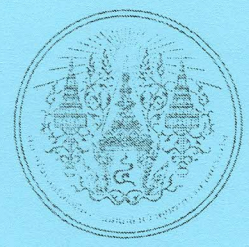


รายงานการวิจัยฉบับสมบูรณ์

การสร้างรายชื่อตัวละครโดยอัตโนมัติสำหรับงานระบุผู้พูด

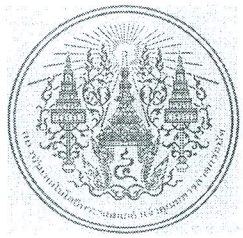
Automatic Avatar Construction for Speaker Identification Task

รศ.ดร. พรฤดี เนติโสภากุล

ได้รับทุนสนับสนุนงานวิจัยจากเงินรายได้ประจำปีงบประมาณ 2557

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง



รายงานการวิจัยฉบับสมบูรณ์

การสร้างรายชื่อตัวละครโดยอัตโนมัติสำหรับงานระบุผู้พูด

Automatic Avatar Construction for Speaker Identification Task

รศ.ดร. พรฤดี เนติโสภากุล

ได้รับทุนสนับสนุนงานวิจัยจากเงินรายได้ประจำปีงบประมาณ 2557

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

กิตติกรรมประกาศ

งานวิจัยเรื่อง “การสร้างรายชื่อตัวละครโดยอัตโนมัติสำหรับงานระบุผู้พูด” ได้รับทุนสนับสนุนการวิจัยจากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง จากแหล่งเงินรายได้คณะเทคโนโลยีสารสนเทศประจำปีงบประมาณ พ.ศ.2557 ผู้วิจัยขอขอบพระคุณคณะฯ และสถาบันฯ ที่ให้การสนับสนุนทุนวิจัยมา ณ ที่นี้

รศ.ดร. พรฤดี เนติโสภากุล

ชื่อโครงการ (ภาษาไทย) การสร้างรายชื่อตัวละครโดยอัตโนมัติสำหรับงานระบุผู้พูด
ชื่อโครงการ (ภาษาอังกฤษ) Automatic Avatar Construction for Speaker Identification Task
แหล่งเงิน คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ประจำปีงบประมาณ 2557 จำนวนเงินที่ได้รับการสนับสนุน 100,000 บาท
ระยะเวลาการทำวิจัย ตั้งแต่ 1 ตุลาคม พ.ศ. 2556 ถึง 30 กันยายน พ.ศ. 2557
ชื่อ-สกุล หัวหน้าโครงการ
รศ.ดร. พรฤดี เนติโสภากุล
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

งานวิจัยนี้ได้นำเสนอวิธีการแก้ปัญหาการระบุตัวละครอัตโนมัติ และการแก้สรรพนามจากนิทานเด็กภาษาไทย สำหรับการสกัดตัวละครอัตโนมัติ วิธีการที่ใช้คือการสกัดคำนามคัดเลือกจากข้อความโดยใช้ชุดของกฎนิพจน์ปกติ และการควบคุมจำนวนครั้งของการปรากฏซ้ำ ส่วนการแก้สรรพนามอัตโนมัติ ได้นำทฤษฎีเซนเตอร์ริ่งมาประยุกต์ใช้งาน การทดลองได้ถูกออกแบบมาเพื่อวัดประสิทธิภาพของวิธีการดังกล่าว พบว่า การสกัดตัวละครอัตโนมัติโดยใช้กฎนามวลี ให้ค่าการเรียกคืนที่ดีที่สุดประมาณ 78% สำหรับตัวละครหลัก ส่วนการแก้สรรพนามอัตโนมัติโดยใช้ทฤษฎีเซนเตอร์ริ่งที่ปรับปรุงแล้ว ให้ค่าความถูกต้องที่ดีที่สุดเทียบกับการแก้สรรพนามด้วยวิธีเดียวกันโดยมนุษย์อยู่ที่ 79 % เมื่อตัดวลีคำพูดออกไป

คำสำคัญ: การประมวลผลภาษา, การสกัดข้อความ, การระบุตัวละคร, การแก้คำสรรพนาม

Research Title: Automatic Avatar Construction for Speaker Identification Task
Researcher: Assoc. Prof. Ponrudee Netisopakul, Ph.D.
Faculty: Information Technology. **Department:** Information Technology.

ABSTRACT

This research devised two methods to solved two main problems related to automatic avatar identification and pronoun resolution in children stories in Thai language. For automatic avatar extraction, candidate noun phrases are extracted using a set of regular expression rules and controlled number of repetition. For automatic pronoun resolution, a centering theory and its variation is implemented. The experiment are designed to measure the effectiveness of these methods. Automatic avatar extraction using noun phrases rules achieved the best recall around 78% on main characters extraction; while automatic pronoun resolution using centering theory, comparing to centering process by human, achieved the best correctness around 79% when skipping quote phrases.

Keywords: Natural Language Processing, Information Extraction, Avatar Identification, Pronoun Resolution

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

งานสกัดรายชื่อตัวละคร (Avatar extraction) เป็นขั้นตอนหนึ่งและเป็นปัญหาที่สำคัญในงานวิจัยที่เกี่ยวข้องกับการระบุผู้พูด (Speaker Identification) จากหนังสือนิทานเด็ก หนังสือนิยาย หรือบทละครต่างๆ วัตถุประสงค์เพื่อให้คอมพิวเตอร์สามารถกำกับตัวละครโดยอัตโนมัติ เพื่อนำไปใช้ประยุกต์ในงานอื่นๆ เช่น การสร้างหนังสือนิทานเสียงโดยอัตโนมัติ [1] แม้ว่าการสกัดชื่อตัวละครจะเป็นเพียงขั้นตอนหนึ่ง แต่โดยตัวเองก็เป็นงานที่มีความซับซ้อนมาก ประกอบด้วยงานย่อยหลายงาน ได้แก่ การสกัดคำนามที่อาจเป็นชื่อตัวละคร การจำแนกลักษณะคำนามที่อาจเป็นตัวละครออกจากคำนามที่ไม่อาจเป็นชื่อตัวละคร การแก้คำสรรพนามว่าระบุถึงตัวละครหรือกลุ่มตัวละครใด รวมถึงการเรียกชื่อตัวละครด้วยคำที่แตกต่างกันแต่หมายถึงตัวละครเดียวกัน และการเรียกตัวละครที่ต่างกันด้วยคำหรือกลุ่มคำเดียวกัน

จะพบว่างานการสกัดรายชื่อตัวละครเพื่อใช้ระบุผู้พูดยังไม่มีผู้วิจัยค้นคว้าเท่าที่ควร งานวิจัยเกี่ยวกับการระบุผู้พูดที่ผ่านมา ส่วนมากจะทำการสร้างรายชื่อตัวละครโดยมนุษย์ก่อน [2,3,4] ในภาษาไทยพบงานระบุผู้พูดจากข้อความ [5] ซึ่งสร้างรายชื่อตัวละครโดยมนุษย์ก่อนเช่นกัน จากนั้นจึงนำรายชื่อตัวละคร (avatar list) ดังกล่าว มาเรียนรู้จดจำรูปแบบด้วยวิธีการต่างๆ เพื่อระบุผู้พูดต่อไป

ส่วนงานอื่นๆ ที่เกี่ยวข้องแต่ไม่ตรงนัก ได้แก่ งานการสกัดชื่อเฉพาะ (Named Entity Extraction) [6] และ งานการสกัดชื่อเฉพาะภาษาไทย [7,8,9,10] งานดังกล่าว อาจสกัดคำนามเฉพาะเชิงเวลา เชิงสถานที่ หรือชื่อบุคคล โดยมักมีสมมติฐานว่าชื่อบุคคล มักเป็นชื่อที่ไม่มีอยู่ในพจนานุกรม และอาจจะนำหน้าด้วยคำว่า นาย นาง นางสาว หรือ ชื่อยศ ตำแหน่ง บรรดาศักดิ์ หรือคำนำหน้าอื่นๆ โดยมักจะตามหลังด้วยคำกริยา อย่างไรก็ตาม สมมติฐานดังกล่าว ใช้ไม่ได้กับการสกัดชื่อตัวละครจากหนังสือนิทาน เนื่องจาก ในนิทาน ตัวละครมักเป็นคำทั่วไปที่พบได้ในพจนานุกรม เช่น หนูน้อย ผึ้งงาน นกกระสา กระต่าย ต้นไม้ เป็นต้น โดยคำระบุตัวละครดังกล่าว ไม่มีความแตกต่างที่สังเกตได้ชัดเจนจากคำนามอื่นๆ ที่ไม่ใช่ตัวละคร

อีกปัญหาสำคัญในการสกัดชื่อตัวละคร คือการแก้ปัญหาว่า สรรพนามต่างๆ ในนิทานหมายถึงตัวละครใด (Pronoun Resolution) โดยจะสำรวจเทคนิคการแก้สรรพนามโดยอัตโนมัติ และประเมินผลการทำงานเพื่อนำไปประยุกต์ใช้ในงานวิจัยเกี่ยวกับการระบุผู้พูดจากหนังสือนิทาน หนังสือนิยาย หรือบทละครต่างๆ ต่อไป

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 เพื่อคิดค้นอัลกอริทึมในสกัดรายชื่อตัวละครจากหนังสือนิทานเด็กโดยอัตโนมัติ

1.2.2 เพื่อศึกษาเทคนิคการแก้สรรพนามและปรับปรุงประยุกต์ใช้เทคนิคดังกล่าวในหนังสือนิทานเด็ก

1.2.3 เพื่อวัดประสิทธิภาพของอัลกอริทึมข้างต้น

1.3 ขอบเขตการวิจัย

1.3.1 อัลกอริทึมต้องสามารถสกัดตัวละครจากหนังสือนิทานเด็กได้ถูกต้องอย่างน้อย 70%

1.3.2 อัลกอริทึมต้องสามารถแยกตัวละครที่มีบทบาทออกจากตัวละครที่ไม่สำคัญ โดยมีประสิทธิภาพอย่างน้อย 60%

1.3.3 การทดลองและวัดประสิทธิภาพ ให้เทียบกับมนุษย์ โดยใช้ข้อมูลจากหนังสือนิทานจำนวนอย่างน้อย 40 เรื่อง

1.4 วิธีดำเนินการวิจัย

1.4.1 ศึกษางานวิจัยที่เกี่ยวข้องกับการสกัดชื่อตัวละครและการอ้างถึงตัวละคร

1.4.2 ศึกษางานวิจัยที่เกี่ยวข้องกับการแก้สรรพนาม

1.4.3 คิดค้นและปรับปรุงอัลกอริทึมที่ใช้ในการสร้างรายชื่อตัวละครอัตโนมัติ

1.4.4 คิดค้นและปรับปรุงอัลกอริทึมเพื่อแก้ปัญหา pronoun resolution

1.4.5 รวบรวมข้อมูลเพื่อสร้างชุดทดสอบและชุดเทียบผล

1.4.6 ทดลองและวัดประสิทธิภาพอัลกอริทึมที่ใช้ทั้งหมด

1.4.7 เขียนบทความวิจัยเผยแพร่ในงานประชุมนานาชาติ

1.4.8 จัดทำรายงานผลการดำเนินการ

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ได้อัลกอริทึมที่ใช้ในการสร้างรายชื่อตัวละครอัตโนมัติ

1.5.2 อัลกอริทึมที่ได้เป็นส่วนสำคัญในงานการระบุผู้พูดจากนิทานเด็ก

1.5.3 ระเบียบวิธีที่ได้สามารถประยุกต์ใช้กับงานวิจัยอื่นๆ ต่อไป

บทที่ 2
แนวคิด ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในงานวิจัยที่เกี่ยวข้องกับการระบุผู้พูดจากหนังสือนิทาน หนังสือนิยาย หรือบทละครต่างๆ จะมีขั้นตอนที่สำคัญขั้นตอนหนึ่งคือ การสกัดชื่อและกำกับชื่อตัวละครที่ปรากฏอยู่ภายในเนื้อเรื่องของนิทาน นิยาย หรือบทละคร ซึ่งเป็นส่วนหนึ่งของการสกัดชื่อเฉพาะ โดยมีงานวิจัยเกี่ยวกับการสกัดชื่อเฉพาะออกมาเป็นจำนวนมาก สามารถจำแนกแนวทางในการศึกษาได้เป็น 3 กลุ่ม [6] คือ

- ระบบที่ใช้กฎ ซึ่งเป็นระบบที่ต้องใช้ผู้เชี่ยวชาญในการสร้างกฎสำหรับการสกัดชื่อเฉพาะ
- ระบบที่ใช้วิธีทางสถิติและเทคนิคการเรียนรู้
- ระบบที่ใช้เทคนิคผสมระหว่างการสร้างกฎและการใช้เทคนิคการเรียนรู้

เมื่อทำการสกัดชื่อเฉพาะแล้ว ในหนังสือนิทาน หนังสือนิยาย หรือบทละคร ยังมีปัญหาที่สำคัญคือ ตัวละครตัวๆ หนึ่งมักถูกเรียกได้หลายๆชื่อ หรือแทนด้วยคำอื่นๆ ที่หมายถึงตัวละครนั้นๆ ดังตัวอย่างนิทานในภาพที่ 2.1

> "กินอาหารกลางวันที่ต่างจังหวัด--น่าสนใจเป็นที่สุด!" **หนูบ้าน**อุทานเมื่อได้รับบัตรเชิญจากญาติ**ของมัน** ซึ่งผู้นำข่าวนี้มาแจ้งก็คือคุณไปรษณีย์นกนั่นเอง "อากาศสดชื่น แสงแดดอ่อนๆ มีแต่ความเงียบและสงบสุข..." **หนูน้อย**ยังคงฝันเฟื่องต่อไป ขณะที่มองออกไปนอกหน้าต่างแล้วเห็นความวุ่นวายตรงถนนข้างล่าง "เชื่อเลยว่าอาหารต้องมีรสชาติวิเศษสุด เวลาที่เราพักผ่อนอยู่ในสวนดอกไม้ใต้ท้องฟ้ากว้าง"

ภาพที่ 2.1 แสดงตัวอย่างเนื้อเรื่องนิทานเรื่อง หนูบ้านกับหนูนา

จากภาพที่ 2.1 เมื่อมนุษย์อ่านนิทานเรื่องนี้ก็จะทราบได้ทันทีว่าตัวละคร "หนูน้อย" นั้นหมายถึงตัวละคร "หนูบ้าน" ที่ปรากฏในประโยคก่อนหน้า และคำว่า "ญาติของมัน" จะหมายถึงตัวละครตัวใด มนุษย์ก็จะสามารถทราบได้เมื่ออ่านเนื้อเรื่องต่อไปเรื่อยๆ แต่การจะทำให้คอมพิวเตอร์สามารถเข้าใจได้เหมือนมนุษย์เป็นงานที่ยุ่งยากซับซ้อน

2.1 ทฤษฎีเซ็นเตอร์ริ่ง (Centering theory: CT)

เนื่องจากเกิดความคลุมเครือการปัญหา Pronoun Resolution ที่ใช้วิธีที่ให้คำสรรพนามหมายถึงนามวลีล่าสุดที่ปรากฏ (History List) ซึ่งในบางประโยคอาจเกิดความคลุมเครือ เช่น "**Jack** drank **the wine** on **the table**. **It** was brown and round" เมื่อใช้ History List ในการแก้ปัญหาจะได้ว่า "**It**"

อ้างถึง “the table” แต่ในความเป็น “It” อ้างถึง “the wine” ดังนั้นจึงเกิดความคิดเกี่ยวกับคำที่เป็นศูนย์กลางในการเชื่อมโยง (discourse focus หรือ center) ระหว่างประโยค (local context หรือ utterance) นั้นก็คือทฤษฎีเซ็นเตอร์ริ่ง โดยทฤษฎีเซ็นเตอร์ริ่งจะสมมุติว่าประโยคในปริเฉท (discourse) จะมีคำที่เป็นศูนย์กลาง (CB: backward looking center) และ เซ็ตของนามวลีที่คาดว่าจะเป็นศูนย์กลาง (center) ตัวต่อไป (CF: forward looking center) โดยนามวลีใน CF จะเรียงตามเงื่อนไขของหลักไวยากรณ์ ซึ่งทั่วไปจะเรียงจากนามวลีที่เป็นประธาน (subject) กรรมตรง (direct object) กรรมรอง (indirect object) และนามวลีอื่นๆ นอกจากนี้ยังมีตัวแปรที่สำคัญ คือ นามวลีลำดับแรกจาก CF ที่ผ่านการเรียงลำดับแล้ว (CP: preferred next center) ซึ่งหมายถึงนามวลีที่มีโอกาสเป็นศูนย์กลางตัวต่อไปมากที่สุด กฎและเงื่อนไขของทฤษฎีเซ็นเตอร์ริ่งมีดังนี้

เงื่อนไขสำหรับแต่ละประโยค (U_i) ในปริจเฉท (U_1, U_2, U_3, ...,U_n) เมื่อ i =1, 2, 3, ... ,n

1. ต้องมี CB ที่ชัดเจนตัวเดียว
2. นามวลีทุกตัวใน CF_i ต้องสามารถถูกอ้างถึงได้
3. คำที่เป็นศูนย์กลาง (CB_i) คือ นามวลีตัวแรกของ CF_{i-1} ที่ถูกอ้างถึงในประโยค U_i

กฎของทฤษฎีเซ็นเตอร์ริ่ง คือ

1. ถ้านามวลีตัวใน CF_i ถูกอ้างถึงด้วยคำสรรพนามในประโยค U_{i+1} แล้วศูนย์กลาง CB_{i+1} ต้องถูกอ้างถึงด้วยคำสรรพนามเช่นเดียวกัน นั้นก็คือคำสรรพนาม 2 อ้างถึงสิ่งเดียวกัน
2. ลำดับของความสำคัญของสถานะคือ continue, retain, smooth-shift และ rough-shift การกำหนดสถานะ กำหนดตามตารางที่ 2.1

ตารางที่ 2.1 กฎของการกำหนดสถานะ

	$CB_i = CB_{i-1}$ หรือ CB_i = NULL	$CB_i \neq CB_{i-1}$
$CB_i = CP_i$	CONTINUE	SMOOTH-SHIFT
$CB_i \neq CP_i$	RETAIN	ROUGH-SHIFT

ตัวอย่างที่ 1

a. *Jack$_1$ saw him$_2$ in the park$_3$*

b. *He$_4$ was riding a bike$_5$*

ภาพที่ 2.2 ตัวอย่างที่ 1 สำหรับการแก้สรรพนามด้วย CT

เมื่อพิจารณาตัวอย่างที่ 1 ดังภาพที่ 2.2 ด้วยทฤษฎีเซ็นเตอร์ริ่งจะได้ว่าCF_aคือ “Jack”, “him”, “the park” ตามลำดับ ดังนั้น CP_a คือ “Jack” และ CB_a คือ “him” แล้วเมื่อพิจารณา U_b ถ้าดูจากความหมาย

"He_4" อาจจะหมายถึง "Jack" หรือ "him" แต่อย่างไรก็ตาม เมื่อพิจารณาตามทฤษฎีเซ็นเตอร์ริ่งแล้ว "He_4" นั้นหมายถึง "him" เนื่องจากมีสถานะเป็น continue ทำให้ CB ไม่เปลี่ยน

ตัวอย่างที่ 2

a. *While $Jack_1$ was walking in the $park_2$, he_1 met Sam_3*

b. *He_4 invited him_5 to the $party_6$*

ภาพที่ 2.3 ตัวอย่างที่ 2 สำหรับการแก้สรรพนามด้วย CT

จากตัวอย่างที่ 2 ดังภาพที่ 2.3 CB_a คือ "Jack" เนื่องจากมีคำสรรพนามเพียงตัวเดียวในประโยคดังนั้น "he" จึงเป็น CB_a ซึ่งก็คือ "Jack" นั้นเอง เมื่อพิจารณา U_b "He" อาจจะหมายถึง "Jack-continue" หรือ "Sam-smooth shift" จากกฎข้อที่ 2 ของเซ็นเตอร์ริ่ง "He_4" นั้นหมายถึง "Jack"

2.2 งานวิจัยที่เกี่ยวข้อง

สำหรับงานวิจัยที่เกี่ยวกับการระบุผู้พูดจากหนังสือนิทาน นิยาย หรือบทละครที่ผ่านมา งานวิจัย "Identifying Speakers in Children's Stories for Speech Synthesis" [2] มีการใช้เทคนิค pattern matching ในการสกัดชื่อเฉพาะ หรือนามวลีที่อาจจะเป็นตัวละครในเรื่องก่อน แล้วจึงทำการระบุผู้พูด แต่ไม่ได้มีการแก้ปัญหาตัวละครตัวหนึ่งมีชื่อเรียก หรือคำที่อ้างถึงอื่นๆ

ถัดมาได้มีงานวิจัยเกี่ยวกับการระบุผู้พูดที่สามารถระบุผู้พูดได้ดีมากขึ้น คือ "Hierachical Rule Generalisation for Speaker Identification in Fiction Book" [3] และ "A naïve salience-based method for speaker identification in fiction books" [4] งานวิจัย [3] เป็นการระบุผู้พูดโดยใช้เทคนิค hierarchical rule pattern matching ในการระบุผู้พูดซึ่งต้องอาศัยข้อมูลชุดฝึกสอนก่อน ทำให้งานวิจัยนี้เหมาะกับหนังสือที่มีผู้แต่งคนเดียวกันกับหนังสือที่ใช้เป็นชุดฝึกสอน งานวิจัย [4] เป็นงานวิจัยที่แก้ปัญหาของงานวิจัย [3] โดยการระบุผู้พูดในการวิจัยนี้ไม่จำเป็นต้องอาศัยชุดฝึกสอน ถึงอย่างไรก็ตาม งานวิจัยทั้งสองเลือกใช้วิธีการสร้างรายชื่อตัวละคร (avatar list) และคำที่อ้างถึงอื่นๆ ด้วยมือ เพื่อช่วยแก้ปัญหาดังกล่าว

สำหรับงานวิจัยการระบุผู้พูดในภาษาไทย "Semi-automatic Novel Text Classification based on Character" [5] ซึ่งใช้แบบจำลองภาษา N-gram ในการระบุผู้พูด ยังคงมีการระบุตัวละครในนิยายล่วงหน้าด้วยมือเช่นเดียวกัน

ดังนั้น การสกัดรายชื่อตัวละครโดยอัตโนมัติจึงเป็นหัวข้อวิจัยที่ใหม่และท้าทาย

นอกจากนี้ ในปัญหาเรื่องการเรียกชื่อหรือสรรพนามที่ต่างกัน แต่อ้างอิงไปยังตัวละครเดียวกันนั้น งานวิจัยด้านนี้เรียกว่า Coreference Resolution หรือ Pronoun Resolution

ที่ผ่านมา "A Machine Learning Approach to Coreference Resolution of Noun Phrases" [11] เป็นงานวิจัยแรกที่นำเสนอการใช้เทคนิคการเรียนรู้เพื่อแก้ปัญหา Coreference Resolution ซึ่งงานวิจัยนี้ไม่เพียงแต่แก้ปัญหาสรรพนามที่ใช้อ้างถึงได้ แต่ยังแก้ปัญหาคำนามทั่วไปที่ใช้อ้างถึงได้อีกด้วย โดยคุณสมบัติ (features) ที่ใช้สำหรับการเรียนรู้ด้วย Decision Tree มีทั้งหมด 12 คุณสมบัติ (features) คือ

ให้ i คือ คำที่เกิดขึ้นก่อน (the potential antecedent)

j คือ คำที่อาจอ้างถึง (the anaphor)

1. Distance Feature (DIST) ระยะทางระหว่างประโยคที่ i และ j ปรากฏ ซึ่งมีค่าที่เป็นไปได้คือ 0, 1, 2, 3, ...
2. i-Pronoun Feature (I_PRONOUN) ถ้า i เป็นคำสรรพนาม ค่าที่เป็นไปได้คือ จริงหรือเท็จ
3. j-Pronoun Feature (J_PRONOUN) ถ้า j เป็นคำสรรพนาม ค่าที่เป็นไปได้คือ จริงหรือเท็จ
4. String Match Feature (STR_MATCH) i และ j เป็นคำเดียวกันเมื่อเอาคำนำหน้านามออกค่าที่เป็นไปได้คือ จริงหรือเท็จ
5. Definite Noun Phrase Feature (DEF_NP) ถ้า j เป็นนามวลี (noun phrase) ค่าที่เป็นไปได้คือ จริงหรือเท็จ
6. Demonstrative Noun Phrase Feature (DEM_NP) ถ้า j เป็นนามวลีที่ชี้เฉพาะ (noun phrase) ค่าที่เป็นไปได้คือ จริงหรือเท็จ
7. Number Agreement Feature (NUMBER) ถ้า i และ j เป็นพหูพจน์หรือเอกพจน์ เหมือนกัน ค่าที่เป็นไปได้คือ จริงหรือเท็จ
8. Semantic Class Agreement Feature (SEMCLASS) ถ้า i และ j ถูกจัดอยู่ในคลาสเดียวกันตาม ISA hierarchy ค่าที่เป็นไปได้คือ จริง เท็จ และ unknown
9. Gender Agreement Feature (GENDER) i และ j เป็นเพศเดียวกัน หรือคนละเพศ หรือไม่สามารถระบุได้ ค่าที่เป็นไปได้คือ จริง เท็จ และ unknown
10. Proper-Names Feature (PROPER_NAME) ถ้า i และ j คือชื่อเฉพาะทั้งคู่ ค่าที่เป็นไปได้คือ จริงหรือเท็จ
11. Alias Feature (ALIAS) ถ้า i เป็นชื่อย่อหรือนามแฝงของ j หรือในทางกลับกัน j เป็นชื่อย่อหรือนามแฝงของ i ค่าที่เป็นไปได้คือ จริงหรือเท็จ
12. Appositive Feature (APPOSITIVE) ถ้า j เป็นกลุ่มคำนามที่ทำหน้าที่ขยาย i ค่าที่เป็นไปได้คือ จริงหรือเท็จ

ถัดมา ได้มีงานวิจัยสำหรับแก้ปัญหา Coreference Resolution คือ “Employing the Centering Theory in Pronoun Resolution from the Semantic Perspective” [12] เป็นงานวิจัยที่เทคนิคการเรียนรู้ที่ได้ปรับปรุงเพิ่มเติมจากงานวิจัย [11] โดยใช้เพิ่ม features ที่ใช้ในการเรียนรู้ ซึ่ง features ที่ได้มาจากแนวคิดของ Centering Theory

ถึงอย่างไรก็ตาม งานวิจัย [11][12] ทำการทดลองกับคลังข้อมูลที่เป็นข่าวภาษาอังกฤษ ส่วนงานวิจัยที่เราได้นำเสนอจะเป็นโดเมนนิทานภาษาไทย ซึ่งมีข้อแตกต่างหลายประการ เช่น ภาษาไทยไม่มีข้อมูลที่บ่งบอกชื่อเฉพาะ ไม่มีการเว้นวรรคหรือใช้อักขระพิเศษในการแบ่งคำ หลักเกณฑ์ในการสร้างคำนามค่อนข้างคลุมเครือ ทำให้ features บางอย่างอาจใช้ไม่ได้ และอาจต้องเพิ่มเติม features อื่นๆ ซึ่งงานวิจัยที่ได้ทบทวน Centering Theory สำหรับภาษาไทย โดยเน้นที่การแก้สรรพนามที่ละไว้ (Zero anaphora) คือ “Zero Pronoun Resolution in Thai: A Centering Approach” [13] อย่างไรก็ตาม เนื่องจากงานดังกล่าว เน้นที่ zero anaphora เท่านั้น จึงยังไม่มีความชัดเจนว่า ประสิทธิภาพของ Centering Theory สำหรับการแก้สรรพนามที่มีการเอ่ยถึงเป็นเท่าไร

สังเกตว่า ส่วนการทดสอบใน [13] มิได้ใช้การทำงานโดยอัตโนมัติ แต่ใช้ผู้เชี่ยวชาญมนุษย์ในการทดสอบ ว่าการแก้สรรพนามเป็นไปตามทฤษฎีหรือไม่ นับว่าเป็นการทดสอบความน่าเชื่อถือของทฤษฎีโดยมนุษย์ ซึ่งน่าสงสัยว่า ทฤษฎีแก้สรรพนามด้วย Centering Theory สามารถใช้แก้สรรพนามด้วยคอมพิวเตอร์โดยอัตโนมัติหรือไม่

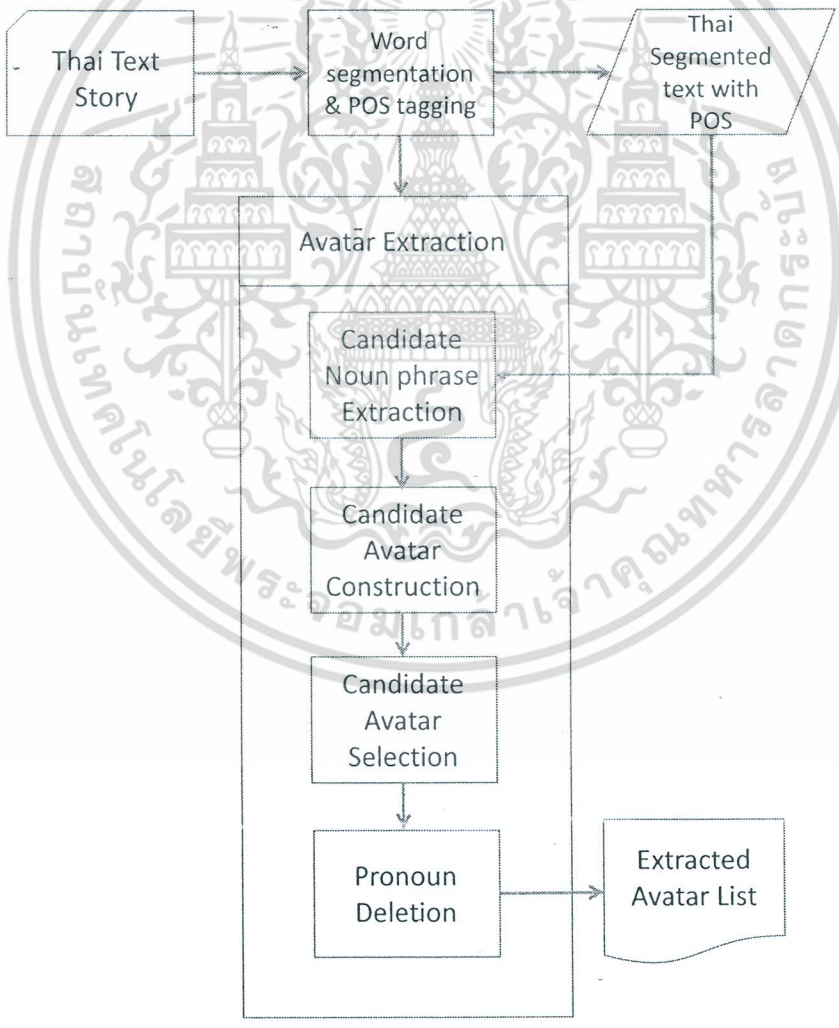
ดังนั้น งานวิจัยนี้ จะคิดค้นอัลกอริทึมในการสกัดรายชื่อตัวละคร รวมถึงทดสอบทฤษฎี Centering Theory โดยออกแบบอัลกอริทึมตามหลักการดังกล่าว

บทที่ 3
วิธีดำเนินการวิจัย

การทำงานวิจัยนี้แบ่งเป็นสองส่วนคือ ส่วนการสกัดตัวละคร และส่วนแก้สรรพนาม ดังนี้

3.1 อัลกอริทึมสกัดตัวละครอัตโนมัติจากนิทานเด็ก

การสร้างอัลกอรึทึมสกัดตัวละครอัตโนมัติตั้งอยู่บนสมมุติฐานที่ว่า ส่วนของคำที่เรียงกันเป็นตัวละครจะเป็นคำนามที่มีรูปแบบของ part of speech (POS) อยู่รูปแบบหนึ่ง โดยรูปแบบดังกล่าวจะปรากฏซ้ำๆ กันในนิทานส่วนใหญ่ โดยเฉพาะถ้าใช้อัลกอรึทึมนี้สำหรับการสกัดตัวละครหลัก การปรากฏของตัวละครหลักจะเกิดขึ้นหลายครั้งในนิทานเรื่องนั้น แต่จะเกิดซ้ำน้อยลงสำหรับตัวละครอื่นๆ



ภาพที่ 3.1 ขั้นตอนการสกัดตัวละครอัตโนมัติจากนิทานเด็ก

ดังนั้น จากภาพที่ 3.1 ขั้นตอนการสกัดตัวละครอัตโนมัติจากนิทานเด็ก ประกอบด้วยขั้นตอนหลัก 2 ขั้นตอน คือ ขั้นตอนการเตรียมข้อมูลจากข้อความภาษาไทย ประกอบด้วยการตัดคำและการกำกับ part of speech (POS) ผลลัพธ์จะได้ข้อความภาษาไทยที่มีการตัดคำและกำกับ POS ให้กับแต่ละคำแล้ว มีข้อสังเกตคือการตัดคำดังกล่าว มีส่วนที่ตัดคำไม่ถูกต้องอยู่มากพอสมควร เช่น คำว่า ลิลลี่ทุ่ง ตัดเป็น ลิ/ncn ลลี่ทุ่ง/ncn ซาร่าห์แสนสวย ตัดเป็น ซา/npn ราห์แสน/npn สวย/vi เป็นต้น อย่างไรก็ตาม ในการทดลองนี้จะไม่มีการแก้ไขการตัดคำและกำกับข้อความที่ผิดพลาดแต่อย่างใด

ขั้นตอนถัดมาคือ การสกัดตัวละคร (Avatar Extraction) โดยอ่านประโยคในนิทานเข้ามาทีละประโยค หากประโยคดังกล่าวเป็นคำพูดจะตัดทิ้งไป การประมวลผลประโยคประกอบด้วย 4 ขั้นตอนย่อย คือ การสกัดกลุ่มคำนามคัดเลือก (Candidate Noun Phrase Extraction) การสร้างตัวละครคัดเลือก (Candidate Avatar Construction) การคัดเลือกตัวละครเป้าหมาย (Candidate Avatar Selection) รายละเอียดของ 3 ขั้นตอนย่อยนี้จะอธิบายเพิ่มเติมในส่วนถัดไป และสุดท้ายคือการตัดสรรพนามออกไปจากตัวละครที่คัดเลือกมา เนื่องจากสรรพนามโดดๆ ไม่อาจจัดเป็นตัวละคร

3.1.1 การสกัดกลุ่มคำนามคัดเลือก (Candidate Noun Phrase Extraction)

เป็นการสกัดกลุ่มคำนามที่มีสิทธิ์เป็นตัวละครหรือเป็นส่วนหนึ่งของคำที่ประกอบกันเป็นตัวละคร เช่น (หนู/ncn บ้าน/ncn) (เจ้า/ntit ลา/vt) (ต้น/ncn มะกอก/npn) การสกัดในขั้นตอนนี้ใช้หลักการจับคู่นิพจน์ปกติ (Regular expression matching) ซึ่งหากยิ่งมีกฎคำนามของนิพจน์ปกติจำนวนมาก จำนวนคำนามคัดเลือกที่สกัดออกมาได้ก็ยิ่งมีมาก อย่างไรก็ตาม หากมีจำนวนกฎมากเกินไป ก็จะทำให้ได้คำนามคัดเลือกมากเกินไป ส่งผลให้ค่า precision ต่ำมาก ตารางที่ 3.1 แสดงกฎนิพจน์ปกติที่ใช้และตัวอย่างคำนามคัดเลือกที่สกัดโดยกฎดังกล่าว

นอกจากนี้ ในอัลกอริทึมนี้ การเรียงลำดับของกฎมีผลต่อการสกัด เนื่องจากการทำงานของอัลกอริทึมที่ใช้จะทำการจับคู่เพียงครั้งเดียวตามลำดับที่เรียงไว้ ดังนั้น คำนามที่สกัดได้แล้วด้วยกฎที่มาก่อนจะไม่ถูกสกัดอีกครั้งด้วยกฎต่อๆ มา

ตารางที่ 3.1 ตัวอย่างกฎที่ใช้ในการสกัดคำนามคัดเลือก

rule #	regular expression	example extracted noun phrase
1	<NTIT><NCN>	(เจ้า/ntit หญิง/ncn) (คุณ/ntit ปู่/ncn)
2	<NTIT><NPN>	(เจ้า/ntit หนู/npn) (เจ้า/ntit สะพานลอย/npn) (เจ้า/ntit ตุ่น/npn) (เทพเจ้า/ntit เฮอร์มีส/npn) (พระ/ntit อาทิตย์/npn)

rule #	regular expression	example extracted noun phrase
3	<PREF3><NCN>	(ชาว/pref3 นา/ncn) (ชาว/pref3 เมือง/ncn)
4	<PREF2><NCN>	(ผู้/pref2 เฒ่า/ncn)
5	<NTIT><VT>	(เจ้า/ntit ลา/vt)
6	<NCN><NCN><NCN>	(หมาจิ้งจอก/ncn จอม/ncn เจ้าเล่ห์/ncn)
7	<NCN><VT><NCN>	(คน/ncn ตัด/vt ไม้/ncn)
8	<NCN><PPER>	(เนิน/ncn เขา/pper)
9	<NCN><VI>	(ยาย/ncn แห้ง/vi)
10	<NCN><NCN>	(หญิง/ncn สาว/ncn) (ลม/ncn เหนือ/ncn)
11	<NCN><NPN>	(ต้น/ncn หลิว/npn) (นก/ncn กระสา/npn) (หญิง/ncn ชรา/npn)
12	<PPER>	(สะพานลอย/pper)
13	<NCN>	(ราชสีห์/ncn)
14	<NPN>	(หมาจิ้งจอก/npn)

อย่างไรก็ตาม พบคำนามที่กฎยังไม่ครอบคลุมจำนวนหนึ่ง ซึ่งต่อมาปรากฏเป็นตัวละคร ดังแสดงในตารางที่ 3.2 ข้อสังเกตคือ คำนามที่ขึ้นต้นด้วย <NCN> <NTIT> และ <PPER> มีโอกาสเป็นตัวละครยิ่งกว่านั้น ปรากฏว่า ตัวละครบางตัวกลับถูกกำกับด้วย <VI>

ตารางที่ 3.2 ตัวอย่างคำนามที่กฎไม่ครอบคลุมแต่เป็นตัวละคร

ต้น/ncn อ้อ/int	ต้น/ncn หลิว/vt	เมฆ/ncn น้อย/adv
กวี/ncn หนุ่ม/adj	เจ้า/ncn ด่าง/adj	แมลง/ncn ทับ/vt ชรา/npn
เจ้า/ntit หญิง/ncn ปุยฝ้าย/vt	เจ้า/ntit ลูก/ncn ม้า/ncn	ราชินี/npn ผึ้ง/ncn
เทพเจ้า/ntit ซี/npn อุส/npn	เจ้า/ntit ไมกี้ก็/part	เจ้า/ntit โฟแลตก็/part

คุณ/pper ยาย/ncn	หนู/pper น้อย/vi	เจ้า/pper ลา/vt
สะพานลอย/vi เกเร/vi	กระต่าย/vi	หมาป่า/vi

3.1.2 การสร้างตัวละครคัดเลือก (Candidate Avatar Construction)

ขั้นตอนนี้เป็นการรวมเอาคำนามคัดเลือกที่ยังถูกแยกด้วย POS เข้าด้วยกัน โดยตัดส่วน POS ออกไป ทำให้ได้คำนามที่ครบถ้วนเป็นตัวละครเท่านั้น เนื่องจากถัดจากนี้เราจะทำงานกับข้อความที่อาจเป็นตัวละครเท่านั้น เรียกว่า ตัวละครคัดเลือก ผลลัพธ์ของขั้นตอนนี้คือรายการตัวละครคัดเลือกในนิทานแต่ละเรื่อง โดยแต่ละตัวละครคัดเลือกจะปรากฏซ้ำๆ กันหลายครั้ง เพื่อใช้ในขั้นตอนถัดไป

3.1.3 การคัดเลือกตัวละครเป้าหมาย (Candidate Avatar Selection)

ขั้นตอนนี้ จะคัดตัวละครจากจำนวนครั้งที่ปรากฏ โดยจะตัดตัวละครคัดเลือกที่ปรากฏน้อยกว่าจำนวนครั้งที่กำหนดออกไป โดยในการทดลองวัดประสิทธิภาพจะกำหนดจำนวนครั้งขั้นต่ำไว้ที่มากกว่า 2 ครั้ง 3 ครั้ง และ 4 ครั้งขึ้นไป โดยจะตัดคำสรรพนามโดด เช่น เขา มัน เธอ ออกไปจากตัวละครเป้าหมายด้วย

3.2 อัลกอริทึมการแก้สรรพนาม (Pronoun Resolution)

การทำงานของอัลกอรีทึมนี้ จะอิมพลีเมนต์ทฤษฎีเซ็นเตอร์ริ่ง (Centering theory) ให้ทำงานโดยอัตโนมัติ นอกจากนี้ ยังมีการปรับปรุงทฤษฎีเซ็นเตอร์ริ่ง (Centering theory) ตามภาพที่ 3.2 และภาพที่ 3.3 เป็นดังนี้

อินพุตของระบบเป็นไฟล์ดอทเอ็กซ์เอ็มแอล (.XML) ที่มีการแบ่งวลี (phrase) การตัดคำ(word segmentation) กำกับหน้าที่ของคำ (POS tagging) และกำกับชื่อตัวละคร (Avatar annotation) เรียบร้อยแล้ว จากนั้นแปลงให้อยู่ในรูปโครงสร้างข้อมูล W โดยสมาชิกของ W แต่ละตัวตามโครงสร้างของข้อมูลจะประกอบด้วย คำในวลี (words) และ นามวลี (noun phrases) ซึ่งหมายถึงชื่อตัวละครหรือคำสรรพนาม (pronoun) ที่ใช้เรียกตัวละคร จำนวนสมาชิของ W ขึ้นอยู่กับจำนวนของวลีเรื่องนั้นๆ โดยอัลกอริทึมจะทำการปรับปรุงรายการต่อไปนี้

Forward Centers (CFi) หมายถึง รายการตัวละครในประโยคปัจจุบันที่อาจถูกอ้างอิงโดยคำสรรพนามปัจจุบัน

Preferred Center (CPi) หมายถึง ตัวละครที่สำคัญที่สุดจากรายการคำนาม CFi โดยปกติจะเลือกตัวละครที่ถูกเอ่ยถึงตัวแรกในประโยคปัจจุบัน

Backward Center (CBi) หมายถึง ตัวละครที่ถูกอ้างอิงถึงโดยคำสรรพนาม และเป็นเซนเตอร์ปัจจุบันของเนื้อความ

การทำงานจะเริ่มแก้สรรพนาม (pronoun) ที่ละตัวในแต่ละวลี (Wi) ตามทฤษฎีเซ็นเตอร์ริ่งที่ถูกปรับปรุงแล้ว คือ

1. เมื่อวลีแรกที่มีนามวลีปรากฏอยู่ ซึ่งโดยทั่วไปจะไม่พบคำสรรพนามอยู่ในวลีแรกที่นามวลีปรากฏจะพบแต่นามวลีที่เป็นชื่อตัวละคร เนื่องจากต้องมีการกล่าวถึงชื่อตัวละครก่อน จึงมีการใช้คำสรรพนามแทนตัวละคร ดังนั้นตัวแปร CFi ของวลีนี้จะมีค่าเป็นรายชื่อของตัวละครที่ปรากฏในวลีนี้, ตัวแปร CPi ก็จะเป็นชื่อตัวละครตัวแรกของรายชื่อตัวละครของวลีนี้ ตามทฤษฎี และตัวแปร CBi จะมีค่าเป็น null แล้วทั้งสามตัวแปรจะถูกเก็บในตัวแปร listofelementupdate ตามตำแหน่งของค่าตัวแปร CB_index ซึ่งจะมีค่าเพิ่มขึ้นตามลำดับเมื่อพบนามวลี
2. เมื่อวลีต่อมามีชื่อตัวละครหรือคำสรรพนามปรากฏอยู่ แบ่งเป็น 2 ขั้นตอนคือ
   2.1 เมื่อพบคำสรรพนาม อัลกอริทึมจะทำการแก้ pronoun resolution ของคำสรรพนามที่พบในวลีที่ละตัว เริ่มจากการจำลอง CF ที่เป็นไปได้ของวลีที่กำลังพิจารณาอยู่ (ตำแหน่งที่ i) (ตัวแปร CFi_potential) โดยที่ค่าของตัวแปร CFi_potential จะมีค่าเท่ากับรายชื่อนามวลีของวลี เมื่อคำสรรพนามนั้นเป็นคำสรรพนามแรกที่พบ ถ้าไม่ใช่ ตัวแปร CFi_potential จะมีค่าเท่ากับรายชื่อของนามวลีที่คำสรรพนามก่อนหน้าได้ถูกแก้ไขแล้ว นั้นก็คือค่าของ CF ในตัวแปร listofelementupdate[CB_index-1] จากนั้นทำการหาชื่อตัวละครที่เป็นไปได้สำหรับสรรพนามที่กำลังพิจารณาอยู่ โดยใช้นามวลีจาก CF จากวลีที่มีนามวลีก่อนหน้าวลีปัจจุบัน ตามกฎข้อที่ 1 ของทฤษฎีเซ็นเตอร์ริ่ง ร่วมกับชื่อตัวละครอื่นในวลีที่มีการกล่าวถึงก่อนคำสรรพนามนั้น โดยเรียงลำดับชื่อตัวละครที่อยู่ในวลีก่อนชื่อตัวละครที่จาก CF จากวลีก่อนหน้า ผลลัพธ์ที่ได้จะเก็บไว้ในตัวแปร np_potentail_list แล้วทำการจับคู่คำสรรพนามและชื่อตัวละครที่เป็นได้ทั้งหมด พร้อมจำลอง CF ที่เป็นไปได้ เมื่อสรรพนามถูกแทนด้วยชื่อตัวละครที่เป็นไปได้ โดยจะเก็บข้อมูลไว้ในตัวแปร CFi_update จากนั้นนำข้อมูลของคำสรรพนาม, ชื่อตัวละครที่มีโอกาสเป็นคำตอบ และ CFi_update ของคู่สรรพนามและชื่อตัวละครนั้น ในโครงสร้างข้อมูล rule1workinglist แล้วส่งไปประมวลผลที่ฟังชันก์ backwardlooking() ซึ่งในฟังชันก์นี้จะทำการหาสถานะของคำสรรนามกับชื่อตัวละครแต่ละคู่ว่ามีสถานะเป็น continue, retain, smooth-shift หรือ rough-shift แล้วทำการเปรียบเทียบเพื่อหาสถานะที่มีความสำคัญมากที่สุด ตามกฎข้อที่ 3 ของทฤษฎี คือจะเลือกจาก continue, retain, smooth-shift และ rough-shift ตามลำดับ แต่งานวิจัยนี้ได้เพิ่มกฎฮิวริสติก (Heuristic) คือ คำสรรพนามที่อยู่ในวลีเดียวกันแต่ต่างกัน ต้องอ้างถึงตัวละครที่ต่างกันด้วย จากนั้นฟังชันก์ backwardlooking() จะส่งคู่สรรพนามและชื่อตัวละครที่เป็นคำตอบ พร้อมค่าในตัวแปร CFi_update นั้นก็คือ CFi

ที่ทำการแก้ไขคำสรรพนามเรียบร้อย เพื่อใช้ในการแก้ไขคำสรรพนามตัวต่อ โดยจะถูกเก็บค่าลงในตัวแปร listofelementupdate โดยที่ CPi ก็คือ ชื่อตัวละครตัวแรกในตัวแปร CFi_update และ CBi คือชื่อตัวละครที่ระบบเลือกเป็นคำตอบของคำสรรพนามนี้

2.2 ถ้าไม่มีคำสรรพนามปรากฏในวลี ดังนั้น ตัวแปร CFi ของวลีนี้จะมีค่าเป็น รายชื่อของตัวละครที่ปรากฏในวลีนี้, ตัวแปร CPi ก็จะเป็นชื่อตัวละครตัวแรกของรายชื่อตัวละครของวลีนี้ตามทฤษฎี และตัวแปร CBi จะมีค่าตามตัวแปร CPi จากการตีความตามทฤษฎีของผู้วิจัย แล้วเก็บทั้งสามตัวแปรในตัวแปร listofelementupdate ตามตำแหน่งของ ค่าตัวแปร CB_index

3. เมื่อวลีไม่มีชื่อตัวละครหรือคำสรรพนามใดให้ข้ามไปในวลีถัดไป

ในตารางที่ 3.3 จะแสดงการเปรียบเทียบอัลกอริทึมกับทฤษฎีเซ็นเตอร์ริ่ง เพื่อให้สังเกตความเกี่ยวข้องและแสดงในส่วนของอัลกอริทึมที่ทำการปรับปรุง

ตารางที่ 3.3 ความสัมพันธ์ของอัลกอริทึมกับทฤษฎีเซ็นเตอร์ริ่ง และอัลกอริทึมที่ปรับปรุง

อัลกอริทึมที่ตรงกับทฤษฎีเซ็นเตอร์ริ่ง (CA)	อัลกอริทึมที่ได้ปรับปรุงเพิ่มเติม (UCA)
1. ชื่อตัวละครที่เป็นไปได้ที่อาจจะเป็น CB มาจาก CF ของวลีก่อนหน้า ซึ่งตรงกับ กฎข้อที่ 1 ของ ทฤษฎีเซ็นเตอร์ริ่ง ที่ว่า "If any element of $CF(U_n)$ is realized by a pronoun in U_{i+1}, then the $CB(U_{i+1})$ must be realized by a pronoun also" [13] 2. ทฤษฎีเซ็นเตอร์ริ่งทำการเลือกชื่อตัวละครที่มีความสำคัญของสถานะมากที่สุดตาม กฎข้อที่ 2 ของ ทฤษฎีเซ็นเตอร์ริ่ง "continue > retain > smooth-shift > rough-shift" 3. เมื่อไม่มีคำสรรพนามปรากฏในวลีที่กำลังพิจารณาอยู่ CB จะมีค่าเท่ากับ CP ของวลีนั้นตามเงื่อนไขที่ว่า "The center, $CB(U_i)$, is the highest-ranked element of $CF(U_{i-1})$ that is realized in U_i "	1. ทำการปรับปรุงอัลกอริทึมเซ็นเตอร์ริ่ง โดยการเพิ่มรายชื่อตัวละครที่ปรากฏในวลีเดียวกับสรรพนาม และรายชื่อตัวละครที่มากจากรายการ CF ของวลีก่อนหน้า ในรายการตัวละครที่คำสรรพนามอาจอ้างอิงถึงได้ โดยเรียงลำดับความสำคัญจากชื่อตัวละครในวลีปัจจุบัน ก่อนชื่อตัวละครที่จาก CF ของวลีก่อนหน้า 2. เพิ่มกฎฮิวริสติกที่ว่า "คำสรรพนามที่อยู่ในวลีเดียวกันแต่ต่างกัน ต้องอ้างถึงตัวละครที่ต่างกันด้วย"

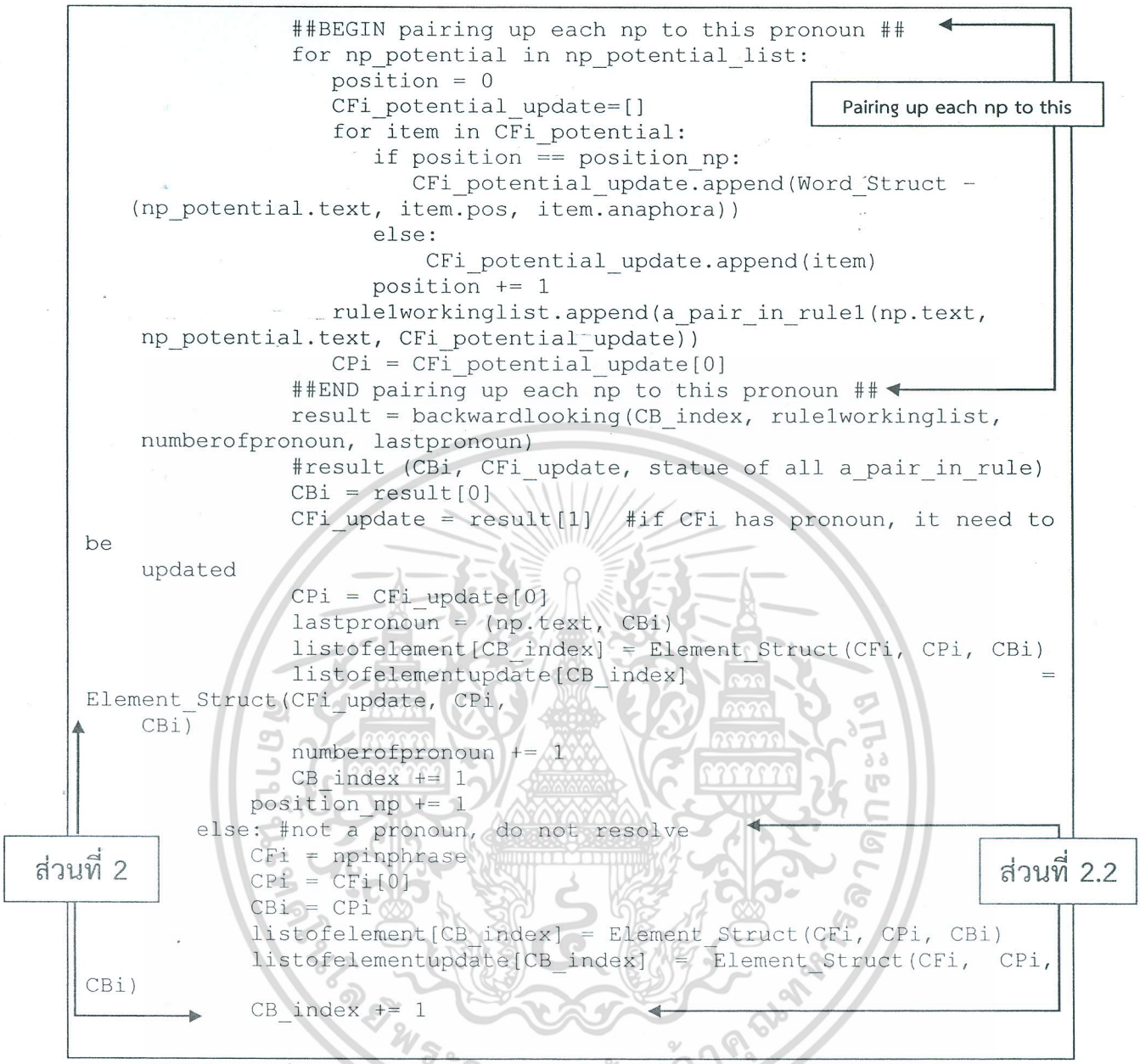
```
# Main process
listofelement = {}
# keep {(1: class element_struct1), (2: class element_struct2),...}
listofelementupdate = {}
# same as above but update pronoun CFi to be NP CFi
#structure of W (as input of this program)
# Length of W is the number of phrases in the story
# Each phase contains a pair of (list of WORDS in a phrase, list of
NPs in a phrase)
# WORDS contains (text, pos, is_pronoun (Y/N))
# NPS contains (text, pos, is_pronoun (Y/N)) as subset of WORDS
----BEGIN main process ----
  W = NP_Entity(story_id)
  #list of pairs [(list of word, list of NP in phrase)]; list of word
= [(word,pos,anaphora)], list of NP = [(NP or pper, pos, anaphora)]
  CB_index = 0 #is a number of pronouns to resolve so far for this
story
  for i in range(0,len(W)):
     CFi = [] # variable keep CF of centering theory
     CPi = '' # variable keep CP of centering theory
     CBi = '' # variable keep CB of centering theory
     phrase = W[i].words
     npinphrase = W[i].npinphrase  # list of NPS of Wi
     if CB_index == 0 and len(npinphrase) > 0:
     # process the first NP phrase of the story only
        # CFi = NPinphrase, CPi = Frist NP in CFi, CBi=null
        listofelement[CB_index] = Element_Struct(CFi, CPi, '')
        listofelementupdate[CB_index] = Element_Struct(CFi, CPi, '')
        CB_index += 1
     elif len(npinphrase) > 0 : #process from the second NP phrase
        if len([noun for noun in npinphrase if noun.anaphora ==
'yes']) > 0:
           position_np = 0 # position of NP in CFi
           numberofpronoun = 0 # number of pronoun in current phrase
           lastpronoun = () #a pair of pronoun and np that was
resolved
           for np in npinphrase:
              if np.anaphora == 'yes':#found pronoun, must resolve
                 #CFi_potential keep a potential of CF current phrase
                 if numberofpronoun == 0:
                    CFi_potential = npinphrase
                 else:
                    CFi_potential = listofelementupdate[CB_index-
1].CF

              CFi_1 = listofelementupdate[(CB_index-1) -
numberofpronoun].CF
              # a adjacent previous CF
              np_potential_list =
find_prior_potential_np(position_np, CFi_potential, CFi_1) # keep only
NP and pronoun occur before
              rule1workinglist = [] # variable for pairing up each np
in np_potential list with current pronoun
```

ส่วนที่ 1

ส่วนที่ 2

NPs are in the same phrase with pronoun

ภาพที่ 3.2 อัลกอริทึมที่ใช้ในการแก้ปัญหา pronoun resolution



ภาพที่ 3.2 (ต่อ) อัลกอริทึมที่ใช้ในการแก้ปัญหา pronoun resolution

```
# function backwardlooking()
# input current position of CB (CB_index), a pair of pronoun and NP,
number of pronoun in phrase and the last pronoun
# output CB, updated CF
def backwardlooking(CB_index, rule1workinglist, num, lastpronoun):
    CFi_1 = listofelementupdate[CB_index-1].CF
    CPi_1 = listofelementupdate[CB_index-1].CP
    CBi_1= listofelementupdate[CB_index-1].CB
    CB_status = [] # variable for finding preference of each pair
    for a_pair in rule1workinglist:
        CFi = a_pair.CFi_potential
        CPi = CFi[0] # CPi is a first element of CFi_potential, which
is np[0]
        CBi_potential = a_pair.np # set CBi as np[0] from previous step
        result = centeringtable(CBi_potential, CPi, CPi_1, CBi_1)
        listofallstatue.append(dict_state[result])
        CB_status.append((a_pair.pronoun, CBi_potential, CFi, result))
    #sort CB_status by its dict_state
    CB_status = sorted(CB_status, key=lambda status: status[3])
    ## BEGIN SElECTION CBi AND UPDATE CFi##
    #Rule ถ้ามี pronoun ที่ต่างกันในphrase แล้ว np ที่อ้างถึงต้องไม่เหมือนกัน
    if len(lastpronoun) <= 0: #The frist pronoun in phrase, num is 0
        CBi = CB_status[0][1]
        CFi_update = CB_status[0][2]
    else:
        pronoun = CB_status[0][0]
        resloved = CB_status[0][1]
        if pronoun != lastpronoun[0]:
            match = 0
            for a_pairofresult in CB_status:
                if a_pairofresult[1] != lastpronoun[1]:
                    CBi = a_pairofresult[1]
                    CFi_update = a_pairofresult[2]
                    match = 1
                    break
                else:
                    CBi = CB_status[0][1]
                    CFi_update = CB_status[0][2]
        else: #update pronoun with noun phrase and append resolved
pronoun to the list
            CBi = CB_status[0][1]
            CFi_update = CB_status[0][2]
    return (CBi, CFi_update)
```

Heuristic Rule

ภาพที่ 3.3 อัลกอริทึมที่ใช้ในการแก้ปัญหา pronoun resolution (ฟังชันก์ backwardlooking)

บทที่ 4
การออกแบบและวัดประสิทธิภาพ

4.1 การออกแบบการทดลองสำหรับงานสกัดตัวละครอัตโนมัติ

ได้รันอัลกอริทึมกับนิทาน 40 เรื่อง โดยไม่มีการแก้ไขการตัดคำและการกำกับ POS ที่ผิดพลาด โดยกำหนดเงื่อนไขจำนวนซ้ำเป็นมากกว่า 2 ครั้ง มากกว่า 3 ครั้ง และมากกว่า 4 ครั้ง ตามลำดับ

4.2 ผลการทดลองวัดประสิทธิภาพสำหรับงานสกัดตัวละครอัตโนมัติ

ตารางที่ 4.1 ผลการสกัดตัวละครจากนิทานโดยกำหนดเงื่อนไขจำนวนซ้ำเป็น 2, 3, และ 4 ครั้ง

		# extracted characters		
		> 2	> 3	> 4
# main characters	91	71	62	54
# supporting characters	85	33	26	17
correct extracted		104	88	71
incorrect extracted		167	88	47
total	176	271	176	118
precision		38.38%	50.00%	60.17%
recall		59.09%	50.00%	40.34%
recall of main characters extracted		78.02%	68.13%	59.34%
recall of support characters extracted		38.82%	30.59%	20.00%

จากตารางที่ 4.1 ผลการทดลองสกัดตัวละครจากนิทาน โดยกำหนดเงื่อนไขจำนวนซ้ำที่ต้องการเป็นมากกว่า 2 ครั้ง 3 ครั้ง และ 4 ครั้งตามลำดับ พบว่า หากเพิ่มเงื่อนไขจำนวนซ้ำมากขึ้น จะสกัดจำนวนตัวละครที่ถูกต้องได้น้อยลง โดยจากตัวละครหลัก 91 ตัวละคร หากกำหนดเงื่อนไขเป็น 2 จะสกัดได้ 71 ตัวละคร หากกำหนดเงื่อนไขเป็น 3 จะสกัดได้ 62 ตัวละคร และหากกำหนดเงื่อนไขเป็น 4 จะสกัดได้เพียง 54 ตัวละคร นั่นคือคำนวณค่า recall ได้เป็น 78.02% 68.13% และ 59.34% ตามลำดับ สำหรับตัวละครสนับสนุน ผลการสกัดจะไม่ดีนัก โดยจะสกัดได้ 33, 26 และ 17 ตัวละครสนับสนุน จาก 85 ตัวละครสนับสนุน ทำให้คำนวณค่า recall ได้เป็น 38.82% 30.59% และ 20.00% ตามลำดับ และได้ค่า recall รวมเป็น 59.09% 50.00% และ 40.34% ตามลำดับ

อย่างไรก็ตาม ถ้าคิดค่า precision พบว่า การเพิ่มจำนวนซ้ำ จะทำให้ได้ค่า precision เพิ่มขึ้น คือมีส่วนที่สกัดผิดน้อยลงเรื่อยๆ ทำให้ precision เพิ่มขึ้นจาก 38.38% เป็น 50.00% และ 60.17% เมื่อกำหนดค่าจำนวนซ้ำเป็น 2, 3, และ 4 ตามลำดับ

4.3 การวิเคราะห์ผลการสกัดตัวละครจากนิทาน

สำหรับการวิเคราะห์ผลการสกัดตัวละคร พบว่ามีปัญหาสองแบบ คือ ตัวละครที่ไม่สามารถสกัดได้ และตัวละครที่สกัดได้บางส่วนแต่ไม่ถูกต้อง ตัวอย่างตัวละครที่ไม่สามารถสกัดได้ ที่เป็นชื่อเฉพาะ ได้แก่ *ลูอิส ลีนา ไมกี้ เจ้าโตโต้ โอลิเวียร์* ซึ่งปัญหาเกิดจากการตัดคำรวมกับการกำกับชนิดคำไม่ถูกต้อง หรือพบว่าตัวละครถูกเอ่ยถึงในนิทานเพียงครั้งเดียวหรือสองครั้ง จากนั้นได้ใช้สรรพนาม ได้แก่ เธอและม้าแทนลีนาและไมกี้ ในการกล่าวถึงในครั้งถัดๆ ไป ซึ่งอัลกอรึทึมส่วนแรกได้สกัดคำสรรพนามออกมาแทน ส่วนชื่อเฉพาะบางชื่อ เช่น จิม ซาร่าห์ กามีย์ ผลการสกัดผิดพลาด คือ สกัดได้เพียง จิ ซา และ ย์ ตามลำดับ ซึ่งเป็นปัญหาที่ต่อเนื่องจากการตัดคำที่ไม่ถูกต้อง อย่างไรก็ตาม มีชื่อเฉพาะที่สกัดได้ถูกต้องเช่นกัน เช่น จอห์น มาร์แตง อัลเบิร์ต เป็นต้น กลุ่มที่สองที่ไม่สามารถสกัดได้ คือ ตัวละครที่ไม่ระบุชื่อเฉพาะแต่ใช้คำขยายคำนามสามัญให้รู้ว่าแตกต่าง เช่น *คนตัดไม้อีกคน กบอีกตัว เทวดาประจำป่า หนังสือกฎหมาย* ซึ่งคำนามขยายดังกล่าวมักถูกเอ่ยถึงเพียงครั้งหรือสองครั้งเช่นกัน จากนั้นใช้เพียงคำนามหลักในการเอ่ยถึง ทำให้การสกัดไม่ถูกต้อง

นอกจากนี้ จากการทดลองพบว่า ความครอบคลุมของกฎคำนามและการเรียงลำดับของกฎที่ใช้ในการสกัดมีส่วนในการสกัดตัวละครเช่นกัน เช่น คำว่า ต้นเกาลัด ถูกเอ่ยถึง 2 ครั้งในนิทาน ต่อมาถูกเอ่ยถึงเป็นลุงเกาลัด และแต่ละครั้งมีการกำกับ POS ที่แตกต่างกัน เช่นเดียวกับคำว่าต้นสน คนเดินทาง และคำอื่นๆ

อย่างไรก็ตาม พบว่า คำที่สกัดนอกจากสามารถสกัดตัวละครหลักได้ดีแล้ว ยังสามารถบ่งบอกธีม (theme) ของนิทานได้ค่อนข้างดี เช่น เช่น เหตุการณ์เกิดในบ้าน ในป่า ในห้องสมุด บนถนน ฤดูกาล เกี่ยวกับจดหมาย เจ้าหญิงเจ้าชายหรือดินแดน และมีอะไรเกี่ยวข้องบ้าง เช่น ขวาน หนังสือ เสียง สัตว์ต่างๆ เสื้อ เป็นต้น ซึ่งคำเหล่านี้ที่ถูกสกัดออกมา แม้จะไม่ใช่ตัวละคร แต่เป็นสิ่งที่มีความสำคัญในการดำเนินเรื่องในนิทานดังกล่าว

4.4 การออกแบบการทดลองสำหรับงานแก้สรรพนามด้วย Centering Theory

การวัดประสิทธิภาพของทฤษฎีเซ็นเตอร์ริ่งที่ได้ทำการปรับปรุง (updated centering algorithm - UCA) เพื่อตรวจสอบว่าการปรับปรุงทฤษฎีเซ็นเตอร์ริ่ง ช่วยเพิ่มประสิทธิภาพมากน้อยเพียงใดเมื่อเทียบกับทฤษฎีเซ็นเตอร์ริ่งเดิม (centering algorithm - CA) สำหรับข้อมูลที่ใช้ในการทดลองจะใช้หนังสือนิทานภาษาไทยสำหรับเด็กจำนวน 40 เรื่อง ซึ่งมีคำสรรพนามที่ปรากฏในส่วนของบทบรรยาย 824 คำ จากส่วนบรรยาย 3075 วลี และคำสรรพนาม 841 คำ จากบทพูด (quote) 825 วลี โดยจะทำการเปรียบเทียบ

ผลลัพธ์จาก CA กับ UCA กับผลลัพธ์การแก้คำสรรพนามตามทฤษฎีเซ็นเตอร์ริ่งโดยมนุษย์ การวัดผลจะแบ่งเป็น 2 ส่วน คือ

4.4.1 วัดความถูกต้องของสถานะคำตอบ ตามทฤษฎี centering (Centering Status Correctness)

สถานะต่างๆ ตามทฤษฎีเซ็นเตอร์ริ่ง คือ continue, retain, smooth-shift และ rough-shift โดยถ้าหากมนุษย์ไม่สามารถใช้ทฤษฎีเซ็นเตอร์ริ่งแก้สรรพนามได้ มนุษย์จะกำกับสถานะเป็น unclassified คือส่วนที่ทฤษฎี centering ใช้ไม่ได้ผล ผลการทดลองเปรียบเทียบสถานะของคำตอบแบบอัตโนมัติ (Automatic) ด้วยอัลกอริทึม CA และ UCA กับสถานะที่กำกับโดยมนุษย์ (Manual) แสดงใน**ตารางที่ 4.2**

4.4.2 วัดความถูกต้องของการแก้สรรพนาม (Pronoun Resolution Correctness)

โดยจะแสดงความถูกต้องของการระบุคำนาม (ชื่อตัวละคร) ให้กับคำสรรพนาม ในส่วนนี้จะมีการวัดความถูกต้องในส่วนย่อยๆ คือ การวัดความถูกต้องเมื่อทำการแก้ปัญหา Pronoun Resolution ทุกวลี (a และ b) และการแก้สรรพนามเฉพาะวลีที่เป็นส่วนของการบรรยาย คือตัดบทพูดออกไป (c และ d) เพื่อสำรวจความแตกต่างและผลกระทบของการตัดวลีในบทพูด ที่มีต่อการแก้สรรพนาม โดยวัดความถูกต้องสองค่า คือค่าความถูกต้องทั้งหมด และความถูกต้องเมื่อคำนวณให้ค่าความถูกต้องของมนุษย์เป็น 100% นั่นคือ ตัดส่วนที่เป็น unclassified ออกไปจากตัวหาร (Minus centering theory not applied: Minus CT n/a) ผลการทดลองแสดงใน**ตารางที่ 4.3**

4.5 ผลการทดลองวัดประสิทธิภาพการแก้สรรพนาม

จากตารางที่ 4.2 พบว่าสถานะของคำตอบของมนุษย์มี continue 767 คำ ซึ่งมีมากที่สุด retain 86 คำ smooth-shift 135 คำ rough-shift 111 คำ และคำสรรพนามที่ไม่สามารถแก้ได้ด้วยทฤษฎีเซ็นเตอร์ริ่ง (unclassified) 566 คำ

การประมวล CA พบว่าสถานะของคำตอบที่ได้คือ continue 945 คำ, retain 145 คำ และไม่มีสถานะ smooth-shift และ rough-shift เมื่อเปรียบเทียบกับการประมวลผลโดยมนุษย์ คิดเป็นค่า precision 50.61% และ 36.07% ตามลำดับ โดยมี precision เฉลี่ยอยู่ที่ 49.01% คิดเป็นค่า recall 97.78% และ 76.74% ตามลำดับ โดยมีค่า recall เฉลี่ยอยู่ที่ 74.25%

ผลของการของ UCA พบว่าสถานะของคำตอบที่ได้คือ continue 911 คำ, retain 133 คำ, smooth-shift 17 คำ และ rough-shift 38 คำ เมื่อเปรียบเทียบกับการประมวลผลโดยมนุษย์ คิดเป็นค่า precision 53.12%, 41.46%, 22.58 และ 51.61% ตามลำดับ โดยมี precision เฉลี่ยอยู่ที่ 51.35% คิดเป็นค่า recall 97.52%, 79.07%, 5.19% และ 28.83% ตามลำดับ โดยมีค่า recall เฉลี่ยอยู่ที่ 77.80%

ตารางที่ 4.2 ผลการทดลองการวัดความถูกต้องของสถานะคำตอบ ตามทฤษฎี centering (Centering Status Correctness)

	Manual / Automatic	Manually					
		Continue	Retain	Smooth	Rough	Unclassified	Total
CA	Continue	750	20	87	97	528	**1482**
	Retain	17	66	48	14	38	**183**
	Smooth-shift	0	0	0	0	14	**0**
	Rough-shift	0	0	0	0	0	**0**
	Unclassified	0	0	0	0	0	**0**
	Total	**767**	**86**	**135**	**111**	**566**	**1665**
UCA	Continue	748	15	84	64	497	**1408**
	Retain	15	68	42	8	31	**164**
	Smooth-shift	1	2	7	7	14	**31**
	Rough-shift	3	1	2	32	24	**62**
	Unclassified	0	0	0	0	0	**0**
	Total	**767**	**86**	**135**	**111**	**566**	**1665**

จากตารางที่ 4.3 เมื่อวัดความถูกต้องของคำตอบจากการแก้ปัญหาสรรพนามในทุกวลี (All Phrase) เมื่อประมวลผลด้วย CA มีค่าความถูกต้อง 53.03% และเมื่อประมวลผลด้วย UCA มีความถูกต้อง 54.23% และค่าความถูกต้องเมื่อตัดคำสรรพนามที่ไม่สามารถใช้ทฤษฎีเซ็นเตอร์ริ่งแก้ปัญหาได้ (Minus CT n/a) พบว่า เมื่อประมวลผลด้วย CA มีค่าความถูกต้อง 77.34% และเมื่อประมวลผลด้วย UCA มีความถูกต้อง 79.62% ซึ่งพบว่าเมื่อปรับปรุงทฤษฎีเซ็นเตอร์ริ่งแล้วสามารถเพิ่มความถูกต้องได้เพียง 1-2%

เมื่อทำการแก้ปัญหาสรรพนามในทุกวลีแล้ว ผู้วิจัยได้ทำการทดลองเพิ่มเติมโดยการใช้วิธีการเดียวกับในการแก้ปัญหาสรรพนามกับวลีที่เป็นการบรรยาย (Minus Quote Phrase) พบว่า เมื่อประมวลผลด้วย CA กับคำสรรพนามทุกสถานะค่าความถูกต้อง 65.78% และมีค่าความถูกต้อง 77.25% เมื่อตัดคำสรรพนามที่มีสถานะ unclassified ไม่นำมาคำนวณ ส่วนการประมวลด้วย UCA กับคำสรรพนามทุกสถานะสามารถระบุ

ได้ถูกต้อง 66.26% และมีความถูกต้อง 77.99% เมื่อตัดคำสรรพนามที่มีสถานะ unclassified ไม่นำมาคำนวณ เมื่อเปรียบเทียบผลลัพธ์การแก้ปัญหาสรรพนามในทุกวลีกับการแก้ปัญหาเฉพาะสรรพนามที่อยู่ในส่วนบรรยาย พบว่ามีความถูกต้องเพิ่มขึ้นประมาณ 12 % และเมื่อเปรียบเทียบความถูกต้องเมื่อนำคำสรรพนามที่มีสถานะ unclassified มาคำนวณและไม่นำมาคำนวณพบว่ามีความถูกต้องเพิ่มขึ้น 11-25 %

ตารางที่ 4.3 ผลการทดลองเมื่อวัดความถูกต้องของการปัญหา Pronoun Resolution

	Experiments	pronoun	Result (All)			Result minus CT n/a			
			✓	✗	%	CT n/a	✓	✗	%
ALL Phrases	CA	1665	883	782	53.03	566	850	249	77.34
	UCA	1665	903	762	54.23	566	875	224	79.62
Minus Quote Phrase	CA	824	542	282	65.78	147	523	154	77.25
	UCA	824	546	278	66.26	147	528	149	77.79

4.6 การวิเคราะห์ผลการทดลองการแก้สรรพนาม

จากการทดลองในหัวข้อ 4.4.1 วัดความถูกต้องของสถานะคำตอบ จะสังเกตได้ว่าสถานะของคำตอบที่ได้จาก CA ไม่มีสถานะ smooth-shift และ rough-shift เนื่องจากว่าอัลกอริทึมเซนเตอร์ริ่งนั้นจะประมวลผลต่อเนื่องกันไปเรื่อยๆ โดยจะใช้ศูนย์กลางก่อนหน้า (CB_{i-1}) และศูนย์กลางปัจจุบัน (CB_i) เพื่อหาศูนย์กลางใหม่ที่ดีที่สุดตามตารางที่ 2.1 โดยเลือกเรียงจากสถานะ continue, retain, smooth-shift และ rough-shift ลำดับ ส่งผลให้สถานะ continue และ retain ถูกเลือกก่อนเสมอ ดังนั้น คำตอบของ CA จึงมีแต่สถานะ continue และ retain ส่วนสถานะตำตอบของ UCA ที่มีสถานะคำตอบอื่นๆเพิ่มขึ้นมา เนื่องจากมีการใช้กฎฮิวริสติกที่ว่า “คือ คำสรรพนามที่อยู่ในวลีเดียวกันแต่ต่างกัน ต้องอ้างถึงตัวละครที่ต่างกันด้วย” จึงทำให้มีโอกาสเปลี่ยนศูนย์กลางมากกว่า CA ทำให้ผลการแก้สรรพนามดีขึ้นด้วย ส่วนสถานะคำตอบที่ถูกกำกับเป็น unclassified โดยมนุษย์ อัลกอริทึม CA และ UCA จะไม่มีโอกาสทำงานได้เลย เนื่องจากคำนามที่ถูกอ้างถึง ไม่อยู่ในขอบข่ายที่ทฤษฎีครอบคลุมถึง ส่วนสาเหตุที่สถานะคำตอบเป็น unclassified นั้นเนื่องจากว่า

1. นามวลีที่ถูกอ้างถึง อยู่ในประโยคก่อนหน้าที่ไกลเกินกว่ารายการนามวลีใน CF_{i-1} เนื่องด้วยภาษาไทยไม่มีขอบเขตของประโยคที่แน่นอน ดังนั้นการแบ่งวลีก็มีผลกระทบในส่วนนี้ด้วย

2. มีการใช้สรรพนามอ้างถึงนามวลีที่หมายถึงสิ่งต่างๆที่นอกเหนือจากตัวละครภายในเรื่อง เช่น การใช้คำสรรพนามแทนผู้แต่งหรือผู้อ่าน

3. มีคำสรรพนามที่อ้างถึงนามวลีแบบกลุ่ม เช่น พวกเขาอ้างถึงจิมกับจอห์น พวกมันอ้างถึงเหล่าหนังสือทั้งหลาย ซึ่งนามวลีแบบกลุ่มเหล่านี้ อาจไม่ปรากฏเป็นข้อความอย่างชัดเจน จึงไม่อยู่ในขอบข่ายของการทำงานของอัลกอริทึม เช่น นามวลีถูกเอ่ยแยกกัน อาจไม่ได้อยู่ในประโยคเดียวกันหรือประโยคก่อนหน้าของคำสรรพนามแบบกลุ่มที่อ้างถึง เป็นต้น

จากผลการทดลองตารางที่ 4.2 และ ตารางที่ 4.3 พบว่าค่าความถูกต้องของสถานะคำตอบที่ได้จากการประมวลผล CA และความถูกต้องของการแก้สรรพนามด้วย CA มีค่าใกล้เคียงกัน คือ 74.25% กับ 77.34% ตามลำดับ ขณะที่ความถูกต้องของสถานะคำตอบที่ได้จากการประมวลผลด้วย UCA และความถูกต้องของการแก้สรรพนามด้วย UCA คือ 77.79% กับ 79.62% ตามลำดับ สาเหตุที่ทำให้ความถูกต้องที่ได้ทั้งสองแบบมีค่าต่างกันเนื่องจาก คอมพิวเตอร์จะพยายามให้ค่าสถานะใดสถานะหนึ่งจากสี่สถานะแก่สรรพนามที่กำลังแก้อยู่เสมอ ขณะที่ในความเป็นจริง สรรพนามนั้นๆ ไม่สามารถแก้ได้ด้วยทฤษฎีเซนเตอร์ริ่ง ทำให้สถานะต่อๆ มาของสรรพนามตัวถัดๆ ไปผิดพลาดไปด้วย แม้ว่าอาจจะแก้สรรพนามว่าอ้างอิงถึงตัวละครใดได้ถูกต้องก็ตาม

ในที่นี้ เฉพาะในส่วนที่คอมพิวเตอร์แก้สรรพนามได้ถูกต้อง แต่มีผลลัพธ์การให้สถานะของคอมพิวเตอร์แตกต่างจากมนุษย์ ปรากฏว่ามี 107 คำ ดังตัวอย่างในภาพที่ 4.1 คำว่า “เขา” หมายถึง หนูนา และ “ฉัน” หมายถึง หนูบ้าน อย่างไรก็ตาม ขณะนี้หนูนาไม่อยู่ในบริบทใกล้เคียงพอที่จะแก้ด้วยทฤษฎีเซนเตอร์ริ่งได้ มนุษย์จึงกำกับด้วย unclassified แต่อัลกอริทึมจะเลือกสถานะโดยใช้ CB_{i-1} ตามตัวอย่างคือ CB_13:หนูบ้านและกำกับสถานะเป็น continue ผลที่ตามมาคือ คำว่า “ฉัน” อัลกอริทึมแก้สรรพนามเป็น หนูบ้าน ได้ถูกต้อง แต่มีสถานะที่ไม่ถูกต้อง ดังนั้น จะเห็นได้ว่า เมื่อมีการประมวลผลที่ผิดพลาดจากวลีก่อนหน้านี้อาจส่งผลให้สถานะผิดพลาดต่อเนื่องไปกับวลีต่อๆไป

Phrase 13: หนูบ้าน พูด

CF_13 = {หนูบ้าน}, CB_13 = หนูบ้าน

Phrase 14: ช่วย บอก **เขา** ด้วยว่า **ฉัน** ยินดี ที่ จะ ไป **ฉัน** จะ ไป ถึงที่ นั่น ตอนเที่ยง วันพรุ่งนี้

เขา	-> CF_14 = {เขา, ฉัน, ฉัน}	
	-> หนูบ้าน/continue	คอมพิวเตอร์
	-> หนูนา/unclassified	มนุษย์
ฉัน	-> CF_15 = {หนูบ้าน, ฉัน, ฉัน}	
	-> หนูบ้าน/continue	คอมพิวเตอร์
	-> หนูบ้าน/rough-shift	**มนุษย์**

ภาพที่ 4.1 ตัวอย่างของการประมวลผลโดยคอมพิวเตอร์ เมื่อคำตอบถูกต้องแต่สถานะของคำตอบไม่ตรงกับการประมวลผลโดยมนุษย์

จากการทดลองในหัวข้อ 4.4.2 วัดความถูกต้องของการแก้สรรพนาม พบว่าเมื่อทำการแก้สรรพนามเฉพาะคำสรรพนามที่อยู่ในวลีบรรยายมีความถูกต้องมากกว่าการแก้สรรพนามในทุกวลีประมาณ 12% เนื่องจากว่าในบทสนทนาจะมีการใช้สรรพนามอ้างถึงตัวละครที่เป็นคู่สนทนา ซึ่งปรากฏในการบรรยายเนื้อเรื่องที่อยู่ไกลจากบทสนทนานั้น และการอ้างถึงผู้พูดเอง ในบางครั้งปรากฏหลังจากการใช้คำสรรพนามแทนตัวละครที่เป็นผู้พูด ดังแสดงตัวอย่างในภาพที่ 4.2 สรรนาม “ฉัน” ในวลีที่ 104 หมายถึง “หนูนา” ในวลี 105 และ “ฉัน” ในวลีที่ 106 หมายถึง “หนูบ้าน” ในวลี 107 ซึ่งเป็นชื่อตัวละครที่ปรากฏขึ้นภายหลังจากคำสรรนาม และเมื่อไม่นำคำสรรพนามที่มีสถานะเป็น unclassified มาคำนวณร่วมด้วย ทำให้ความถูกต้องเพิ่มขึ้นประมาณ 24%

P100: หนูบ้านถาม
P101: "ไม่มีแซนด์วิช หรือมัฟฟิน หรือชีสบ้างหรือ"
P102: หนูนาส่ายหัว
P103: "แล้วเค้กช็อกโกแลต ทาร์ตแยม หรือว่าไอศกรีมล่ะ"
P104: "ฉันยังไม่เคยชิมด้วยซ้ำ"
P105: หนูนาตอบ
P106: "โถ ญาติที่รักของฉัน"
P107: หนูบ้านพูด

ภาพที่ 4.2 ตัวอย่างการใช้สรรพนามอ้างถึงตัวละครที่ตามหลังสรรพนาม

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

งานวิจัยนี้มีจุดประสงค์เพื่อที่จะออกแบบอัลกอริทึมเพื่อสกัดตัวละครจากหนังสือนิทานเด็ก และเพื่อแก้สรรพนามว่าอ้างอิงถึงตัวละครใด ผลการวัดประสิทธิภาพการสกัดตัวละครพบว่า มีค่าความครบถ้วน (recall) สำหรับตัวละครหลักสูงสุดที่ประมาณ 78% และมีค่าความแม่นยำ (precision) มากขึ้นเมื่อเพิ่มค่าจำนวนคำซ้ำถึงระดับหนึ่ง อย่างไรก็ตาม สำหรับงานนี้ ค่าทั้งสองค่านี้จะเพิ่มขึ้นลดลงในทิศทางตรงข้ามกัน จึงต้องหาจุดสมดุลที่เหมาะสม

สำหรับการแก้ปัญหาการอ้างอิงสรรพนาม ได้ทดลองโดยให้มนุษย์แก้สรรพนามด้วยทฤษฎี centering พบว่า ทฤษฎีดังกล่าวไม่สามารถแก้สรรพนามได้ทั้งหมด โดยเมื่อทำตามทฤษฎีสามารถแก้สรรพนามได้ 1099 คำจาก 1665 คำ คิดเป็นประมาณ 66% เมื่ออิมพลีเมนต์อัลกอริทึมดังกล่าว พบว่าควรตัดวลีที่เป็นคำพูดออกไปก่อนจะได้ความถูกต้องมากกว่า โดยมีความถูกต้องประมาณ 65% และหากเทียบเฉพาะคำสรรพนามที่มนุษย์แก้ได้ จะคิดเป็นความถูกต้องประมาณ 77% การปรับปรุงอัลกอริทึม ให้ผลลัพธ์ที่ดีขึ้นประมาณ 1-2% โดยผลลัพธ์ที่ดีที่สุดอยู่ที่ประมาณ 79%

5.2 ข้อเสนอแนะ

ข้อเสนอแนะเพื่อปรับปรุงงานวิจัยเรื่องการสร้างรายชื่อตัวละครโดยอัตโนมัติสำหรับงานระบุผู้พูด มีดังต่อไปนี้

5.2.1 ปรับปรุงส่วนการสกัดตัวละคร ส่วนที่หนึ่ง การเพิ่มจำนวนกฎในการสกัดตัวละคร โดยเฉพาะกฎที่สกัดตัวละครประเภทชื่อเฉพาะ และตัวละครไม่มีชื่อที่ใช้คำนามขยาย ส่วนที่สอง การปรับให้อัลกอริทึมสามารถปรับเปลี่ยนจำนวนคำซ้ำได้แบบยืดหยุ่น (adaptive repetition size) ตามลักษณะของนิทานแต่ละเรื่อง เพื่อให้ได้ค่าความแม่นยำและค่าความครบถ้วนที่ดีที่สุด ส่วนที่สาม หากกฎสองกฎสามารถสกัดคำนามที่ต่อเนื่องกัน ก็ควรรวมคำนามจากสองกฎเข้าด้วยกันเป็นตัวละครหนึ่งตัวได้ นอกจากนี้ อาจใช้อัลกอริทึมการเรียนรู้เพื่อเรียนรู้ลักษณะเด่นที่ช่วยในการสกัดตัวละคร

5.2.2 ปรับปรุงส่วนการแก้สรรพนามดังนี้

- เพิ่มกฎฮิวริสติก เพื่อให้สรรพนามในบทพูดสามารถอ้างถึงนามวลีที่เกิดขึ้นหลังคำสรรพนามได้ และขยายให้คำสรรพนามสามารถอ้างถึงนามวลีที่อยู่ไกลเกินกว่า CF_{i-1}

- ปรับปรุงการแบ่งขอบเขตของวลีให้มีความถูกต้องมากขึ้น เนื่องจากการแบ่งขอบเขตของวลีมีผลกระทบในการแก้สรรพนามด้วย
- อาจมีการใช้เทคนิคของ Machine Learning รวมกับกฎ เพื่อเพิ่มประสิทธิภาพในการแก้สรรพนาม

5.2.3 งานวิจัยนี้ยังไม่ได้แก้ปัญหา เรื่อง ตัวละครตัวเดียวกันที่เรียกด้วยชื่อต่างกัน และปัญหาการรับรู้ตัวละครคนละตัวที่เรียกคล้ายๆ กัน ซึ่งยังเป็นปัญหาที่สำคัญในการให้คอมพิวเตอร์สามารถกำกับตัวละครให้ถูกต้องต่อไป

บรรณานุกรม

[1] Netisopakul, P., Woraratpanya, K., Wangsiripitak, S., & Pasupa, K. (2012). "Emotional Speech Synthesis for Visibility Impaired: From-Book-to-Speech", Research Project Funding Application submitted to National Research Council of Thailand.

[2] Zhang, Y, J., Black, W, A., & Sproat, R. (2003). "Identifying Speaker in Children's Stories for Speech Synthesis". In proceedings of EUROSPEECH 2003 (pp. 2041–2044). Geneva, Switzerland.

[3] Glass, K., & Bangay, S. (2006). "Hierarchical Rule Generalisation for Speaker Identification in Fiction Books". In proceedings of SAICSIT'06 (pp. 31–40). South African: South African Institute for Computer Scientists and Information Technologists.

[4] Glass, K., & Bangay, S. (2007). "A Naïve, Salience-Based Method for Speaker Identification in Fiction Books". In proceedings of the 18th International Symposium of the Pattern Recognition Association of South Africa (pp. 1–6). Pietermaritzburg, South Africa: PRASA.

[5] ณัฐธิดา เตชะนภารักษ์, สุภรณี กัลยาณกุล และ อรณี นิลศรีไพรวัลย์. (2011). "Semi-automatic Novel Text Classification based on Character". การแข่งขันสุดยอดซอฟแวร์ประมวลผลภาษาไทย 2011. ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ.

[6] Sasidhar, B. et al. (2011). "A Survey on Named Entity Recognition in Indian Languages with particular reference to Telugu". IJCSI International Journal of Computer Science Issues. 8(2). p 438-443.

[7] Sutheebanjard, P., and Premchaiswadi, W., (2009) "Thai Personal Named Entity Extraction without using Word Segmentation or POS Tagging". Proceeding of eight international symposium of natural language processing, pp 221-226.

[8] Tongtep, N., and Theeramunkong, T., (2010). "Pattern-based Extraction of Named Entities in Thai News Documents", Thammasat Int. J. Sc. Tech., Vol. 15, No. 1, January-March 2010.

[9] Sutheebanjard, P., and Premchaiswadi, W., (2010) "Disambiguation of Thai Personal Name from Online News Articles"

[10] Chanlekha, H., Kawtrakul, A., (2004). "Thai Named Entity Extraction by incorporating Maximum Entropy Model with Simple Heuristic Information. In: Proceedings of IJCNLP-2004, Hainan, China pp. 49–55.

[11] Soon, W.M., Ng, H.T., and Lim, D., (2001) "A Machine Learning Approach to Coreference Resolution of Noun Phrases", Computational Linguistics, 2001, 27(4):521-544.

[12] Fang, K., GuoDong, Z., and Qiaoming, Z., (2009) "Employing the Centering Theory in Pronoun Resolution from the Semantic Perspective". Proceedings of the 2009 Conference on Empirical on Empirical Method in Natural Language Processing, Singapore, 6-7 Aug 2009, P 987-996.

[13] Aroonmanakun, W., (2000). "Zero Pronoun Resolution in Thai: A Centering Approach", In Burnham, Denis, Interdisciplinary Approaches to Language Processing: The International Conference on Human and Machine Processing on Human and Machine Processing of Language and Speech. NECTEC: Bangkok, 127-147, 2000.

[14] ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ, "LexTo : เล็กซ์โต - โปรแกรมตัดคำสำหรับข้อความภาษาไทย." [ออนไลน์]. เข้าถึงได้จาก: http://www.sansarn.com/lexto/.

[15] Bovy, J., Gloahec le, F., Gollier, J., Deregnaucourt, E., Saffre, P., & Fransolet, M-C. (2551). นิทานฝันหวาน เล่มหนึ่ง แปลและเรียบเรียงโดย
ตุลนาท. พิมพ์ครั้งที่ 8. กรุงเทพฯ : นานมีบุ๊คส์คิดดี้.

[16] Legrand, M., Gollier, J., Dekelper, I., Gloahec le, F., Laere Van, M., & Bovy, J.. (2551). นิทานฝันหวาน เล่มสอง แปลและเรียบเรียงโดย
ตุลนาท. พิมพ์ครั้งที่ 7. กรุงเทพฯ : นานมีบุ๊คส์คิดดี้.

[17] Laclaverie, M., Dekelper, I.,& Gollier, J. (2549). นิทานฝันหวาน เล่มสาม แปลและเรียบเรียงโดย ตุลนาท. พิมพ์ครั้งที่ 6. กรุงเทพฯ : นานมีบุ๊คส์คิดดี้.

[18] Milbourne, A. (2551). อมตะนิทานอีสป เล่มหนึ่ง แปลและเรียบเรียงโดย
ปานนภา ตั้งกุลธวัช. พิมพ์ครั้งที่ 3. กรุงเทพฯ : นานมีบุ๊คส์คิดดี้.

[19] Pirotta, S. (2551). อมตะนิทานอีสป เล่มสาม แปลและเรียบเรียงโดย
รุ่งอรุณ สัมปัชชลิต. พิมพ์ครั้งที่ 2. กรุงเทพฯ : นานมีบุ๊คส์คิดดี้.

ภาคผนวก

ภาคผนวก ก
ชื่อและตัวละครในนิทานภาษาไทยสำหรับเด็กทั้ง 40 เรื่อง

ตารางที่ ก.1 ชื่อนิทานภาษาไทยสำหรับเด็กและชื่อตัวละคร 40 เรื่อง

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
1	หนูบ้านกับหนูนา	**หนูบ้าน: หนูน้อย, เจ้าหนู** **หนูนา: ญาติของมัน(ญาติของหนูบ้าน)** นกไปรษณีย์: คุณไปรษณีย์นก, ผู้นำข่าว
2	ลากับหมาป่า	**หมาป่า: หมาป่าท่าทางดุร้าย** **เจ้าลา: ลา**
3	ราชสีห์กับหนู	**หนูน้อย: หนู** **ราชสีห์: ราชสีห์ขนาดมหึมา**
4	นกกระสากับหมาจิ้งจอก	**หมาจิ้งจอก: หมาจิ้งจอกจอมเจ้าเล่ห์** **นกกระสา: ผู้มาใหม่** กระรอก: -
5	ทำไมผึ้งจึงมีเหล็กใน	**ราชินีผึ้ง: ราชินีตัวน้อย** **เทพเจ้าซีอุส: เทพเจ้า** ผึ้งน้อย: ผึ้งน้อยทั้งหมด, พวกผึ้ง, แมลงมีขน, ฝูงผึ้ง, ฝูงผึ้งทั้งหมด, บรรดาผึ้ง, เหล่าผึ้ง, ผึ้งทั้งหมด, ประชากรผึ้ง, ประชากรผึ้งทั้งหมด คนรับใช้: -
6	เต่าบินได้	เต่า: เต่าตัวหนึ่ง เพื่อนของเต่า: เพื่อนเต่า, เต่าอีกตัว, เพื่อน นกอินทรี: - หนู: หนูนา
7	ต้นอ้อผู้บอบบางกับต้นมะกอก	**ต้นมะกอก: -** **ต้นอ้อ: อ้อ, อ้อน้อย**
8	คนตัดไม้ผู้ซื่อสัตย์	**คนตัดไม้: คนตัดต้นไม้** **เทพเจ้าเฮอร์มีส: เทพเจ้า** **คนตัดไม้อีกคน: คนตัดไม้อีกคนหนึ่ง, ชายผู้นั้น**
9	ต้นไม้แปลงร่าง	**ต้นเกาลัด: ต้นเกาลัดใหญ่, ต้นไม้ชรา, ลุงเกาลัด, ลุง** **ต้นสน: ต้นสนต้นหนึ่ง, สนน้อย, ต้นสนน้อย, หนูน้อย, เด็กน้อย**

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
10	ห้องสมุดบินได้	**หนังสือนิทาน: เจ้าหนังสือนิทาน, หนังสือนิทานเล่มน้อย, เจ้าหนังสือนิทานเล่มน้อย, เจ้านิทานเล่มน้อย** **หนังสือประวัติศาสตร์: คุณตา, คุณตาหนังสือประวัติศาสตร์, หนังสือประวัติศาสตร์เก่าคร่ำคร่าเล่มหนึ่ง,หนังสือเก่าแก่ที่สุด** **เด็กผู้ชาย: เด็กผู้ชายคนหนึ่ง, เด็กชายผู้เคยร้องไห้, เด็กคนนั้น** **คุณยาย: คุณยายใจดี, คุณยายบรรณารักษ์** นักวิทยาศาสตร์: - คุณป้าขายข้าวแกง: คุณป้า ชายหนุ่มคนหนึ่ง: วิศวกร นักบิน: นักบินวัยกลางคน หนังสือกฎหมาย: - หนังสือตำราอาหาร, ตำราอาหาร หนังสือนิยาย: - หนังสือเกี่ยวกับสุขภาพ: - หนังสือทุกเล่ม: บรรดาหนังสือ หนังสือธรรมะ: - หนังสือวิทยาศาสตร์: - หนังสือภาพดอกไม้: - หนังสือศิลปะ: - หนังสือสิ่งแวดล้อม: -
11	สะพานลอยเกเรกับเจ้าด่าง	**เจ้าด่าง: เจ้าหมาน้อย, หมาจรจัด** **สะพานเกเร: สะพานลอย, สะพานลอยนิสัยเกเร, เจ้าสะพานลอยจอมเกเร, เจ้าสะพานลอย, สะพานลอยจอมเกเร**
12	กระต่ายกับเต่า	**กระต่าย: -** **เต่า: -** หอยทากสวนตัวเล็ก: หอยทากสวน, หอยทากน้อย, หอยทาก หมาจิ้งจอก: -

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
		ไก่ฟ้า: -
13	พระอาทิตย์กับลมเหนือ	**ลมเหนือ: -** **พระอาทิตย์: -** **ผู้ชายคนหนึ่ง: ชายหนุ่ม, ชายคนนั้น**
14	หมาป่ากับสุนัขบ้าน	**สุนัขบ้าน: สุนัขบ้านสีเข้มตัวหนึ่ง, สุนัขบ้านขนปุย, สุนัขบ้านขนปุยตัวใหญ่, สุนัขบ้านตัวนั้น, เพื่อนของมัน** **หมาป่า: หมาป่าตัวหนึ่ง, หมาป่าหนุ่ม, เจ้าหมาป่า** ชายชราคนที่ถือไม้เท้า: ชายชราที่ถือไม้เท้า นักเดินทาง: -
15	เจ้าหญิงปุยฝ้ายกับเจ้าชายสายลม	**เจ้าชายสายลม: เจ้าชาย** เจ้าตุ่น: - **เจ้าหญิงปุยฝ้าย: เจ้าหญิง, ปุยฝ้าย, ปิศาจ, ปิศาจสีขาว** ชาวเมือง: ชาวบ้าน คนหาปลา: -
16	กบผู้เฒ่านั่งเฝ้ากอบัว	กบผู้เฒ่า: - กบหนุ่ม: กบหนุ่มตัวหนึ่ง กบตัวหนึ่ง: - กบอีกตัว: - กบทั้งฝูง: กบฝูงหนึ่ง, กบทั้งหลาย
17	ยายแห้งลอยฟ้า	**สายลม: -** **ยายแห้ง: เด็กผู้หญิงที่ผอมมากคนหนึ่ง, เด็กผู้หญิง**
18	แมงมุมใยทอง	**เจ้าแมงมุม: เจ้าแมงมุมน้อย, แมงมุม, แมงมุมตัวหนึ่ง, แมงมุมตัวนี้** องค์หญิง: - แมลงทับ: แมลงทับชรา
19	ม้าแกลบน้อยของลีน่า	**ลีน่า: -** **เจ้าไมกี้: ไมกี้**

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
		เจ้าของฟาร์มเลี้ยงม้า: เจ้าของฟาร์ม, พ่อของลีน่า, พ่อ, ครูสอนขี่ม้า คุณลุง: คุณลุงบาสเตียน, ลุงบาสเตียน, หัวหน้าหน่วยกู้ภัย เจ้าโตโต้: โตโต้, สุนัข, สุนัขคู่ใจ
20	แพะอยากกินน้ำอัดลม	**เจ้าของฟาร์ม: -** **เจ้าโฟแลต: โฟแลต, แพะตัวเมีย, เจ้าแพะ, แพะ, แพะตัวเมียตัวหนึ่ง** **ภรรยาของเจ้าของฟาร์ม: ภรรยา** พนักงาน: พนักงานหนุ่ม
21	พรวิเศษจากนกน้อย	**กามีย์: -** **เจ้านกซิลแวง: นกซิลแวง, ซิลแวง, เจ้านก, เจ้านกน้อย, เทวดาน้อย, เด็กผู้ชาย, นกตัวหนึ่ง, นกน้อย, นก** จูเลียน: - คุณปู่: ปู่ของจูเลียน, คนเลี้ยงแกะ ซานตาคลอส: -
22	ผีน้อยลูแตง	จิม: - จอห์น: - **จิมกับจอห์น: เด็กทั้งสอง, พี่น้องฝาแฝด, หลานสองคนนี้, หลานทั้งสอง** **นกเค้าแมว: เจ้านก, นกเค้า, นก** นักดนตรีตาบอด: เพื่อนคนนั้น, เพื่อนคนหนึ่ง คุณยาย: - เจ้าผีน้อย: เจ้าผี, ผีน้อย, ผีน้อยผู้น่ารัก, ผีน้อยลูแตง, ลูแตง
23	หงส์ขนทอง	**หงส์: หงส์ผู้เย่อหยิ่ง, หงส์ผู้ยิ่งใหญ่, พญาหงส์, เจ้าหงส์ขนทอง, เจ้าหงส์** เป็ดน้อยตัวหนึ่ง: เป็ดน้อย, ผู้ช่วยชีวิตตัวน้อย, ลูกเป็ดน้อยสีเทา พวกเป็ด: เป็ดตัวอื่นๆ, เป็ดตัวอื่น

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
24	สร้อยคอนกเขา	**นกเขา: นกเขาที่สวยงาม, เจ้านกเขา, เจ้านกเขาตัวนี้, นกที่เย่อยิ่ง, เจ้านก** ปู่: - โอลิเวียร์: - สุนัข: - นกกางเขน: -
25	ลิลลี่ทุ่งผู้แสนดี	**มาร์แตง: คนเลี้ยงแพะ** **ลิลลี่ทุ่ง: -** **ซาราห์: ลูกแพะน้อย, แพะน้อย, แพะที่งามที่สุด, แพะดื้อ** ชาวไร่: - ต้นหญ้า: -
26	ใบไม้วิเศษ	**อัลเบิร์ต: หนูน้อยอัลเบิร์ต,** **เจ้านาย: เจ้าของปราสาท, เจ้าของปราสาทอีกหลัง, หัวหน้า** ทหารยาม: - อัศวิน: -
27	นิทานหิ่งห้อย	**พระอาทิตย์: พระอาทิตย์ที่แจ่มใส่** **สายฝน: ฝนที่แสนเศร้า** หิ่งห้อย: หนอนบางตัว, หนอนพวกนี้,
28	น้ำตาของเมฆน้อย	**เมฆน้อย: ก้อนเมฆน้อยๆ, ก้อนเมฆ, นักเดินทาง, เมฆก้อนน้อย, เมฆน้อยก้อนหนึ่ง**
29	ห่านออกไข่เป็นทองคำ	**หญิงชรา: แม่มด** **ชาวนา: สามี** **ภรรยา: ภรรยาชาวนา, คุณนาย** **ห่าน: ห่านตัวหนึ่ง, ห่านตัวนั้น, ห่านวิเศษ, ห่านทองคำ**
30	ต้นไม้ร้องไห้	**ต้นหลิว: ต้นไม้ร้องไห้, หลิว, ต้นหลิวร้องไห้, ต้นหลิวหัวเราะ** **กวีหนุ่ม: กวี**

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
		นกกระจอก: นกกระจอกตัวหนึ่ง,นก
31	เสื้อกันหนาวสีแดงของลาน้อย	**มารี: เด็กหญิง, ลูกสาวของเจ้าของฟาร์ม, ลูกสาว** **ลาน้อยตัวหนึ่ง: เจ้าลาน้อย, ลาน้อย, เจ้าลา** แม่: - พ่อ: - ลุยส์: พี่ชายคนโต ม้าตัวหนึ่ง: - ม้าอีกตัว: - ม้าตัวอื่นๆ: -
32	สุนัขจิ้งจอกกับนายพราน	**สุนัขจิ้งจอก: สุนัขจิ้งจอกตัวหนึ่ง, สุนัขจิ้งจอกสีขาว,** **สุนัขจิ้งจอกเจ้าเล่ห์, เจ้าจิ้งจอก, เจ้าสุนัขจิ้งจอก** ชาวบ้าน: ชายคนหนึ่ง เทวดาประจำป่า: -
33	แมลงหวี่น้อยผู้ยิ่งใหญ่	**แมลงหวี่น้อย: แมลงหวี่, แมลงหวี่ตัวน้อย,แมลงหวี่** **ผู้เคราะห์ร้าย, ลูกรัก** **แมงมุม: แมงมุมตัวหนึ่ง, เจ้าแมงมุม** นกนางแอ่น: กระจุกขนในกรงเล็บ นกอินทรี: พญาแห่งนก, พญาอินทรี, ราชาแห่งนก
34	บุรุษไปรษณีย์ผู้น่ารัก	**ลูอิส: ลูอิสน้อย, ลูกสาว, เด็กหญิง, หญิงสาว** **เฟลิกซ์: บุรุษไปรษณีย์หนุ่มน้อย, บุรุษไปรษณีย์** ป้า: หญิงชรา, ยายจู้จี้, คุณยายจู้จี้ พ่อค้าร้านขายของชำ: พ่อค้า, พ่อ, ผู้เป็นพ่อ
35	นกสาลิกากับนกยูง	**นกสาลิกาหนุ่ม: นกสาลิกา, นกสาลิกาตัวใหญ่ที่สุด,** **นกสาลิกาน้อย** แม่นกสาลิกา: แม่ของนกสาลิกา พ่อ: พ่อนกสาลิกา, พ่อของนกสาลิกา นกยูงตัวหนึ่ง: นกยูง คนรับใช้: -
36	ของขวัญของมาร์ติน	**มาร์ติน: -** **คุณปู่: ปู่, ปู่ของมาร์ติน**

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
		ผู้ใหญ่คนหนึ่ง: - ผู้หญิงอีกคนหนึ่ง: ผู้หญิงคนนั้น เจ้าหน้าที่: - นายตรวจ: นายตรวจคนนั้น
37	การผจญภัยของลูกค้างคาว	**ลูกค้างคาวตัวหนึ่ง: เจ้าค้างคาวน้อย, ค้างคาวน้อย, เจ้าค้างคาว** ชายชราคนหนึ่ง: ชายชรา
38	กระรอกน้อยกับนกเค้าแมว	**เจ้ากระรอก: กระรอกสีน้ำตาลอมแดง, กระรอกสีน้ำตาลอมแดงตัวหนึ่ง, กระรอกน้อย, เจ้ากระรอกน้อย, กระรอก** **เจ้านกเค้าแมว: นกเค้า, นกเค้าแมว, เจ้านกเค้าแมวจอมโหด, เจ้านก**
39	กระดิ่งผูกคอแมว	**หนูผู้เฒ่า: -** **แมว: พวกแมว, เจ้าแมว, เจ้าแมวตัวนั้น** **หนูขาวตัวแรก: -** **หนูสีขาวตัวที่สอง: -** หนูฉลาดตัวที่สาม: - หนูบางตัว: - หนูในทุ่งนา: - หนูในโรงงานเก็บเมล็ดข้าว: - หนูในโรงรีดนม: - หนูในโรงรีดนม: - หนูทุกตัว: - เจ้าของไร่: -
40	กบเลือกนาย	**กบที่อาวุโสที่สุด: กบที่อาวุโส** **พวกกบ: ทุกตัว, กบทุกตัว, กบจำนวนมาก** **ผู้นำตัวใหม่: สัตว์ร้าย, สัตว์ประหลาด, เจ้านายคนใหม่, พระราชาองค์ใหม่, พระราชาองค์แรก** นกนางแอ่น: นกนางแอ่นตัวหนึ่ง, นกตัวหนึ่ง กบรุ่นเยาว์: กบรุ่นเยาว์สองตัว, พวกกบรุ่นเยาว์ พวกกบแก่ๆ: -

เรื่องที่	ชื่อเรื่อง	ตัวละครหลัก/ตัวละครรอง
		เทพซีอุส: ซีอุส

ภาคผนวก ข

ผลงานวิจัยที่ตีพิมพ์

Speaker Identification from Multiple Short Stories without an Avatar List

Ponrudee Netisopakul **Patipan Wikaha**

Knowledge Management and Knowledge Engineering Laboratory
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
Email: ponrudee@it.kmitl.ac.th, patipan.x@gmail.com

Abstract

A task of identifying speakers from text is usually done by selecting a right speaker from a list of actors in a story. This actor list is called an avatar list, which so far, have been manually created for each story. This paper proposes a new methodology to identify speaker without using an avatar list. This can be achieved by incorporating multiple natural language processing techniques. Those are noun phrase chunking, speaker candidate extraction using speech verbs, and heuristic rules for identifying the speaker of each quote text.

Keywords: speaker identification, multiple children stories, text annotation, speech verbs

1 Motivation

In early 2012, a group of instructors and researchers at Faculty of Information Technology, KMITL initiated a project called ESSVIBS – Emotional Speech Synthesis for Visibility Impaired: From-Book-to-Speech [1]. The project is meant as a framework to explore a number of interesting issues related to speech and text processing, including Thai OCR, natural language text understanding, machine learning and speech synthesis.

In brief, input to ESSVIBS are Thai short children story books, output is a computer generated speech with multiple voices representing males, females, children and adults voices. At the initial state, forty Thai children stories are collected for experiments.

One of problems which is a focus in this paper is a problem of marking speakers in a story. We have examined some related works in a field of speaker identification from text [2, 3, 4, 5] and found that:

1. Most of the researches work with a long story with leading actor/actress and supporting actors/actresses.
2. For most of the work above except one, before a speaker can be identified, a list of actors/actresses above must be manually created and input to the speaker identification process – this list is called *an avatar list*.
3. Only the work in [2] identified actors using name entity extraction.

At this point, it is clear to us that an approach with an avatar list is inappropriate for our work because, first, we aim for our algorithm to work with a large collection of short stories, even with stories have not seen before. Second, Thai names in children stories are not very different from common noun phrases, hence, a name entity extraction may not work very well.

For these reasons, our research proposed a novel method to extract actors from multiple Thai short stories and using this list instead of an avatar to identify speakers.

Figure 1 shows an example input and desired output of our system - a Speaker Identification from Multiple Short Story without an Avatar – SIMS-woA. An input is excerpt from a short story. The brackets are English translation. An output is in xml format with speakers of each quote identified.

Hence, the research questions are:

- How to extract noun phrases which are actors in the story without extracting every common noun? This is an automatic avatar list construction task. When this is done, we can use the list to annotate the actors in the story. This is called an actor annotation task.

- How to correctly assign a speaker to each quote that appear in the story, especially when there are more than one actor adjacent to the quote or when there is no actor in the same paragraph of the quote? This is called a speaker assignment task.

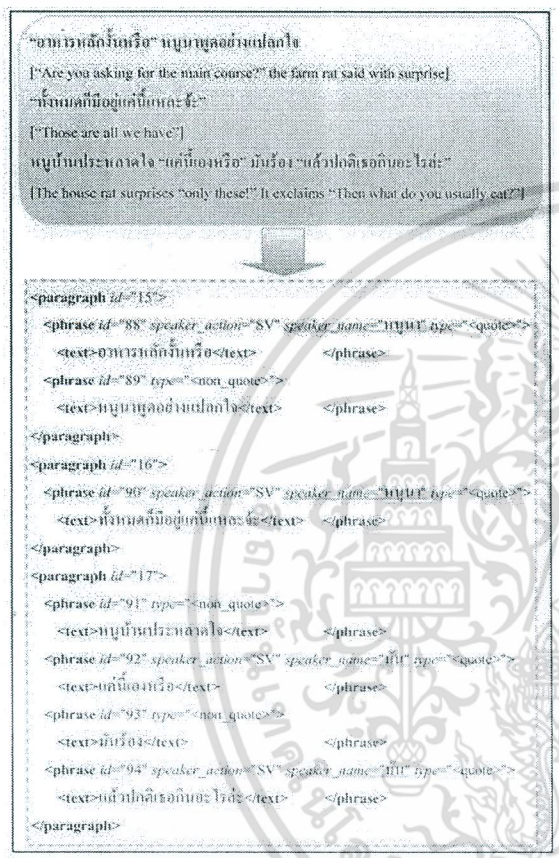


Figure 1 Example of input and desired output

The outline of this paper is as follow. Section 2 reviews approaches by previous research. Section 3 explains in detail our proposed methodology. Section 4 explains experiments and results of evaluating our approach. Conclusion and future work are presented in section 5.

2 Related Work

There are common processes for the task of identifying speakers from text [2, 3, 4]. First, quote texts are identified then actors are identified. Finally, speaker of each quote are identified. The differences among these researches are mainly the methods used to identifying actors. The work in [2] used pattern matching to extract proper names (or character names in the story) from the whole text. Assuming that the speaker must be in the same paragraph as the quote, [2] assigned a speaker from a preceding phrase or a following phrase of the quote. This work resolved neither a missing speaker issue when no character name is found around the quote, nor a multiple candidate speaker issue when there are more than one character names found around the quote.

The work in [3] created hierarchical phrase structure of each paragraph, beginning by separating quote text from narrative text. Narrative texts are further processed and tagged as noun phrase (subject), main verb, punctuation and so on. These structures are manually built from seeding text and used as input to learn (or merge) a set of generalized rules for finding speech verbs, actors and speakers, respectively. The paper showed that this approach did not obtain good result for a different author test set.

The work in [4] improved upon [3] by applying a scoring technique, for each phase of annotation, based on set of features. For example, features such as main verb, hypernyms, adjacent sentence, and proximity to quote are used for speech verb annotation. Features such as subject or object, noun or pronoun, proper noun, abbreviation and distance from verb are used for actor annotation. In addition, a hand-code decision tree is used to resolve speaker ambiguous.

Note that, even with these complicate processes in [3, 4], they both must still employ manually created avatar list to reduce errors when choosing speaker for each quote.

The work in [5] is the only speaker identification work found processing Thai language. Training set composed of adjacent quote phrases with POS tags and manually tagged actors and speakers. The learning features are "language model" and n-gram. The work can only identify speaker from a simple sentence. A speaker previously identified is not taken into account to identify a next speaker.

In our work, we improve upon previous works in the following aspects.

1. We classify verbs into 4 classes and prioritized them based on their possibility of being pointers to speakers.
2. When an avatar list is absent, an actor list can be automated created instead using verb classes as clues.
3. This actor list can be used to scan for candidate speakers of quote texts, when speech

verbs are absent from those phrases.

4. Speaker selection decision is based on a simple verb class priority and proximity to quote text.

3 Proposed Methodology

Since our intended system must work with a number of short Thai children books. Each has its own set of characters, which conventionally must be manually identified for each story in advance. However, we propose that a labor work of creating an avatar list for each story can be avoided. An alternative method to automatically identify actors and speakers of multiple short stories without the need to manually create many avatar lists is laid out in detail in this section.

An overall idea is based on a heuristic that a noun phrase representing an actor who speaks usually appears adjacent to the quote, either in the same paragraph or in the adjacent paragraph, as shown in Figure 2. Therefore, the first main task is to identify a set of candidate actors from paragraphs with quote and adjacent to quote. This list can be used in place of an avatar list. Next main task is to identify only candidate speakers for each quote. Last task is to use a heuristic to assign a speaker of that quote from the previous list of candidates.

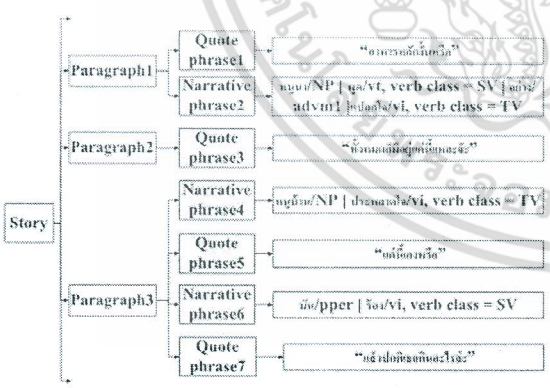


Figure 2 Hierarchical view of text structure in SIMS-woA

Figure 2 shows the hierarchical view of text structure, our system - SIMS-woA - working with. A story composes of many paragraphs; SIMSwoA focuses on paragraphs with quote text and adjacent to quote text. It is reasonable to assume that to find a speaker of a quote; the process should only look nearby the quote. It should not have to process every paragraph.

A paragraph may have both quote text and narrative text or it may have only quote text or narrative text. Figure 2 shows a paragraph with a quote text following by a narrative text. In paragraph 1, the quote text is "อาหารหลักงั้นหรือ" and the narrative text is "หนูนาพูดอย่างแปลกใจ". In order to know who speak the quote text, first, the system identifies pairs of (noun phrase, verb class) which either precede or follow the quote text, within the same paragraph or in the adjacent paragraph. The noun phrases from these pairs are candidate actors for the quote. These candidates are prioritized based on their associate verb class.

To automatically construct a list of actors without using manually created avatar, one of the novelties of SIMS-woA is to classify verbs into four verb classes and prioritize them based on their possibilities to be an indicator for a candidate speaker.

These verb classes are speech verbs (SV), thinking/feeling verbs (TV), action verbs (AV), and other verbs. Table 1 shows the number of each verb classes. There are 43 speech verbs, 10 thinking/feeling verbs and 54 action verbs. Speech verbs are verbs indicating the action of saying, such as พูด (say) ว่า (said that) กล่าว (state) ประกาศ (pronounce) พึมพำ (mumble) บ่น (complain) ดุ (rebuke) ชม (praise). If a pair of (noun phrase, SV) is found next to a quote, it is pretty clear that the noun phrase has a high possibility of being a speaker of that quote. Therefore, this noun phrase is given the highest priority to be assigned as a speaker of the quote.

However, from our preliminary investigation, we found that a verb class SV alone is not enough to detect actors or speakers in the story. Many narrative phrases adjacent to quote texts do not have any explicit speech verb, but they do have either thinking verbs or action verbs. Thinking verbs are verbs indicate mental actions of actors, such as คิด(think) นึก(contemplate) รำพึง(cogitate) สงสัย(doubt). In a narrative children story, when an actor thinks, the quote text must also be read out loud. In this sense, a thinking verb class (TV) has a second highest priority to be assigned as a speaker.

Action verbs are verbs indicating actions of actors, such as ยิ้ม (smile) พยักหน้า (nod) เสนอ (offer). When telling a story, instead of a speech verb, an action verb is often place there to indicate tone,

feeling or movement of the speaker. Hence, a noun phrase with an action verb class (AV) has the next priority for a speaker assignment. For those verbs not belong to these three verb classes, the process labeling them as 'other' verbs.

Table 1 The number of each verb classes

Classes	Count	Examples
SV	43	พูด (say), ว่า (said that), …
TV	10	คิด(think), สงสัย(doubt), …
AV	54	ยิ้ม (smile), พยักหน้า (nod), …
Other	-	คอย (wait), วิ่ง (ran), …

Note that list of verbs can be collected during preprocessing step but their associated verb class as defined here is currently done manually. We are investigating whether this can be done automatically using pre-labeled learning set.

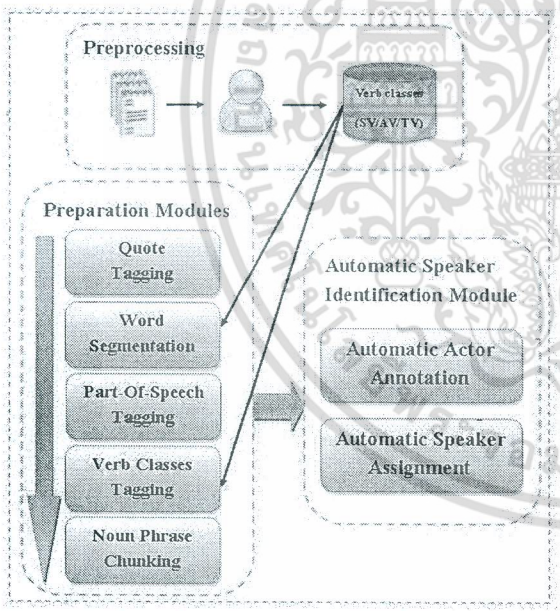


Figure 3 Overall architecture of SIMS-woA

Figure 3 shows an overall architecture of SIMS-woA. A preparation module on the left is an NLP pipeline to prepare Thai text for automatic speaker identification module on the right.

Input to the preparation module is a story in text form. First, quote text are marked everywhere and will not be processed further. The rest of the text goes through word segmentation and POS tagging using a publicly available Thai segmentation tool [6] and a Thai POS tagger [7,8]. Then, verbs are mapped into verb classes as previously explained.

One last important step during preparation is noun phrase chunking. We do not use name entity extraction technique to find an actor here because in most of children stories, actors are common nouns with attributes such as หนูนา (farm rat) หนูบ้าน (house rat) คนตัดต้นไม้ (woodsman). Therefore, more than a hundred regular expression rules, as partially shown in Figure 4, are used to construct a noun phrase from word tags.

For example, "หนู/ncn บ้านncn" are chunked into "หนูบ้าน/NP" and "คน/ncn" "ตัด/vt" "ต้นไม้/ncn" are chunked into "คนตัดต้นไม้/NP".

```
NP    : {<ncn><ncn><ncn><cl><adj>}
      : {<ncn><prep><ncn><ncn>}
      : {<ncn><vt><ncn>}
      : {<ncn><ncn><adj>}
      : {<ncn><cl><adj>}
      : {<ncn><ncn>}
      : {<ncn><npn>}
      : {<ntit><ncn>}
      : {<npn>}
      : {<ncn>}
```

Figure 4 Example of regular expression rules for NP chunking

On the right of Figure 3, an automatic speaker identification module has two sub-modules: automatic actor annotation and automatic speaker assignment. For actor annotation, Figure 5 shows that there are two tasks for finding actors, who will be candidates for speakers. Those are actor extraction task and actor annotation task.

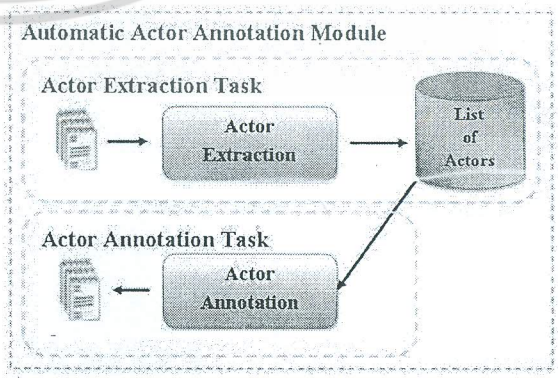


Figure 5 Automatic actor annotation task

Figure 6 are rules used to extract actors from

phrases adjacent to a quote text. These rules match (partial) phrases with three verb classes previously described. Note that phrases with these verb classes mostly have only one noun phrase. However, there can be many adjacent phrases to the quote text. Therefore, there can be many actors, which need to be decided which one is most likely to be a speaker in a later module. For now, actors extracted are stored in an actor list to be used in an actor annotation task.

Rule1:	{(NP|pper) (SV|TV|AV)}
Rule2:	{(NP|pper) (vi|vt|prev|neg|adv|conj) (SV|TV|AV)}
Rule3:	{(NP|pper) (vt) (conj|adv) (SV|TV|AV)}
Rule4:	{(NP|pper) (prev) (prev) (SV|TV|AV)}
Rule5:	{(NP|pper) (prel) (vt) (SV|TV|AV)}
Rule6:	{(NP|pper) (conj) (prev|vi|vt|adv) (SV|TV|AV)}
Rule7:	{(NP|pper) (conj) (vt) (pref1) (SV|TV|AV)}
Rule8:	{(NP|pper) (conj) (prev) (prev) (SV|TV|AV)}
Rule9:	{(NP|pper) (prev) (vt) (conj) (SV|TV|AV)}
Rule10:	{(NP|pper) (vi) (vpost) (vi) (SV|TV|AV)}
Rule11:	{(NP|pper) (vi) (vi) (vpost) (SV|TV|AV)}
Rule12:	{(NP|pper) (vt) (vpost) (conj) (SV|TV|AV)}
Rule13:	{(NP|pper) (conj) (vt) (prec) (SV|TV|AV)}

Figure 6 Actor extraction rules

Sometimes there is no narrative phrase adjacent to a quote or those phrases do not matched rules above. Hence, the extracted actor list is used to annotate actors from other parts of text in the story. This is called actor annotation task. This step increases a chance of finding a right speaker. Note that a speaker of a particular quote may have been mentioned in a previous paragraph far from a quote or even in the beginning of the story.

Another task during actor annotation is to assign an action to each actor. This task is important because its verb class information will be used to priority actors for a speaker assignment sub-module. That is SV has a highest priority, then TV and AV.

Assigning an action to an actor is straightforward if there is only one actor and only one verb in the phrase. Then the action of the actor is the type of that verb class, such as SV, TV, AV or other verb. The task is more complicate for phrases with many actors or actions. For example, "มากินพายแอปเปิ้ลกัน" หนูบ้านพูดให้หนูนาคลายความกลัว, the action of "หนูบ้าน" is "พูด" and the action of "หนูนา" is "คลายความกลัว".

The algorithm is shown in Figure 7. It processes a word token from left to right for each phrase. If other types of token except an actor and a verb are found, the algorithm just moves to the next token. If an actor is found with an action already assigned, it also skips to the next token; otherwise, a current actor is kept. Then a next token with of type verb is kept in a variable *best_action*; when a next verb token is found, its verb class priority is compared to *best_action*'s priority. The verb with highest priority replaces the old one and finally is assigned to be an action of the current actor, which happen either when a new actor is found or when a phrase is ended.

```
Actor's Action Assignment
Input: word token of a phrase
Output: association of (actor_i, action_i) for every
actor_i in the phrase

#set a pair of current actor and best action
cur_actor = null
best_action = null
Loop until no word in the phrase
  for each word w in the phrase:
    if w is actor with no action assigned :
      #check if there is already a pair found
      if cur_actor and best_action is not null
        action(cur_actor) = best_action
        clear values of cur_actor and best_action
      # then start with a new current actor
      cur_actor = w
      # if w is of type verb, set value of best_action
      else if  w is a {ST/TV/AV} verb
        if priority(w) > priority(best_action)
        best_action = w
End loop
action(cur_actor) = best_action
clear values of cur_actor and best_action
```

Figure 7 Actor's action assignment algorithm

After actors and their actions are determined in a story, the last step is to assign speaker for each quote text.

Figure 8 shows the speaker assignment algorithm. First, we try to select a speaker from actors (or candidate speakers) in the same paragraph with the quote. The candidates are spilt into a list of candidates found in previous phrases (ListActorBefore) and a list of candidates found in the phrase following the quote (ListActorAfter). Candidates in both lists are sorted by priorities of their verb classes and their proximity to the quote. The priority of verb classes is "SV", "TV", "AV" and "Other", respectively. The highest priority actors from both lists are compared and chosen. However, if the first actors from both lists have the same priori-

ties, the first actor from ListActorAfter is chosen.

In case that there is no candidate speaker found in the same paragraph, the algorithm uses actors from a previous paragraph to find a speaker. Except that if the previous paragraph is also a quote, the algorithm assign a second last speaker found to a current quote. If everything else fails, the last speaker is assigned to the quote.

```
Speaker Assignment Algorithm
Input: tagged text structure and candidate speakers
Output: association of (quote_i, speaker_i) for each quote_i

# set of (quote, speaker) pairs in a story
Q = {(q_1, s_1), (q_2, s_2) ..., (q_n, s_n)}

Loop for each quote q_i
 ListActorBefore = list of actors from previous phrase
 ListActorAfter = list of actors from following phrase
 if the paragraph with quote has a candidate(s)
   sort ListActorBefore and ListActorAfter
   s_i = the highest priority actor from both lists
 else #no candidate in the same paragraph
   #get actors from a previous paragraph
   ListActorBefore = previous paragraph list of actors
   sort ListActorBefore
   s_i = the highest priority actor in ListActorBefore
 endif
 if s_i is empty and a previous phrase is a quote
   then #assign second last speaker as a speaker
     s_i = s_i-2 when s_i-2 not same as s_i-1
   else #everything else fails
     # assign last speaker as a speaker
     s_i = s_i-1
 endif
End loop
```

Figure 8 Speaker assignment algorithm

4 Experiments

4.1 Experiment Design

Experiments are designed to evaluate efficiency and correctness of the proposed algorithms. A collection composes of forty short children stories collected from story books. A collection has 1086 paragraphs, 825 quote phrases, with 3078 narrative phrases and 1020 adjacent phrases. There are 24839 word tokens.

In order to evaluation the effects of each actor identification tasks toward to the correctness of a speaker assignment task, the measurement are done in three stages. In stage 1, only an actor extraction module is applied to the collection, and then the actor list resulting from this step is used as candidate speakers directly applied to a speaker assignment task. In stage 2, after an actor extraction task, an actor list is compiled for each story. The list is used to match against adjacent quote phrases to find more candidate speakers, before applying it to a speaker assignment task. Stage 3 is similar to stage 2, except that an extracted actor list is used to scan against the whole story opposed to scan against only adjacent quote phrases.

4.2 Results and Discussion

Table 2 Results of the three actor annotation tasks and their effects to a speaker assignment task

	Actor Annotation Task		Speaker Assignment Task	
			Correct	(%)
Stage 1	Number of Actors	831	683	82.79
	Retrieved	586		
	Correct	546		
	Precision (%)	93.17		
	Recall (%)	65.70		
Stage 2	Number of Actors	831	703	85.21
	Retrieved	858		
	Correct	807		
	Precision (%)	94.06		
	Recall (%)	97.11		
Stage 3	Number of Actors	1512	706	85.58
	Retrieved	1569		
	Correct	1469		
	Precision (%)	93.63		
	Recall (%)	97.16		

Table 2 shows the results of the three stages. Out of 831 actors, stage 1, focusing on actors with the three verb classes, extracts correctly 546 actors and incorrectly 40 actors, resulting in assigning speaker correctly 683 quotes out of 825 quote or about 82.79%. Note that the recall rate, 65.70%, is not very high at this stage.

When the algorithm is improved with an actor annotation task in stage 2, it drastically increases the correctness of actor identification. From 831 actors found in phrases adjacent to quote, the algorithm identified correctly 807 actors or 97.11%, resulting in identify speakers correctly for 703 quotes or about 85.21%. Note that the number of retrieved actors is more than the

number of actors because the algorithm also extracts noun phrases which are not actors.

Stage 3 follows the same pattern, from 1512 actors in the whole text of forty stories, 1469 actors are correctly identified and resulting in correctly assign speakers of 706 quotes or about 85.58%.

Note that although the algorithm in stage 3 improves the overall result, its effect is not as much as the improvement from stage 1 to stage 2.

In addition, the precision rates of three actor annotation tasks only remain about the same, from 93.17, 94.06, to 93.63%, although the recall rates improve from 65.70% in stage 1 to 97.11% in stage 2 and 97.16% in stage 3. This shows that increasing the number of actors is not always resulting in increasing the number of correct speaker assignment. Only *true* candidate actor detection could result in improving a chance to find a *true* speaker of the quote.

5 Conclusion and Future Work

This work proposed a methodology to annotate speakers in multiple Thai children stories. The framework works well without avatar list with an average of 85% correctness.

However, future work is still needed to resolve anaphora. In addition, problems of same character with different aliases and different characters with the same title must also be resolved.

Acknowledgment

This research is partially supported by Faculty of Information Technology Research Fund, King Mongkut Institute of Technology Ladkrabang.

References

[1] Netisopakul, P., Woraratpanya, K., Wangsiripitak, S., & Pasupa, K. (2012). *Emotional Speech Synthesis for Visibility Impaired: From-Book-to-Speech,* Research Project Funding Application submitted to National Research Council of Thailand.

[2] Zhang, Y, J., Black, W, A., & Sproat, R. (2003). Identifying Speaker in Children's Stories for Speech Synthesis. In *proceedings of EUROSPEECH 2003* (pp. 2041–2044). Geneva, Switzerland.

[3] Glass, K., & Bangay, S. (2006). Hierarchical Rule Generalisation for Speaker Identification in Fiction Books. In *proceedings of SAICSIT'06* (pp. 31–40). South African: South African Institute for Computer Scientists and Information Technologists.

[4] Glass, K., & Bangay, S. (2007). A Naïve, Salience-Based Method for Speaker Identification in Fiction Books. In *proceedings of the 18th International Symposium of the Pattern Recognition Association of South Africa* (pp. 1–6). Pietermaritzburg, South Africa: PRASA.

[5] Tachanaparak, N., Kunlayanakual, S., & Niinsripaiwan, A. (2011). *Semi-automatic Novel Text Classification based on Character* (BEST Working paper 13p33c002). National Electronics and Computer Technology Center. (in Thai) Retrieved August 08, 2013, from http://thailang.nectec.or.th/halloffame/images/stories/best/download/13p33c002.pdf

[6] Tungkualtaveesub, W., Ratmanee, P., & Jindaphitak, T. (2010). *Thai Word Segmentation a Hybrid Approach* (BEST Working paper 34S001). National Electronics and Computer Technology Center. (in Thai) Retrieved August 08, 2013, from http://thailang.nectec.or.th/halloffame/images/stories/best/download/best2010_12p34s001.pdf

[7] Satayamas, V. (2012). *Part-of-speech tagger for Thai Language*. Retrieved August 08, 2013, from http://veer66.wordpress.com/tag/pos/

[8] NAiST Lab, Kasetsart University. (2011). *Jitar model and Jitar 20100224*. Retrieved August 08, 2013, from http://naist.cpe.ku.ac.th/pkg/

[9] Soon, Meng, W., Ng, Tou, H., & Lim, Yong , Chung, D. (2001). A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27(4), 521–544.

[10] Kong, F., GuoDong, Z., & Zhu Qiaoming. (2009). Employing the Centering Theory in Pronoun Resolution from the Semantic Perspective. In *proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing* (pp. 987–996). Singapore: Association for Computational Linguistics Stroudsburg, PA, USA.

[11] Sutheebanjard, P., & Premchaiswadi, W. (2009). Thai Personal Named Entity Extraction without using Word Segmentation or POS Tagging. In *8th International Symposium on Natural Language Processing* (pp. 221–226). Bangkok.

ประวัตินักวิจัย

ชื่อ - นามสกุล (ภาษาไทย) รศ.ดร พรฤดี เนติโสภากุล

ชื่อ - นามสกุล (ภาษาอังกฤษ) Assoc. Prof. Ponrudee Netisopakul, Ph.D.

สถานที่ติดต่อ คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

หมายเลขโทรศัพท์ ไปรษณีย์อิเล็กทรอนิกส์ ponrudee@it.kmitl.ac.th

ประวัติการศึกษา

- Ph.D. (Computer Information and Science), Case Western Reserve University, Cleveland, OH, USA.
- M.S. (Computer Information Science), University of Delaware, Newark, DE, USA.
- M.S. (Computer Science), University of Southern California, Los Angeles, CA, USA.
- B.S. with Honor (Statistics), Chulalongkorn University, Bangkok, THAILAND.

ประสบการณ์งานวิจัยที่เกี่ยวข้อง

- Phucharasupa, K., Netisopakul, P., "Classifying Thai Action-Verb Classes Based on Paraphrasing Behavior", International Computer Science and Engineering Conference (ICSEC 2014), July 30 – August 1, 2014, Khon Kaen, Thailand, pp 52-57.
- Kuptabut, S., Netisopakul, P., "Multiple Event Extraction from a Sentence: Case Study in a Football Domain", International Computer Science and Engineering Conference (ICSEC 2014), July 30 – August 1, 2014, Khon Kaen, Thailand, pp 331-336.
- Kanungsukkasem, N., Netisopakul, P., Leelanupab, T., "Recognition of NASDAQ Stock Symbols in Tweets", International Conference on Knowledge and Smart Technology (KST 2014), January 30-31, 2014, Chonburi, Thailand, pp 12-16.

- Lertsuksakda, R., Netisopakul, P., Pasupa, K., "Thai Sentiment Terms Construction using Hourglass of Emotions", International Conference on Knowledge and Smart Technology (KST 2014), January 30-31, 2014, Chonburi, Thailand, pp 46-50.
- Netisopakul, P., Chainapaporn, P., "MA_THR: Multi-Agent Thai Herb Recommendation from Heterogeneous Data Sources", Information Search, Integration, and Personalization in Communications in Computer and Information Science, Volume 421, 2014, pp 103-118.
- Netisopakul, P., Wikaha, P., "Speaker Identification from Multiple Short Stories without an Avatar List", Symposium on Natural Language Processing (SNLP-2013), October 28-30, 2013, Phuket, Thailand, pp 24-30.
- Chainapaporn, P., Netisopakul, P., "Word Similarity algorithm for Merging Thai Herb Information from Heterogeneous Data Sources", International Conference on Information Technology and Electrical Engineering (ICITEE 2013), October 7-8, 2013, Indonesia, pp. 159-163.
- Phucharasupa K., Netisopakul, P., "Thai Sentence Paraphrasing from the Lexical Resource", 26th Pacific Asia Conference on Language Information and Computation (PACLIC 26), Bali, November 8-10, 2012, pp 415-424.
- Chainapaporn, P., Netisopakul, P., "Thai Herb Information Extraction from Multiple Websites", International Conference on Knowledge and Smart Technology, July 7-8, 2012, Burapha University, Thailand. pp 16-23.
- Wiroteyakun A., Netisopakul, P., "Web-based Database Support System for Particle Monitoring Reports" Journal of the National Research Council of Thailand, Vol. 42, No. 2, July-December 2011. pp 63-81.
- Wiroteyakun, A., Netisopakul, P., "Improving Software Maintenance Size Metrics A Case Study: Automated Report for Partical Monitoring in Hard Disk Drive Industry", The 9th International Conference on Computer Science and Software Engineering, May 30-June 1, 2012. Thailand. pp 335-340.
- Pasupa K., Netisopakul, P., "Thai Paragraph Shortening Based on Binary Classification Model", the Joint International Symposium on Natural Language Processing and

Agricultural Ontology Service 2011, February 9-10, 2012, Bangkok, Thailand. pp 181-185.

- Kuptabut, S., and Netisopakul, P., "Ambiguity Resolution for Semantic Annotated using Transformation-Based Learning" the Joint International Symposium on Natural Language Processing and Agriculture Ontological Service 2011. February 9-10, 2012. Bangkok, Thailand.
- Phucharasupa K., Netisopakul, P., "Classification of Thai Sentence Paraphrase", the Joint International Symposium on Natural Language Processing and Agricultural Ontology Service 2011, February 9-10, 2012, Bangkok, Thailand. pp 197-203.
- Netisopakul, P., Kuptabut, S., "Frame Structure Construction using Ontology-Guided Semantic Analysis Process: Case Study in a Football Domain", Proceedings of the 26th International Conference on Circuits/Systems, Computers and Communications, 19-22 June 2011, Gyeongju, Korea. pp. 539-542.
- Comejina, P., Netisopakul, P., "Reducing Ambiguity in Thai Text to Thai Lanna Text Translation System using Viterbi Algorithm" Proceedings of the 7th National Conference on Computing and Information Technology. May 11-12, 2011. Bangkon, Thailand. Pp. 812-817.
- Chainapaporn P., Netisopakul, P., "Multi-Agent Architecture for Thai Herb Recommendation", the Joint International Symposium on Natural Language Processing and Agricultural Ontology Service 2011, February 9-10, 2012, Bangkok, Thailand. pp 1-6.
- Suksom, N., Buranarach, M., Thein, Y. M., Supnithi, T., & Netisopakul, P. "A Knowledge-based Framework for Development of Personalized Food Recommender System" Proceedings of the 5th International Conference on Knowledge, Information and Creativity Support Systems. Nov. 25-27, 2010. Chiang Mai, Thailand. pp. 274-277.
- Kuptabut, S., Netisopakul, P., "Ontology Directed Semantic Annotation Process", Proceedings of the 3th International Conference on Information Sciences and Interaction Sciences, Chengdu, China, June 23-25, 2010. pp. 251-255.

- Netisopakul, P., Sappajit, W., "Prediagnosis Doctor Simulation Using Case-Based Techniques", Proceedings of 2009 World Congress on Computer Science and Information Engineering (CSIE 2009). March 31-April 2, 2009. Los Angeles, USA. pp. 318-321.
- Kuptabuth, S., Netisopakul, P., "On Factors Affect Document Clustering: Comparison of Summary versus Full Documents", Proceedings of the 6th International Joint Conference on Computer Sciences and Software Engineering, May 13-15, 2009. Phuket, Thailand. pp. 236-241.
- Lertlitlitrungroj W., Netisopakul, P., "Simulation Modeling for Tollway Collection Decision Support System", Proceedings of the 6th International Conference on Computer Sciences and Software Engineering, May 13-15, 2009. Phuket, Thailand. Pp. 63-69.
- Lertlilitrungroj, W., Netisopakul, P., "Implementation of RFID-Based Electronic Toll Collection System: Case Study at Mukdahan Thai-Loas Bridge", Proceedings of the 2nd National Conference on Information Technology, November 6-7, 2008. Bangkok, Thailand. pp. 301-305.
- Leelapatra, W., Netisopakul, P., "Improving Query Expansion Using Link Analysis", Proceedings of ECTI-CON 2008, May 14-17, 2008, pp. 165-168.
- Netisopakul., P., Lertvikool, S., "Development of Vendor Managed Inventory Using Web Service", Annual Meeting of Global Academy of Business and Economic Research, September, 17-19, 2008, USA, pp. 342-351.
- Netisopakul, P., Kaewwan, K., "Thai Sentence Segmentation using M-ATN", Proceedings of the 7th Symposium of Natural Language Processing 2007. December 13-15, 2007. Chonburi, Thailand, pp. 91-96.
- Netisopakul, P., Siriumpankul, N., "Educational Service Web Database Prototype", in Communications in Computer and Information Science, 2007, Volume 2, Part 11, 479-488. Advanced Intelligent Computing Theories and Applications, With Aspects of Contemporary Intelligent Computing Techniques, Proceedings of Third International Conference on Intelligent Computing, August 2007, pp. 479-488.

- Netisopakul, P., "Web Metrics Support System (WMSS): Case Study at Faculty of Architecture, Chiang Mai University", Hawaii International Conference on Business, May 25-28, 2006, Honolulu, Hawaii, pp. 3761-3771.
- Netisopakul, P., Jaiyen, S., Intarapaiboon, P., Leenawong, C., "Experiments on Role-POS Tagging Learning Factors", Pacific Association for Computational Linguistics (PACLING 2005), August 24-27, 2005, Japan, pp. 269-273.
- Netisopakul, P., Leenawong, C., "Application of Nearest Neighbor Algorithm in E-Tourism Advisory System", International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), July 4-7, 2005, Korea, pp. 1141-1142.
- Netisopakul, P., Jaiyen, S., Intarapaiboon, P., "Role Assignment to Thai Word using Multilayer Perceptron Network", International Symposium on Mathematical, Statistical and Computer Sciences (MSCS 2005), February 19-20, 2005, Thailand, pp. 353-358.