โครงการ
การเรียนการสอนเพื่อเสริมประสบการณ์

ชื่อโครงการ โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย

A Question Answering program from Thai Wikipedia

ชื่อนิสิต นางสาวณัฐภาศ์ แจ้งสว่าง เลขประจำตัวนิสิต 5833624223

นางสาวปณิตา วิโรจน์วงษ์ชัย เลขประจำตัวนิสิต 5833640223

ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์

ปีการศึกษา 2561

โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย

นางสาวณัฐภาศ์ แจ้งสว่าง
นางสาวปณิตา วิโรจน์วงษ์ชัย

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2561

A Question Answering program from Thai Wikipedia

Naspa Changsawang

Panita Virojwongchai

A Project Submitted in Partial Fulfillment of the Requirements

for the Degree of Bachelor of Science Program in Computer Science

Department of Mathematics and Computer Science

Faculty of Science

Chulalongkorn University

Academic Year 2018

ชื่อโครงงาน (ภาษาไทย)	โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย
ชื่อโครงงาน (ภาษาอังกฤษ)	A Question Answering program from Thai Wikipedia
ผู้ดำเนินการ	1. นางสาวณัฐภาศ์ แจ้งสว่าง เลขประจำตัวนิสิต 5833624223
	2. นางสาวปณิตา วิโรจน์วงษ์ชัย เลขประจำตัวนิสิต 5833640223
สาขาวิชา	วิทยาการคอมพิวเตอร์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.ฑิตยา หวานวารี

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับโครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาบัณฑิต ในรายวิชา 2301499 โครงงานวิทยาศาสตร์ (Senior Project)

..... หัวหน้าภาควิชาคณิตศาสตร์

(ศาสตราจารย์ ดร.กฤษณะ เนียมมณี) และวิทยาการคอมพิวเตอร์

คณะกรรมการสอบโครงงาน

..... อาจารย์ที่ปรึกษาโครงงานหลัก

(ผู้ช่วยศาสตราจารย์ ดร.ฑิตยา หวานวารี)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.ธีรพงษ์ พงษ์พัฒนเจริญ)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.จารุโลจน์ จงสถิตย์วัฒนา)

นางสาวณัฐภาศ์ แจ้งสว่าง, นางสาวปณิตา วิโรจน์วงษ์ชัย: โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย (A Question Answering program from Thai Wikipedia) อ.ที่ปรึกษาโครงงานหลัก: ผศ.ดร.ทิตยา หวานวารี, 62 หน้า

ในยุคสมัยนี้ข้อมูลข่าวสารมีความจำเป็นต่อการดำเนินชีวิตมากขึ้น การค้นหาข้อมูลจึงเป็นสิ่งที่มีความสำคัญ ในปัจจุบันงานด้านการค้นหาข้อมูล ส่วนมากไม่ได้ตอบสนองผู้ใช้โดยตรง ผู้ใช้จะต้องค้นหาคำตอบที่ต้องการจากรายการเอกสาร แต่ในความเป็นจริงผู้ใช้ต้องการคำตอบเป็นข้อความสั้น ๆ โครงงานฉบับนี้จึงมีวัตถุประสงค์เพื่อพัฒนาโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย โดยผลลัพธ์ของคำถามเป็นช่วงคำสั้น ๆ ที่สืบค้น ซึ่งแบ่งออกเป็น 2 ขั้นตอน ได้แก่ การค้นคืนเอกสารและการสกัดคำตอบ จากการทดลองวิธีการจับคู่สายอักขระมีผลลัพธ์ในการค้นคืนเอกสารมากที่สุด 43 % และวิธีการคำนวณคะแนนจากตำแหน่งของคำโดยที่ k=10 และมีค่ากำหนด มีผลลัพธ์ในการค้นคืนคำตอบที่ถูกต้องเป็นอันดับแรก 18 %
ดังนั้นการรวมการค้นคืนเอกสารและการสกัดคำตอบเข้าด้วยกันสามารถตอบคำตอบถูกต้อง 5%

ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์ ลายมือชื่อนิสิต.....

ลายมือชื่อนิสิต.....

สาขาวิชา วิทยาการคอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาโครงงานหลัก.....

ปีการศึกษา 2561

5833624223, 5833640223: MAJOR COMPUTER SCIENCE

KEYWORDS : question answering, Thai language, NLP

Naspa Changsawang, Panita Virojwongchai: A Question Answering program from Thai Wikipedia. ADVISOR: ASSOC. PROF. Dittaya Wanvarie, Ph.D., 62pp.

Information plays an important role in the world since we are now in the information era. Hence, searching for data is important. Search engines on the internet do not directly give what users need. Users need to read through the documents to get the specific answer. To solve this problem, we have developed a question answering system which retrieves a short answer for a Thai question based on knowledge in Thai Wikipedia. The system consists of document retriever and answer extraction modules. From the experiment, the string matching method has the highest results in 43% of document retrieval. The result of calculating score from the position of the word, where k = 10 and set a threshold has the highest results in 18%. Thus, a question answering system extracts correct answer 5%.

Department: Mathematics and Computer Science Student's Signature Naspa Changsawang

Student's Signature Panita Virojwongchai

Field of Study: Computer Science Advisor's Signature.

Academic Year: 2018

กิตติกรรมประกาศ

โครงงาน “โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย” ได้รับการสนับสนุนอย่างเต็มที่ ทั้งนี้เนื่องจากได้รับความอนุเคราะห์และความช่วยเหลือจากคณาจารย์และบุคลากรต่าง ๆ หลายท่าน ซึ่งทำให้งานวิจัยนี้สำเร็จลุล่วงไปด้วยดี นอกจากนี้ยังมีผู้มีพระคุณอีกมากมายที่คอยสนับสนุนคอยให้คำปรึกษาอยู่เบื้องหลังงานวิจัยชิ้นนี้ ผู้เขียนรู้สึกเป็นเกียรติและปลื้มใจอย่างยิ่ง จึงขอขอบพระคุณอย่างสูงกับบุคลากรต่อไปนี้

ขอขอบพระคุณ ผศ.ดร.ฑิตยา หวานวารี อาจารย์ที่ปรึกษาโครงงานฉบับนี้และ ดร.นฤมล ประทานวณิช ที่คอยให้คำปรึกษา และและข้อเสนอแนะ ซึ่งช่วยพัฒนาประสิทธิภาพของโครงงานและยังช่วยแก้ไขแนวทางในการทำงานให้มีประสิทธิภาพที่ดียิ่งขึ้น

ขอขอบพระคุณครอบครัวที่คอยให้กำลังใจตลอดการพัฒนาโครงงาน แม้ในยามที่ต้องเจอกับปัญหาและอุปสรรคต่าง ๆ และเชื่อมั่นในตัวผู้เขียนเสมอมา

ขอขอบคุณเพื่อน ๆ คณะวิทยาศาสตร์ ภาควิชาคณิตศาสตร์ สาขาวิชาวิทยาการคอมพิวเตอร์ที่เข้าอกเข้าใจ คอยให้คำปรึกษา และเป็นกำลังใจให้กันจนงานชิ้นนี้สำเร็จลุล่วงไปด้วยดี

สารบัญ

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและเหตุผลการวิจัย

การสืบค้นข้อมูลข่าวสารเป็นสิ่งสำคัญอย่างยิ่งไม่ว่าจะยุคสมัยไหน ในปัจจุบันเทคโนโลยีสารสนเทศเข้ามามีบทบาทในชีวิตประจำวันของคนเรามากขึ้น สามารถค้นหาข้อมูลจากเครือข่ายอินเทอร์เน็ตได้ทุกที่ทุกเวลา โดยไม่ต้องเสียเวลาไปห้องสมุด ค้นหาจากวารสาร หรือสอบถามจากบุคคลผู้รู้อื่นๆ มนุษย์มีการสืบค้นข้อมูลตลอดช่วงอายุคน เพื่อนำมาใช้ประโยชน์ในด้านต่าง ๆ อย่างมากมาย ทั้งด้านการเรียน การทำงาน หรือในชีวิตประจำวัน ดังนั้น ความสะดวกรวดเร็วและความถูกต้องของข้อมูลจึงเป็นสิ่งที่ผู้ใช้ต้องการ

ในอินเทอร์เน็ตมีข้อมูลสารสนเทศมากมาย ผู้ใช้สามารถสืบค้นได้ ทั้งข้อความ รูปภาพ สื่อมัลติมิเดีย ภาพเคลื่อนไหว วีดิโอ และข้อมูลอื่น ๆ ได้ตามต้องการ การที่ผู้ใช้จะค้นหาสิ่งที่ต้องการในข้อมูลจำนวนมากนั้นเป็นเรื่องยาก จึงต้องอาศัยเครื่องมือที่เรียกว่า ระบบค้นหาข้อมูล (Search Engine) ซึ่งเป็นระบบที่สามารถนำไปประยุกต์ใช้งานได้อย่างกว้างขวาง แต่ในปัจจุบันระบบค้นหาข้อมูลไม่ได้ตอบสนองผู้ใช้โดยตรง เนื่องจากระบบจะค้นหาเอกสารที่เกี่ยวข้องกับคำสำคัญ (Keyword) ที่ผู้ใช้ค้น หลังจากนั้นผู้ใช้ต้องใช้เวลาในการอ่านเอกสารและหาคำตอบเองซึ่งเสียเวลา โปรแกรมตอบคำถามจะช่วยเพิ่มประสิทธิภาพของระบบค้นหาข้อมูลให้มีความรวดเร็วมากยิ่งขึ้น

การสืบค้นข้อมูลและเข้าถึงข้อมูลสารสนเทศสามารถทำผ่านเว็บเบราว์เซอร์ทั่วโลก เว็บไซต์ที่ให้บริการค้นหาข้อมูลมีมากมาย เช่น sanook yahoo และ blogger เป็นต้น ซึ่งเว็บไซต์ที่รวบรวมข้อมูลจำนวนมากและได้รับความนิยมมากที่สุดมีชื่อว่า วิกิพีเดีย เป็นสารานุกรมที่มีเนื้อหาเสรีครอบคลุมหลายหัวเรื่องอย่างกว้างขวาง และมีหลากหลายภาษา แต่ละภาษาจะมีโครงสร้างที่แตกต่างกัน ทั้งอักขระที่ใช้และไวยากรณ์ เราจึงต้องดัดแปลงระบบสำหรับแต่ละภาษา เช่น ข้อความภาษาไทยจะมีโครงสร้างภาษาต่างจากภาษาอังกฤษ การเรียงลำดับคำ การเว้นเครื่องหมายวรรคตอน และภาษาไทยจะมีสระและวรรณยุกต์เข้ามาผสมในคำหรือประโยค ดังนั้นคณะผู้จัดทำจึงต้องการพัฒนาโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย เพื่อแก้ปัญหาที่กล่าวมาข้างต้น

1.2 วัตถุประสงค์

เพื่อพัฒนาโปรแกรมตอบคำถามโดยการค้นหาคำตอบจากวิกิพีเดียภาษาไทย เพื่อให้ตอบคำถามสะดวกรวดเร็วมากขึ้น

1.3 ขอบเขตโครงงาน

1. โครงงานนี้ศึกษาเฉพาะการค้นคืนคำตอบจากวิกิพีเดียภาษาไทย
2. ผลลัพธ์ของคำถามเป็นช่วงคำสั้น ๆ ที่สืบค้น มาจากวิกิพีเดียภาษาไทย ณ วันที่รวบรวมข้อมูลเท่านั้น
3. ระบบสามารถตอบเฉพาะคำถามเกี่ยวกับข้อเท็จจริงเท่านั้น ระบบไม่สามารถตอบคำถามที่เป็นความคิดเห็นได้ รวมถึงไม่สามารถตอบคำถามแบบถูกหรือผิดได้
4. ระบบสามารถตอบคำถามที่เป็นภาษาไทยเท่านั้น โดยคำถามจะต้องมีการสะกดคำที่ถูกต้องตามพจนานุกรมฉบับราชบัณฑิตยสถาน

1.4 ขั้นตอนการวิจัย

1. ค้นหาและศึกษางานวิจัยรวมถึงเนื้อหาที่เกี่ยวข้องกับการถามตอบ โดยอ้างอิงขั้นตอนการทำงานจากงานวิจัยของ Chen และคณะ [4] ซึ่งเป็นระบบที่ใช้วิกิพีเดียภาษาอังกฤษเป็นฐานข้อมูลในการค้นคืนเอกสารและหาคำตอบ และใช้ฐานข้อมูล SQuad ในการฝึกสอนและทดสอบระบบ
2. ศึกษาเครื่องมือ โปรแกรมและเทคนิคที่ใช้ในงานวิจัย เพื่อเป็นประโยชน์ในการทำโครงงาน โดยศึกษาการตัดคำในคลังโปรแกรม deepcut [7] และ tltk [8] ศึกษาการแปลงคำเป็นเวกเตอร์ การแปลงคำเป็นตัวเลข โดยอาศัยตัวแบบโครงข่ายประสาทเทียมแบบจำสั้นจำยาว (Long Short Term Memory) ซึ่งเป็นระบบโครงข่ายประสาทเทียมแบบวนซ้ำ (Recurrent Neural Network)การกำกับหมวดคำ (Part Of Speech Tagging) และ การสกัดนิพจน์ระบุนาม (Name Entity Recognition) การกำหนดน้ำหนักคำ (TF-IDF) เพื่อหาคำที่มีความสำคัญ
3. รวบรวมข้อมูล เพื่อกำหนดขอบเขตของโครงงาน
4. วิเคราะห์และออกแบบวิธีการ แบ่งการทำงานออกเป็น 3 ส่วนหลัก ๆ ได้แก่ การค้นคืนหัวข้อเอกสารที่เกี่ยวข้องกับคำถาม การอ่านเอกสาร และการหาคำตอบ
5. พัฒนาระบบถามตอบ
6. ตรวจสอบความถูกต้องและวัดประสิทธิภาพโดยตรวจสอบความถูกต้องของคำตอบจากชุดข้อมูลทดสอบ
7. สรุปผลการดำเนินงาน
8. จัดทำเอกสาร

ตารางเวลาการดำเนินงาน

ขั้นตอนการดำเนินงาน	ปี 2561					ปี 2562			
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.
1.ค้นหาและศึกษาบทความรวมถึงเนื้อหาที่เกี่ยวข้องกับการถามตอบ									
2.ศึกษาเครื่องมือ โปรแกรมและเทคนิคที่ใช้ในงานวิจัย									
3.รวบรวมข้อมูล เพื่อกำหนดขอบเขตของโครงงาน									
4.วิเคราะห์และออกแบบวิธีการ									
5.พัฒนาระบบถามตอบ									
6.ตรวจสอบความถูกต้องและวัดประสิทธิภาพ									
7.สรุปผลการดำเนินงาน									
8.จัดทำเอกสาร									

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ต่อผู้พัฒนา

1. มีความรู้ ความเข้าใจในการทำระบบถามตอบ
2. เพิ่มพูนทักษะการเขียนโปรแกรมและการพัฒนาระบบ
3. เรียนรู้การคิดวิเคราะห์วางแผนการทำงานอย่างเป็นระบบแบบแผน เพื่อให้เกิดประโยชน์สูงสุดตามทรัพยากรที่มีอยู่
4. ฝึกการเรียนรู้ด้วยตนเอง การยอมรับฟังความคิดเห็นของผู้อื่น ความตรงต่อเวลา ความรับผิดชอบในหน้าที่

ประโยชน์ต่อผู้ใช้ระบบ

1. เป็นทางเลือกให้ผู้ใช้งานเลือกในการค้นหาข้อมูลสารสนเทศ
2. ผู้ใช้งานสามารถค้นหาและเข้าถึงข้อมูลได้สะดวกรวดเร็ว ผลลัพธ์ที่ได้จากการค้นหาจะเป็นข้อความสั้น ๆ ซึ่งผู้ใช้งานไม่ต้องเสียเวลามาอ่านเอกสารเพื่อหาคำตอบ รวมถึงบอกชื่อเอกสารที่มาของคำตอบด้วย
3. ผลลัพธ์จากการค้นหามีความถูกต้องและแม่นยำ

1.6 โครงสร้างของรายงาน

โครงงานฉบับนี้ประกอบไปด้วยเนื้อหา 5 บทดังต่อไปนี้

บทที่ 1 กล่าวถึงบทนำ

บทที่ 2 กล่าวถึงงานวิจัยและความรู้พื้นฐานที่เกี่ยวข้อง

บทที่ 3 กล่าวถึงการออกแบบระบบและขั้นตอนการดำเนินงานการค้นคืนเอกสารและการสกัดคำตอบ

บทที่ 4 กล่าวถึงผลการทดสอบของระบบ

และบทที่ 5 กล่าวถึงข้อสรุป ปัญหาและอุปสรรคที่เกิดขึ้นในระหว่างการทำงาน และข้อเสนอแนะ

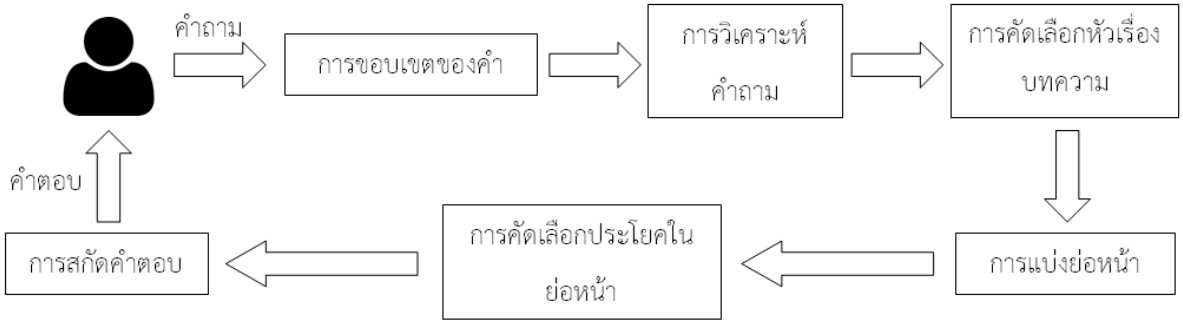
บทที่ 2
งานวิจัยและความรู้พื้นฐานที่เกี่ยวข้อง

2.1 ระบบคำถาม คำตอบ

จากรูปที่ 2.1 เป็นภาพรวมระบบงานวิจัยของ Ryu และคณะ [1] ใช้การตัดคำและกำกับหมวดคำในการวิเคราะห์คำถาม การวิเคราะห์คำถามแบ่งเป็น 3 เกณฑ์ คือ

1. รูปแบบของคำตอบ เช่น ข้อเท็จจริง (factoid) รายการ และคำตอบเชิงพรรณา
2. เนื้อหาของคำตอบ แบ่งว่าคำตอบเป็นอะไร เช่น บุคคล สถานที่ วันเวลา
3. เป้าหมายของคำถามและคุณสมบัติของเป้าหมายนั้น เช่น คำถามว่า “มาบุญครองอยู่ที่ไหน” มี “มา บุญครอง” เป็นเป้าหมายของคำถาม และคุณสมบัติที่ต้องการคือสถานที่ที่มาบุญครองตั้งอยู่

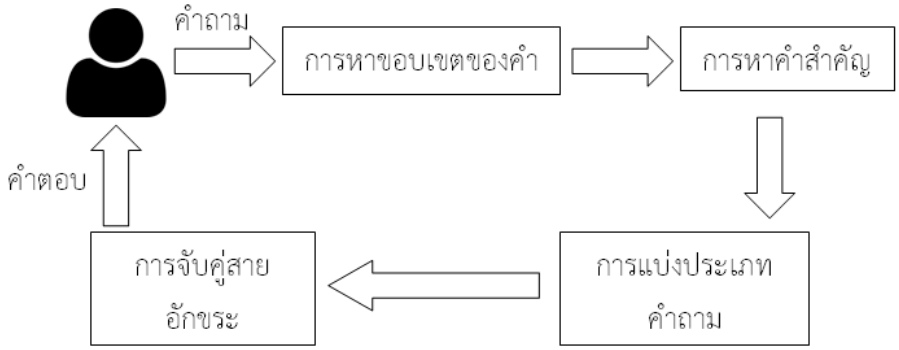
ระบบประกอบด้วย 3 ขั้นตอน ดังนี้ การวิเคราะห์คำถาม การค้นคืนเอกสาร และ การตอบคำถาม โดยระบบจะวิเคราะห์คำถาม แล้วเทียบคำถามกับหัวข้อของบทความ ถ้าคำถามและหัวข้อบทความตรงกันค่าคะแนนจะมีค่าเป็น 1 หมายความว่า เลือกค้นคืนเอกสารนี้ ต่อมาแบ่งบทความออกเป็นบทความย่อย ๆ แล้ววัดระยะระหว่างคำในคำถามคำในคำถามกับคำที่น่าจะเป็นคำตอบ เพื่อดูว่าประโยคใดในบทความตรงกับคำถามมากที่สุด สุดท้ายระบบจะนำคะแนนมาคูณกัน แล้วเลือกคำตอบที่มีคะแนนสูงที่สุด ผลการทดลองพบว่า ระบบมีความถูกต้อง 71% ในกลุ่มคำตอบแบบข้อเท็จจริง ส่วนกลุ่มคำตอบแบบรายการ และคำตอบเชิงพรรณา ใช้ F1-score เป็นเกณฑ์ ได้ 48% และ 33% ตามลำดับ



รูปที่ 2.1 ภาพรวมระบบงานวิจัยของ Ryu และคณะ

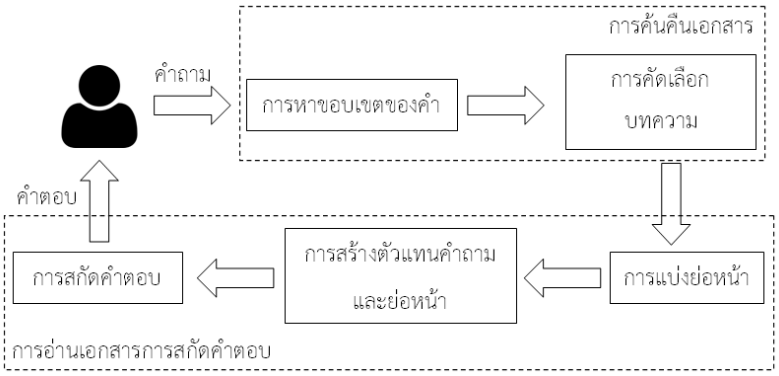
จากรูปที่ 2.2 เป็นภาพรวมระบบงานวิจัยของ Decha และ Patanukhom [2] ใช้โปรแกรม LexTo [3] ซึ่งเป็นการจับคู่คำแบบยาวที่สุด (longest matching) ในการตัดคำ และตัดแบ่งประโยคโดยใช้ข่ายงานประสาทเทียม ซึ่งพิจารณาจากการเว้นวรรค โดยนับคำที่อยู่หน้าและหลังเว้นวรรค จากนั้นจัดแบ่งแยกจำนวน ชนิดคำนาม คำกริยาและชนิดตัวเลข ต่อมาจัดกลุ่มคำถามโดยแบ่งคำถามออกเป็น 7 ประเภท คือ เท่าไร ใคร ที่ไหน เมื่อใด อย่างไร แบบใดและ สิ่งใด แล้วตัดคำที่ไม่จำเป็น (stop word) ออก เพื่อนำคำสำคัญของคำถาม

และบริบทมาใช้ในการหาคำตอบที่น่าจะเป็นไปได้ด้วยการจับคู่คำ ผลการทดลอง แบ่งเป็น 2 ส่วน ได้แก่ การแบ่งประโยคซึ่งใช้การตรวจสอบแบบไขว้ 10 กลุ่ม (10-fold cross validation) ในการวัดประสิทธิภาพ โดยมีค่าเฉลี่ยความถูกต้อง 81.31% ค่าความเที่ยงตรง (precision) 81.36% ค่าการค้นคืน (recall) 81.33% ส่วนการตอบคำถาม มีค่า MRAR (mean reciprocal answer rank) เท่ากับ 0.657 และค่าความถูกต้องเท่ากับ 75.71%



รูปที่ 2.2 ภาพรวมระบบงานวิจัยของ Decha และ Patanukhom

จากรูปที่ 2.3 เป็นภาพรวมระบบงานวิจัยของ Chen และคณะ [4] ระบบแบ่งเป็น 2 ส่วน ได้แก่ การค้นคืนเอกสาร (Document Retriever) เริ่มจากตัดคำโดยใช้ Stanford CoreNLP toolkit จากนั้นระบบจะเลือกบทความ 5 บท ที่เกี่ยวข้องกับคำถาม โดยใช้ดัชนีแบบอินเวอร์ต (inverted index) และค่า TF-IDF ซึ่งมีประสิทธิภาพมากกว่าการใช้ ElasticSearch [5] และใช้การแฮช 2-แกรม (bigram hashing) ในการเพิ่มประสิทธิภาพการค้นคืน เมื่อได้บทความที่เกี่ยวข้องแล้ว ขั้นตอนการอ่านเอกสาร (Document Reader) จะแบ่งบทความออกเป็นย่อหน้าแล้วใช้ตัวแบบ LSTM สร้างตัวแทนของย่อหน้า โดยใช้เวกเตอร์แทนคำของ Glove [6] รวมถึงหมวดคำ นิพจน์ระบุนาม และความถี่ของคำ เป็นข้อมูลเข้าของการสร้างตัวแทนย่อหน้า ต่อมาทำตัวแทนคำถาม โดยใช้ข่ายงานประสาทเทียมแบบวนซ้ำโดยป้อนข้อมูลเข้าเป็นเวกเตอร์แทนคำเช่นเดียวกับขั้นตอนการอ่านเอกสาร หลังจากนั้นใช้ตัวจำแนกคำต้นประโยคและท้ายประโยค ผลการทดลองพบว่าระบบมีความถูกต้อง 70.0% และ ค่า F1-score 79.0%



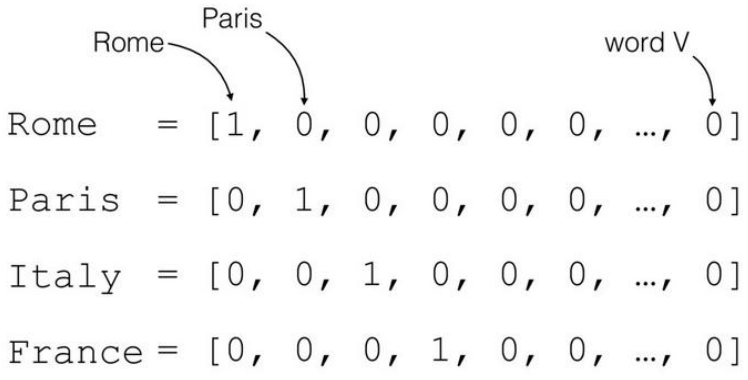
รูปที่ 2.3 ภาพรวมระบบจากงานวิจัยของ Chen และคณะ

2.2 การแทนคำและข้อความด้วยเวกเตอร์

มนุษย์สามารถเข้าใจข้อความ ภาพและเสียง แต่คอมพิวเตอร์สามารถประมวลผลได้แต่ตัวเลข จึงต้องมีการแปลงคำเป็นตัวเลขแล้วเก็บค่าในรูปแบบเวกเตอร์ ซึ่งมีหลายวิธีดังนี้

1. One Hot Vector เป็นวิธีการที่ง่ายที่สุดในการแปลงคำให้เป็นตัวเลขอยู่ในรูปของเวกเตอร์ฐานสอง (Binary Vector) ซึ่งเวกเตอร์ของคำต่าง ๆ จะดูแค่ว่ามีคำนั้น ๆ อยู่หรือไม่เท่านั้น ตามรูปที่ 2.4

   ตัวอย่าง



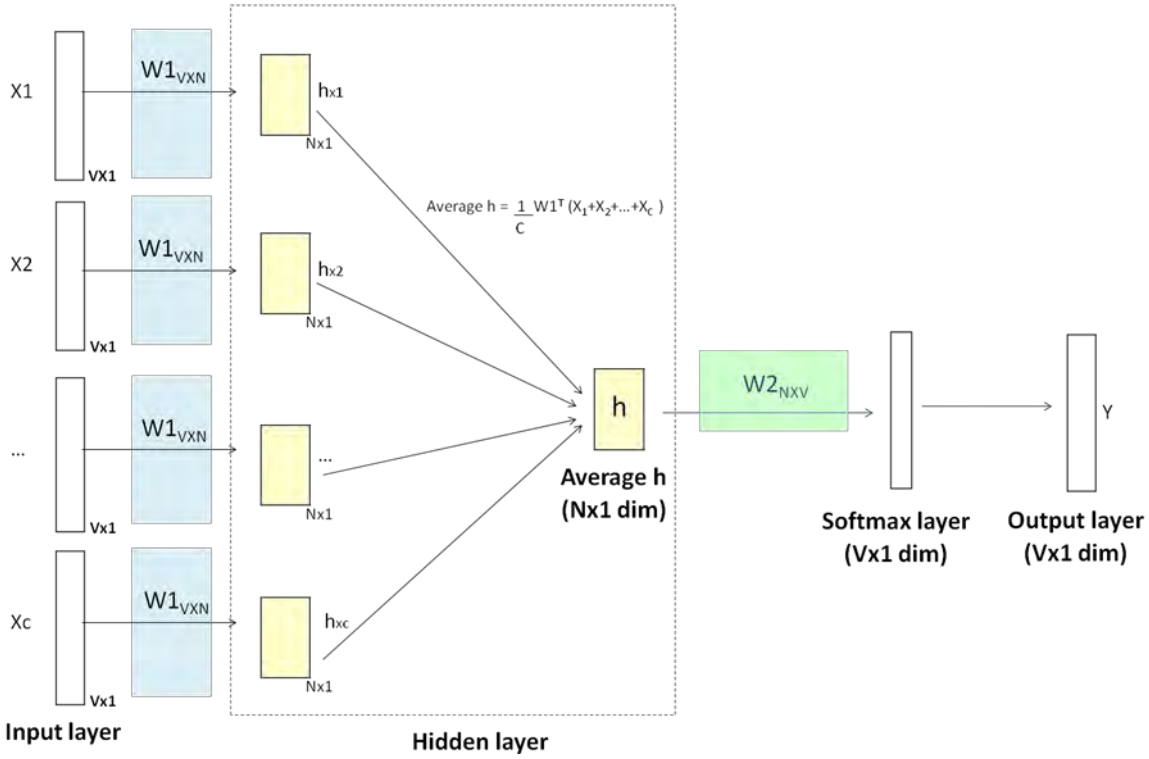
รูปที่ 2.4 ตัวอย่าง One Hot vector

นำมาจาก https://medium.com/@athif.shaffy/one-hot-encoding-of-text-b69124bef0a7

ข้อเสียของวิธีนี้คือ หากคำในชุดข้อมูลมีจำนวนมาก ขนาดเวกเตอร์ที่ใช้แทนคำก็จะมีขนาดใหญ่ขึ้น ทำให้ต้องใช้เวลาประมวลผลมากขึ้น

2. Word2Vec [12], [13], [14] เป็นการแปลงคำเป็นตัวเลขโดยอาศัยโครงข่ายประสาทเทียมแบบป้อนหน้า (Feed forward network) เข้ามาช่วยในการฝึกสอนให้ตัวแบบเรียนรู้ แล้วเก็บค่าตัวเลขในรูปแบบเวกเตอร์ ซึ่งเวกเตอร์ของคำต่าง ๆ ถูกคำนวณจากคำที่ห้อมล้อมคำนั้น ๆ นอกจากนี้ยังสามารถหาความคล้ายคลึงของคำผ่านเวกเตอร์ของคำได้ โดยอาศัยผลคูณจุด (dot product) ของเวกเตอร์ของคำทั้งสอง การฝึกสอนจะใช้ข้อมูลที่เป็นเอกสารซึ่งประกอบด้วยคำต่าง ๆ เรียงกันอยู่เป็นลำดับ จากนั้นนำกรอบหน้าต่าง (window) ขนาดคงที่ค่าหนึ่งซึ่งเปรียบเสมือนช่วงวลีของคำที่เราสนใจและคำรอบข้าง ต่อมา เลื่อนกรอบหน้าต่างดังกล่าวไปเรื่อย ๆ เพื่อฝึกสอนคำทั้งหมดที่ปรากฏในเอกสาร ในการเลื่อนกรอบหน้าต่าง 1 ครั้งแบบจำลองจะนำคำที่สนใจไปแปลงเป็นเวกเตอร์ในรูปแบบ One Hot Vector เพื่อป้อนเป็นข้อมูลขาเข้าแก่ตัวแบบโครงข่ายประสาทเทียม วิธีการฝึกสอนแบ่งออกเป็น 2 รูปแบบ

2.1 Continuous Bag of Words (CBOW) คือ การนำคำบริบทมาทำนายคำเป้าหมาย (target word) ซึ่งมีการทำงานตามรูปที่ 2.5



รูปที่ 2.5 ภาพรวมโครงข่ายประสาทเทียมในการสร้างเวกเตอร์แทนคำแบบ CBOW

- ชั้นข้อมูลขาเข้า (input layer) มี One Hot Vector X1 ,X2 ,...,Xc เป็นบัพนำเข้า (node) จำนวน C บัพ โดยแต่ละบัพเป็นเวกเตอร์ที่มีมิติเท่ากับจำนวนคำศัพท์ (V) ซึ่งเป็น One hot vector ของคำบริบท
- ชั้นซ่อนตัว (hidden layer) กำหนดให้เวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นมีมิติ N โดย N มีค่าน้อยกว่าจำนวนคำศัพท์ (V) โดยมีเวกเตอร์ตัวแทนของคำ (h_x) เป็นบัพซ่อน (node) ซึ่งการเชื่อมจากบัพนำเข้ามาเป็นบัพซ่อน จะใช้การนำเมทริกซ์น้ำหนักของชั้นซ่อนตัว ($W1_{VxN}$)ที่เป็น embedding layer คูณกับค่าบัพนำเข้า จากนั้นนำค่าทุกบัพซ่อนมาหาค่าเฉลี่ย (Average h)
  Average h = $\frac{1}{C} W1^{T} (X_1+X_2+...+X_C) = \frac{1}{C} (h_{X1}+h_{X2}+...+h_{Xc})$
- ชั้นข้อมูลขาออก (output layer) มีมิติของเวกเตอร์เท่ากับจำนวนคำศัพท์ (V) โดยเวกเตอร์ชั้นซ่อนตัวจะคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออก ($W2_{NxV}$) ซึ่งค่าที่ได้จากชั้นนี้จะมีการปรับค่าโดยใช้ฟังก์ชัน softmax และปล่อยค่าเวกเตอร์ในรูปแบบ One Hot Vector ของคำเป้าหมายที่เราสนใจ

การฝึกสอนระบบจะต้องการทั้งตัวอย่างที่ถูก (positive samples) คือ ทำนายคำที่อยู่ตรงกลางของกรอบหน้าต่างถูก และตัวอย่างที่ผิด (negative samples) คือ ทำนายคำที่ไม่ใช่คำตรงกลางของกรอบหน้าต่างซึ่งเอกสารจะสร้างได้เฉพาะตัวอย่างที่ถูก ส่วนตัวอย่างที่ผิดต้องสร้างเอง โดยเลือกคำอื่น ๆ ที่ไม่อยู่ในกรอบหน้าต่างมาใช้ แต่เนื่องจากขนาดคำศัพท์ (V) ใหญ่มาก และคำที่ถูกซึ่งอยู่ตรงกลางของกรอบหน้าต่างมีเพียงคำเดียว หากใช้คำอื่น ๆ ทั้งหมดมาสร้างตัวอย่างที่ผิด จะมีตัวอย่างที่ผิดมากจนเกินไป ส่งผลให้ระบบใช้เวลาในการประมวลผลมาก ดังนั้นระบบจึงจะสุ่มคำศัพท์มาเพียงเท่ากับจำนวนคำในกรอบหน้าต่าง เพื่อสร้างเป็นตัวอย่างที่ผิด

ขั้นตอนการฝึกสอนมีดังนี้

1. แปลงคำบริบทให้อยู่ในรูปแบบ One Hot Vector
2. นำ One Hot Vector ของคำบริบท มาคูณกับเมทริกซ์น้ำหนักของชั้นซ่อนตัวจะได้เวกเตอร์แทนคำบริบทแต่ละคำ ซึ่งจะเอาแถวที่ตรงกันออกมา เนื่องจากแต่ละแถวของเมทริกซ์น้ำหนัก จะเป็นเวกเตอร์ตัวแทนของแต่ละคำ
3. นำเมทริกซ์ที่ได้จากข้อ 2 มาหาค่าเฉลี่ยและนำมาคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออกจะได้เวกเตอร์คำเป้าหมาย
4. นำเวกเตอร์คำเป้าหมายมาปรับค่าโดยใช้ฟังก์ชัน softmax แล้วแปลงเป็น One Hot Vector

<u>ตัวอย่าง</u> ประโยค "I am student" จำนวนคำศัพท์เท่ากับ 3 สมมติให้กรอบหน้าต่างเท่ากับ 1 และมิติของเวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นเท่ากับ 2 ดังรูปที่ 2.6

กำหนด One hot vector ของแต่ละคำและเมทริกซ์น้ำหนักของชั้นซ่อนตัว (W1) ดังนี้

One hot vector ของ I : 1 0 0

One hot vector ของ am : 0 1 0

One hot vector ของ student : 0 0 1

เมทริกซ์น้ำหนักของชั้นซ่อนตัว (W1)

0.1	**0.3**
0.2	**0.1**
0.3	**0.2**

โดยที่ แถว 1 คือ เวกเตอร์ตัวแทนของ I

แถว 2 คือ เวกเตอร์ตัวแทนของ am

แถว 3 คือ เวกเตอร์ตัวแทนของ student

ขั้นขาเข้า : คำบริบท I

- แปลงคำบริบท I ให้อยู่ในรูปแบบของ One hot vector จากที่กำหนดไว้จะได้

$$1 \quad 0 \quad 0$$

ซึ่งเป็น One hot vector ของ I เป็นค่าบัพนำเข้า

ขั้นซ่อนตัว

- นำค่าบัพนำเข้าคูณกับเมทริกซ์น้ำหนัก จะได้เวกเตอร์ตัวแทนของ ‘I’ เป็นค่าบัพซ่อน

$$\begin{matrix}1 & 0 & 0\end{matrix} \times \begin{matrix}0.1 & 0.3\\ 0.2 & 0.1\\ 0.3 & 0.2\end{matrix} = \begin{matrix}0.1 & 0.3\end{matrix}$$

ขั้นข้อมูลขาออก

- นำค่าเวกเตอร์ที่ได้จากขั้นซ่อนตัว คูณกับเมทริกซ์น้ำหนักขั้นซ่อนตัว จะได้ค่าเวกเตอร์ขาออก

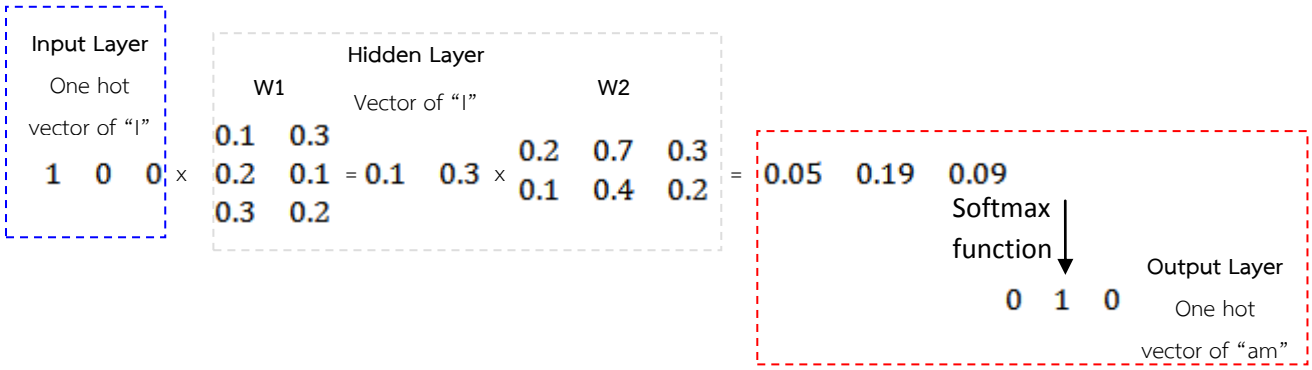
$$\begin{matrix}0.1 & 0.3\end{matrix} \times \begin{matrix}0.2 & 0.7 & 0.3\\ 0.1 & 0.4 & 0.2\end{matrix} = \begin{matrix}0.05 & 0.19 & 0.09\end{matrix}$$

- นำค่าเวกเตอร์ขาออกใช้ softmax function ปล่อยค่า One hot vector ของคำทำนายคำเป้าหมาย

$$\begin{matrix}0.05 & 0.19 & 0.09\end{matrix} \xrightarrow{\text{Softmax function}} \begin{matrix}0 & 1 & 0\end{matrix}$$

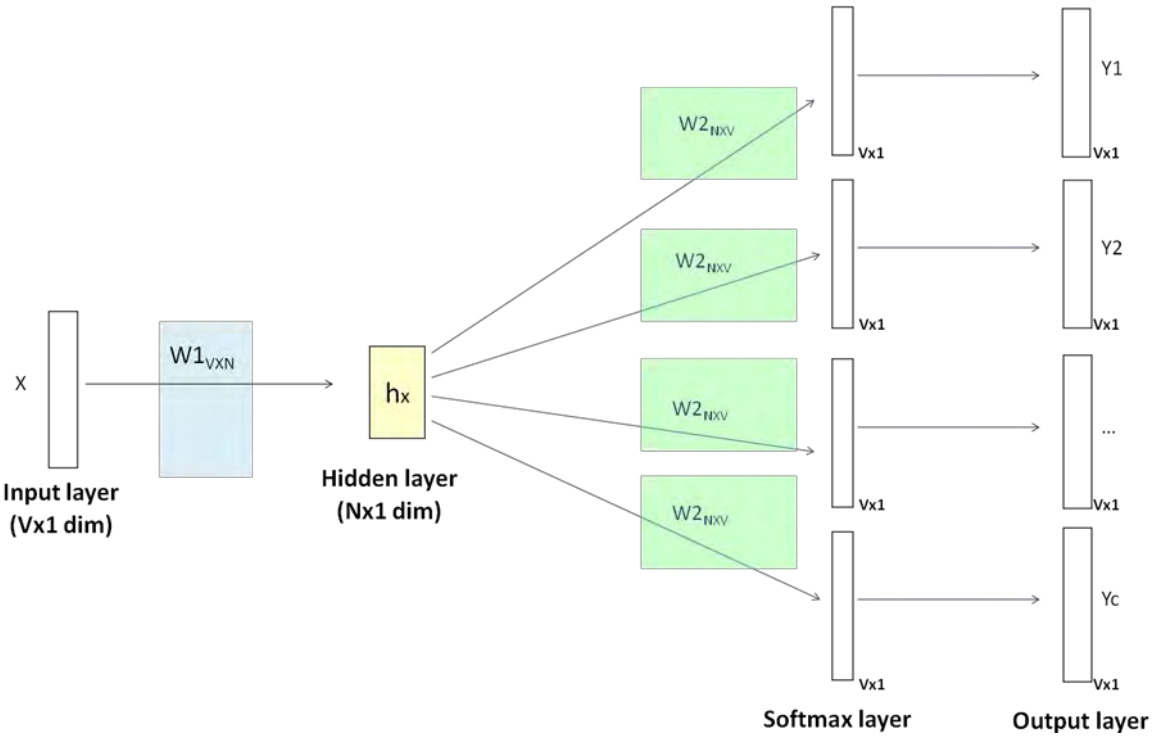
ซึ่งเป็นค่า One hot vector ของ am จากค่าที่กำหนดไว้ข้างต้น

ภาพโดยรวมตัวอย่างการคำนวณข้างต้น



รูปที่ 2.6 ตัวอย่างการคำนวณเวกเตอร์ตัวแทนแบบ CBOW

2.2 Skip gram คือ การนำคำเป้าหมายมาทำนายคำบริบท Skip gram จะใช้เวลาในการประมวลผลนานกว่า CBOW ซึ่งคำที่มีบริบทใกล้เคียงกัน จะถูกแปลงเป็นเวกเตอร์ตัวแทนที่มีหน้าตาคล้ายกัน มีการทำงานตามรูปที่ 2.7



รูปที่ 2.7 ภาพรวมโครงข่ายประสาทเทียมในการสร้างเวกเตอร์แทนคำแบบ Skip gram

โครงสร้างของโครงข่ายประสาทเทียมมี ดังนี้

- ชั้นข้อมูลขาเข้า(input layer) มี บัพนำเข้า (node) X เป็น One hot vector ของคำเป้าหมายที่มีมิติเท่ากับจำนวนคำศัพท์(V)
- ชั้นซ่อนตัว (hidden layer) กำหนดให้เวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นมีมิติ N โดย N มีค่าน้อยกว่าจำนวนคำศัพท์ (V) มีเวกเตอร์ตัวแทนของคำ (h_x) เป็นบัพซ่อน (hidden node) โดยเชื่อมจากขั้นนำเข้าด้วยเมทริกซ์น้ำหนักของชั้นซ่อนตัว ($W1_{VxN}$) ซึ่งเป็น embedding layer
- ชั้นข้อมูลขาออก (output layer) มีจำนวนบัพ C บัพ โดยค่าแต่ละบัพได้จากการที่ได้จากชั้นนี้จะมีการปรับค่าโดยใช้ฟังก์ชัน softmax และปล่อยค่าเวกเตอร์ในรูปแบบ One Hot Vector ของคำต่าง ๆ ที่ถูกทำนายว่าเป็นคำบริบทของคำเป้าหมายที่เราสนใจ โดยแต่ละเวกเตอร์จะมีขนาดเท่ากับจำนวนคำศัพท์ (V)

ขั้นตอนการฝึกสอนมีดังนี้

1. แปลงคำเป้าหมายให้อยู่ในรูปแบบ One Hot Vector
2. นำ One Hot Vector ของคำเป้าหมายมาคูณกับเมทริกซ์น้ำหนักของชั้นซ่อนตัวจะได้เวกเตอร์แทนคำเป้าหมาย ซึ่งจะเอาแถวที่ตรงกันออกมา เนื่องจากแต่ละแถวของเมทริกซ์น้ำหนัก จะเป็นเวกเตอร์ตัวแทนของแต่ละคำ
3. นำเมทริกซ์ที่ได้จากข้อ 2 มาคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออกจะได้เวกเตอร์คำบริบททำนาย

4. นำเวกเตอร์คำบริบททำนายมาปรับค่าโดยใช้ฟังก์ชัน softmax แล้วแปลงเป็น One Hot Vector เทียบกับคำบริบทที่ต้องการ ปรับค่าเมทริกซ์น้ำหนักทั้งหมด แล้วทำข้อ 2-4 ซ้ำจนครบทุกคำบริบทที่ต้องการ

<u>ตัวอย่าง</u> ประโยค "I am student" จำนวนคำศัพท์เท่ากับ 3 สมมติให้กรอบหน้าต่างเท่ากับ 1 และมิติของเวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นเท่ากับ 2 ดังรูปที่ 2.8

กำหนด one hot vector ของคำศัพท์และเมทริกซ์น้ำหนักของชั้นซ่อนตัว (W1) ดังนี้

One hot vector ของ I : 1 0 0

One hot vector ของ am : 0 1 0

One hot vector ของ student : 0 0 1

เมทริกซ์น้ำหนักของชั้นซ่อนตัว (W1)

$$\begin{matrix} 0.1 & 0.3 \\ 0.2 & 0.1 \\ 0.4 & 0.3 \end{matrix}$$

โดยที่ แถวที่ 1 คือ เวกเตอร์ตัวแทนของ I

แถวที่ 2 คือ เวกเตอร์ตัวแทนของ am

แถวที่ 3 คือ เวกเตอร์ตัวแทนของ student

ขั้นขาเข้า : คำเป้าหมาย am

- แปลงคำเป้าหมาย am ให้อยู่ในรูปแบบของ One hot vector เป็นค่าบัพนำเข้า

$$0 \quad 1 \quad 0$$

ขั้นซ่อนตัว

- นำค่าบัพนำเข้าคูณกับเมทริกซ์น้ำหนัก จะได้เวกเตอร์ตัวแทนของ ‘am’ เป็นค่าบัพซ่อน

$$\begin{matrix} 0 & 1 & 0 \end{matrix} \times \begin{matrix} 0.1 & 0.3 \\ 0.2 & 0.1 \\ 0.4 & 0.3 \end{matrix} = \begin{matrix} 0.2 & 0.1 \end{matrix}$$

ขั้นข้อมูลขาออก

- นำค่าเวกเตอร์ที่ได้จากขั้นซ่อนตัว คูณกับเมทริกซ์น้ำหนักขั้นซ่อนตัว จะได้ค่าเวกเตอร์ขาออก

$$\begin{matrix} 0.2 & 0.1 \end{matrix} \times \begin{matrix} 0.2 & 0.1 & 0.3 \\ 0.8 & 0.3 & 0.7 \end{matrix} = \begin{matrix} 0.12 & 0.08 & 0.13 \end{matrix}$$

- ใช้ softmax function ปรับค่าเวกเตอร์ขาออก ให้เป็น One hot vector
  จะได้ค่า one hot vector ของคำเป้าหมาย student

0.12 0.08 0.13 → (Softmax function) → 0 0 1

ภาพรวมตัวอย่างการคำนวณข้างต้น

รูปที่ 2.8 ตัวอย่างการคำนวณเวกเตอร์ตัวแทนแบบ Skip gram

ขั้นตอนการใช้งาน Word2Vec

ผลลัพธ์หลังจากการฝึกสอนจะได้เวกเตอร์ที่เป็นตัวแทนของคำเป้าหมายแต่ละคำ จากแต่ละแถวของเมทริกซ์น้ำหนักของชั้นซ่อนตัว และเวกเตอร์ตัวแทนคำบริบทจากแต่ละหลักของเมทริกซ์น้ำหนักของชั้นข้อมูลขาออก มาใช้ในการหาค่าความคล้ายระหว่างคำ มีดังนี้

1. แปลงคำที่สนใจให้อยู่ในรูปแบบ One Hot Vector
2. นำ One Hot Vector ของคำที่สนใจ มาคูณกับเมทริกซ์น้ำหนักของชั้นซ่อนตัวจะได้เวกเตอร์แทนคำ เนื่องจากแต่ละแถวของเมทริกซ์น้ำหนัก เป็นเวกเตอร์ตัวแทนของแต่ละคำ
3. หาค่าความคล้ายของคำทั้งสอง โดยหาค่าผลคูณจุดระหว่างเวกเตอร์แทนคำ 2 คำที่ต้องการเปรียบเทียบ

<u>ตัวอย่าง</u> ประโยค "I am student" สมมติให้มิติของเวกเตอร์ตัวแทนคำที่ต้องการสร้างขึ้นเท่ากับ 2 โดยผลลัพธ์จาการฝึกสอน ได้เมทริกซ์น้ำหนัก W1 ดังนี้

W1

0.1	0.3
0.2	0.1
0.3	0.2

การหาค่าความคล้ายระหว่างคำว่า am กับ คำว่า student สามารถทำได้โดยการนำค่าเวกเตอร์ตัวแทนของ am ทำผลคูณจุดกับเวกเตอร์ตัวแทนของ student

หาค่าเวกเตอร์ตัวแทนของ am โดยการนำ one hot vector ของ am คูณกับค่าเมทริกซ์น้ำหนัก W1

$$\begin{matrix}0 & 1 & 0\end{matrix} \times \begin{matrix}0.1 & 0.3\\ 0.2 & 0.1\\ 0.3 & 0.2\end{matrix} = \begin{matrix}0.2 & 0.1\end{matrix}$$

หาค่าเวกเตอร์ตัวแทนของ student โดยการนำ one hot vector ของ student คูณกับค่าเมทริกซ์น้ำหนัก W1

$$\begin{matrix}0 & 0 & 1\end{matrix} \times \begin{matrix}0.1 & 0.3\\ 0.2 & 0.1\\ 0.3 & 0.2\end{matrix} = \begin{matrix}0.4 & 0.3\end{matrix}$$

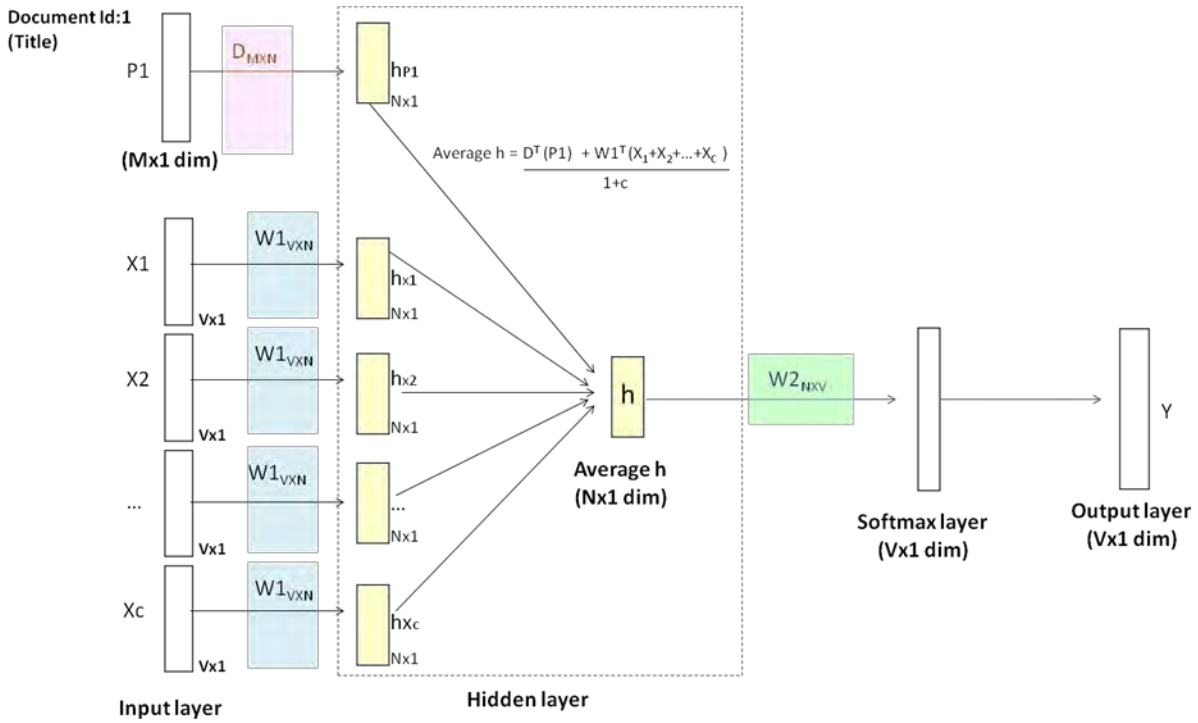
คำนวณผลคูณจุดระหว่าง เวกเตอร์ตัวแทนของ am และ เวกเตอร์ตัวแทนของ student

$$\begin{matrix}0.2 & 0.1\end{matrix} \bullet \begin{matrix}0.4 & 0.3\end{matrix} = 0.11$$

ดังนั้น จะได้ค่าความคล้ายระหว่างคำว่า am และ student เท่ากับ 0.11

3. การแทนข้อความด้วยเวกเตอร์ (Doc2Vec/Paragraph2Vec) [15] ถูกคิดค้นมาจากการต่อยอดของ Word2Vec แบ่งออกเป็น 2 รูปแบบ

3.1 Distributed Memory (PV-DM) มีการทำงานคล้ายกับ CBOW คือ ทำนายคำเป้าหมายจากคำบริบท แต่จะมีการสร้างเวกเตอร์เพิ่มขึ้นมาหนึ่งตัว เพื่อจดจำข้อมูลของคำทุกคำในเอกสารหรือย่อหน้านั้น ๆ ซึ่งมีการทำงานตามรูปที่ 2.9



รูปที่ 2.9 ภาพรวมโครงข่ายประสาทเทียมในการสร้างเวกเตอร์เอกสารแบบ Distributed Memory

โครงสร้างของโครงข่ายประสาทเทียมมี ดังนี้

- ชั้นข้อมูลขาเข้า (input layer) มี P1 และ X1, X2, ... , Xc เป็นบัพนำเข้า (node) โดย P1 เป็นค่า one hot vector ของหมายเลขหัวเรื่องซึ่งมีมิติเท่ากับจำนวนเอกสาร(M)และ X1,X2,...,Xc เป็นค่า one hot vector ของคำบริบทแต่ละคำซึ่งมีมิติเท่ากับจำนวนคำศัพท์ (V)
- ชั้นซ่อนตัว (hidden layer) กำหนดให้เวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นมีมิติ N โดยมีบัพซ่อนคือเวกเตอร์ตัวแทนของเอกสาร (h_{P1}) และเวกเตอร์ตัวแทนของคำ (h_x) เชื่อมจากบัพนำเข้าโดยผ่านเมทริกซ์น้ำหนักของเอกสาร (D_{MxN}) และเมทริกซ์น้ำหนักของคำชั้นซ่อนตัว ($W1_{VxN}$) ตามลำดับ ดังนี้

เวกเตอร์ตัวแทนของเอกสาร (h_{P1}) ได้จากการนำเวกเตอร์ชั้นข้อมูลขาเข้าของเอกสารคูณกับเมทริกซ์น้ำหนักของเอกสาร (D_{MxN}) และเวกเตอร์ตัวแทนของคำ (h_x) ได้จากการนำเวกเตอร์ชั้นข้อมูลขาเข้าของคำศัพท์คูณกับเมทริกซ์น้ำหนักของคำชั้นซ่อนตัว ($W1_{VxN}$) จากนั้นนำทุกบัพซ่อนมาหาค่าเฉลี่ย (Average h)

Average h = $\frac{1}{1+C} D^T (P1) + W1^T(X_1+X_2+...+X_C) = \frac{1}{1+C}(h_{P1}+h_{X1}+h_{X2}+...+h_{Xc})$

ซึ่งเมทริกซ์น้ำหนักของเอกสารและเมทริกซ์น้ำหนักของคำชั้นซ่อนตัวเป็น embedding layer

- ชั้นข้อมูลขาออก (output layer) มีมิติของเวกเตอร์เท่ากับจำนวนคำศัพท์ (V) โดยเวกเตอร์ชั้นซ่อนตัวจะคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออก ($W2_{NxV}$) ซึ่งค่าที่ถูกปล่อยจากชั้นนี้เปรียบเสมือนเวกเตอร์ในรูปแบบ One Hot Vector ของคำเป้าหมายที่เราสนใจ

หากมีเอกสารใหม่ระบบจะทำการเพิ่มหลักในเมทริกซ์น้ำหนักของเอกสารของชั้นซ่อนตัว และสุ่มค่าน้ำหนักใส่ในหลักที่เพิ่มขึ้นมานั้น เป็นเวกเตอร์ของเอกสารใหม่ จากนั้นจึงทำการอนุมานเวกเตอร์ฝั่งตัวของเอกสารนั้นๆ และปรับค่าเมทริกซ์น้ำหนักของคำบริบท

ขั้นตอนการฝึกสอนมีดังนี้

1. แปลงหมายเลขหัวเรื่องและคำบริบทให้อยู่ในรูปแบบ One Hot Vector
2. นำ One Hot Vector ของหมายเลขหัวเรื่องและคำบริบท มาคูณกับเมทริกซ์น้ำหนักของชั้นซ่อนตัวของแต่ละตัว จะได้เวกเตอร์แทนหัวเรื่องและคำบริบท
3. นำเวกเตอร์ตัวแทนของหัวเรื่องและคำบริบทมาหาค่าเฉลี่ยหรือนำมาต่อกัน
4. นำเมทริกซ์ที่ได้จากข้อ 3 มาคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออกจะได้เวกเตอร์คำเป้าหมาย
5. นำเวกเตอร์คำเป้าหมายมาปรับค่าโดยใช้ฟังก์ชัน softmax แล้วแปลงเป็น One Hot Vector

<u>ตัวอย่าง</u> หัวเรื่องที่ 1 “School” ประโยค "I am student" จำนวนคำศัพท์เท่ากับ 3 สมมติให้มีทั้งหมด 3 เอกสาร กรอบหน้าต่างเท่ากับ 1 และมิติของเวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นเท่ากับ 2 ดังรูปที่ 2.10

กำหนด ค่า One hot vector ของเอกสารและคำศัพท์ รวมถึง เมทริกซ์น้ำหนักของเอกสาร (D) และเมทริกซ์น้ำหนักของคำ(W1) ดังนี้

One hot vector ของคำศัพท์

One hot vector ของ I : 1 0 0

One hot vector ของ am : 0 1 0

One hot vector ของ student : 0 0 1

One hot vector ของเอกสาร

One hot vector ของ d1 : 1 0 0

One hot vector ของ d2 : 0 1 0

One hot vector ของ d3 : 0 0 1

เมทริกซ์น้ำหนักของเอกสาร (D)

$$\begin{matrix} 0.3 & 0.2 \\ 0.4 & 0.5 \\ 0.1 & 0.01 \end{matrix}$$

โดยที่ แถวที่ 1 คือ เวกเตอร์ตัวแทนของ d1

แถวที่ 2 คือ เวกเตอร์ตัวแทนของ d2

แถวที่ 3 คือ เวกเตอร์ตัวแทนของ d3

เมทริกซ์น้ำหนักของชั้นซ่อนตัว (W1)

$$\begin{matrix} 0.1 & 0.3 \\ 0.2 & 0.1 \\ 0.4 & 0.3 \end{matrix}$$

โดยที่ แถวที่ 1 คือ เวกเตอร์ตัวแทนของ I

แถวที่ 2 คือ เวกเตอร์ตัวแทนของ am

แถวที่ 3 คือ เวกเตอร์ตัวแทนของ student

ขั้นชั้นขาเข้า : เอกสาร d1 , คำ I

- แปลงหมายเลขหัวเรื่องเอกสารและคำเป็น One hot vector เป็นค่าบัพนำเข้า

d1 **1 0 0**

I **1 0 0**

ขั้นชั้นซ่อนตัว :

- นำค่า One hot vector ของ d1 คูณกับเมทริกซ์น้ำหนักขั้นซ่อนตัวของเอกสาร (D) จะได้เวกเตอร์ตัวแทนเอกสาร d1

$$\begin{matrix}1 & 0 & 0\end{matrix} \times \begin{matrix}0.3 & 0.2\\ 0.4 & 0.5\\ 0.1 & 0.01\end{matrix} = \begin{matrix}0.3 & 0.2\end{matrix}$$

- นำค่า One hot vector ของ I คูณกับเมทริกซ์น้ำหนักขั้นซ่อนตัวของคำ (W1) จะได้เวกเตอร์ตัวแทนของ ‘ I ’

$$\begin{matrix}1 & 0 & 0\end{matrix} \times \begin{matrix}0.1 & 0.3\\ 0.2 & 0.1\\ 0.3 & 0.2\end{matrix} = \begin{matrix}0.1 & 0.3\end{matrix}$$

- คำนวณค่าเฉลี่ยระหว่างค่าเวกเตอร์ตัวแทนของคำและเวกเตอร์ตัวแทนของเอกสาร จะได้ Average h = 0.2 0.25

ขั้นข้อมูลขาออก

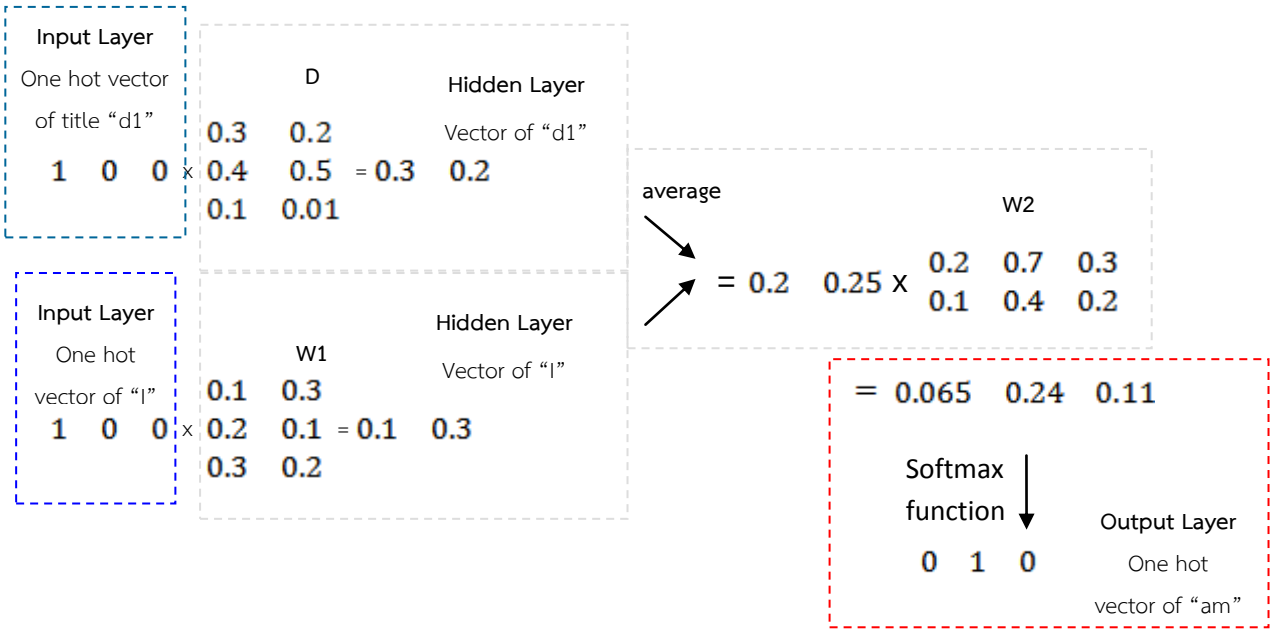
- ค่าเวกเตอร์ขาออกได้จากการนำ Average h คูณกับเมทริกซ์น้ำหนักขั้นขาออก (W2)

$$\begin{matrix}0.2 & 0.25\end{matrix} \times \begin{matrix}0.2 & 0.7 & 0.3\\ 0.1 & 0.4 & 0.2\end{matrix} = \begin{matrix}0.065 & 0.24 & 0.11\end{matrix}$$

- ใช้ softmax function ปรับค่าเวกเตอร์ขาออก ให้เป็น One hot vector จะได้ค่า one hot vector ของคำเป้าหมาย am

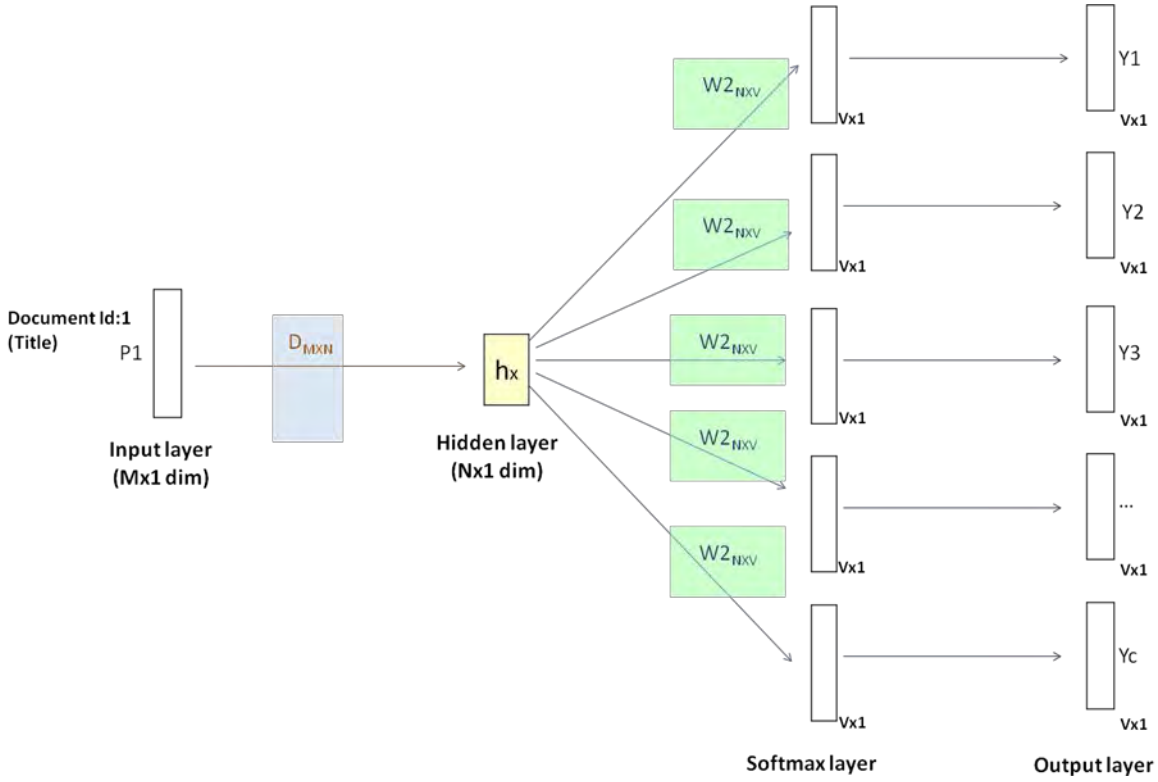
$$\begin{matrix}0.065 & 0.24 & 0.11\end{matrix} \xrightarrow{\text{Softmax function}} \begin{matrix}0 & 0 & 1\end{matrix}$$

ภาพรวมของตัวอย่างการคำนวณข้างต้น



รูปที่ 2.10 ตัวอย่างการคำนวณเวกเตอร์ตัวแทนแบบ Distributed Memory

3.2 Distributed Bag of Words (DBOW) มีการทำงานคล้ายกับ Skip gram แต่เปลี่ยนจากทำนายคำบริบทจากคำเป้าหมาย เป็นทำนายคำในเอกสารจากเอกสารนั้นๆ ซึ่งมีการทำงานตามรูปที่ 2.11



รูปที่ 2.11 ภาพรวมโครงข่ายประสาทเทียมในการสร้างเวกเตอร์เอกสารแบบ Distributed Bag of Words

โครงสร้างของโครงข่ายประสาทเทียมมี ดังนี้

- ชั้นข้อมูลขาเข้า (input layer) มี P1 เป็นบัพนำเข้า (node) โดย P1 เป็นค่า one hot vector ของหมายเลขหัวเรื่องซึ่งมีมิติเท่ากับจำนวนเอกสาร(M)
- ชั้นซ่อนตัว (hidden layer) กำหนดให้เวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นมีมิติ N โดยมีเวกเตอร์ตัวแทนของเอกสาร (h_x) เป็นบัพซ่อน (node) เชื่อมจากบัพชั้นข้อมูลขาเข้าโดยผ่านเมทริกซ์น้ำหนักของเอกสาร (D_{MxN}) ซึ่งเป็น embedding layer
- ชั้นข้อมูลขาออก (output layer) มีจำนวนบัพเท่ากับสองเท่าของจำนวนคำที่อยู่ในกรอบหน้าต่างในหัวเรื่องที่เราสนใจ ซึ่งแต่ละบัพเป็นเวกเตอร์ที่มีมิติเท่ากับจำนวนคำศัพท์ (V) โดยเวกเตอร์ชั้นซ่อนตัวจะคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออก (W_{2NxV}) ซึ่งค่าที่ถูกปล่อยจากชั้นนี้เปรียบเสมือนเวกเตอร์ในรูปแบบ One Hot Vector

ขั้นตอนการฝึกสอนมีดังนี้

1. แปลงหมายเลขหัวเรื่องให้อยู่ในรูปแบบ One Hot Vector
2. นำ One Hot Vector ของหมายเลขหัวเรื่อง มาคูณกับเมทริกซ์น้ำหนักของชั้นซ่อนตัว จะได้เวกเตอร์แทนหัวเรื่อง
3. นำเมทริกซ์ที่ได้จากข้อ 2 มาคูณกับเมทริกซ์น้ำหนักของชั้นข้อมูลขาออกจะได้เวกเตอร์คำที่อยู่ในหัวเรื่องที่เราสนใจ
4. นำเวกเตอร์คำมาปรับค่าโดยใช้ฟังก์ชัน softmax แล้วแปลงเป็น One Hot Vector

<u>ตัวอย่าง</u> หัวเรื่องที่ 1 “School” ประโยค "I am student" จำนวนคำศัพท์เท่ากับ 3 สมมติให้มีทั้งหมด 3 เอกสาร กรอบหน้าต่างเท่ากับ 1 และมิติของเวกเตอร์ตัวแทนที่ต้องการสร้างขึ้นเท่ากับ 2 ดังรูปที่ 2.12 และให้ค่าของเมทริกซ์น้ำหนักเอกสาร D , W1 เหมือนในตัวอย่างของ PV-DM

ขั้นชั้นขาเข้า : เอกสาร d1

- แปลงหมายเลขหัวเรื่องเอกสารค่าบัพนำเข้า

d1 1 0 0

ขั้นชั้นซ่อนตัว :

- นำค่า One hot vector ของ d1 คูณกับเมทริกซ์น้ำหนักขั้นซ่อนตัวของเอกสาร (D) จะได้เวกเตอร์ตัวแทนเอกสาร d1 เป็นค่าบัพซ่อนตัว

$$\begin{matrix}1 & 0 & 0\end{matrix} \times \begin{matrix}0.3 & 0.2\\ 0.4 & 0.5\\ 0.1 & 0.01\end{matrix} = \begin{matrix}0.3 & 0.2\end{matrix}$$

ขั้นข้อมูลขาออก

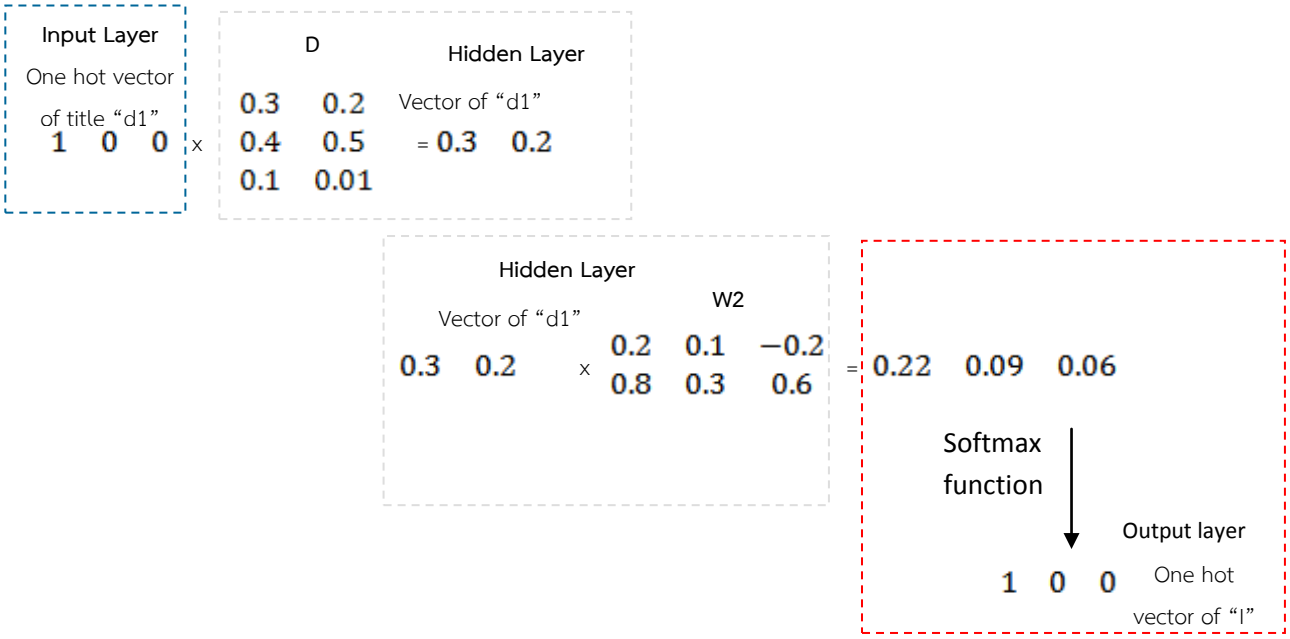
- ค่าเวกเตอร์ขาออกได้จากการนำ ค่าบัพซ่อนตัวคูณกับเมทริกซ์น้ำหนักขั้นขาออก (W2)

$$\begin{matrix}0.3 & 0.2\end{matrix} \times \begin{matrix}0.2 & 0.1 & -0.2\\ 0.8 & 0.3 & 0.6\end{matrix} = \begin{matrix}0.22 & 0.09 & 0.06\end{matrix}$$

- ใช้ softmax function ปรับค่าเวกเตอร์ขาออก ให้เป็น One hot vector จะได้ค่า one hot vector ของ I

$$\begin{matrix}0.22 & 0.09 & 0.06\end{matrix} \xrightarrow{\text{Softmax function}} \begin{matrix}1 & 0 & 0\end{matrix}$$

ภาพรวมของตัวอย่างการคำนวณข้างต้น



รูปที่ 2.12 ตัวอย่างการคำนวณเวกเตอร์ตัวแทนแบบ Distributed Bag of Words

ขั้นตอนการใช้งาน Doc2Vec มีดังนี้

ผลลัพธ์หลังจากการฝึกสอนจะได้เวกเตอร์ตัวแทนของเอกสารแต่ละเอกสาร จากแต่ละแถวของเมทริกซ์น้ำหนักเอกสาร (D) และเวกเตอร์ตัวแทนของแต่คำศัพท์ จากแต่ละแถวของเมทริกซ์น้ำหนักคำศัพท์ของคำชั้นซ่อนตัว (W1) รวมถึงจะได้เวกเตอร์ตัวแทนของแต่ละคำบริบทจากแต่ละหลักของเมทริกซ์น้ำหนักของชั้นข้อมูลขาออก (W2) มาใช้ในการหาเอกสารที่มีความใกล้เคียงกัน

1. แปลงหมายเลขหัวเรื่องที่สนใจให้อยู่ในรูปแบบ One Hot Vector
2. นำ One Hot Vector ของหมายเลขหัวเรื่องมาคูณกับเมทริกซ์น้ำหนักเอกสาร จะได้เวกเตอร์แทนหัวเรื่อง
3. หาค่าความคล้ายของเอกสารทั้งสอง โดยหาค่าผลคูณจุดระหว่างเวกเตอร์แทนเอกสาร 2 เอกสารที่ต้องการเปรียบเทียบ

<u>ตัวอย่าง</u> จำนวนเอกสารเท่ากับ 3 เอกสาร ได้แก่ d1 , d2 , d3 สมมติให้มิติของเวกเตอร์ตัวแทนเอกสารที่ต้องการสร้างขึ้นเท่ากับ 2 โดยกำหนดค่าเมทริกซ์น้ำหนักของเอกสาร และ One hot vector ดังนี้

One hot vector ของเอกสาร

One hot vector ของ d1 : 1 0 0

One hot vector ของ d2 : 0 1 0

One hot vector ของ d3 : 0 0 1

เมทริกซ์น้ำหนักของเอกสาร (D)

$$\begin{matrix} 0.3 & 0.2 \\ 0.4 & 0.5 \\ 0.1 & 0.01 \end{matrix}$$

โดยที่ แถวที่ 1 คือ เวกเตอร์ตัวแทนของ d1

แถวที่ 2 คือ เวกเตอร์ตัวแทนของ d2

แถวที่ 3 คือ เวกเตอร์ตัวแทนของ d3

คำนวณหาค่าความคล้ายระหว่างเอกสาร d1 และ d2 ได้จากผลคูณจุดของเวกเตอร์ตัวแทนเอกสาร d1 และ ผลคูณจุดเวกเตอร์ตัวแทนเอกสาร d2

หาเวกเตอร์ตัวแทนเอกสาร d1 ได้จากการนำ One hot vector ของเอกสาร d1 คูณกับเมทริกซ์น้ำหนักของเอกสาร (D)

$$\begin{matrix} 1 & 0 & 0 \end{matrix} \times \begin{matrix} 0.3 & 0.2 \\ 0.4 & 0.5 \\ 0.1 & 0.01 \end{matrix} = \begin{matrix} 0.3 & 0.2 \end{matrix}$$

หาเวกเตอร์ตัวแทนเอกสาร d2 ได้จากการนำ One hot vector ของเอกสาร d2 คูณกับเมทริกซ์น้ำหนักของเอกสาร (D)

$$\begin{matrix} 0 & 1 & 0 \end{matrix} \times \begin{matrix} 0.3 & 0.2 \\ 0.4 & 0.5 \\ 0.1 & 0.01 \end{matrix} = \begin{matrix} 0.4 & 0.5 \end{matrix}$$

คำนวณผลคูณจะระหว่างเวกเตอร์ตัวแทนเอกสาร d1 และ เวกเตอร์ตัวแทนเอกสาร d2

$$\begin{matrix} 0.3 & 0.2 \end{matrix} \bullet \begin{matrix} 0.4 & 0.5 \end{matrix} = 0.22$$

ดังนั้น ค่าความคล้ายระหว่างเอกสาร d1 และเอกสาร d2 เท่ากับ 0.22

4. Gensim [7] เป็นคลังโปรแกรมที่ออกแบบมาเพื่อประมวลผลข้อความที่ไม่มีโครงสร้าง ซึ่งประกอบด้วยอัลกิริทึมต่าง ๆ ที่ใช้กับการจัดการข้อความ เช่น การหา tf-idf การสร้างเวกเตอร์ของคำ นอกจากนี้สามารถนำมาใช้ประโยชน์ในการสร้างระบบต่าง ๆ เช่น ระบบแนะนำ ระบบคำถามคำตอบ

2.3 การตัดคำ

1. Deepcut [8] เป็นคลังโปรแกรมที่ใช้อัลกอริทึม ข่ายงานประสาทเทียมเชิงลึก (Deep Neural Network) มีการตัดคำโดยใช้การกรองแบบคอนโวลูชัน (Convolutional filters) และใช้การจำแนกคำ (binary classification) ว่าเป็นจุดเริ่มต้นของคำหรือไม่ ข้อดี คือ สามารถฝึกสอนให้คลังโปรแกรมตัดคำตามความต้องการได้ ข้อเสีย คือ ใช้เวลาในการฝึกสอนเป็นเวลานาน

2. Tltk [9] เป็นคลังโปรแกรมที่ใช้ในการหาขอบเขตของคำโดยการตัดคำให้ได้จำนวนคำน้อยที่สุด (maximum matching) ในคลังข้อมูลพจนานุกรม ซึ่งใช้อัลกอริทึม 3-แกรม ของคลังข้อมูลที่ตัดพยางค์แล้วช่วยเลือกการตัดพยางค์ที่ดีสุดก่อนจะนำไปรวบพยางค์เป็นคำภายหลัง และมีการเก็บคำศัพท์ในคลังข้อมูล 630,000 คำ จากหนังสือพิมพ์ แต่ไม่มีการเก็บคำที่เป็นคำเฉพาะ ข้อดี คือ สามารถเพิ่มคำที่ไม่มีในคลังข้อมูลพจนานุกรมได้

3. PyThaiNLP [10] เป็นคลังโปรแกรมที่ใช้ในการหาขอบเขตของคำโดยค้นหาคำในคลังข้อมูลพจนานุกรม มีการใช้วิธีการประมาณค่าความน่าจะเป็นของสายคำที่เกิดขึ้นร่วมกันว่ามีค่าเท่ากับผลคูณของความน่าจะเป็นที่จะพบคำทีละ 1 ตัว ติดกันในสายคำนั้น (unigram) ในการกำกับหมวดคำ นอกจากนี้คลังโปรแกรมยังมีการรวบรวมคำไม่สำคัญ เพื่อใช้ในการลบคำไม่สำคัญทิ้ง

4. Stop word คือ คำไม่สำคัญ เช่น เป็น หรือ ของ ได้ บาง ไว้ โดย และ แบบ เอง จะ จึง เป็นต้น ในบทความมักจะมีการใช้คำไม่สำคัญบ่อยครั้ง เนื่องจากเป็นคำที่ใช้ประกอบและเชื่อมประโยคให้สละสลวยมากยิ่งขึ้น ซึ่งการลบคำที่ไม่สำคัญออกจะทำให้ระบบสามารถประมวลผลข้อความได้เร็วขึ้น

2.4 นิพจน์ปกติ (Regular Expressions)

Regular Expressions หรือ regex คือ การกำหนดรูปแบบหรือกลุ่มคำ เพื่อเอาไว้ใช้ค้นหาอักษรหรือข้อความต่าง ๆ ตามที่ต้องการว่าตรงตามเงื่อนไข (pattern) ที่เรากำหนดไว้หรือไม่ สามารถค้นหาได้ทั้งอักขระธรรมดา และสัญลักษณ์ สามารถใช้ในการค้นหาข้อความ การแทนที่ข้อความ การตรวจสอบรูปแบบข้อความ เป็นต้น Regular Expressions มีอยู่เกือบทุกภาษาโปรแกรม แต่อาจจะแตกต่างกันออกไปเล็กน้อย ซึ่งโครงงานนี้ใช้ Regular Expressions ในการจัดการข้อมูล เนื่องจากข้อมูลเอกสารของวิกิพีเดียใช้ป้ายกำกับระบุ

หัวเรื่องและเนื้อหาอยู่ จึงคัดแยกส่วนชื่อหัวเรื่องและเนื้อหาออกมาจากข้อมูลดั้งเดิม เพื่อนำไปใช้ในการพัฒนาโปรแกรม

2.5 ระยะทางเลเวนชเตย์น (Levenshtein ratio and distance)

[11] เป็นขั้นตอนวิธีการวัดหาค่าความต่างกันของสายอักขระสองชุด ระหว่างชุดแรกที่เป็นต้นแบบ และ ชุดที่สองที่เป็นชุดเปรียบเทียบ โดยค่าความต่างกันจะวัดจากจำนวนของการที่จะต้องทำการตัดออก แทรก และแทนที่ ของอักขระในชุดเปรียบเทียบจนกระทั่งมีลักษณะเหมือนชุดอักขระที่เป็นต้นแบบทุกประการโดยมีรายละเอียด ดังนี้

1.การแทรก เป็นการนำเอาอักขระตัวใด ๆ มา เพื่อให้ชุดอักขระชุดนั้นเหมือนกับอีกชุดอักขระหนึ่งในภายหลัง เช่น run ⟶ ruin จะแทรกตัว i ให้กับ run เพื่อให้ run กลายเป็น ruin เป็นต้น

2.การตัดออก เป็นการตัดอักขระออกครั้งละ 1 ตัว จากชุดอักขระตัวหนึ่ง เพื่อให้ชุดอักขระชุดนั้นเหมือนกับอีกชุดอักขระหนึ่งในภายหลัง เช่น dog ⟶ do จะตัดตัว g ออก เพื่อให้ dog กลายเป็น do เป็นต้น

3.การแทนที่ เป็นการนำอักขระของชุดอักขระหนึ่งไปแทนอักขระของอีกชุดอักขระหนึ่ง เพื่อให้ชุดอักขระชุดนั้นเหมือนกับอีกชุดอักขระหนึ่งในภายหลัง เช่น cat ⟶ rat จะแทนที่ r ด้วย c เพื่อให้ cat กลายเป็น rat หรืออาจมองในทางกลับกันก็ได้ เป็นต้น

ขั้นตอนวิธีการวัด ใช้กำหนดการพลวัต (dynamic programming) ในการแก้ปัญหา

<u>ตัวอย่าง</u>

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	4	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

รูปที่ 2.13 ตัวอย่างการคำนวณระยะทางเลเวนชเตย์น

นำมาจาก https://en.wikipedia.org/wiki/Levenshtein_distance

จากรูปที่ 2.13 จะเห็นว่าคำว่า Sunday เป็นชุดต้นแบบ และ Saturday เป็นชุดเปรียบเทียบ มีค่าความต่างกันอยู่คือ 3 โดยได้มาจาก

- การตัด a และ t ออก ดำเนินการ 2 ครั้ง
- การแทนที่ r ด้วย n ดำเนินการ 1 ครั้ง

บทที่ 3
ขั้นตอนการดำเนินงาน

ในบทนี้จะกล่าวถึงวิธีการค้นคืนเอกสารและการสกัดคำตอบจากคลังข้อมูลวิกิพีเดียภาษาไทย เพื่อใช้ในการคัดเลือกคำตอบ โดยอาศัยองค์ความรู้ต่างๆในการคัดเลือกคำตอบที่เหมาะสมที่สุด โดยจะอธิบายถึงรายละเอียดของชุดข้อมูลที่ใช้ในการฝึกสอนระบบ แนวคิดและขั้นตอนการออกแบบตัวแบบสำหรับการคัดเลือกหัวเรื่องและคำตอบ

3.1 เครื่องมือที่ใช้ในการวิจัย

การฝึกสอนตัวแบบในโครงงานนี้ใช้ภาษาไพทอน (Python) เป็นหลักในการเขียนโปรแกรมของระบบทั้งหมด โดยใช้ชุดข้อมูลของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย (National Software Contest : NSC) นำมาจาก http://copycatch.in.th/thai-qa-task.html ในการฝึกสอนตัวแบบ

3.2 รายละเอียดชุดข้อมูลสำหรับการฝึกสอน

งานวิจัยฉบับนี้ใช้ชุดข้อมูลของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย สำหรับการฝึกสอนโปรแกรมตอบคำถาม ชุดข้อมูลดังกล่าวแบ่งเป็น 2 ส่วน คือ ชุดข้อมูลเอกสารวิกิพีเดียภาษาไทยซึ่งในเอกสารประกอบด้วยเนื้อหาหรือบทความเขียนต่อกันเป็นย่อหน้าเดียว ในเนื้อหามีทั้งคำภาษาไทยและภาษาอังกฤษ ดังรูปที่ 3.1 และชุดข้อมูลตัวอย่างคำถามคำตอบดังรูปที่ 3.2 บางคำถามมีทั้งคำภาษาไทยและภาษาอังกฤษ และบางคำตอบต้องตอบเป็นคำภาษาอังกฤษ งานวิจัยฉบับนี้จะวิเคราะห์คำตอบที่เป็นภาษาไทยและตัวเลขเท่านั้น เนื่องด้วยข้อจำกัดในด้านเวลา การฝึกสอนและทดสอบจะสุ่มคัดเลือกบทความในวิกิพีเดียออกมาเพียงจำนวนหนึ่งเท่านั้นสำหรับการฝึกสอน และมีรายละเอียดชุดข้อมูล ดังตารางที่ 3.1

จำนวนเอกสารทั้งหมด	125,000 เอกสาร
จำนวนเอกสารที่ใช้ฝึกสอน (ชุดที่ 1)	100 เอกสาร
จำนวนเอกสารที่ใช้ฝึกสอน (ชุดที่ 2)	5000 เอกสาร
จำนวนคำถามคำตอบ	100 ชุด
จำนวนคำเฉลี่ยที่ปรากฏในเอกสารเฉพาะภาษาไทยและตัวเลข	735 คำ

ตารางที่ 3.1 รายละเอียดชุดข้อมูลสำหรับการฝึกสอน

<doc id="473065" url="https://th.wikipedia.org/wiki?curid=473065" title="เอเลี่ยน 2 ฝูงมฤตยูนอกโลก">เอเลี่ยน 2 ฝูงมฤตยูนอกโลก เอเลียนส์ เป็นภาพยนตร์นิยายวิทยาศาสตร์แอคชั่นในปี ค.ศ. 1986 กำกับการแสดงโดย เจมส์ แคเมรอน และนำแสดงโดย Sigourney Weaver ,Carrie Henn ,Michael Biehn ,Lance Henriksen ,William Hope และPaul Reiser เป็นภาคต่อจาก ภาพยนตร์ปี ค.ศ. 1979 เอเลียน สร้างโดยทเวนตีท์เซนจูรีฟอกซ์</doc>

รูปที่ 3.1 ตัวอย่างชุดข้อมูลเอกสาร

"question_id": 1,
"question": "เอเลียนส์เป็นภาพยนตร์แนวไหน",
"answer": "นิยายวิทยาศาสตร์แอคชั่น",
"answer_begin_position ": 149,
"answer_end_position": 172,
"article_id": 473065

รูปที่ 3.2 ตัวอย่างชุดข้อมูลคำถามคำตอบ

3.3 การทำความสะอาดข้อมูล (Data Cleansing)

เนื่องจากข้อมูลของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทยอยู่ในรูปแบบ

<doc id="article_id" url="link" title = ""> ...(content)....</doc>

ซึ่งไม่สะดวกต่อการใช้งาน จึงใช้การกำหนดรูปแบบกลุ่มคำ ในการจัดการข้อมูลของแต่ละเอกสารให้อยู่ในรูปแบบของ Dictionary คือ

{ 'article_id' : 135329, 'article' : ชิงชัน, 'content' : ชิงชัน เป็นชื่อของไม้ยืนต้นขนาดกลางจนถึงขนาดใหญ่ประเภทไม้ผลัดใบชนิดหนึ่ง, 'words_list' : 'ชิงชัน', 'เป็น', 'ชื่อ', 'ของ', 'ไม้', 'ยืนต้น', 'ขนาด', 'กลาง', 'จน', 'ถึง', 'ขนาด', 'ใหญ่', 'ประเภท', 'ไม้', 'ผลัด', 'ใบ', 'ชนิด', 'หนึ่ง'

ซึ่งกำหนดให้ article_id, article, content, words_list เป็นกุญแจ (key) ทำให้สามารถเข้าถึงข้อมูลได้รวดเร็ว เพราะว่า ข้อมูลได้ถูกทำดัชนีไว้อัตโนมัติโดยใช้ Key นอกจากนี้ Dictionary ยังมีเมธอดและฟังก์ชันอำนวยความสะดวกสำหรับการทำงานทั่วไป โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย จะสนใจเนื้อหาเฉพาะภาษาไทยและตัวเลข จึงแทนที่ตัวอักษรอื่น ๆ ด้วยช่องว่าง ดังรูปที่ 3.3 และมีการตัดคำโดยใช้คำสั่ง “deepcut” และจัดเก็บลงในค่า keys 'words_list' นอกจากนี้ขั้นตอนในการตัดเฉพาะข้อมูลรูปแบบคำที่ต้องการมาเป็น values ในแต่ละค่า Keys ของ Dictionay ที่จัดเก็บ ใช้นิพจน์ปกติในการเลือกจากรูปแบบ เช่น รูปแบบการเลือก article_id จะใช้รูปแบบ

“id[^\’]+\’([^\’]+)\’” , “title[^\’]+\’([^\’]+)\’”

และการเลือกบทความ ใช้รูปแบบ

<doc [^>]+ >(.+)</doc>

ซึ่งใช้คลังโปรแกรม re ช่วยในการจัดการ

เอเลี่ยน 2 ฝูงมฤตยูนอกโลก เอเลียนส์ 2เป็นภาพยนตร์นิยายวิทยาศาสตร์แอคชั่นในปี ค ศ 1986 กำกับการแสดงโดย เจมส์ แคเมรอน และนำแสดงโดย

และ เป็นภาคต่อจากภาพยนตร์ปี ค ศ 1979 เอเลียน สร้างโดยทเวนตีท์ เซนจูรีฟอกซ์

รูปที่ 3.3 ตัวอย่างชุดข้อมูลเอกสารหลังจากทำความสะอาดข้อมูล

3.4 ภาพรวมระบบ

โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย จะแบ่งระบบออกเป็น 2 ส่วน ดังนี้

3.4.1 การค้นคืนเอกสาร

การค้นคืนเอกสาร เป็นขั้นตอนที่ใช้ในการคัดเลือกหัวเรื่อง ซึ่งจะแบ่งออกเป็น 2 ส่วน คือ การฝึกสอนระบบ และการทดสอบระบบ

1. การฝึกสอนระบบ

ผู้จัดทำจะฝึกสอนการแทนคำและข้อความด้วยเวกเตอร์ ผ่านการใช้เจนซิมซึ่งเป็นคลังโปรแกรม ที่ใช้ในการฝึกสอนการแทนคำและข้อความด้วยเวกเตอร์ โดยแบ่งเป็น **2** วิธี

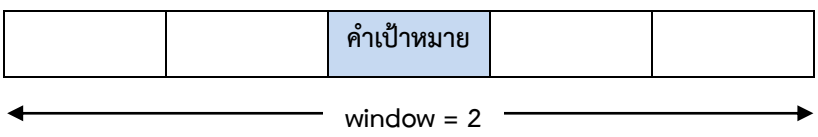
วิธีที่ 1 Doc2vec

ข้อมูลที่ใช้ฝึกสอนระบบประกอบด้วยชุดคำถามคำตอบ 100 ข้อ และเอกสาร n เอกสาร ซึ่งรวมถึงเอกสารที่เป็นคำตอบด้วย มีขั้นตอนการทำงานดังนี้

1. ตัดคำในคำถามและคำในเอกสารโดยใช้เครื่องมือการตัดคำ “Deepcut”

2. เก็บคำในเอกสารในรูปแบบป้ายชื่อเอกสารดังรูปที่ 3.5 เนื่องจากสามารถเรียกใช้งานของแต่ละเอกสารได้สะดวก ต่อมาเปลี่ยนคำในเอกสารเป็นเวกเตอร์โดยใช้โมดูล doc2vec จากคลังโปรแกรมเจนซิม ซึ่งมีพารามิเตอร์ ดังนี้

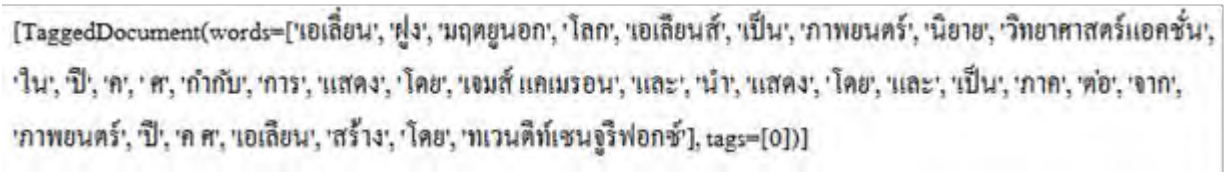
- ขนาดของเวกเตอร์แทนข้อความ (vectot size) เป็น 200
- ระยะทางสูงสุดระหว่างคำเป้าหมายและคำบริบทที่ถูกทำนายภายในประโยค (window) เป็น 2 ดังรูปที่ 3.4 จากงานวิจัยของ Levy และ Goldberg แสดงให้เห็นว่าระยะทางสูงสุดระหว่างคำเป้าหมายและคำบริบทเป็น 2 สร้างเวกเตอร์ได้ดีกว่าระยะทางสูงสุดระหว่างคำเป้าหมายและคำบริบทเป็น 5



รูปที่ 3.4 ระยะทางสูงสุดระหว่างคำเป้าหมายและคำบริบทที่ถูกทำนายภายในประโยค

- อัตราการเรียนรู้ (learning rate) เป็น 0.025 จากการทดลองสุ่มค่าอัตราการเรียนรู้ ปรากฏว่าอัตราการเรียนรู้เป็น 0.025 ให้การค้นคืนหัวเรื่องได้ดีที่สุด

- ความถี่ที่ต่ำที่สุดของคำที่จะนำมาคิดคำนวณ (min_count) เป็น 1 ซึ่งหมายถึงใช้ทุกคำโดยไม่ตัดคำใดทิ้ง เนื่องจากมีคำที่สำคัญที่คาดว่ามีผลในการเลือกหัวเรื่อง ที่มีความถี่คำเป็น 1



รูปที่ 3.5 ตัวอย่างการเก็บข้อมูลป้ายชื่อเอกสาร (Tagged document)

วิธีที่ 2 Word2vec

ข้อมูลที่ใช้ฝึกสอนระบบประกอบด้วยชุดคำถามคำตอบ 100 ข้อ และหัวเรื่อง n หัวเรื่อง ซึ่งรวมถึงหัวเรื่องที่เป็นคำตอบด้วย มีขั้นตอนการทำงานดังนี้

1. ตัดคำในคำถามโดยใช้เครื่องมือการตัดคำ “Deepcut”

2. เปลี่ยนคำถามและหัวเรื่องเป็นเวกเตอร์โดยใช้คลังโปรแกรม word2vec จากคลังโปรแกรมเจนซิม

2. การทดสอบระบบ

การทดสอบระบบจะแบ่งเป็น 3 วิธี ดังนี้

วิธีที่ 1 Doc2vec

หลังจากฝึกสอนชุดคำถามและเอกสารด้วยโมดูล doc2vec แล้ว ต่อมานำมาหาค่าความเหมือนโดยใช้คำสั่ง model.docvecs.most_similar() ระหว่างเวกเตอร์ของคำถามกับเวกเตอร์ของเอกสารที่เหมือนกันมากสุด (most similarity)

วิธีที่ 2 Word2vec

หลังจากฝึกสอนชุดคำถามและหัวเรื่องด้วยคลังโปรแกรม word2vec แล้ว ต่อมานำมาทดสอบโดยมีขั้นตอนการทำงานดังนี้

1. หาค่าความคล้ายโคไซน์ (cosine similarity) ระหว่างเวกเตอร์หัวเรื่องและเวกเตอร์คำทุกคำในคำถาม โดยทำการหาผลคูณจุดของเวกเตอร์ของคำทั้งสอง แล้วนำค่าความคล้ายโคไซน์มาบวกกัน

2. หาหัวเรื่องที่มีค่าความคล้ายโคไซน์ที่มากที่สุด

วิธีที่ 3 String Matching ข้อมูลที่ใช้ทดสอบระบบประกอบด้วยชุดคำถามคำตอบ 100 ข้อ และหัวเรื่อง n หัวเรื่อง ซึ่งรวมถึงหัวเรื่องที่เป็นคำตอบด้วย โดยจะวัดค่าความต่างกันของสายอักขระสองชุดตามทฤษฎีระยะทางเลเวนชเตย์นที่กล่าวในบทที่ 2 ซึ่งเมทริกซ์จะมีขนาด (m+1)x(n+1) โดยที่คอลัมน์ (m) เป็นคำถาม (ชุดต้นแบบ) และแถว (n) เป็นหัวเรื่อง (ชุดเปรียบเทียบ) ถ้าค่าความต่างน้อย หมายความว่า สายอักขระสองชุดมีความเหมือนกันมาก ดังนั้นเลือกหัวเรื่องที่มีค่าความต่างกับคำถามน้อยที่สุด

3.4.2 การสกัดคำตอบ

การสกัดคำตอบจะแบ่งเป็น 2 ส่วน ดังนี้

1. **การหาคำที่น่าจะเป็นคำตอบ** มีขั้นตอนดังนี้

1. ตัดคำในคำถามและคำในเอกสารโดยใช้เครื่องมือการตัดคำ “Deepcut”

2. หารายการของคำในเอกสารที่มีอยู่ในคำถาม

3. หาคำบริบทของคำในเอกสารที่ตรงกับคำถามโดยกำหนดคำข้างหน้า k คำและคำข้างหลัง k คำ และเก็บเฉพาะตำแหน่งของคำที่เป็นจำนวนเต็มบวก

4. เก็บเฉพาะคำที่มีตำแหน่งของคำในเอกสารที่น้อยกว่าระยะห่างจากคำเป้าหมายที่กำหนดไว้ เนื่องจากคาดว่าตำแหน่งของคำในเอกสารที่มากกว่าค่ากำหนดซึ่งเป็นค่าที่โดดออกไปไม่น่าจะเป็นคำตอบและเพื่อลดเวลาในการคำนวณคะแนน

$$\text{ระยะห่างจากคำเป้าหมาย} = \frac{\text{ผลรวมของตำแหน่งคำบริบทที่ได้จากข้อ 3}}{\text{จำนวนคำบริบทที่ได้จากข้อ 3}} + \text{ส่วนเบี่ยงเบนมาตรฐาน}$$

2. **การให้คะแนน**

การให้คะแนนจะแบ่งเป็น 2 วิธีหลัก ๆ ดังนี้

วิธีที่ 1 การคำนวณคะแนนจากตำแหน่งของคำ โดยต้องการตำแหน่งที่มีคำตรงกับคำถามมาใช้ในการหาคำบริบทที่คาดว่าน่าจะเป็นคำตอบ

หลังจากหาคำที่น่าจะเป็นคำตอบแล้ว นำตำแหน่งของคำในเอกสารที่ได้จากการหาคำที่น่าจะเป็นคำตอบ มาลบกับ ตำแหน่งของแต่ละคำในเอกสารที่ตรงกับคำถาม แล้วหาคำที่มีผลรวมค่าคะแนนของคำในเอกสารที่น้อยที่สุด

ตัวอย่าง

บทความ: หม่อมราชวงศ์กุลปราโมทย์ จิรประวัติ (ราชสกุลเดิม: สวัสดิกุล; เกิด: 2 มิถุนายน พ.ศ. 2458 — ตาย: 30 กรกฎาคม พ.ศ. 2528) เป็นหม่อมในพระเจ้าวรวงศ์เธอ พระองค์เจ้าเฉลิมพลฑิฆัมพร พระโอรสในสมเด็จพระเจ้าบรมวงศ์เธอ เจ้าฟ้ายุคลฑิฆัมพร กรมหลวงลพบุรีราเมศร์กับพระเจ้าวรวงศ์เธอ พระองค์เจ้าเฉลิมเขตรมงคล และต่อมาเป็นหม่อมในหม่อมเจ้าขจรจิรพันธ์ จิรประวัติ พระโอรสในพระเจ้าบรมวงศ์เธอ พระองค์เจ้าจิรประวัติวรเดช กรมหลวงนครไชยศรีสุรเดชกับหม่อมเจ้าสุมนมาลย์ จิรประวัติประวัติ ประวัติ หม่อมราชวงศ์กุลปราโมทย์ สวัสดิกุล เป็นธิดาของหม่อมเจ้าวิบูลย์สวัสดิ์วงศ์ สวัสดิกุล หม่อมเกื้อ สวัสดิกุล ณ อยุธยา (สกุลเดิม:ณ ระนอง) เกิดเมื่อวันที่ 2 มิถุนายน พ.ศ. 2458 ต่อมาเสกสมรสเป็นหม่อมคนแรกของพระเจ้าวรวงศ์เธอ พระองค์เจ้าเฉลิมพลฑิฆัมพรเมื่อ พ.ศ. 2475 ได้ประสูติพระธิดา 1 พระองค์ ต่อมาทรงหย่ากับพระองค์เจ้าเฉลิมพลฑิฆัมพรและเสกสมรสใหม่กับหม่อมเจ้าขจรจิรพันธ์ จิรประวัติเมื่อ พ.ศ. 2482 และพำนักที่วังหม่อมเจ้าขจรจิรพันธ์ตราบจนถึงแก่กรรม หม่อมราชวงศ์กุลปราโมทย์ สวัสดิกุล ถึงแก่กรรมเมื่อวันที่ 30 กรกฎาคม พ.ศ. 2528 สิริอายุ 70 ปีสมรส สมรส หม่อมราชวงศ์กุลปราโมทย์ สวัสดิกุล สมรสครั้งแรกกับพระเจ้าวรวงศ์เธอ พระองค์เจ้าเฉลิมพลฑิฆัมพร พระโอรสในสมเด็จพระเจ้าบรมวงศ์เธอ เจ้าฟ้ายุคลฑิฆัมพร กรมหลวงลพบุรีราเมศร์กับพระเจ้าวรวงศ์เธอ พระองค์เจ้าเฉลิมเขตรมงคล เมื่อ พ.ศ. 2475 มีพระธิดา 1 พระองค์ คือ- หม่อมเจ้าจามเทพี ยุคล (23 ธันวาคม พ.ศ. 2476 - 26 มกราคม พ.ศ. 2485) ต่อมาสมรสใหม่กับหม่อมเจ้าขจรจิรพันธ์ จิรประวัติพระโอรสในพระเจ้าบรมวงศ์เธอ พระองค์เจ้าจิรประวัติวรเดช กรมหลวงนครไชยศรีสุรเดชกับหม่อมเจ้าสุมนมาลย์ จิรประวัติ พ.ศ. 2482 มีธิดา 1 คน- หม่อมราชวงศ์หญิงสุรภี จิรประวัติ สมรสกับ หม่อมราชวงศ์วุฒิสวัสดิ์ สวัสดิวัตน์- หม่อมหลวงหญิงเสาวรส สวัสดิวัตน์ - หม่อมหลวงหญิงกุลยา สวัสดิวัตน์สาแหรก

คำถาม: หม่อมราชวงศ์กุลปราโมทย์ จิรประวัติ ราชสกุลเดิมคืออะไร

ตำแหน่งคำในเอกสารที่ตรงกับคำถาม: 'หม่อมราชวงศ์': [0, 2, 35, 74, 88], 'กุลปราโมทย์ จิรประวัติ': [1, 3] 'ราชสกุล': [4], 'เดิม': [5, 42]

คำบริบทของคำที่ตรงกับคำถามและมีตำแหน่งน้อยกว่าค่ากำหนด: 'สวัสดิกุล': [6], '2': [7, 44], 'ประวัติ': [33, 34], 'กุลปราโมทย์ สวัสดิกุล': [36], 'ธิดา': [37], 'สวัสดิกุล ณ อยุธยา': [40], 'สกุล': [41], 'ระนอง': [43]

ค่าคะแนนของคำที่น่าจะเป็นคำตอบ: (6, 86), (7, 90), (33, 194), (34, 198), (36, 208), (37, 214), (40, 232), (41, 238), (43, 252), (44, 260)

สมมติคำว่า 'สวัสดิกุล' อยู่ตำแหน่งที่ 6

ค่าคะแนนจะเป็น (6-0) +(6-1)+(6-2)+(6-3)+ (6-4)+(6-5)+ (35-6)+(6-42) = 86 ซึ่งเป็นคำที่มีผลรวมค่าคะแนนที่น้อยที่สุด

คำตอบ: สวัสดิกุล

วิธีที่ 2 Word2vec

1. แทนคำถามและคำในเอกสารด้วยเวกเตอร์ โดยใช้คลังโปรแกรม word2vec จากคลังโปรแกรมเจนซิม

2. หาค่าความคล้ายโคไซน์ (cosine similarity) ระหว่างเวกเตอร์คำในเอกสารที่ได้จากข้อ 5 และเวกเตอร์คำทุกคำในคำถาม โดยทำการหาผลคูณจุดของเวกเตอร์ของคำทั้งสอง แล้วนำค่าความคล้ายโคไซน์มาบวกกัน

3. หาคำในเอกสารที่มีค่าความคล้ายโคไซน์ที่มากที่สุด

บทที่ 4
ผลการวิจัย

ในบทนี้จะกล่าวถึงผลของวิธีการค้นคืนเอกสารและการสกัดคำตอบที่นำเสนอ และผลจากการรวมส่วนค้นคืนเอกสารและการสกัดคำตอบเข้าด้วยกัน โดยการค้นคืนเอกสารจะวัดผลว่าสามารถค้นคืนเอกสารที่ถูกต้องมากน้อยเท่าไหร่ และการสกัดคำตอบจะวัดผลโดยดูจากการค้นคืนคำตอบที่ถูกเป็นอันดับแรก

4.1 ผลการฝึกสอนเพื่อการค้นคืนเอกสาร

การทดสอบจะแบ่งชุดข้อมูลบทความออกเป็น 2 ชุด คือ ชุดที่ 1 หรือ D_{100} จะเป็นบทความ 100 บทความ โดยใน 1 บทความจะมีคำตอบของคำถาม 1 คำถาม และ ชุดที่ 2 หรือ D_{5000} มี 5,000 บทความโดยมีบทความทุกชิ้นใน D_{100} และบทความอื่น ๆ ซึ่งไม่มีคำตอบ เพื่อดูว่าหากมีบทความมากขึ้น จะส่งผลกระทบอย่างไรต่อประสิทธิภาพด้านการค้นคืน ดังตารางที่ 4.1

จำนวนชุดบทความ	**อันดับที่ค้นหาหัวเรื่องถูก**	Doc2vec Distributed Memory	Doc2vec Distributed Bag of Words	Word2vec Continuous Bag of Words	Word2vec Skip Gram	String Matching
D_{100}	**อันดับแรก**	5 %	60 %	18 %	16 %	**72 %**
	10 อันดับแรก	17 %	90 %	29 %	33 %	-
	20 อันดับแรก	31 %	95 %	43 %	45 %	-
D_{5000}	**อันดับแรก**	4 %	25 %	11 %	11 %	**62 %**
	10 อันดับแรก	9 %	56 %	16 %	16 %	-

จำนวนชุดบทความ	อันดับที่ค้นหาหัวเรื่องถูก	Doc2vec Distributed Memory	Doc2vec Distributed Bag of Words	Word2vec Continuous Bag of Words	Word2vec Skip Gram	String Matching
D_{5000}	20 อันดับแรก	10 %	69 %	17 %	16 %	-
D_{125000}	อันดับแรก	-	-	-	-	**43 %**
	2 อันดับแรก	-	-	-	-	50 %
	3 อันดับแรก	-	-	-	-	51 %
	4 อันดับแรก	-	-	-	-	54 %
	5 อันดับแรก	-	-	-	-	59 %

ตารางที่ 4.1 ผลของการค้นคืนเอกสาร

วิธีที่ 1 Doc2vec แบ่งออกเป็น 2 รูปแบบ คือ Distributed Memory และ Distributed Bag of Words

จากตารางที่ 4.1 การวัดผลของ Doc2vec ทั้ง 2 รูปแบบ แสดงให้เห็นว่า แบบ Distributed Bag of Words มีประสิทธิภาพด้านการค้นคืนเอกสารได้มากกว่า Distributed Memory แต่จากการประมวลผลบน Google Colab RAM 12.72 GB CPU @ 2.30GHz แสดงให้เห็นว่า Distributed Bag of Words มีการประมวลผลเร็วกว่าซึ่งทำงานบน Google Colab ดังตารางที่ 4.2

	เวลาที่ใช้ในการประมวลผล
Distributed Memory	50 นาที
Distributed Bag of Words	45 นาที

ตารางที่ 4.2 เวลาที่ใช้ในการประมวลผลของ Doc2vec บน D_{5000}

วิธีที่ 2 Word2vec แบ่งออกเป็น 2 รูปแบบ คือ Continuous Bag of Words และ Skip Gram

จากตารางที่ 4.1 การวัดผลของ Word2vec ทั้ง 2 รูปแบบ แสดงให้เห็นว่า แบบ Continuous Bag of Words มีประสิทธิภาพด้านการค้นคืนเอกสารได้ใกล้เคียงกับ Skip Gram

วิธีที่ 3 การจับคู่สายอักขระ (String Matching) เนื่องจากวิธีการจับคู่สายอักขระ เป็นวิธีที่สามารถค้นหัวเรื่องได้มากที่สุด จึงมีการนำชุดข้อมูลเอกสารทั้งหมดจำนวน 125,000 เอกสาร มาเป็นชุดทดสอบชุดที่ 3 หรือ D_{125000} ร่วมด้วย

จากตารางที่ 4.1 การวัดผลของการจับคู่สายอักขระ แสดงให้เห็นว่า การเพิ่มจำนวนของชุดข้อมูลเอกสาร ส่งผลให้ประสิทธิภาพด้านการค้นคืนเอกสารลดลงประมาณ 30% ซึ่งถือว่าประสิทธิภาพลดลงมาก

4.2 ผลการฝึกสอนเพื่อการสกัดคำตอบ

การสกัดคำตอบจะทดสอบโดยมีข้อมูลนำเข้าเป็นชุดข้อมูลบทความหรือ D_{5000} มี 5,000 บทความ และชุดคำถาม 100 คำถาม จากขั้นตอนการดำเนินงานจะมีการหาคำบริบทของคำในบทความที่ตรงกับคำถามโดยกำหนดคำข้างหน้า k คำและคำข้างหลัง k คำ การวัดผลจึงมีการเปรียบเทียบผลโดยการกำหนด k = 3 และ k = 10 นอกจากนี้ยังมีการวัดผลว่าการตัดคำที่มีตำแหน่งมากกว่าค่ากำหนดส่งผลต่อประสิทธิภาพในการวัดผลหรือไม่

	การคำนวณคะแนนจากตำแหน่งของคำ				**Word2vec**			
	k = 3		**k = 10**		**k = 3**		**k = 10**	
	มีค่ากำหนด	**ไม่มีค่ากำหนด**	**มีค่ากำหนด**	**ไม่มีค่ากำหนด**	**มีค่ากำหนด**	**ไม่มีค่ากำหนด**	**มีค่ากำหนด**	**ไม่มีค่ากำหนด**
ตอบถูกอันดับแรก	15 %	16 %	**18 %**	16 %	8 %	**10 %**	7 %	4 %
หาคำตอบได้	63 %	73 %	78 %	**87 %**	73 %	80 %	86 %	**92 %**

ตารางที่ 4.3 การวัดผลของการคำนวณคะแนนจากตำแหน่งของคำ

วิธีที่ 1 การคำนวณคะแนนจากตำแหน่งของคำ

จากตารางที่ 4.3 การวัดผลของการคำนวณคะแนนจากตำแหน่งของคำ แสดงให้เห็นว่า การกำหนดคำข้างหน้า 10 คำและคำข้างหลัง 10 คำและการตัดคำที่มีตำแหน่งมากกว่าค่ากำหนดส่งผลให้มีประสิทธิภาพด้านการค้นคืนคำตอบที่ถูกต้องเป็นอันดับแรก แต่การกำหนดคำข้างหน้า 10 คำและคำข้างหลัง 10 คำและการไม่ตัดคำที่มีตำแหน่งมากกว่าค่ากำหนดสามารถค้นคืนคำตอบได้มากกว่า

วิธีที่ 2 Word2vec

จากตารางที่ 4.3 การวัดผลของ Word2vec แสดงให้เห็นว่า การกำหนดคำข้างหน้า 3 คำ และคำข้างหลัง 3 คำและการไม่ตัดคำที่มีตำแหน่งมากกว่าค่ากำหนดส่งผลให้มีประสิทธิภาพด้านการค้นคืนคำตอบที่ถูกต้องเป็นอันดับแรก แต่การกำหนดคำข้างหน้า 10 คำและคำข้างหลัง 10 คำและการไม่ตัดคำที่มีตำแหน่งมากกว่าค่ากำหนดสามารถค้นคืนคำตอบได้มากกว่า

จากตารางที่ 4.3 แสดงให้เห็นว่า วิธีที่สกัดคำตอบได้ถูกต้องเป็นอันดับแรกมากที่สุดคือ การคำนวณคะแนนจากตำแหน่งของคำโดยการกำหนดคำข้างหน้า 10 คำและคำข้างหลัง 10 คำและการตัดคำที่มีตำแหน่งมากกว่าค่ากำหนด ซึ่งค่าเฉลี่ยของจำนวนคำที่น่าจะเป็นคำตอบเป็น 40 คำ จากคำในเอกสารทั้งหมดที่ปรากฏในเอกสารเฉพาะภาษาไทยและตัวเลข 735 คำ

4.3 ผลของโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย

จากผลของการค้นคืนเอกสารแสดงให้เห็นว่า วิธีการจับคู่สายอักขระมีผลลัพธ์ในการค้นคืนเอกสารมากที่สุด และผลการสกัดคำตอบแสดงให้เห็นว่า วิธีการคำนวณคะแนนจากตำแหน่งของคำโดยที่ k=10 และมีค่ากำหนด มีผลลัพธ์ในการค้นคืนคำตอบที่ถูกต้องเป็นอันดับแรกมากที่สุด จึงทำการรวมการค้นคืนเอกสารและการสกัดคำตอบเข้าด้วยกัน แล้วทดสอบบน D_{5000} ซึ่งมี 5,000 บทความ และ D_{125000} ซึ่งมี 125,000 บทความ มีผลการค้นคืนดังตารางที่ 4.4

	ชุดที่ 1	**ชุดที่ 2**
ตอบถูกอันดับแรก	9 %	5 %

ตารางที่ 4.4 ผลของโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย

จากตารางที่ 4.4 ผลของโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย แสดงให้เห็นว่าหากมีวิธีการสกัดคำตอบที่ดีกว่านี้ อาจส่งผลให้โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทยสามารถค้นคืนคำตอบที่ถูกต้องเป็นอันดับแรกมากยิ่งขึ้น

บทที่ 5

ข้อสรุปและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงผลสรุป ซึ่งเป็นผลการทดสอบประสิทธิภาพวิธีการค้นคืนเอกสารและการสกัดคำตอบจากคลังข้อมูลวิกิพีเดียภาษาไทยที่ได้นำเสนอ และผลจากการรวมส่วนค้นคืนเอกสารและการสกัดคำตอบเข้าด้วยกัน รวมถึงปัญหาและข้อเสนอแนะจากการทำโครงงานชิ้นนี้

5.1 สรุปผลการวิจัย

จากการทดสอบประสิทธิภาพวิธีการค้นคืนเอกสารแบบการจับคู่สายอักขระและการสกัดคำตอบจากคลังข้อมูลวิกิพีเดียภาษาไทยแบบการคำนวณคะแนนจากตำแหน่งของคำโดยที่ k=10 และมีค่ากำหนด จากเอกสารจำนวน 125000 เอกสาร เป็นวิธีที่ดีที่สุดโดยสามารถเลือกคำตอบได้ถูกอันดับหนึ่งได้ 5% และมากกว่าการสุ่มเลือกคำตอบที่มีความน่าจะเป็น 0.000008% อยู่ 4.999992%

5.2 ปัญหาและอุปสรรคระหว่างการดำเนินงาน

- ข้อมูลของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทยมีรูปแบบเป็นไฟล์ข้อความที่ไม่มีการแยกหัวเรื่องเอกสารและไม่มีการแบ่งย่อหน้าของเอกสาร ซึ่งไม่ตรงกับขั้นตอนการดำเนินงานแรกที่วางแผนไว้ที่จะมีการคัดเลือกย่อหน้า จึงทำการดูดข้อมูลจากวิกิพีเดียมาใช้ซึ่งได้ข้อมูลที่พร้อมใช้งานมากกว่า ในรูปแบบเจสัน (json) แต่เนื่องจากข้อมูลจากวิกิพีเดียไม่สอดคล้องกับชุดข้อมูลคำถามคำตอบของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย จึงต้องกลับไปใช้ข้อมูลของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย และปรับเปลี่ยนขั้นตอนการดำเนินงาน
- ข้อมูลของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทยบางเอกสารมีข้อมูลไม่ครบถ้วนมีแต่หมายเลขเอกสาร ไม่มีเนื้อหาข้อมูล
- ชุดข้อมูลคำถามคำตอบของการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย บางคำถามประกอบด้วยภาษาไทยและภาษาอังกฤษ และบางคำถามต้องตอบเป็นภาษาอังกฤษ ซึ่งงานวิจัยฉบับนี้จะวิเคราะห์เฉพาะข้อมูลที่เป็นภาษาไทยและตัวเลข ทำให้ประสิทธิภาพในการตอบคำถามถูกต้องลดลง

- ชุดข้อมูลเอกสารมีจำนวนมาก ทำให้การทำความสะอาดข้อมูลและการตัดคำในเอกสารใช้เวลานาน

- การตัดคำพวกคำเฉพาะและคำทับศัพท์ ไม่สามารถหาขอบเขตของคำได้ถูก เช่น มูวี่อวอร์ด จังหวัดนะระ เป็นต้น จากการตัดคำที่ไม่สมบูรณ์ทำให้ความหมายของคำหลังจากถูกตัดคำแล้วเปลี่ยนแปลงไป ซึ่งส่งผลให้ประสิทธิภาพในการสกัดคำตอบไม่ค่อยถูกต้อง

- การทำงานมีการประมวลผลบนโคแลป (Google Colab) ซึ่งสะดวกต่อการใช้งาน เนื่องจากสามารถติดตั้งคลังโปรแกรมไลบรารี่ได้ง่ายและสามารถเข้าถึงไฟล์จากกูเกิลไดรฟ์ (Google Drive) ได้ นอกจากนี้ยังสามารถจดบันทึกข้อความได้ แต่มีปัญหาด้านเวลาในการทำงาน (runtime) เนื่องจากบางครั้งใช้เวลาในการทำงานได้ถึง 12 ชั่วโมง แต่บางครั้งก็ใช้เวลาในการทำงานได้ไม่นานและหลุดค่อนข้างบ่อย ทำให้การประมวลผลไม่สมบูรณ์ ขณะที่คลาวด์ (Google Cloud) มีพื้นที่จัดเก็บข้อมูลที่จำกัดรวมถึงเครดิต (ค่าใช้งาน) ในการใช้งานอย่างจำกัด จึงเลือกใช้คลาวด์เฉพาะงานที่ต้องใช้เวลาในการทำงานนาน เช่น การตัดคำ แล้วนำผลลัพธ์มาบันทึกในกูเกิลไดรฟ์เพื่อใช้งานต่อในภายหลัง

5.3 ข้อเสนอแนะ

หลังจากพัฒนาโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย พบข่อเสนอแนะคือ หากมีชุดคำถามคำตอบในการฝึกสอนมากยิ่งขึ้น และมีวิธีในการสกัดคำตอบเพิ่มเติมที่สามารถค้นคืนคำตอบที่ถูกต้องเป็นอันดับแรก จะสามารถเพิ่มประสิทธิภาพให้กับโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย

รายการอ้างอิง

[1] Pum-Mo Ryu,Myung-Gil Jang and Hyun-Ki Kim.Open domain question answering using Wikipedia-based knowledge model. Information Processing & Management volume 50, Issue 5 (September 2014) : 683-692

[2] Decha, Hatsanai and Karn Patanukhom. Development of thai question answering system.ICCIP'17 Proceedings of the 3rd International Conference on Communication and Information Processing (2017) :124-128

[3] NECTEC.LexTo [online].2016. Available from : http://www.sansarn.com/lexto [2018,August 1]

[4] Danqi Chen, Adam Fisch, Jason Weston,and Antoine Bordes.Reading Wikipedia to Answer Open-Domain Questions. arXiv:1704.00051v2 (2017)

[5] Clinton Gormley and Zachary Tong. Elasticsearch: The Definitive Guide. O'Reilly Media, Inc.,(2015) cited in Danqi Chen, Adam Fisch, Jason Weston,and Antoine Bordes.Reading Wikipedia to Answer Open-Domain Questions. arXiv:1704.00051v2 (2017)

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation.Empirical Methods in Natural Language Processing (EMNLP) (2014) : 1532-1543

[7] Radim Rehurek. Gensim.[online]. 2018. Available from : https://radimrehurek.com/gensim/ [2018,August 8]

[8] Rakpong Kittinaradorn. Deepcut.[online]. 2018. Available from : https://github.com/rkcosmos/deepcut [2018,August 8]

[9] Wirote Aroonmanakun. tltk.[online]. 2018. Available from : https://pypi.org/project/tltk [2018,August 8]

[10] Wannaphong Phatthiyaphaibun. PyThaiNLP.[online]. 2018. Available from : https://pypi.org/project/pythainlp/ [2018,August 8]

[11] Antti Haapala. Levenshtein.[online]. 2018. Available from : https://pypi.org/project/python-Levenshtein/ [2018,August 8]

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv :1301.2781v3 (2013)

[13] Xin Rong. word2vec Parameter Learning Explained. arXiv:1411.2738v4 (2016)

[14] Yoav Goldberg and Omer Levy. word2vec Explained: Deriving Mikolov et al.arXiv:1402.3722v1(2014)

[15] Tomas Mikolov and Quoc Le. Distributed Representations of Sentences and Documents. arXiv:14053v2 (2014)

ภาคผนวก

แบบเสนอหัวข้อโครงงาน รายวิชา 2301399 Project Proposal
ปีการศึกษา 2561

ชื่อโครงงาน (ภาษาไทย)	โปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย
ชื่อโครงงาน (ภาษาอังกฤษ)	A Question Answering program from Thai Wikipedia
อาจารย์ที่ปรึกษา	ผศ.ดร.ทิตยา หวานวารี
ผู้ดำเนินการ	1. นางสาวณัฐภาศ์ แจ้งสว่าง เลขประจำตัวนิสิต 5833624223
	2. นางสาวปณิตา วิโรจน์วงษ์ชัย เลขประจำตัวนิสิต 5833640223
	สาขาวิชา วิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
	คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

หลักการและเหตุผล

การสืบค้นข้อมูลข่าวสารเป็นสิ่งสำคัญอย่างยิ่งไม่ว่าจะยุคสมัยไหน ในปัจจุบันสามารถค้นหาข้อมูลจากเครือข่ายอินเทอร์เน็ตได้ทุกที่ทุกเวลาโดยไม่ต้องเสียเวลาไปห้องสมุด มนุษย์มีการสืบค้นข้อมูลตลอดช่วงอายุคนเพื่อนำมาใช้ประโยชน์ในด้านต่าง ๆ อย่างมากมาย ทั้งด้านการเรียน การทำงาน หรือในชีวิตประจำวัน ดังนั้นความสะดวกรวดเร็วและความถูกต้องของข้อมูลจึงเป็นสิ่งที่ผู้ใช้ต้องการ

ในอินเทอร์เน็ตมีข้อมูลสารสนเทศมากมาย ผู้ใช้สามารถสืบค้นได้ ทั้งข้อความ รูปภาพ สื่อมัลติมิเดีย ภาพเคลื่อนไหว วีดิโอ และข้อมูลอื่น ๆ ได้ตามต้องการ การที่ผู้ใช้จะค้นหาสิ่งที่ต้องการในข้อมูลจำนวนมากนั้นเป็นเรื่องยาก จึงต้องอาศัยเครื่องมือที่เรียกว่า ระบบค้นหาข้อมูล (Search Engine) แต่ในปัจจุบันระบบค้นหาข้อมูลไม่ได้ตอบสนองผู้ใช้โดยตรง เนื่องจากระบบจะค้นหาเอกสารที่เกี่ยวข้องกับคำสำคัญ (Keyword) ที่ผู้ใช้ค้น หลังจากนั้นผู้ใช้ต้องใช้เวลาในการอ่านเอกสารหาคำตอบเองซึ่งเสียเวลา โปรแกรมตอบคำถามจะช่วยเพิ่มประสิทธิภาพของระบบค้นหาข้อมูลให้มีความรวดเร็วมากยิ่งขึ้น

การสืบค้นข้อมูลและเข้าถึงข้อมูลสารสนเทศสามารถทำผ่านเว็บบราว์เซอร์ทั่วโลก เว็บไซต์ที่รวบรวมข้อมูลจำนวนมากซึ่งได้รับความนิยมมากที่สุดมีชื่อว่า วิกิพีเดีย เป็นสารานุกรมที่มีเนื้อหาเสรีและมีหลากหลายภาษา แต่ละภาษาจะมีโครงสร้างที่แตกต่างกัน ทั้งอักขระที่ใช้และไวยากรณ์ เราจึงต้องดัดแปลงระบบสำหรับแต่ละภาษา เช่น ข้อความภาษาไทยจะมีโครงสร้างภาษาต่างจากภาษาอังกฤษ การเรียงลำดับคำ และภาษาไทยจะมีสระและวรรณยุกต์เข้ามาผสมในคำหรือประโยค ดังนั้นคณะผู้จัดทำจึงต้องการพัฒนาโปรแกรมตอบคำถามจากคลังข้อมูลวิกิพีเดียภาษาไทย เพื่อแก้ปัญหาที่กล่าวมาข้างต้น

วัตถุประสงค์

เพื่อพัฒนาโปรแกรมตอบคำถามโดยการค้นหาคำตอบจากวิกิพีเดียภาษาไทย เพื่อให้ตอบคำถามสะดวกรวดเร็วมากขึ้น

งานวิจัยที่เกี่ยวข้อง

จากงานวิจัยของ Ryu และคณะ [1] ใช้การตัดคำและกำกับหมวดคำในการวิเคราะห์คำถาม การวิเคราะห์คำถามแบ่งเป็น 3 เกณฑ์ คือ 1. รูปแบบของคำตอบ เช่น ข้อเท็จจริง (factoid) รายการ และคำตอบเชิงพรรณา 2. เนื้อหาของคำตอบ แบ่งว่าคำตอบเป็นอะไร เช่น บุคคล สถานที่ วันเวลา 3. เป้าหมายของคำถามและคุณสมบัติของเป้าหมายนั้น เช่น คำถามว่า “มาบุญครองอยู่ที่ไหน” มี “มาบุญครอง” เป็นเป้าหมายของคำถาม และคุณสมบัติที่ต้องการคือสถานที่ที่มาบุญครองตั้งอยู่ ระบบประกอบด้วย 3 ขั้นตอน ดังนี้ การวิเคราะห์คำถาม การค้นคืนเอกสาร และ การตอบคำถาม โดยระบบจะวิเคราะห์คำถาม แล้วเทียบคำถามกับหัวข้อของบทความ ถ้าคำถามและหัวข้อบทความตรงกันค่าคะแนนจะมีค่าเป็น 1 หมายความว่าเลือกค้นคืนเอกสารนี้ ต่อมาแบ่งบทความออกเป็นบทความย่อย ๆ แล้ววัดระยะระหว่างคำในคำถาม เพื่อดูว่าประโยคใดในบทความตรงกับคำถามมากที่สุด สุดท้ายระบบจะนำคะแนนมาคูณกัน แล้วเลือกคำตอบที่มีคะแนนสูงที่สุด ผลการทดลองพบว่า ระบบมีความถูกต้อง 71% ในกลุ่มคำตอบแบบข้อเท็จจริง ส่วนกลุ่มคำตอบแบบรายการ และคำตอบเชิงพรรณา ใช้ F-score เป็นเกณฑ์ ได้ 48% และ 33% ตามลำดับ

จากงานวิจัยของ Decha และ Patanukhom [2] ใช้โปรแกรม LexTo [3] ซึ่งเป็นการจับคู่คำแบบยาวที่สุด (longest matching) ในการตัดคำ และตัดแบ่งประโยคโดยใช้ข่ายงานประสาทเทียม ซึ่งพิจารณาจากการเว้นวรรค โดยนับคำที่อยู่หน้าและหลังเว้นวรรค จากนั้นจัดแบ่งแยกจำนวนชนิดคำนาม คำกริยาและชนิดตัวเลข จัดกลุ่มคำถามโดยแบ่งคำถามออกเป็น 7 ประเภท คือ เท่าไร ใคร ที่ไหน เมื่อใด อย่างไร แบบใดและ สิ่งใด แล้วตัดคำที่ไม่จำเป็น (stop word) ออก เพื่อนำคำสำคัญของคำถามและบริบทมาใช้ในการหาคำตอบที่น่าจะเป็นไปได้ด้วยการจับคู่คำ ผลการทดลอง แบ่งเป็น 2 ส่วน ได้แก่ การแบ่งประโยคซึ่งใช้การตรวจสอบแบบไขว้ 10 กลุ่ม (10-fold cross validation) ในการวัดประสิทธิภาพ โดยมีค่าเฉลี่ยความถูกต้อง 81.31% ค่าความเที่ยงตรง (precision) 81.36% ค่าการค้นคืน (recall) 81.33% ส่วนการตอบคำถาม มีค่า MRAR (mean reciprocal answer rank) เท่ากับ 0.657 และค่าความถูกต้องเท่ากับ 75.71%

จากงานวิจัยของ Chen และคณะ [4] ระบบแบ่งเป็น 2 ส่วน ได้แก่ การค้นคืนเอกสาร (Document Retriever) เริ่มจากตัดคำโดยใช้ Stanford CoreNLP toolkit จากนั้นระบบจะเลือกบทความ 5 บท ที่เกี่ยวข้องกับคำถาม โดยใช้ดัชนีแบบอินเวอร์ต (inverted index) และค่า TF-IDF ซึ่งมีประสิทธิภาพมากกว่าการใช้ ElasticSearch [5] และใช้การแฮช 2-แกรม (bigram hashing) ในการเพิ่มประสิทธิภาพการค้นคืน เมื่อได้บทความที่เกี่ยวข้องแล้ว ขั้นตอนการอ่านเอกสาร (Document Reader) จะแบ่งบทความออกเป็นย่อหน้าแล้วใช้ตัวแบบ LSTM สร้างตัวแทนของย่อหน้า โดยใช้เวกเตอร์แทนคำของ Glove [6] รวมถึงหมวดคำ นิพจน์ระบุนาม และความถี่ของคำ เป็นข้อมูลเข้าของการสร้างตัวแทนย่อหน้า ต่อมาทำตัวแทนคำถาม โดยใช้ข่ายงานประสาทเทียมแบบวนซ้ำโดยป้อนข้อมูลเข้าเป็นเวกเตอร์แทนคำเช่นเดียวกับขั้นตอนการอ่านเอกสาร

หลังจากนั้นใช้ตัวจำแนกคำต้นประโยคและท้ายประโยค ผลการทดลองพบว่าระบบมีความถูกต้อง 70.0% และค่า F1-score 79.0%

ความรู้ที่เกี่ยวข้อง

Word vector / Word embedding (Mikolov, 2013) คือการแปลงคำเป็นตัวเลขแล้วเก็บค่าในรูปแบบเวกเตอร์ ซึ่งเวกเตอร์ของคำต่าง ๆ ถูกคำนวณจากคำที่ห้อมล้อมคำนั้น ๆ ในส่วนนี้คณะผู้จัดทำจะฝึกสอน (train) เอง นอกจากนี้ยังสามารถหาความคล้ายคลึงของคำผ่านเวกเตอร์ของคำได้ โดยอาศัยผลคูณจุด (dot product) ของเวกเตอร์ของคำทั้งสอง

คลังโปรแกรมที่เกี่ยวข้อง

- **Gensim** เป็นคลังโปรแกรมซึ่งประกอบด้วยอัลกิริทึมต่าง ๆ ที่ใช้กับการจัดการข้อความ เช่น การหา tf-idf การสร้างเวกเตอร์ของคำ โปรแกรมถามตอบนี้จะอาศัยคลังโปรแกรม Gensim ในการสร้างเวกเตอร์ของคำ เพื่อใช้ในการกรองหัวข้อของเอกสารที่เกี่ยวข้อง และใช้ในการเลือกย่อหน้าที่น่าจะเป็นคำตอบ
- **Deepcut** เป็นคลังโปรแกรมที่ใช้อัลกอริทึม ข่ายงานประสาทเทียมเชิงลึก (Deep Neural Network) มีการตัดคำโดยใช้การกรองแบบคอนโวลูชัน (Convolutional filters) และใช้การจำแนกคำ (binary classification) ว่าเป็นจุดเริ่มต้นของคำหรือไม่
- **Tltk** เป็นคลังโปรแกรมที่ใช้ในการหาขอบเขตของคำโดยการจับคู่คำที่ยาวที่สุด (maximum matching) ในคลังข้อมูลพจนานุกรม ซึ่งใช้อัลกอริทึม 3-แกรม ของคลังข้อมูลที่ตัดพยางค์แล้วช่วยเลือกการตัดพยางค์ที่ดีสุดก่อนจะนำไปรวบพยางค์เป็นคำภายหลังและมีการเก็บคำศัพท์ในคลังข้อมูล 630000 คำ จากหนังสือพิมพ์ แต่ไม่มีการเก็บคำที่เป็นคำเฉพาะ
- **Pythainlp** เป็นคลังโปรแกรมที่ใช้ในการหาขอบเขตของคำโดยค้นหาคำในคลังข้อมูลพจนานุกรม มีการใช้วิธี unigram ในการทำ การกำกับหมวดคำ นอกจากนี้คลังโปรแกรมยังมีการรวบรวมคำไม่สำคัญ เพื่อใช้ในการลบคำไม่สำคัญทิ้ง

ภาพรวมของระบบ

จากรูปที่ 1 จะแบ่งเป็นการทำงานดังนี้

1. การค้นคืนเอกสาร (Document Retriever) เพื่อหาบทความที่น่าจะมีคำตอบ

1.1 หาขอบเขตของคำในประโยคคำถาม โดยใช้คลังโปรแกรมของไพธอนชื่อว่า “Deepcut” เปรียบเทียบคลังโปรแกรม 3 วิธี

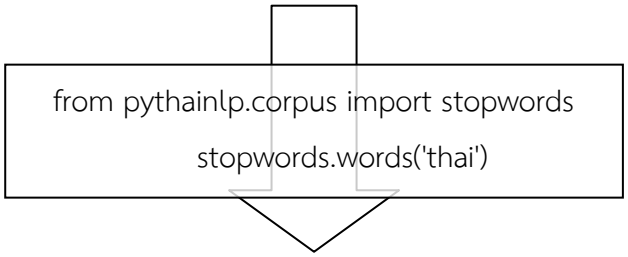
Deepcut: ['เฮโจเกียว', ' ', 'หรือ', ' ', 'เฮเซเกียว', ' ', 'เป็น', 'เมือง', 'หลวง', 'โบราณ', 'ของ', 'ประเทศ', 'อะไร']

Tltk: เฮ|โจ|เกียว|||<s/>|หรือ|<s/>|เฮ|เซ|เกียว||<u/>เป็น|เมืองหลวง|โบราณ|ของ|ประเทศ|อะไร<u/>

Pythainlp: ['เฮ', 'โจเกียว', ' ', 'หรือ', ' ', 'เฮ', 'เซ', 'เกียว', ' ', 'เป็น', 'เมืองหลวง', 'โบราณ', 'ของ', 'ประเทศ', 'อะไร']

1.2 ตัดคำที่ไม่จำเป็น (stop word) ออกจากคำถาม เพื่อหาคำสำคัญโดยใช้คลังโปรแกรมของไพธอนชื่อว่า “Pythainlp”

['เฮโจเกียว', ' ', 'หรือ', ' ', 'เฮเซเกียว', ' ', 'เป็น', 'เมือง', 'หลวง', 'โบราณ', 'ของ', 'ประเทศ', 'อะไร']



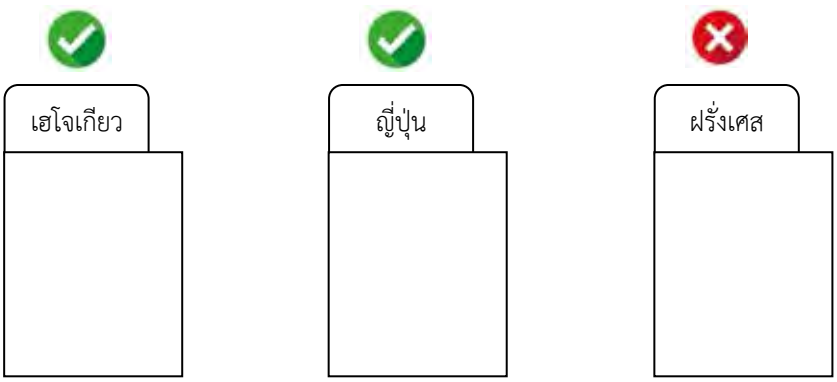
['เฮโจเกียว', 'เฮเซเกียว', 'เมือง', 'หลวง', 'โบราณ', 'ประเทศ‘]

1.3 นำคำสำคัญจากคำถามที่ได้มาใช้ในกรองหัวเรื่อง (title) เพื่อคัดเลือกบทความที่เกี่ยวข้อง โดยเปรียบเทียบเวกเตอร์แทนคำสำคัญของคำถาม และหัวเรื่อง เช่น เวกเตอร์แทนคำสำคัญของคำถาม 1 คำจะกรองหัวเรื่อง ได้ k หัวเรื่อง แล้วหาผลรวมของเวกเตอร์

<u>ตัวอย่าง</u>

คำสำคัญของคำถาม : เฮเซเกียว

หัวเรื่อง : **เฮโจเกียว** ญี่ปุ่น



2. การอ่านเอกสาร (Document Reader)

2.1 แบ่งบทความออกเป็นย่อหน้า เนื่องจากมีประสิทธิภาพมากกว่าประโยค

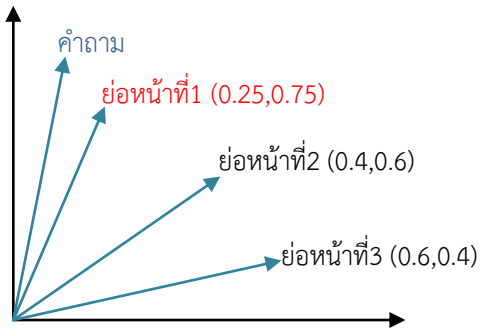
2.2 สร้างลักษณะสำคัญแทนย่อหน้า เพื่อนำไปหาคำตอบ เช่น ใช้เวกเตอร์แทนคำเพื่อสร้างตัวแทนของย่อหน้าหรือดูจำนวนคำนาม

เฮโจเกียว

เฮโจเกียว หรือ เฮเซเคียว เป็นเมืองหลวงโบราณของประเทศญี่ปุ่น อีกนัยหนึ่งก็คือเมืองหลวงในยุคนารานั่นเอง ปัจจุบันตั้งอยู่ใน อำเภอเมืองนารา และ อำเภอโอยามาโตะ จังหวัดนารา

มีรูปทรงเป็นแบบ เวียง คือรูปทรงสี่เหลี่ยมพื้นผ้าตามแบบจีน หรือ ตามแบบที่นิยมสร้างกันในอาณาจักรล้านนา และ พม่า ซึ่งเฮโจเกียวนี้ได้นำแบบแผนมาจากเมืองซีอานประเทศจีน เมืองหลวงของจีนในขณะนั้น

ในปี พ.ศ. 1250 (ค.ศ. 707) ได้มีเหตุการณ์ย้ายเมืองหลวง จากฟูจิวะระเคียว ไปสู่เฮโจเคียว ซึ่งเป็นพระราชเสาวนีย์ของ พระจักรพรรดินีเกนเม แต่ก็มีการย้ายไปที่อื่นอีก คือ คุนิเกียว หรือ คุนิโนมิยะ หลังจากนั้นอีกไม่นาน ก็ได้กลับมาเป็นเมืองหลวงเหมือนเดิมในปี พ.ศ. 1283 (ค.ศ. 745)



3. การสกัดคำตอบ

3.1 เลือกย่อหน้าที่อาจเป็นคำตอบ

3.2 สกัดคำตอบโดยทำนายจุดเริ่มต้นและสิ้นสุดของคำตอบในย่อหน้า เช่น ใช้วิธีการจับคู่คำ (String Matching) หรือ เวกเตอร์แทนคำเพื่อสกัดคำตอบ

<u>ตัวอย่างการตอบคำถาม</u>

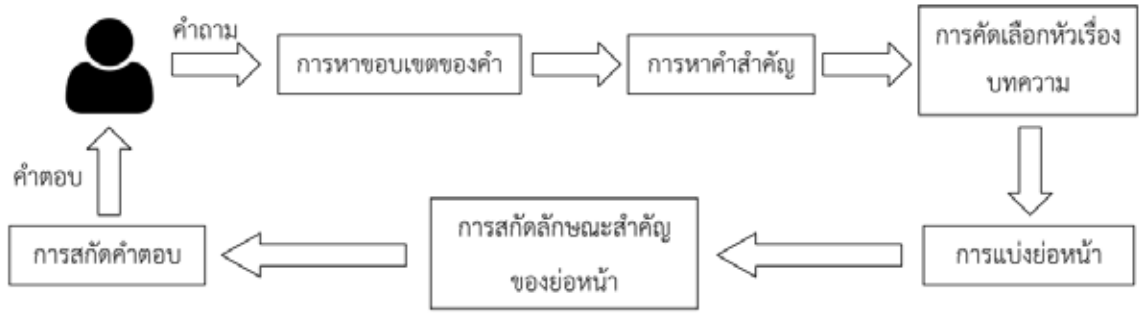
คำถาม : เฮโจเกียว หรือ เฮเซเกียว เป็นเมืองหลวงโบราณของประเทศอะไร

คำตอบ : ประเทศญี่ปุ่น

โดยลักษณะของการเลือกคำตอบจะเป็น ดังนี้

"answer" : "ประเทศญี่ปุ่น" , "answer_begin_position " : 142,

"answer_end_position" : 155 , article_id" : 32031



รูปที่ 1 ภาพรวมของระบบ

ขอบเขตของโครงงาน

1. โครงงานนี้ศึกษาเฉพาะการค้นคืนคำตอบจากวิกิพีเดียภาษาไทย
2. ผลลัพธ์ของคำถามเป็นช่วงคำสั้น ๆ ที่สืบค้น มาจากวิกิพีเดียภาษาไทย ณ วันที่รวบรวมข้อมูลเท่านั้น
3. ระบบสามารถตอบเฉพาะคำถามเกี่ยวกับข้อเท็จจริงเท่านั้น ระบบไม่สามารถตอบคำถามที่เป็นความคิดเห็นได้ รวมถึงไม่สามารถตอบคำถามแบบถูกหรือผิดได้
4. ระบบสามารถตอบคำถามที่เป็นภาษาไทยเท่านั้น โดยคำถามจะต้องมีการสะกดคำที่ถูกต้องตามพจนานุกรมราชบัณฑิตยสถาน

วิธีการดำเนินงาน

1. ค้นหาและศึกษางานวิจัยรวมถึงเนื้อหาที่เกี่ยวข้องกับการถามตอบ โดยอ้างอิงขั้นตอนการทำงานจากงานวิจัยของ Chen และคณะ [4] ซึ่งเป็นระบบที่ใช้วิกิพีเดียภาษาอังกฤษเป็นฐานข้อมูลในการค้นคืนเอกสารและหาคำตอบ และใช้ ฐานข้อมูล SQuad ในการเทรนและทดสอบระบบ
2. ศึกษาเครื่องมือ โปรแกรมและเทคนิคที่ใช้ในงานวิจัย เพื่อเป็นประโยชน์ในการทำโครงงาน โดยศึกษาการตัดคำในคลังโปรแกรม deepcut [7] และ tltk [8] ศึกษาการแปลงคำเป็นเวกเตอร์ การทำ word embedding โดยใช้ Long Short Term Memory ซึ่งเป็น Recurrent Neural Network model การทำ Part Of Speech Tagging และ Name Entity Recognition การหา TF-IDF
3. รวบรวมข้อมูล เพื่อกำหนดขอบเขตของโครงงาน
4. วิเคราะห์และออกแบบวิธีการ แบ่งการทำงานออกเป็น 3 ส่วนหลัก ๆ ได้แก่ การค้นคืนหัวข้อเอกสารที่เกี่ยวข้องกับคำถาม การอ่านเอกสาร และการหาคำตอบ
5. พัฒนาระบบถามตอบ
6. ตรวจสอบความถูกต้องและวัดประสิทธิภาพโดยตรวจสอบความถูกต้องของคำตอบจากชุดข้อมูลทดสอบและ F-score
7. สรุปผลการดำเนินงาน
8. จัดทำเอกสาร

ตารางเวลาการดำเนินงาน

ขั้นตอนการดำเนินงาน	ปี 2561					ปี 2562			
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.
1.ค้นหาและศึกษาบทความรวมถึงเนื้อหาที่เกี่ยวข้องกับการถามตอบ									
2.ศึกษาเครื่องมือ โปรแกรมและเทคนิคที่ใช้ในงานวิจัย									
3.รวบรวมข้อมูล เพื่อกำหนดขอบเขตของโครงงาน									
4.วิเคราะห์และออกแบบวิธีการ									
5.พัฒนาระบบถามตอบ									
6.ตรวจสอบความถูกต้องและวัดประสิทธิภาพ									
7.สรุปผลการดำเนินงาน									
8.จัดทำเอกสาร									

ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ต่อผู้พัฒนา

5. มีความรู้ ความเข้าใจในการทำระบบถามตอบ
6. เพิ่มพูนทักษะการเขียนโปรแกรมและการพัฒนาระบบ
7. เรียนรู้การคิดวิเคราะห์วางแผนการทำงานอย่างเป็นระบบแบบแผน เพื่อให้เกิดประโยชน์สูงสุดตามทรัพยากรที่มีอยู่
8. ฝึกการเรียนรู้ด้วยตนเอง การยอมรับฟังความคิดเห็นของผู้อื่น ความตรงต่อเวลา ความรับผิดชอบในหน้าที่

ประโยชน์ต่อผู้ใช้ระบบ

4. เป็นทางเลือกให้ผู้ใช้งานเลือกในการค้นหาข้อมูลสารสนเทศ
5. ผู้ใช้งานสามารถค้นหาและเข้าถึงข้อมูลได้สะดวกรวดเร็ว ผลลัพธ์ที่ได้จากการค้นหาจะเป็นข้อความสั้นๆ ซึ่งผู้ใช้งานไม่ต้องเสียเวลามาอ่านเอกสารเพื่อหาคำตอบ รวมถึงบอกชื่อเอกสารที่มาของคำตอบด้วย
6. ผลลัพธ์จากการค้นหามีความถูกต้องและแม่นยำ

อุปกรณ์และเครื่องมือที่ใช้

1. ฮาร์ดแวร์
   1.1 แล็ปท็อปคอมพิวเตอร์ รุ่น Intel(R) Core(TM) i3-4005U ความเร็ว 1.70 GHz RAM 4.00 GB
   1.2 แล็ปท็อปคอมพิวเตอร์ รุ่น Intel(R) Core(TM) i5-2450M ความเร็ว 2.50 GHz RAM 4.00 GB
   1.3 Google Cloud Platform compute engine machine type n1-standard-8 vCPU 30 GB
2. ซอฟแวร์
   2.1 Google Chrome
   2.2 Mozilla Firefox
   2.3 Microsoft Office 365
   2.4 Spyder python 3.6

งบประมาณ

1. กระดาษถ่ายเอกสารขนาด A4 80 แกรม	ราคา	300 บาท
2. หมึกพิมพ์	ราคา	1500 บาท
3. ค่าถ่ายเอกสารและค่าทำรูปเล่ม	ราคา	300 บาท
4. SSD	ราคา	4000 บาท
5. ค่าเช่า Google Cloud Platform, Firebase, and APIs	ราคา	4000 บาท
6. ค่าอุปกรณ์จัดเก็บบันทึกข้อมูล	ราคา	3500 บาท
7. ค่าอุปกรณ์อิเล็กทรอนิกส์	ราคา	3000 บาท
8. ค่าบำรุงรักษาอุปกรณ์	ราคา	2000 บาท
	รวม	18600 บาท

<u>หมายเหตุ</u> ทั้งนี้งบประมาณอาจเปลี่ยนแปลงตามความเหมาะสมและขอถัวเฉลี่ยทุกรายการ

เอกสารอ้างอิง

[1] Pum-Mo Ryu,Myung-Gil Jang and Hyun-Ki Kim.Open domain question answering using Wikipedia-based knowledge model. Information Processing & Management volume 50, Issue 5 (September 2014) : 683-692

[2] Decha, Hatsanai and Karn Patanukhom. Development of thai question answering system.ICCIP'17 Proceedings of the 3rd International Conference on Communication and Information Processing (2017) :124-128

[3] NECTEC.LexTo [online].2016. Available from : http://www.sansarn.com/lexto [2018,August 1]

[4] Danqi Chen, Adam Fisch, Jason Weston,and Antoine Bordes.Reading Wikipedia to Answer Open-Domain Questions. arXiv:1704.00051v2 (2017)

[5] Clinton Gormley and Zachary Tong. Elasticsearch: The Definitive Guide. O'Reilly Media, Inc.,(2015) cited in Danqi Chen, Adam Fisch, Jason Weston,and Antoine Bordes.Reading Wikipedia to Answer Open-Domain Questions. arXiv:1704.00051v2 (2017)

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation.Empirical Methods in Natural Language Processing (EMNLP) (2014) : 1532-1543

[7] Rakpong Kittinaradorn. Deepcut.[online]. 2018. Available from : https://github.com/rkcosmos/deepcut [2018,August 8]

[8] Wirote Aroonmanakun. tltk.[online]. 2018. Available from : https://pypi.org/project/tltk [2018,August 8]

ประวัติผู้เขียน

ชื่อ ณัฐภาศ์ แจ้งสว่าง
การศึกษา สาขาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ที่อยู่ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพ 10330
อีเมล naspa.c@student.chula.ac.th

ชื่อ ปณิตา วิโรจน์วงษ์ชัย
การศึกษา สาขาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ที่อยู่ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพ 10330
อีเมล panita.vi@student.chula.ac.th