู้พัฒนาได้พัฒนาโปรแกรมตัดคำแบบส นใจบริบท โดยอาศัยหลักการผสมผสานระหว่างการใช้พจนานุกรมและหลักการสถิติซึ่งให้ความสำคัญกับคำบริบท โดยประโยคที่นำเข้ามาในระบบจะถูกแบ่งคำโดยพจนุกรมเพื่อหารูปแบบการแบ่งคำที่เป็นไปได้ทั้งหมด และใช้วิธีการเรียนรู้ด้วยวิธีการทางสถิติมาเลือกรูปแบบที่เป็นไปได้มากที่สุ ด วิธีการเรียนรู้ด้วยวิธีการทางสถิติจะถูกฝึกฝนด้วยข้อความจากคลังข้อความที่มีการแบ่งคำไว้ก่อนแล้ว (Annotated Corpus) จากการการทดสอบโปรแกรมพบว่าสามารถแบ่งคำได้ค่าความถูกต้อง 92.59% ที่ความเร็วไม่ต่ำกว่า3,500 คำต่อวินาที

Thai word segmentation is an important phase that effect to Thai data processing because word in Thai have been write contiguously. A word segmentation method in research can divided into two group dictionary based and statistically based but the problems happen in that two groups is cryptic or cloudy word segmentation.

Our team develops Thai word segmentation application that focuses in context using dictionary and statistically. The sentence that input to the system will be divided by dictionary for finding sample space of word segmentation and use statistic for choose the most possible pattern. This method will be trained by use annotated corpus. From test result of our application average percent correctness is 92.59% and speed is not below than 3,500 words per second.

บทนำ

คำในประโยคภาษาไทยจะถูกเขียนติดกันไปโดยไม่มีช่องว่างระหว่า งคำ[1] ซึ่งแตกต่างจาก ภาษาอื่นๆ เช่นภาษาอังกฤษที่มีช่องว่างระหว่างคำชัดเจน และ ภาษาจีนที่คำสามารถเขียนให้อยู่ ในรูปตัวอักษรตัวเดียวได้ ดังนั้น ประโยคภาษาไทยที่จะถูกนำไปผ่านระบบประมวลผลต่างๆ ต้องผ่านการแ บ่งคำเพื่อแบ่งแยกคำในประโยค ถ้ามีการแบ่งคำผิดพลาดจะส่งผลกระทบให้ขั้นตอนอื่นๆ ผิดพลาดตามไปด้วย การแบ่งคำจึงเป็นขั้นตอนสำคัญที่มีผลกระทบต่อความถูกต้องของขั้นตอนอื่นๆ เช่น การจัดรูปแบบเอกสารในการประมวลผลคำ (Word Processing) การทำดรรชนีสำหรับเอกสาร (Document Indexing) การตรวจสอบคำสะกดผิด (Spelling Check) การรู้จำและสังเคราะห์เสียงพูด (Speech Recognition and Synthesis) การสรุปความ (Text Summarization) การแปลภาษาด้วยคอมพิวเตอร์ (Machine Translation) เป็นต้น

วิธีการแบ่งคำโดยอาศัยการเรียนรู้ด้วยเครื่อง ถูกนำมาประยุ กต์ใช้ในการแบ่งคำโดยมองปัญหาการแบ่งคำเป็นการแยกแยะกลุ่มของเวคเตอร์คุณลักษณะ (Features Vector) ออกเป็น 2 กลุ่ม (Binary Classification) คุณลักษณะที่นิยมนำมาใช้เป็นเวคเตอร์คุณลักษณะคือ n-gram ในระดับตัวอักษร (สายอักษรความยาว n ตัวอักษรที่ตำแหน่งต่างๆ ในเอกสาร ซึ่งตำแหน่งจะถูกเลื่อนไปทีละ 1 ตัวอักษรโดยมีส่วนที่ซ้อนทับกัน) ดังนั้นการเรียนรู้ด้วยเครื่องจะแยกแยะเวคเตอร์คุณลักษณะเป็น 2 กลุ่ม คือ กลุ่มที่เป็นตำแหน่งเริ่มต้นของคำและกลุ่มที่เป็นตำแหน่งภายในคำ

จากผลการทดลองใน [6] พบว่า ความถูกต้องของวิธีการแบ่งคำโดยใช้การเรียนรู้ด้วยเครื่องส่วนใหญ่ มีค่าน้อยกว่าวิธีการแบ่งคำโดยการใช้พจนานุกรม โดยมีเพียงเทคนิคการเรียนรู้ด้วยเครื่องแบบ Condition Random Field (CRF) เพียงเทคนิคเดียวเท่านั้นที่มีความถูกต้องสูงกว่าการแบ่งคำโดยใช้พจนานุกรม

จากการประยุกต์ใช้วิธีการเรียนรู้ด้วยเครื่องในงานวิจัยก่อนหน้า พบว่าส่วนใหญ่นำไปประยุกต์ใช้กับคุณลักษณะระดับตัวอักษร ในโครงการนี้จึงสนใจการนำวิธีการเรียนรู้ด้วยเครื่องไปประยุกต์ใช้กับคุณลักษณะระดับคำ ที่ได้จากวิธีการแบ่งคำด้วยพจนานุกรม เพื่อปรับปรุงการแบ่งคำโด ยสนใจบริบทของคำ

สารบัญ

วัตถุประสงค์และเป้าหมาย

วัตถุประสงค์

1. เพื่อปรับปรุงผลลัพธ์การแบ่งคำโดยใช้พจนานกุรมให้มีความถูกต้องสูงขึ้น
2. เพื่อศึกษาถึงวิธีการเรียนรู้ด้วยเครื่อง (Machine Learning) เพื่อใช้เลือกกรณีการแบ่งคำที่น่าจะเป็นไปได้มากที่สุด อ้างอิงจากคลังข้อความที่มีการกำกับข้อมูลไว้
3. เพื่อศึกษาถึงวิธีการที่เหมาะสมในการแก้ปัญหาคำที่ไม่ปรากฏในพจนานกุรม

เป้าหมายและขอบเขต

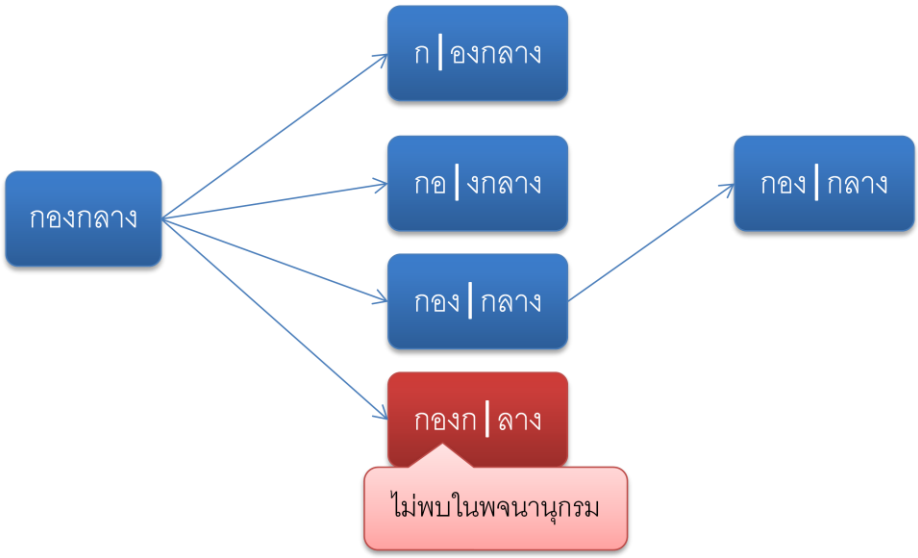
1. แบ่งคำโดยใช้นิยามหน่วยคำตามหลักเกณฑ์ “หน่วยเล็กที่สุดที่มีองค์ประกอบความเป็นคำครบถ้วน” (Minimal Integrity Unit) [5]
2. แก้ปัญหาความกำกวมของการแบ่งคำได้ ด้วยวิธีการเรียนรู้โดยอาศัยหลักการทางสถิติ
3. การเรียนรู้ด้วยเครื่องเป็นแบบ Supervised learning เท่านั้น โดยฝึกฝนด้วยข้อมูลจากคลังข้อความ
4. สามารถลดความผิดพลาดของการแบ่งคำ ที่เกิดจากคำไม่ปรากฏในพจนานุกรมได้

รายละเอียดของการพัฒนา

ทฤษฎีที่เกี่ยวข้อง

- **วิธีการแบ่งคำแบบ** Longest Matching [2]

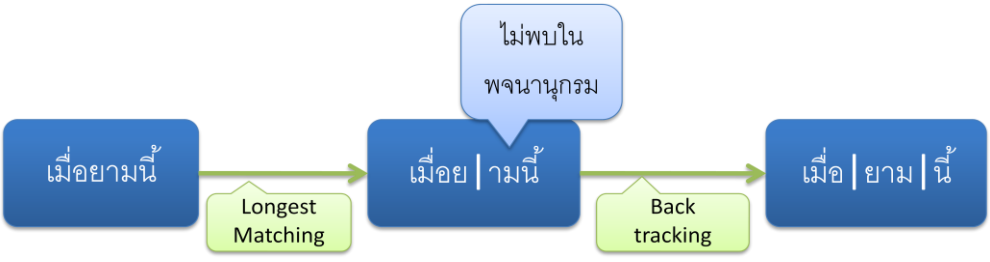
วิธีการแบ่งคำแบบ Longest Matching ทำการแบ่งคำโดยเปรียบเทียบหาคำที่ยาวที่สุดที่พบในพจนานุกรม และทำต่อไปเรื่อยๆ จนจบข้อความ ตัวอย่างการแบบคำแสดงดังรูปที่ 1



รูป 1 ตัวอย่างการแบ่งคำด้วยวิธีการแบ่งแบบ Longest Matching

- **วิธีการแบ่งคำแบบ** Backtracking [3]

จากวิธีการแบ่งคำแบบ Longest Matching ที่เลือกแบ่งคำให้มีความยาวมากที่สุด อาจทำให้ค้นหาคำถัดไปในพจนานุกรมไม่พบ ซึ่งวิธีการ Backtracking สามารถย้อนรอยกลับไปคำก่อนหน้าแล้วเลือกแบ่งคำก่อนหน้าให้มีความยาวสั้นลง เพื่อให้สามารถตัดคำถัดไปได้ดังตัวอย่างในรูปที่ 2

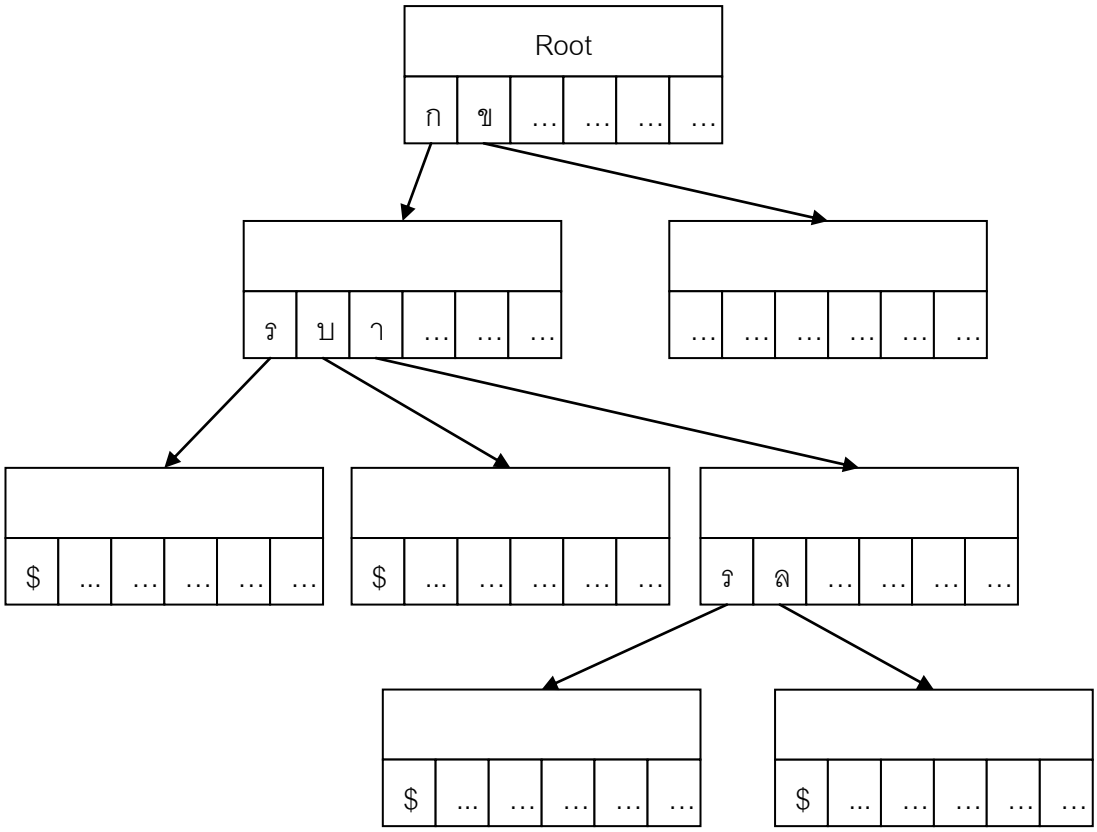


รูป 2 ตัวอย่างการแบ่งคำด้วยวิธีการแบบ Back tracking

ทฤษฎี หลักการ และเทคนิคที่ใช้

- **โครงสร้างข้อมูล** Trie

โครงสร้างข้อมูล Trie เป็นโครงสร้างข้อมูลแบบต้นไม้โดยใช้โหนดแต่ละโหนดจัดเก็บตัวอักษรหนึ่งตัว นิยมใช้เก็บข้อมูลพจนานุกรมเพราะสามารถค้นหาคำศัพท์ที่เก็บได้อย่างรวดเร็ว



รูป 3 ตัวอย่างโครงสร้างข้อมูล Trie

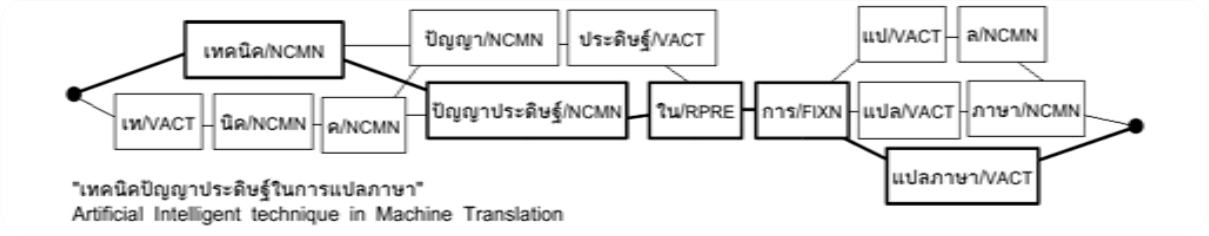
จากรูป 3 เป็นตัวอย่างโครงสร้างข้อมูล Trie ที่จัดเก็บคำศัพท์ “กร” “กบ” “การ” และ “กาล” จะเห็นว่าแต่ละโหนดจะจัดเก็บตัวอักษรที่เป็นไปได้ทั้งหมด และพอยเตอร์ ที่ชี้ไปยังโหนด ถัดไปสำหรับแต่ละตัวอักษร โดยทุกๆ ตัวอักษร ตัวสุดท้าย ของคำศัพท์จะชี้ไปยังโหนดที่มีเครื่องหมายจบคำ (ในที่นี้คือเครื่องหมาย $)

เราสามารถค้นหาคำศัพท์ภายในโครงสร้างข้อมูล Trie โดยเริ่มต้นที่โหนด Root แล้วตรวจสอบว่ามีตัวอักษรตัวแรกอยู่ในโหนดหรือไม่ ถ้ามีให้เดินไปที่โหนดที่พอยเตอร์ของตัวอักษรชี้อยู่ แต่ถ้าไม่มีแสดงว่าไม่พบคำศัพท์ในพจนานุกรม ให้ทำซ้ำไปเรื่อยๆ สำหรับตัวอักษรแต่ละตัว เมื่อนำตัวอักษรทุกตัวของคำศัพท์

มาค้นหมดแล้ว ตรวจสอบว่าโหนดที่ไปหยุดอยู่มีเครื่องหมายจบคำ (เครื่องหมาย $) หรือไม่ ถ้ามีแสดงว่ามีคำศัพท์ในโครงสร้างข้อมูลนี้

- **วิธีสร้างกราฟรูปแบบการแบ่งคำที่เป็นไปได้ทั้งหมด** [4]

ใช้วิธีแบ่งคำแบบ Backtracking เพื่อแบ่งคำออกเป็นคำที่ยาวที่สุดที่มีอยู่ในพจนานุกรม คำที่ถูกแบ่งแล้ว จะถูกตรวจสอบเพื่อแบ่งคำอีกครั้ง จนกระทั่งไม่สามารถแบ่งคำได้อีก รูปที่ 4 แสดงกราฟตัวอย่างการแบ่งคำที่เป็นไปได้ทั้งหมดของประโยค “เทคนิคปัญญาประดิษฐ์ในการแปลภาษา”



รูป 4 กราฟตัวอย่างการแบ่งคำที่เป็นไปได้ทั้งหมด [4]

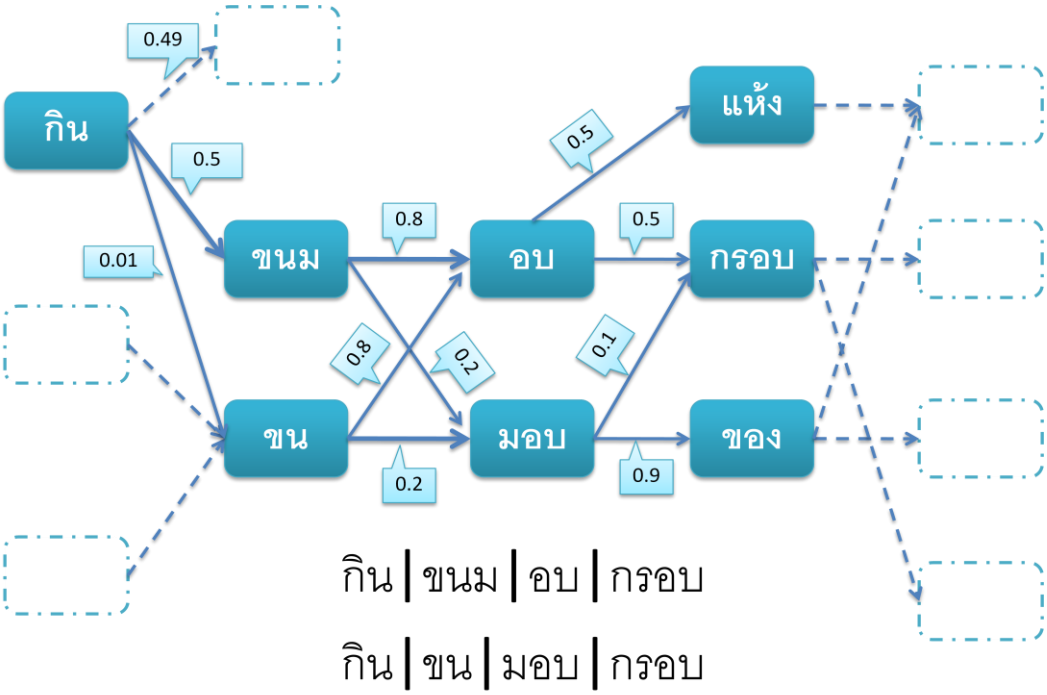
- **วิธีการเลือกรูปแบบการแบ่งคำที่เป็นไปได้มากที่สุด โดยใช้วิธีการเรียนรู้ด้วยเครื่อง**

วิธีการเรียนรู้ด้วยเครื่องแบบโมเด ลมาร์คอพ[7] (Visible Markov model) ถูกใช้กันอย่างแพร่หลายในด้านการวิเคราะห์ลำดับของสถานะได้ค่าความถูกต้องสูง ผู้พัฒนาจึงสนใจที่จะนำวิธีการเรียนรู้ด้วยเครื่องวิธีนี้มาใช้แก้ปัญหาความกำกวมของการ แบ่งคำ โดยเลือกเส้นทางที่มีค่าความน่าจะเป็นมากที่สุดจากการแบ่งคำที่เป็นไปได้ทั้งหมด

โมเดลมาร์คอพจะถูกฝึกฝน (Train) ด้วยข้อความจากคลังข้อความที่ถูกตัดคำแล้ว เพื่อคำนวณหาค่าประมาณของค่าความน่าจะเป็นในการเกิดลำดับคำ สำหรับแต่ละบริบท

โมเดลมาร์คอพจะประกอบไปด้วยโหนดและเส้นทาง (Node & Edge) โดยโหนดแต่ละโหนดในกราฟแทนสถานะ (State) ภายในมาร์คอพโมเดล และเส้นทางแทนการเปลี่ยนสถานะ (State Transition) ที่เป็นไปได้ภายในโมเดลมาร์คอพ โดยมีค่าน้ำหนักตามค่าความน่าจะเป็นในการเปลี่ยนสถานะ (Transition Probability) ในการนำโมเดลมาประยุกต์ใช้จะกำหนดให้คำหนึ่งคำแทนหนึ่งสถานะ และค่าความน่าจะเป็นในการเปลี่ยนสถานะแทนด้วยค่าสถิติของการเกิดลำดับคำในคลังข้อความ

จากรูปที่ 5 เป็นส่วนหนึ่งของโมเดลมาร์คอพ ที่ใช้พิจารณาเพื่อเลือกรูปแบบการแบ่งคำที่เหมาะสมที่สุด สำหรับข้อความ “กินขนมอบกรอบ” โดยเลือกรูปแบบที่ให้ค่าความน่าจะเป็นในการเกิดลำดับคำตามโมเดลมาร์คอพสูงสุด



รูป 5 ส่วนหนึ่งของโมเดลมาร์คอพที่ใช้ในการเลือกรูปแบบการแบ่งคำ

เครื่องมือที่ใช้ในการพัฒนา

- Python 2.5
- Microsoft Visual Studio 2008 (Visual C#)

รายละเอียดโปรแกรมที่ได้พัฒนาในเชิงเทคนิค

Input/Output Specification

Input: ไฟล์เอกสารชนิดข้อความ (text file) ที่ถูกเข้ารหัสแบบ UTF-8

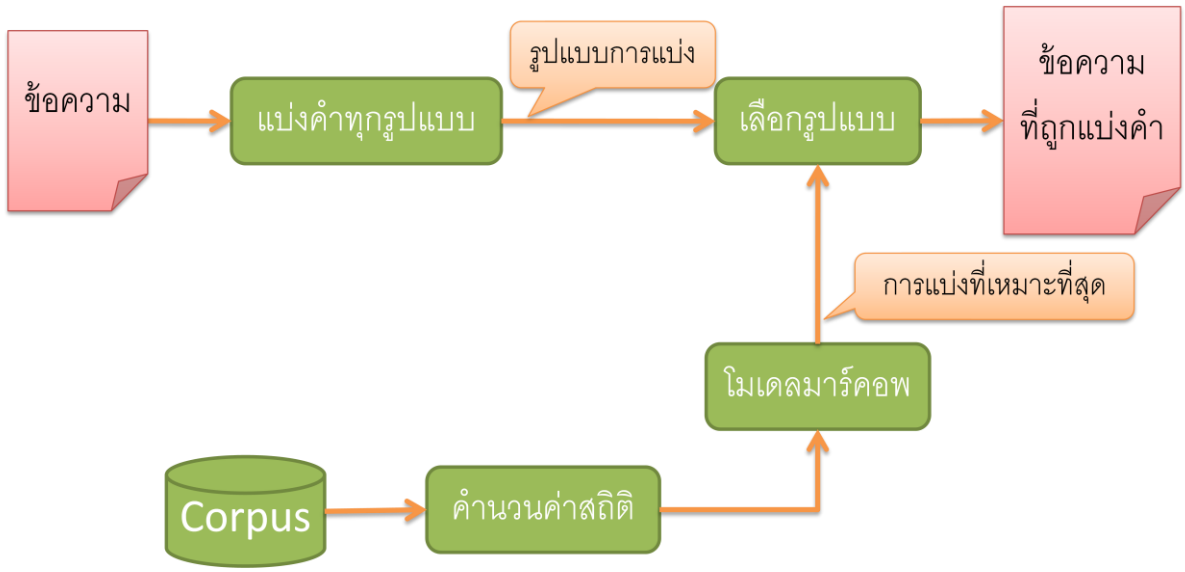
Output: ไฟล์เอกสารชนิดข้อความ (text file) ที่ถูกเข้ารหัสแบบ UTF-8 และมีการกำกับขอบเขตของคำโดยการใส่เครื่องหมาย “|” ระหว่างคำ

Functional Specification

- การฝึกฝน – ผู้ใช้สามารถฝึกฝนระบบด้วยคลังข้อความอื่นๆ ได้
- การแบ่งคำ – เลือกแบ่งคำได้สองแบบ แบบทีละไฟล์ และแบบหลายๆ ไฟล์

โครงสร้างของซอฟต์แวร์ (Design)

ขั้นตอนการแบ่งคำในโปรแกรมที่จะพัฒนาขึ้นแสดงดังรูปที่ 6



รูป 6 โครงสร้างของซอฟต์แวร์

ขอบเขตและข้อจำกัดของโปรแกรมที่พัฒนา

1. โปรแกรมสามารถแทรกตัวแบ่งคำ “|” ในตำแหน่งท้ายคำแต่ละคำได้อย่างมีประสิทธิภาพ
2. ความเร็วในการแบ่งคำไม่ต่ำกว่า 700 คำ/วินาที

กลุ่มของผู้ใช้โปรแกรม

นักภาษาศาสตร์ และบุคคลที่สนใจทั่วไป

ผลของการทดสอบโปรแกรม

จากการฝึกฝนระบบด้วยคลังข้อความซึ่งผู้จัดการแข่งขัน BEST 2009 จัดทำไว้ จำนวนทั้งสิ้น 5,645,984 คำ ได้คำศัพท์ในพจนานุกรมทั้งหมด 40,103 คำ เมื่อนำโปรแกรมไปทดลองแบ่งคำไฟล์ตัวอย่างจำนวน 509 ไฟล์ ได้ค่าความถูกต้อง 92.59% (วัดโดยสนใจความถูกต้องการแบ่งคำภายใน <NE>, <AB>, และ <POEM>) และมีความเร็วขั้นไม่ต่ำกว่า 3,500 คำ/วินาที

ปัญหาและอุปสรรค

1. คลังข้อความที่ผู้จัดการแข่งขัน BEST 2009 มีการกำกับ Tag ผิดพลาด เช่น <//NE> ทำให้ต้องคัดเลือกข้อมูลที่ได้จากคลังข้อความด้วยมนุษย์
2. โปรแกรมยังไม่สามารถตัดคำบางคำได้อย่างถูกต้องเพราะไม่มีลำดับคำดังกล่าวปรากฏในคลังข้อความ

แนวทางในการพัฒนาและประยุกต์ใช้ร่วมกับงานอื่นๆ ในขั้นต่อไป

1. พัฒนาโปรแกรมเป็น Dynamic Link Library (DLL) เพื่อรองรับการติดต่อกับโปรแกรมอื่นๆ ที่จำเป็นต้องใช้การแบ่งคำ
2. เพิ่มส่วน Name Entity Recognition เพื่อให้โปรแกรมสามารถแบ่งคำที่มีชื่อเฉพาะได้แม่นยำมากขึ้น
3. เพิ่มขนาดคลังข้อความที่นำมาใช้ฝึกฝนระบบ

ข้อสรุปและข้อเสนอแนะ

1. ผู้พัฒนาได้พัฒนาโปรแกรมแบ่งคำแบบสนใจบริบท เพื่อแก้ไขปัญหาความกำกวมของการแบ่งคำ โดยนำโมเดลมาร์คอพมาเรียนรู้ลำดับการเกิดของคำในคลังข้อความที่ถูก แบ่งคำแล้ว
2. โปรแกรมแบ่งคำที่ใช้โมเดลมาร์คอพที่สร้างขึ้นจากคลังข้อความขนาด 5,645,984 คำ สามารถแบ่งคำได้โดยมีค่าความถูกต้อง 92.59% ที่ความเร็วไม่ต่ำกว่า 3,500 คำ/วินาที

เอกสารอ้างอิง

1. พระยาอุปกิตศิลปสาร. **หลักภาษาไทย**. สำนักพิมพ์ไทยวัฒนาพาณิชย์, กรุงเทพฯ. 2541.
2. Yuen Poowarawan, **"Dictionary-based Thai Syllable Separation,"** Pro- ceedings of the Ninth Electronics Engineering Conference, 1986.
3. V. Sornlertlamvanich, **"Word Segmentation for Thai in Machine Translation System,"** Machine Translation, National Electronics and Computer Technology Center, Bangkok.
4. C. Kruengkrai and H. Isahara, **"A conditional random field framework for Thai morphological analysis,"** Proc. of the Fifth Int. Conf. on Language Resources and Evaluation (LREC-2006), 2006.
5. Aroonmanakun W., **"Thoughts on Word and Sentence Segmentation in Thai,"** Proceedings of the Seventh International Symposium on Natural Language Processing, 13th-15th September 2007, Pattaya, Thailand, pp. 85-90.
6. Haruechaiyasak, C., Kongyoung, S., and Dailey, M.N., **"A Comparative Study on Thai Word Segmentation Approaches,"** In *Proceedings of ECTI-CON*, 2008
7. S. P. Meyn and R.L. Tweedie, 2005. Markov Chains and Stochastic Stability. Second edition to appear, Cambridge University Press, 2008.

ภาคผนวก

คู่มือติดตั้งโปรแกรม

1. ติดตั้งโปรแกรม Python 2.5.2 จากไดเร็กทอรี่ tools\python-2.5.2.msi

2. คลิกขวาที่ My Computer เลือก Properties คลิก Advanced system settings

3. กดปุ่ม Environment Variables...

4. บริเวณ User variables คลิกปุ่ม New หรือ Edit เพื่อแก้ไขตัวแปร PATH

5. เพิ่ม ;C:\Python25\ ลงในต้วแปร PATH

6. กด OK

7. คัดลอกไดเร็กทอรี่ code\word_break ลงไปในเครื่องคอมพิวเตอร์

คู่มือการใช้งานโปรแกรม

การตัดคำ (ทีละไฟล์)

1. กด Start->Run พิมพ์ cmd แล้วกด Enter เพื่อเรียกหน้าต่าง Command line

2. เปลี่ยนไปที่ไดเร็กทอรี่ที่เก็บไฟล์ context_based.py ที่ได้ทำการคัดลอกไว้

3. เรียกคำสั่ง

```
python context_based.py INPUT_FILE OUTPUT_FILE
```

โดยที่ INPUT_FILE เป็น Path ไปยังไฟล์ที่ต้องการตัดคำ (ต้องมี encoding เป็น UTF-8)

โปรแกรมจะตัดคำ และบันทึกผลลัพธ์ไว้ที่ Path OUTPUT_FILE

การตัดคำ (แบบหลาย ๆ ไฟล์)

1. กด Start->Run พิมพ์ cmd แล้วกด Enter เพื่อเรียกหน้าต่าง Command line

2. เปลี่ยนไปที่ไดเร็กทอรี่ที่เก็บไฟล์ context_based.py ที่ได้ทำการคัดลอกไว้

3. เรียกคำสั่ง

```
python batch FILE_LIST
```

โดยที่ FILE_LIST เป็นไฟล์ที่เก็บรายชื่อไฟล์ที่ต้องการตัดคำ โปรแกรมจะตัดคำและบันทึกไฟล์ที่ตัดคำแล้วไว้ในไดเร็กทอรี่เดียวกัน โดยมีนามสกุลเป็น .cut

ตัวอย่างไฟล์เนื้อหาในไฟล์ FILE_LIST

c:\path\text1.txt

c:\path\text2.txt

c:\path\text3.txt

ผลลัพธ์จากการตัดคำจะเป็น

c:\path\text1.txt.cut

c:\path\text2.txt.cut

c:\path\text3.txt.cut