การปรับปรุงการแยกประเภทข้อความทวีตด้านการจราจรโดยใช้การเสริมข้อความ

IMPROVING A TEXT CLASSIFIER OF ROAD TRAFFIC TWEETS USING THE TEXT AUGMENTATION

ธวัชชัย รักษาชาติ

THAWATCHAI RAKSACHAT

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2567

KMITL-2024-EN-M- 027-243

IMPROVING A TEXT CLASSIFIER OF ROAD TRAFFIC TWEETS USING THE TEXT AUGMENTATION

THAWATCHAI RAKSACHAT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING PROGRAM IN ELECTRICSL AND COMPUTER ENGINEERING

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2024

KMITL-2024-EN-M- 027-243

หัวข้อวิทยานิพนธ์ การปรับปรุงการแยกประเภทข้อความทวีตด้านการจราจรโดยใช้การเสริมข้อความ
นักศึกษา นายธวัชชัย รักษาชาติ
รหัสประจำตัว 64601067
ปริญญา วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์
พ.ศ. 2567
อาจารย์ที่ปรึกษาวิทยานิพนธ์ รศ.ดร. รัฐชัย ชาวอุทัย

บทคัดย่อ

วิทยานิพนธ์ฉบับนี้มีวัตถุประสงค์เพื่อศึกษาการพัฒนาวิธีการที่จะทำให้มีประสิทธิภาพมากขึ้นในการใช้วิธีการเรียนรู้เชิงลึก (Deep learning) เพื่อการจำแนกข้อความในทวีตเตอร์ภาษาไทยที่เกี่ยวข้องกับการรายงานสภาพจราจร การจัดหมวดหมู่ประกอบด้วยสองระดับคือการระบุข้อความอุบัติการณ์ประกอบไปด้วยสองกลุ่มคือข้อความที่ไม่เกี่ยวกับสภาพจราจรและข้อความที่เกี่ยวกับสภาพจราจร ในระดับที่สองเป็นการจำแนกประเภทของข้อความที่เกี่ยวกับสภาพจราจรมีห้าหมวดหมู่ การศึกษาที่ผ่านมาได้ใช้โครงข่ายประสาทแบบคอนโวลูชัน (Convolutional Neural Network : CNN) และการสอนล่วงหน้าแบบเบิร์ต (Bidirectional Encoder Representations from Transformers : BERT Pre-trained) ในการจัดหมวดหมู่ แต่พบว่าจำเป็นต้องมีข้อมูลที่สมดุลเพื่อประสิทธิภาพที่ดีขึ้น ในการแก้ไขปัญหานี้ ผู้วิจัยขอนำเสนอการสร้างข้อความเพิ่มด้วยวิธีหน่วยความจำระยะสั้นระยะยาว (Long Short-Term Memory : LSTM) ผสานกับวิธีห่วงโซ่มาร์คอฟ (Markov chain) เพื่อเพิ่มข้อมูลและคำนวณค่าประสิทธิภาพด้วย BLEU Score ผสาน BERT Score ขั้นตอนถัดไปคือการสร้างชุดข้อมูล (Datasets) ที่สมดุลและนำข้อมูลที่สมดุลแล้วไปฝึกแบบจำลอง (Training model) โดยการใช้การเรียนรู้เชิงลึก CNN ร่วมกับวิธีหน่วยความจำระยะสั้นระยะยาว (Long Short-Term Memory : LSTM) ในการจัดหมวดหมู่ข้อความทั้งสองระดับ การทดลองแสดงให้เห็นว่ามีการปรับปรุงที่สำคัญในการจำแนกหมวดหมู่ข้อความจากทวิตเตอร์ด้วยคะแนน F1-Score ที่เพิ่มขึ้นถึง 18.18% เมื่อเปรียบเทียบกับวิธีอ้างอิง (Baseline)

คำสำคัญ : การเรียนรู้เชิงลึก, การทำส่วนเสริมข้อความ, การวิเคราะห์สภาพจราจรจากทวิตเตอร์, การเกิดอุบัติการณ์บนถนน, การแบ่งกลุ่มข้อความ

Thesis	IMPROVING A TEXT CLASSIFIER OF ROAD TRAFFIC TWEETS USING THE TEXT AUGMENTATION
Student	Mr.Thawatchai. Raksachat
Student ID.	64601067
Degree	Master of Engineering
Program	Electrical and conputer Engineering
Year	2024
Thesis Advisor	Assoc. Prof. Dr. Rathachai Chawuthai

ABSTRACT

Aims to study the development of methods for improving the efficiency of using deep learning techniques to classify text in Thai-language tweets related to traffic reports. The classification is two levels: classifying between non-traffic-related tweets and traffic-related tweets. At the second level, the traffic-related tweets are further categorized into five categories. Previous studies have employed Convolutional Neural Networks (CNN) and Bidirectional Encoder Representations from Transformers (BERT Pre-trained) for categorization. However, it was found that a balanced dataset is essential for better performance. To address this issue, The researcher proposes to create additional text using the long-term short-term memory method combined with the Markov chain method. To add data and calculate performance values with BLEU Score combined with BERT Score. The next step involves creating balanced datasets and training the model using deep learning, combining Convolutional Neural Networks (CNN) with Long Short-Term Memory (LSTM) for text classification at both levels. Experimental results demonstrate significant improvements in text categorization from Twitter, with an F1-Score increase of up to 18.18% when compared to the baseline method.

Keyword: Deep Learning, Text augmentation, Twitter Data Analytics, Road Traffic Incident, Text Classification

กิตติกรรมประกาศ

ขอขอบคุณ การทางพิเศษแห่งประเทศไทย หน่วยงานที่ข้าพเจ้าทำงานที่ได้มอบทุนการศึกษาให้แก่ข้าพเจ้าตลอดจวบจนกระทั่งจบการศึกษา ไม่ว่าจะเป็นค่าลงทะเบียนการศึกษาและค่าใช้จ่ายเพื่อการศึกษาอื่น ๆ ทำให้ข้าพเจ้าศึกษาหาความรู้ได้ด้วยดี

อนึ่ง วิทยานิพนธ์เล่มนี้สำเร็จได้ด้วยความกรุณาจากอาจารย์ที่ปรึกษา รศ.ดร.รัฐชัย ชาวอุทัย ที่ให้ความช่วยเหลือ ให้คำชี้แนะช่วยแก้ปัญหาตลอดจนให้ความรู้และประสบการณ์ที่ดีแก่ข้าพเจ้า

ขอขอบคุณพนักงานกรมทางหลวงที่ได้ให้การช่วยเหลือในส่วนของการแบ่งกลุ่มข้อมูลจนเพื่อนำข้อมูลนั้นมาทำการทดลองจนแล้วเสร็จ

สุดท้าย ขอกราบขอบพระคุณบิดามารดาที่เคารพ ภรรยา ลูกสาวและลูกชายอันเป็นที่รัก ที่คอยช่วยเหลือเป็นกำลังใจเป็นห่วงเป็นใยเสมอมา ตลอดจนเพื่อนร่วมงาน อาจารย์ทุกท่านที่ให้ความรู้ด้านต่าง ๆ ประกอบกันจนข้าพเจ้าสามารถสรรค์สร้างวิทยานิพนธ์เล่มนี้จนแล้วเสร็จได้ด้วยดี

ธวัชชัย รักษาชาติ

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ในทุกวันนี้มีจำนวนผู้ใช้รถยนต์เพิ่มขึ้นอย่างมาก จึงหลีกเลี่ยงไม่ได้ที่จะเกิดเหตุการณ์ไม่ปกติบนถนนไม่ว่าจะเป็นรถติด รถชน รวมถึงเหตุการณ์ทางธรรมชาติล้วนส่งผลให้เกิดปัญหาทางการจราจร ซึ่งกำลังกลายเป็นปัญหาสำคัญในการจัดการจราจรและระบบขนส่งอัจฉริยะ (Intelligent Transport System : ITS) ดังนั้น การรับรู้ถึงปัญหาทางด้านจราจรบนถนนได้อย่างรวดเร็วและครอบคลุมจะทำให้การบริหารจัดการจราจรทำได้ดียิ่งขึ้น ในกรณีที่ต้องการรับรู้ถึงเหตุการณ์จราจรบนถนนนั้นจึงมีหลายวิธี เช่น การรับรู้จากอุปกรณ์การวัด (sensor หรือเซนเซอร์) หรือการรับรู้จากกล้องโทรทัศน์วงจรปิด (Closed Circuit Television : CCTV) ซึ่งทั้งสองวิธีต้องใช้เงินทุนจำนวนมากและอาจไม่ครอบคลุมในหลายๆ พื้นที่การรับรู้ถึงเหตุการณ์บนถนนด้วยข้อความจากทวิตเตอร์ (Twitter) จึงเป็นวิธีการที่สามารถทำได้รวดเร็วและราคาไม่แพง ทวิตเตอร์ คือสื่อสังคมออนไลน์ (Social media หรือโซเชียลมีเดีย) ที่ได้รับความนิยมมากมีการส่งต่อข่าวสารผ่านช่องทางดังกล่าวเป็นอย่างมากสามารถรับรู้ข่าวสารได้เร็วส่งผลให้การบริหารจัดการจราจรสามารถทำได้รวดเร็วมากขึ้นตามไปด้วย [1]

การใช้ข้อความทวิตเตอร์รับรู้เหตุการณ์บนถนนเป็นวิธีการประมวลผลทางภาษาธรรมชาติ (Natural Language Processing : NLP หรือเอ็นเอลพี) การประมวลทางภาษามีวิธีการที่นิยมกันหลายวิธี โดยส่วนใหญ่จะเป็นวิธีการเรียนรู้เชิงลึก (Deep learning) ที่ใช้ข้อความมาฝึกฝน (Train หรือเทรน) เพื่อสร้างเป็นแบบจำลอง (Model หรือโมเดล) การแยกประเภท [1]-[5] งานวิจัยก่อนหน้านี้ [4][5] มีการแบ่งข้อความทวิตเตอร์ออกเป็นเหตุการณ์ต่าง ๆ เพื่อจำแนกประเภทข้อความ ซึ่งถูกแบ่งออกเป็น 5 กลุ่มได้แก่ 1. การรายงานเหตุการณ์จราจร 2. การรายงานเหตุการณ์อุบัติเหตุ 3. การรายงานเหตุการณ์ภัยพิบัติ 4. การรายงานพื้นที่ชุมนุม 5. การรายงานปิดถนนเพื่อซ่อมแซม จากการศึกษาในหลาย ๆ งานวิจัยพบว่าในแต่ละงานวิจัยมีการใช้ข้อมูลสำหรับการสร้างโมเดลเพื่อจำแนกประเภทข้อความที่มีจำนวนแต่ละกลุ่มที่มีความไม่สมดุล (Imbalance) โดยในบางงานวิจัยมีข้อความในบางกลุ่มน้อยกว่ากลุ่มอื่นเกินไปด้วยซ้ำจะเรียกค่าความต่างนี้ว่า อัตราส่วนความไม่สมดุล (Imbalance Ratio : IR หรือไออาร์) [6]

การจำแนกข้อความจากทวิตเตอร์มักจะพบกับปัญหาความไม่สมดุลของข้อมูลอยู่เสมอ การจัดการกับความไม่สมดุลนี้มีทางเลือกหลากหลาย ไม่ว่าจะเป็นการจัดการที่ธรรมดาที่สุดคือการเก็บข้อมูลกลุ่มที่มีน้อยเพิ่มเติม แต่ข้อเสียคือต้องใช้เวลาเพิ่มขึ้นมาก หากต้องการให้ได้ข้อมูลที่รวดเร็วจึงต้องใช้กระบวนการ

ทางสถิติหรือกระบวนการเรียนรู้ทางเครื่อง (Machine learning) เข้ามาช่วยจัดการกับข้อมูลที่ไม่สมดุลดังกล่าว โดยมีวิธีที่ใช้งานกันโดยมากอันดับแรก ๆ คือการสุ่มเลือก (Sampling) ไม่ว่าจะเป็นการสุ่มลดจำนวน (Under-sampling) หรือการสุ่มเพิ่มจำนวน (Over-sampling) [6] วิธีนี้เหมาะกับข้อมูลที่มีมากและเป็นตัวเลข ดังนั้นทางเลือกสำหรับการจำแนกประเภทข้อความจำเป็นต้องเพิ่มวิธีการใหม่ ๆ เข้ามาช่วยด้วยวิธีการนำเอาอัลกอริทึม (Algorithm) ทางคณิตศาสตร์ เช่น การสร้างข้อความใหม่ด้วยวิธีแบบจำลองมาร์คอฟ (Markov model หรือมาร์คอฟโมเดล) [7] คือ การสุ่มเลือกคำที่มีความน่าจะเป็นของคำต่อไปโดยจะเลือกคำที่มีค่าความน่าจะเป็นสูงสุดมาสร้างคำต่อไปเพื่อนำมาต่อกันเป็นข้อความ และมีงานวิจัยอื่น ๆ เช่น การสร้างข้อความด้วยวิธีหน่วยความจำระยะสั้นแบบยาว (Long Short-Term Memory: LSTM หรือแอลเอสทีเอ็ม) โดยมีการสร้างข้อความจากการทำนายคำที่จะเกิดขึ้นต่อด้วยกระบวนการเรียนรู้เชิงลึก [8] การใช้วิธีแอลเอสทีเอ็มเพื่อมาสร้างข้อความมีกระบวนการคือต้องมีคำตั้งต้น (seed word หรือซีดเวิร์ด) ส่งเข้าไปให้แบบจำลอง (Model หรือโมเดล) ทำนายคำต่อไปออกมาเป็นข้อมูลออก (output หรือเอาท์พุท) แล้วนำคำนั้นมาเป็นข้อมูลเข้า (Input หรืออินพุท) เพื่อพยากรณ์ (Predict) คำต่อไป วิธีนี้สามารถสร้างข้อความได้อย่างมหาศาล

จากการศึกษางานวิจัยอื่น ๆ เพิ่มในแนวทางการจัดการกับข้อความไม่สมดุล พบว่าการแก้ปัญหาด้วยวิธีการเพิ่มกลุ่มข้อความที่มีจำนวนน้อยด้วยวิธีการปรับเปลี่ยนคำบางคำในประโยคหรือที่เรียกว่าการเสริมข้อความ (Augmentation หรือออกเมนเทชัน) [9] มีการใช้วิธีเสริมข้อความด้วยการเพิ่มคำรายคำ (Token หรือโทเคน) หลายรูปแบบเพื่อสร้างความแตกต่างให้กับประโยคเล็ก ๆ น้อย ๆ ให้ดูเป็นธรรมชาติคล้ายกับข้อความที่เกิดจากการสร้างจากมนุษย์จริง ๆ จากการศึกษาเพิ่มเติมจากแนวคิดเรื่องความคล้ายคลึงของค่าเส้นสมมุติคำ (word vector หรือเวิร์ดเวคเตอร์) ในคลังข้อความ (Corpus หรือคอร์ปัส) ความคล้ายคลึงของคำจากการหาน้ำหนักของคำนั้นเป็นแนวคิดใหม่ล่าสุด (State-of-the-art) และมีการพัฒนาอย่างต่อเนื่องจนมาปัจจุบันมีการรวบรวมคำมาจากวิกิพีเดีย (wikipedia) มารวมกันเป็นคลังข้อความและมีการเทรนเพื่อแปลงคำเป็นเวิร์ดเวคเตอร์ขนาดใหญ่ที่มีทั้งหมด 275 ภาษา ในชื่อ BPEmb [10]

การจำแนกประเภทข้อความทวิตเตอร์ที่เกี่ยวข้องกับการจราจรถูกศึกษามาหลายงาน เช่น การตรวจจับข้อความอุบัติการณ์บนถนนด้วยการเรียนรู้เชิงลึกที่ทำงานร่วมกันระหว่างโครงข่ายประสาทแบบคอนโวลูชัน (Convolutional Neural Network: CNN หรือซีเอ็นเอ็น) และหน่วยความจำระยะสั้นแบบยาว (Long Short-Term Memory: LSTM หรือแอลเอสทีเอ็ม) เพื่อจำแนกข้อความจราจรแบ่งออกเป็น 5 ประเภท [1] ต่อมามีการศึกษากระบวนการแยกประเภทข้อความทวิตเตอร์ทำงานร่วมกันระหว่างการเรียนรู้เชิงลึกซีเอ็นเอ็นและการสอนล่วงหน้าแบบเบิร์ต (Bidirectional Encoder Representations from Transformers : BERT Pre-trained หรือเบิร์ตพรีเทรน) เพื่อจำแนกข้อความจราจรแบ่งออกเป็น

4 ประเภท [4] ทั้งนี้เมื่อศึกษาอย่างใกล้ชิดพบว่างานทั้งสองชิ้นใช้ชุดข้อมูล (Datasets หรือเดต้าเซ็ต) ที่ไม่สมดุลกันมาสร้างโมเดลส่งผลให้การพยากรณ์บางกลุ่มไม่แม่นยำ

จากปัญหาความไม่สมดุลของข้อความต่าง ๆ ดังที่กล่าวมาจึงได้เกิดเป็นงานวิจัยนี้ขึ้น โดยในการวิจัยนี้ได้นำเทคนิคการสร้างข้อความคำด้วยวิธีแอลเอสทีเอ็มผสานกับวิธีมาร์คอฟโมเดล แล้ววัดผลการเสริมคำด้วยวิธีการหาค่าความคล้ายคลึงคำด้วยค่าคะแนนแบบเบิร์ต (BERT Score หรือเบิร์ตสกอร์) และหาค่าความต่างของคำด้วยค่าคะแนนเบลอ (BLEU Score หรือเบลอสกอร์) แล้วนำข้อความที่ผ่านการจัดการความไม่สมดุลเรียบร้อยแล้วมาจำแนกข้อความด้วยวิธีการเรียนรู้เชิงลึกด้วยซีเอ็นเอ็นร่วมกับแอลเอสทีเอ็ม แล้ววัดผลการจำแนกกลุ่มด้วยการประเมินค่าความแม่น (Accuracy หรือแอคคิวเรซี) และเอฟวัน (F1)

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

ความมุ่งหมายและวัตถุประสงค์ของการศึกษานี้ คือ เพื่อการศึกษาวิธีการสำหรับการปรับปรุงการสร้างโมเดล สำหรับการแยกประเภทข้อความ (Classification) ให้สามารถจำแนกกลุ่มข้อความให้ดีขึ้นด้วยวิธีการเสริมข้อความของกลุ่มที่มีจำนวนน้อยให้มีจำนวนใกล้เคียงกับกลุ่มที่มีจำนวนมาก จนนำมาสู่การสร้างโมเดล เพื่อจำแนกประเภทข้อความด้วยวิธีการเรียนรู้เชิงลึกสำหรับการจำแนกข้อความทั่วไปหรือข้อความอุบัติการณ์ให้มีความถูกต้องเฉลี่ยไม่น้อยกว่า 95 เปอร์เซ็นต์ และเพื่อสร้างโมเดล เพื่อแยกประเภทข้อความด้วยวิธีการเรียนรู้เชิงลึก สำหรับการจำแนกประเภทข้อความออกเป็นข้อความอุบัติการณ์ที่ถูกแบ่งเป็น 5 ประเภท คือ ข้อความการจราจร ข้อความอุบัติเหตุ ข้อความเกี่ยวกับภัยพิบัติ ข้อความกลุ่มผู้ชุมนุมบนถนน ข้อความปิดช่องทางซ่อมถนน ให้มีความถูกต้องเฉลี่ยไม่น้อยกว่า 90 เปอร์เซ็นต์ โดยสรุปงานวิจัยครั้งนี้มีวัตถุประสงค์หลัก ๆ ดังนี้

- เพื่อศึกษาหาวิธีการสร้างข้อความให้เพิ่มมากขึ้น
- เพื่อนำข้อมูลที่สร้างมาเทรนโมเดลการจำแนกข้อความอุบัติการณ์
- เพื่อสร้างโมเดลการจำแนกประเภทข้อความรายงานสภาพจราจรจากทวิตเตอร์

1.3 สมมุติฐานของการศึกษา

ความไม่เท่ากันของข้อความหลายประเภท (Multiple class) ทำให้การสร้างโมเดลเพื่อจำแนกประเภทข้อความมีผลลัพธ์ที่มีความเอนเอียง (bias หรือไบแอส) ไปทางข้อความที่มีจำนวนมากกว่าในทางตรงข้ามกลับทำให้การระบุข้อความที่มีจำนวนน้อยไม่สามารถระบุได้อย่างแม่นยำ

จากปัญหาข้างต้นถ้าทำให้ข้อความหลายประเภทมีจำนวนเท่า ๆ กันแล้วนำข้อความนั้นมาสร้างโมเดลเพื่อการจำแนกข้อความจะสามารถทำให้ผลลัพธ์การระบุประเภทข้อความมีประสิทธิภาพที่ดีขึ้น

1.4 ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย

กรอบแนวความคิดของงานวิจัยนี้เน้นการสร้างระบบสำหรับจำแนกประเภทข้อความบนทวิตเตอร์ ในขั้นแรก ผู้วิจัยใช้วิธีการสร้างโมเดลด้วยวิธีการซีเอ็นเอ็น และการผสมผสานระหว่างเทคนิคซีเอ็นเอ็นและแอลเอสทีเอ็ม พบว่าการใช้วิธีการเหล่านี้ทำให้ค่าความแม่นเพิ่มขึ้นเล็กน้อยจากงานวิจัยก่อนหน้า แต่เมื่อสังเกตุที่ค่าเอฟวัน พบว่าในแต่ละกลุ่มมีค่าที่แตกต่างอย่างมาก เช่น ค่าเอฟวันของกลุ่มที่มีจำนวนมากจะมีคะแนนที่สูง ในขณะที่ค่าค่าเอฟวันของกลุ่มที่มีจำนวนน้อยจะมีคะแนนที่ต่ำ สถานการณ์เช่นนี้เป็นสัญญาณที่ว่าโมเดลมีความเอนเอียงไปทางกลุ่มที่มีขนาดใหญ่กว่า ซึ่งเรียกว่าไบแอส การเอนเอียงนี้จะทำให้โมเดลไม่สนใจข้อความของกลุ่มที่มีจำนวนน้อย

เพื่อแก้ไขปัญหานี้ ผู้วิจัยใช้วิธีการเพิ่มข้อความของกลุ่มที่มีจำนวนน้อยโดยการสร้างประโยคด้วยวิธีการมาร์คอฟโมเดล โดยใช้คำในคอร์ปัสมาเป็นข้อความพื้นฐาน อีกวิธีหนึ่งคือการสร้างข้อความด้วยวิธีการเรียนรู้เชิงลึกด้วยเทคนิคแอลเอสทีเอ็มซึ่งเป็นการสร้างแบบจำลองโมเดลจากการทำนายคำถัดไปโดยอ้างอิงคำก่อนหน้าในประโยค

วิธีการเหล่านี้สามารถสร้างจำนวนข้อความได้มากมายและเมื่อนำข้อมูลที่ถูกสร้างรวมกับกลุ่มข้อมูลเดิม จะทำให้จำนวนของแต่ละกลุ่มมีจำนวนใกล้เคียงกัน แต่เมื่อสังเกตุความหมายของประโยคที่ถูกสร้างขึ้นนั้นจะพบว่ามิได้มีความหมาย ไม่สามารถอ่านและเข้าใจได้เป็นเพียงนำคำมาต่อ ๆ กันจนเกิดประโยค จึงมีแนวคิดดำเนินการงานวิจัยขั้นต่อมาคือการเสริมข้อความเดิมให้มีจำนวนมากขึ้น โดยการปรับเปลี่ยนคำในบางคำให้มีความแตกต่างกัน แต่ยังคงรูปแบบของข้อความเดิมเพื่อให้ความหมายของประโยคไม่เปลี่ยนไป โดยคำที่นำมาเสริมจะใช้วิธีการหาคำที่มีความหมายคล้ายกันมาแทนที่ โดยผู้วิจัยได้ออกแบบกรอบความคิดไว้ดังนี้ เริ่มอินพุทข้อมูลเข้ามาเพื่อทำการเตรียมข้อมูล (Prepare) ประกอบไปด้วยการทำชื่อกลุ่ม (Label หรือเลเบล) การตัดประโยคออกเป็นคำและการลบคำฟุ่มเฟือย จากนั้นเริ่มต้นจัดการกับความไม่สมดุลด้วยการสร้างข้อความเพิ่มและการเสริมคำเพื่อสร้างข้อความเพิ่ม ต่อด้วยการหาลักษณะ (Feature) ของข้อมูล และสุดท้ายนำข้อความที่พร้อมมาเทรนโมเดลการจำแนกข้อความ โดยอธิบายแนวความคิดการทำงานไว้ดังรูปที่ 1.1

รูปที่ 1.1 ภาพรวมกรอบแนวความคิดของงานวิจัย

1.5 ขอบเขตการวิจัย

1.5.1 ใช้ข้อความบนทวิตเตอร์จากบัญชีทางการที่เกี่ยวข้องกับการรายงานสภาพจราจรของประเทศไทยจำนวน 4 บัญชี ในการเก็บข้อมูลระหว่างเดือนกุมภาพันธ์ถึงเดือนเมษายน พ.ศ. 2566

1.5.2 ดำเนินการออกแบบและพัฒนารูปแบบการเสริมข้อมูลให้มีจำนวนใกล้เคียงกันทุกกลุ่ม ซึ่งมีการแบ่งกลุ่มออกเป็น 2 กลุ่มคือข้อความข่าวทั่วไปและข้อความข่าวที่เกี่ยวกับสภาพจราจร และแบ่งกลุ่มข้อความข่าวที่เกี่ยวกับสภาพจราจรออกเป็น 5 กลุ่มคือ ข้อความการจราจร ข้อความอุบัติเหตุ ข้อความเกี่ยวกับภัยพิบัติ ข้อความกลุ่มผู้ชุมนุมบนถนน ข้อความปิดช่องทางซ่อมถนน

1.5.3 ประมวลผลบนเครื่องคอมพิวเตอร์แมคบุ๊ค มีหน่วยประมวลผลกลาง Intel Core i7 2.6 GHz 6-Core และหน่วยความจำหลัก 16 GB 2400 MHz DDR4

1.6 ขั้นตอนของการศึกษา

1.6.1 ศึกษางานวิจัยหรือบทความที่เกี่ยวข้องกับการจำแนกข้อความทวิตเตอร์ที่เกี่ยวกับการจราจรทางบกทั้งภาษาไทยและภาษาอังกฤษ

1.6.2 ศึกษางานวิจัยหรือบทความที่เกี่ยวกับการสร้างและเสริมข้อความให้มีจำนวนมากขึ้น

1.6.3 สร้างระบบสำหรับการดึงข้อมูลจากทวิตเตอร์เข้ามาเก็บเป็นข้อมูลสำหรับการสร้างโมเดล

1.6.4 จัดทำลาเบลของข้อความแต่ละข้อความที่ได้จากการเก็บรวบรวมมาจากทวิตเตอร์ด้วยมือ โดยจะจัดทำลาเบลเป็น 2 ขั้นตอนคือ

1.6.4.1 จัดทำลาเบลของข้อความที่เป็นข้อความอุบัติการณ์จราจรกับข้อความที่เป็นเหตุการณ์ทั่วไป

1.6.4.2 จัดทำลาเบลของข้อความที่เป็นข้อความประเภทของอุบัติการณ์ ประกอบด้วย 5 ประเภท

1.6.5 ขั้นตอนการเตรียมข้อความ (Prepare data) ก่อนการสร้างโมเดล มีขั้นตอนดังนี้

1.6.5.1 ตัดประโยคออกเป็นคำ

1.6.5.2 กรองอักขระพิเศษออกจากข้อความ เช่น “# (แฮชแท็ก), URL, Emoji ออกไป

1.6.5.3 ขั้นตอนการเปลี่ยนคำออกเป็นตัวเลขแบบลำดับ (Text to sequence)

1.6.5.4 ขั้นตอนการทำให้ข้อความทุกข้อความมีความยาวเท่ากัน

1.6.6 การสร้างหรือเสริมข้อความที่มีจำนวนน้อยให้เพิ่มขึ้นจนมีจำนวนใกล้เคียงกับข้อความที่มีจำนวนมาก

1.6.7 ขั้นตอนการทดสอบเพื่อหาวิธีที่ดีที่สุดสำหรับการเสริมข้อความและดำเนิการสร้างข้อความมากขึ้น

1.6.8 ขั้นตอนการสร้างโมเดลสำหรับการจำแนกข้อความประกอบด้วยโมเดลประเภทซีเอ็นเอ็นผสานแอลเอสทีเอ็ม แล้วทดสอบเพื่อหาว่าข้อความจากการสร้างโมเดลด้วยวิธีการเสริมข้อความแบบใดให้ค่าความแม่นยำและค่าเอฟวันดีที่สุด

1.7 คำจำกัดความที่ใช้ในการศึกษา

ในงานวิจัยครั้งนี้มีการใช้คำศัพท์เฉพาะทางซึ่งมีหลายคำศัพท์หรือคำย่อที่มีความหมายได้หลายความหมาย ดังนั้นเพื่อประโยชน์ในการศึกษางานวิจัยนี้ ผู้วิจัยได้สรุปนิยามของคำศัพท์เฉพาะมาดังตารางที่ 1.1

ตารางที่ 1.1 ตารางอภิธานคำศัพท์

คำศัพท์ภาษาอังกฤษ	**คำศัพท์ภาษาไทย**	**คำศัพท์ที่ใช้ในเล่มนี้**
Accuracy	ความแม่นยำ	แอคคิวเรซี
Augmentation	การเสริมคำ	ออกเมนเทชัน
BERT Score	ค่าคะแนนแบบเบิร์ต	เบิร์ตสกอร์
BLEU Score	ค่านะแนนแบบเบลอ	เบลอสกอร์
Word embedding	การฝังคำศัพท์	เวิร์ดเอ็มเบ็ดดิง
Bidirectional Encoder Representations from Transformers: Bert Pre-trained	การสอนล่วงหน้าแบบเบิร์ต	เบิร์ตพรีเทรน
Corpus	คลังข้อความ	คอร์ปัส

ตารางที่ 1.1 ตารางอภิธานคำศัพท์ (ต่อ)

คำศัพท์ภาษาอังกฤษ	**คำศัพท์ภาษาไทย**	**คำศัพท์ที่ใช้ในเล่มนี้**
Convolutional Neural Network (CNN)	โครงข่ายประสาทแบบคอนโวลูชัน	ซีเอ็นเอ็น
Datasets	ชุดข้อมูล	เดต้าเซ็ต
Imbalance Ratio (IR)	อัตราส่วนความไม่สมดุล	ไออาร์
Long Short-Term Memory (LSTM)	วิธีหน่วยความจำระยะสั้นระยะยาว	แอลเอสทีเอ็ม
Natural Language Processing (NLP)	การประมวลผลทางภาษาธรรมชาติ	เอ็นแอลพี
Pre-trained	การสอนไว้ล่วงหน้า	พรีเทรน
platform	ระบบพื้นฐานที่ให้บริการหรือสนับสนุน	แพลตฟอร์ม
Recurrent Neural Network (RNN)	นิวรัลเน็ตเวิร์กแบบวนกลับ	อาร์เอ็นเอ็น
social media	สื่อสังคมออนไลน์	โซเชียลมีเดีย

บทที่ 2
ทฤษฎีพื้นฐานที่เกี่ยวข้อง

ในขั้นตอนการศึกษาทฤษฎีที่เกี่ยวข้องกับการพัฒนาระบบการจำแนกข้อความพบว่ามีการทำงานที่เป็นขั้นตอนหลักสำคัญต่าง ๆ เช่น การรวบรวมข้อมูล การเตรียมข้อมูล การสร้างแบบจำลอง การทดสอบและวัดผล แต่จะมีบางส่วนหรือบางหัวข้อที่แตกต่างกันในแต่ละรายละเอียดของงาน ในบทนี้ผู้วิจัยจะเน้นนำเสนอในส่วนของทฤษฎีที่นำมาใช้เพื่อพัฒนากระบวนการจำแนกข้อความจากทวิตเตอร์ให้ได้ผลตามวัตถุประสงค์และสมุติฐานที่ได้ตั้งไว้

2.1 โซเชียลมีเดีย

โซเชียลมีเดียเป็นแพลตฟอร์ม (Platform) หรือเว็บไซต์ที่ช่วยให้ผู้คนสามารถแลกเปลี่ยนข้อมูล แชร์เรื่องราว ติดตามข่าวสาร และสร้างความสัมพันธ์กับผู้อื่นผ่านการสื่อสารแบบออนไลน์ อย่างเช่น เฟซบุ๊ก (Facebook), ทวิตเตอร์ (Twitter), อินสตาแกรม (Instagram), ไลน์ (Line), ยูทูป (YouTube) และอื่น ๆ การสื่อสารบนแพลตฟอร์มเหล่านี้เป็นไปในรูปแบบข้อความ ภาพถ่าย วิดีโอ หรือเสียง และส่วนใหญ่มีการแสดงความคิดเห็นหรือประกาศข่าวจากผู้ใช้งานอื่น ๆ ภายใต้โพสต์หรือกระทู้ต่าง ๆ

ทวิตเตอร์เป็นแพลตฟอร์มสื่อสังคมออนไลน์ที่เน้นการโพสต์ข้อความสั้นๆ ที่เรียกว่าทวีต (tweet) โดยทวีตสามารถมีขนาดไม่เกิน 280 ตัวอักษร ผู้ใช้งานแต่ละคนสามารถติดตามผู้ใช้งานอื่นเพื่อเข้าใจข่าวสาร แชร์ความคิดเห็น และร่วมสนทนาในรูปแบบที่เรียกว่า “เทรนด์” (trends) ทวิตเตอร์เป็นแพลตฟอร์มที่ได้รับความนิยมมากในหลายประเทศทั่วโลก และในประเทศไทยมีจำนวนผู้ใช้งานเพิ่มขึ้นอย่างต่อเนื่องโดยในปี 2566 มีผู้ใช้ทวิตเตอร์ราว 14.6 ล้านผู้ใช้งานซึ่งส่วนใหญ่ใช้เพื่อการติดตามข่าวสาร ดังนั้น ทวิตเตอร์จึงมีบทบาททางการแจ้งข่าวสารเป็นอย่างมากเนื่องจากการนำเสนอข่าวแบบสั้นๆ ร่วมกับแท็ก (tag) ตามเหตุการณ์รายวัน นอกจากนี้ หน่วยงานราชการที่เกี่ยวข้องกับสภาพจราจรต่างๆ เช่น สถานีวิทยุจราจร สำนักงานตำรวจจราจร รวมถึงหน่วยงานจราจรอื่นๆ สามารถใช้ทวิตเตอร์ในการรายงานสภาพจราจร สื่อสารกับประชาชน และแจ้งข่าวสารเกี่ยวกับการจราจรได้เช่นกัน

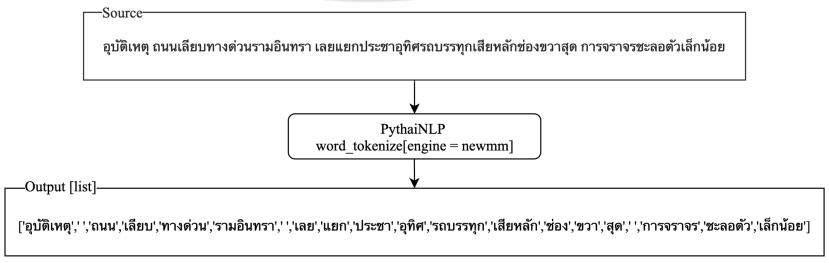
2.2 การเตรียมข้อมูล

การเตรียมข้อมูลเพื่อสร้างโมเดลการเรียนรู้เชิงลึก สำหรับการจำแนกข้อความจากทวิตเตอร์ที่เกี่ยวกับสภาพจราจรบนถนนนั้นเป็นขั้นตอนที่สำคัญเพื่อให้โมเดลที่สร้างมีประสิทธิภาพและสามารถ

ทำงานได้ดี โดยเฉพาะอย่างยิ่งข้อความที่อยู่บนสื่อสังคมออนไลน์นั้นมีผู้ใช้งานหลายประเภท มีการใช้สัญลักษณ์ต่าง ๆ ที่ไม่สามารถนำมาทำเอ็นแอลพีได้หรือแม้แต่การใช้คำฟุ่มเฟือยเยอะเกินไป ซึ่งด้วยภาษาไทยเป็นภาษาที่ไม่มีการเว้นวรรคคำทำให้ยิ่งเพิ่มความยุ่งยากมากขึ้น ด้วยเหตุผลนี้การเตรียมข้อมูลคือการทำให้ข้อมูลนั้นสะอาด สามารถนำไปทำโมเดลได้ง่ายส่งผลให้เพิ่มประสิทธิภาพของการจำแนกข้อความได้มากขึ้น โดยการเตรียมข้อมูลจะมีขั้นตอนหลัก ๆ ดังเนื้อหาต่อไปนี้

2.2.1 การตัดประโยคออกเป็นคำ

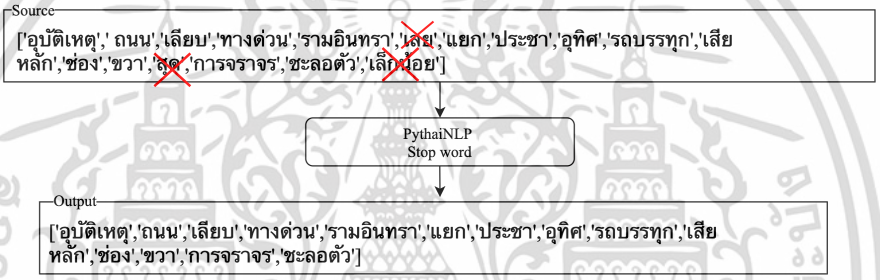
การทำงานในด้านการประมวลผลภาษาธรรมชาติจะมีกระบวนการที่เป็นส่วนเริ่มต้นของการทำงานหลังจากการรวบรวมคำหรือข้อความมาได้แล้ว ขั้นตอนต่อมาคือการแบ่งประโยคนั้นออกมาเป็นคำหรือการตัดคำ (Word segmentation หรือเวิร์ด-เซ็กเมนเทชัน) ขั้นตอนนี้สำคัญมากเนื่องจากภาษาไทยเป็นภาษาที่เขียนติดกันไม่มีตัวขั้นระหว่างคำหรือประโยค การแบ่งคำช่วยให้สามารถนับจำนวนคำ หาความถี่ของคำ เลือกคุณลักษณะของคำ เช่น คำหลัก และสร้างรูปแบบการเขียนข้อความ เพื่อใช้ในการวิเคราะห์หรือการจัดหมวดหมู่ข้อความ ถ้าตัดคำออกมาไม่ถูกต้องจะทำให้การประมวลผลคำผิดเพี้ยนไป เช่น “คุณอาจองนั่งตากลม” อาจจะตัดเป็น “คุณ/อา/จอง/นั่ง/ตาก/ลม” หรือ “คุณ/อาจ/อง/นั่ง/ตา/กลม” วิธีการตัดคำสามารถใช้กฎความสัมพันธ์ระหว่างคำหรือใช้โมเดลเชิงลึกซึ่งถูกฝึกสอนด้วยข้อมูลภาษาเพื่อตัดคำ ตัวอย่างวิธีการตัดคำที่ใช้ในภาษาไทยได้แก่ Maximum Matching, Longest Matching, Conditional Random Fields (CRF) และโมเดลปัญญาประดิษฐ์ (Artificial Intelligence) ที่ถูกฝึกสอนด้วยข้อมูลภาษาไทยอื่น ๆ การตัดคำสำหรับภาษาไทยในปัจจุบันมีชุดคำสั่ง (Library หรือไลบรารีที่ชื่อว่าไพไทยเอ็นแอลพี (Pythainlp) [11] เป็นไลบรารีที่ทำงานบนภาษาไพธอน (Python) ซึ่งในไพไทยเอ็นแอลพีใช้การตัดคำพื้นฐาน (default) เป็นวิธีการนิวเอ็มเอ็ม (newmm engine) ที่พัฒนามาจากเทคนิคการแบ่งส่วนคำศัพท์ภาษาไทยด้วยพจนานุกรม (Dictionary-based) โดยใช้อัลกอริทึมการจับคู่สูงสุด (Maximum Matching) และการจัดกลุ่มของอักขระภาษาไทย (Thai Character Cluster : TCC) ทำงานร่วมกันทำให้การตัดคำภาษาไทยถูกต้องและรวดเร็วมากขึ้น



รูปที่ 2.1 การตัดประโยคออกเป็นคำ

2.2.2 การลบคำฟุ่มเฟือย

สำหรับภาษาไทยมักจะมีคำที่ไม่ค่อยสื่อความหมายหรือไม่จำเป็นสำหรับการประมวลผลภาษาธรรมชาติ เช่น “มี, การ, ได้, ความ” ถือเป็นคำฟุ่มเฟือยเมื่อนำคำเหล่านั้นไปใช้อาจจะทำให้ประมวลผลใช้เวลานานเกินความจำเป็น เนื่องจากเป็นคำที่ไม่ค่อยมีความหมายในการสื่อสารทำให้สิ้นเปลืองทรัพยากรสำหรับการประมวลโดยเปล่าประโยชน์ โดยในทางการประมวลผลภาษาธรรมชาติคำฟุ่มเฟือยเหล่านี้ถูกเรียกว่าคำหยุด (stop word) เพื่อต้องการลบคำหยุดเหล่านั้นให้น้อยลงหรือหมดไปด้วยการใช้ฟังก์ชันที่มีในไพไทยเอ็นแอลพี อย่างไรก็ตามการลบคำหยุดนั้นอาจจะไม่ได้เป็นข้อกำหนดหรือเงื่อนที่ต้องทำ แต่การตัดสินใจที่จะใช้หรือไม่ใช้การลบคำหยุดอาจจะแตกต่างกันในแต่ละบริบท ดังนั้นผู้วิจัยควรวิเคราะห์ถึงความจำเป็นก่อน ซึ่งไพไทยเอ็นแอลพีมีการเตรียมชุดคำศัพท์ของคำหยุดไว้ให้ใช้งาน



รูปที่ 2.2 การลบคำฟุ่มเฟือย

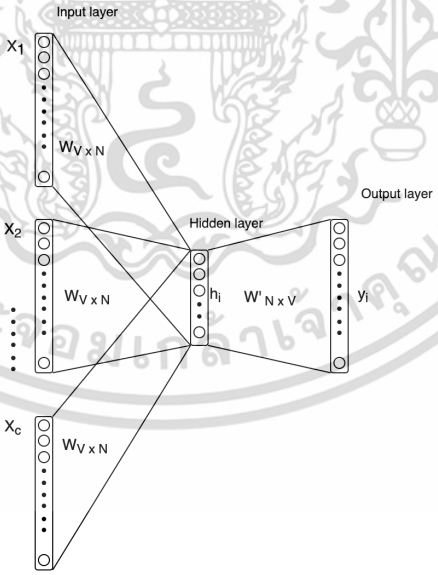
2.2.3 การเปลี่ยนคำให้อยู่ในรูปแบบเวคเตอร์

การฝังคำ (Word embedding หรือเวิร์ดเอ็มเบ็ดดิง) เป็นวิธีการในการแปลงคำให้อยู่ในรูปแบบที่สามารถนำมาประมวลผลเชิงคณิตศาสตร์ได้ โดยมีวัตถุประสงค์หลักในการสร้างตัวแทน (Representation) ที่แทนความหมายของคำในรูปแบบของเวคเตอร์ที่มีมิติต่ำ (low-dimensional vectors) ซึ่งสามารถนำมาใช้ในการประมวลผลข้อมูลขนาดใหญ่ได้อย่างมีประสิทธิภาพ เช่น การค้นหาความสัมพันธ์ระหว่างคำ การหาคำที่คล้ายคลึงกัน หรือการจัดกลุ่มคำตามหมวดหมู่ต่าง ๆ เทคนิคของเวิร์ดเอ็มเบ็ดดิงนั้นมักจะใช้โมเดลการเรียนรู้ทางเครื่องเพื่อสร้างความสัมพันธ์ระหว่างคำและหาเวคเตอร์แทนคำ โดยมักจะใช้ข้อมูลขนาดใหญ่ที่เป็นประโยคหรือเอกสารมาใช้ในกระบวนการเรียนรู้เมื่อคำถูกแปลงให้อยู่ในรูปแบบเวิร์ดเอ็มเบ็ดดิงแล้ว สามารถใช้เวคเตอร์ของคำนั้นในการพยากรณ์หรือวิเคราะห์ข้อมูลต่าง ๆ ได้ เช่น การจัดกลุ่มข้อความ (text clustering) หรือการจัดลำดับความสำคัญของคำในข้อความ (keyword extraction) เทคนิคเวิร์ดเอ็มเบ็ดดิงมีรายละเอียดทางทฤษฎีและการทำงานที่ดี โดยในระยะหลังมีงานวิจัยหลายงานได้ให้ความสำคัญกับวิธีการเวิร์ดทูเว็ค (Word2Vec) ในการเวิร์ดเอ็มเบ็ดดิง

ซึ่งมีวิธีการด้วยกัน 2 วิธีคือสคิป-แกรม (Skip-gram) และวิธีการกระเป๋าคำต่อเนื่อง (Continuous Bag of Words: CBOW หรือซีบีโอดับเบิ้ลยู)

- สคิป-แกรม คือ วิธีการที่เน้นการทำนายโดยใช้คำที่อยู่ตรงกลางเพื่อทำนายคำที่อยู่รอบๆ เช่น หากมีคำว่า “รถชน" และคำในบริบทที่อยู่รอบ ๆ คำ “รถชน” เป็น “บริเวณ”, “เหตุการณ์”, “ถนน” โมเดลสคิป-แกรม จะพยายามทำนายคำเหล่านี้จากคำว่า “บริเวณ”, “เหตุการณ์”, “ถนน” เพื่อสร้างเวคเตอร์ที่สอดคล้องกับความหมายและความสัมพันธ์ระหว่างคำ
- ซีบีโอดับเบิ้ลยู คือ วิธีการที่กลับกันไปในทิศตรงกันข้าม โดยซีบีโอดับเบิ้ลยูจะพยายามทำนายคำศัพท์หนึ่ง โดยใช้คำที่ปรากฏในบริบทที่กำหนด ตัวอย่างเช่น หากมีคำว่า "ถนน", "บริเวณ", "เหตุการณ์" บีโอดับเบิ้ลยูจะทำนายคำว่า “รถชน” ออกมา

ทั้งสองวิธีการนี้จะใช้โครงสร้างโครงข่ายประสาทเทียม (neural network) เพื่อเรียนรู้ความสัมพันธ์ระหว่างคำโดยโครงข่ายประสาทประกอบด้วยเลเยอร์นำเข้า (input layer) เลเยอร์ซ่อน (hidden layer) และเลเยอร์ผลลัพธ์ (output layer) โดยในแต่ละเลเยอร์จะมีจำนวนโหนด (neuron) ที่เป็นเวคเตอร์แสดงค่าการเรียนรู้ของโมเดล แต่จะเลือกใช้วิธีการใดนั้นขึ้ยอยู่กับงานที่ต้องการ เช่นงานสำหรับการสร้างเวิร์ดเอ็มเบ็ดดิงจากข้อความที่มีจำนวนไม่มากนักควรจะใช้วิธีซีบีโอดับเบิ้ลยูเนื่องจากใช้การประมวลผลไม่มากเป็นวิธีการที่เหมาะข้อมูลน้อย



รูปที่ 2.3 โครงสร้างของวิธีการกระเป๋าคำต่อเนื่อง

จากรูปที่ 2.3 สามารถอธิบายได้ดังนี้ เลเยอร์นำเข้าสำหรับบริบทของคำ C คำที่จะถูกแปลงเป็น one-hot vectors $x_1, x_2, \dots, x_C$ เลเยอร์ซ่อน คือ ชั้นบริบทที่แปลงเป็น one-hot vectors จะถูกนำไปคูณด้วย weight matrix W ซึ่งมีขนาด $V \times N$ (V คือจำนวนของคำศัพท์และ N คือขนาดของเลเยอร์ซ่อน) ดังสมการที่ 2.1

$$h = \frac{1}{C}\sum_{i=1}^{C} W x_i \tag{2.1}$$

และเลเยอร์ผลลัพธ์ คือชั้นของการสร้างเวคเตอร์ของคำที่เราต้องการทำนาย โดยใช้ weight matrix W' ที่มีขนาด $N \times V$ และใช้ SoftMax เพื่อแปลงเวคเตอร์ h ให้เป็นความน่าจะเป็นของคำ [18]

ผลลัพธ์ของเวิร์ดทูเว็คเป็นเวคเตอร์ที่สามารถแสดงความหมายและความสัมพันธ์ระหว่างคำได้อย่างมีประสิทธิภาพ โดยเวคเตอร์ของคำที่มีความหมายคล้ายกันจะมีค่าที่ใกล้เคียงกันในเวคเตอร์สเปซ ทำให้สามารถใช้เวคเตอร์นี้เป็นค่าตัวเลขเพื่อแทนค่าของคำซึ่งก็คือการแปลงเป็นตัวเลข มิติของเวคเตอร์นั้นขึ้นอยู่กับขนาดของ “hidden layer” ในโมเดลที่ใช้ ซึ่งสามารถตั้งค่าได้ในขณะที่สร้างโมเดล ในการปฏิบัติงานจริง ขนาดของเวคเตอร์มักจะเป็นค่าระหว่าง 100-300 มิติ ขึ้นอยู่กับความซับซ้อนของงานและขนาดความยาวของข้อมูล เช่นงานการจัดกลุ่มข้อความมักจะใช้จำนวนความยาวสูงสุดของข้อความมาเป็นมิติของคำ โดยในงานวิจัยนี้ได้ใช้ไลบรารี (Library) ที่ชื่อ Gensim หรือเจ็นซิม เข้ามาช่วยในการสร้างการฝังคำ

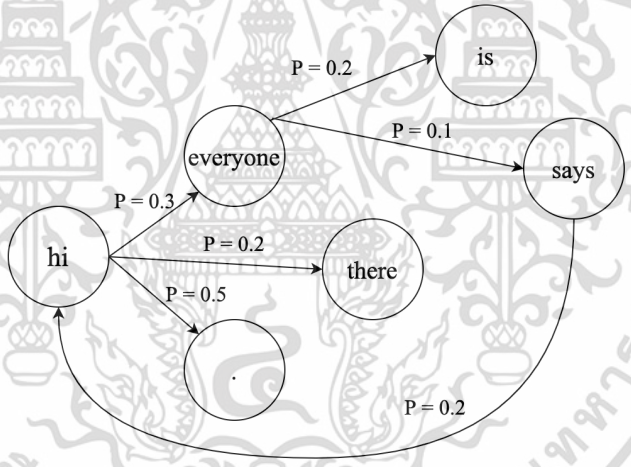
2.3 การจัดการข้อมูลที่จำนวนประเภทไม่เท่ากัน

จากการศึกษางานวิจัยหลายๆ งานที่เกี่ยวกับการจำแนกข้อความสั้นที่ต้องมีการระบุออกเป็นหลายกลุ่มพบว่ามีบางงานวิจัยที่ใช้ตัวอย่างข้อมูลมาเทรนโมเดลด้วยข้อความที่มีจำนวนไม่เท่ากัน ปัญหานี้มักจะก่อให้ผลการทำงานเอนเอียงไปทางกลุ่มที่มีจำนวนข้อความมากกว่าหรือเรียกว่าไบแอส (Bias) ด้วยเหตุนี้จึงมีการวิจัยเพื่อสร้างโมเดลให้มีประสิทธิภาพเพิ่มขึ้นด้วยการสร้างกลุ่มข้อมูลให้มีขนาดเท่ากันก่อนการนำมาเทรนโมเดล โดยในแต่ละงานวิจัยจะใช้วิธีที่แตกต่างกันออกไป สำหรับการจัดการข้อความที่ไม่เท่ากันสามารถสรุปวิธีการที่นิยมได้ ดังนี้

2.3.1 การสร้างข้อความด้วยวิธีตัวแบบลูกโซ่มาร์คอฟ

การสร้างข้อความด้วยวิธีลูกโซ่มาร์คอฟเป็นเทคนิคหนึ่งที่ใช้ในการสร้างข้อความที่มีลักษณะเป็นภาษาธรรมชาติ โดยอิงจากข้อมูลที่มีอยู่และสถิติการเกิดของคำหรือสัญลักษณ์ต่างๆ ในข้อมูลนั้น วิธีการนี้

ถูกนำมาใช้ในหลายด่านโดยมีการนำไปสร้างข้อความที่เกี่ยวข้องกับภาษาธรรมชาติ เช่น การสร้างชื่อสินค้าหรือข้อความสั้นๆ ในการสร้างข้อความด้วยวิธี ลูกโซ่มาร์คอฟ มีขั้นตอนการทำงานหลักๆ ดังนี้ เริ่มต้นด้วยการรวบรวมข้อมูลที่เป็นตัวอย่างข้อความที่ต้องการสร้าง อาจเป็นตัวอักษรหรือคำ และจัดเก็บในรูปแบบที่เหมาะสม ให้อยู่ในรูปแบบของลิสต์ (list) หรือเมทริกซ์ (Matrix) จากข้อมูลที่เตรียมไว้เพื่อสร้างชุดของแต่ละคำที่บอกถึงความน่าจะเป็นในการเกิดข้อความต่อไป โดยจะนับความถี่ของคำหรือสัญลักษณ์ที่ตามหลังคำก่อนหน้า และเก็บเป็นสถิติการเกิดไว้ในเมทริกซ์เดียวกันจากนั้นจะสร้างโมเดลลูกโซ่มาร์คอฟ ที่ใช้สร้างข้อความ โดยการเลือกคำเริ่มต้นแบบสุ่มตามความน่าจะเป็นในเมทริกซ์จากนั้นจึงสร้างข้อความใหม่โดยเลือกคำต่อไปตามลำดับซึ่งสามารถกำหนดการสร้างข้อความได้โดยการสร้างคำต่อไปตามลำดับของโมเดลจนกว่าจะได้ข้อความที่ต้องการ โดยทั้งหมดนี้จะเป็นกระบวนการที่วนซ้ำกันไปเรื่อย ๆ ตามความยาวของข้อความที่ต้องการสร้าง สิ่งที่ต้องระวังคือ ข้อมูลที่ใช้สร้างเมทริกซ์ต้องเป็นตัวอย่างที่เป็นมาตรฐานและเกี่ยวข้องกับข้อความที่ต้องการสร้าง มิฉะนั้น ข้อความที่สร้างขึ้นอาจไม่มีความสมเหตุสมผลหรือแปลกประหลาดได้



รูป 2.4 ลูกโซ่มาร์คอฟการสร้างข้อความ

การสร้างลูกโซ่มาร์คอฟสำหรับการสร้างข้อความ มักจะใช้สมการความน่าจะเป็นแบบมีเงื่อนไข (conditional probabilities) ในรูปแบบเมทริกซ์ สมมติ $S = \{s_1, s_2, \dots, s_n, \}$ เป็นเซ็ตของ “สถานะ” หรือคำในข้อความและ $P(s_i \to s_j)$ คือความน่าจะเป็นที่จะเปลี่ยนไปยังสถานะ s_j จากสถานะ s_i และสมการมาร์คอฟสำหรับสถานะ s_i จะเป็น

$$P(X_{t+1} = s_j | X_t = s_i) = P(s_i \to s_j) \tag{2.2}$$

X_t คือสถานะที่เวลา t

$P(X_{t+1} = s_j | X_t = s_i)$ คือความน่าจะเป็นที่สถานะที่ $t+1$ จะเป็น s_j เมื่อสถานะที่ t จะเป็น s_i ความน่าจะเป็น $P(s_i \rightarrow s_j)$

ในลูกโซ่มาร์คอฟจะถูกคำนวณจากข้อความที่มีอยู่ ดังนี้ นับความถี่ขอคำทั้งหมด (Frequency Counting) จากนั้นจะนับความถี่ของการเกิดของคู่คำที่ติดกัน เช่น ถ้าคำว่า “apple” ถูกตามด้วย “pie” 5 ครั้ง และถูกตามด้วย “tree” 3 ครั้ง จะได้ค่าความถี่ของคู่คำเหล่านั้น จากนั้นคำนวณความน่าจะเป็น (Probability Calculation) ในขั้นตอนนี้จะนำความถี่ของคู่คำไปคำนวณความน่าจะเป็น ดังสมการ

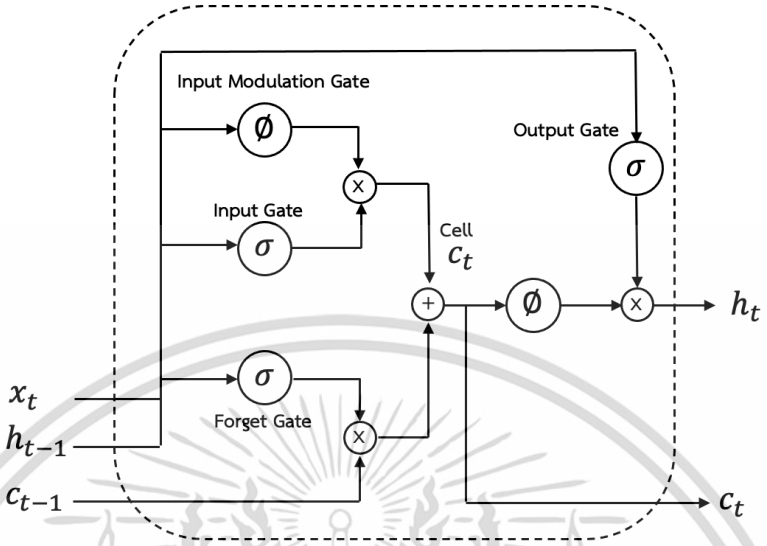
$$P(s_i \rightarrow s_j) = \frac{Frequency(s_i \rightarrow s_j)}{Total\ Frequency(s_i)} \quad (2.3)$$

ที่ $Frequency(s_i \rightarrow s_j)$ คือจำนวนครั้งที่ s_i ถูกตามด้วย s_j

และ $Total\ Frequency(s_i)$ คือจำนวนครั้งทั้งหมดที่ s_i ปรากฏในข้อความโดยจะทำการคำนวณค่านี้สำหรับทุกคู่คำที่ติดกัน

2.3.2 การสร้างข้อความเพิ่มด้วยวิธีแอลเอสทีเอ็ม

แอลเอสทีเอ็ม คือหนึ่งในโครงสร้างของโครงข่ายประสาทเทียมที่เกิดซ้ำ (Recurrent Neural Networks : RNN หรืออาร์เอ็นเอ็น) ที่ถูกออกแบบมาเพื่อรับมือกับปัญหาของอาร์เอ็นเอ็นแบบธรรมดา เช่น ปัญหาการหายไปของเกรเดียน (vanishing gradient) และการเพิ่มขึ้นของเกรเดียน (exploding gradient) โดยแอลเอสทีเอ็มจะทำงานผ่านเซล (cell) ต่างๆ แต่ละเซลของแอลเอสทีเอ็มประกอบด้วยส่วนประกอบหลัก ๆ คืออินพุตเกท (input gate) ฟอร์เกตเกท (forget gate) เอาต์พุตเกท (output gate) และเซลล์สเตต (cell state) โดยแต่ละเกทมีหน้าที่ต่างกันออกไปคือ



รูปที่ 2.5 ส่วนประกอบภายในแอลเอสทีเอ็ม

2.3.2.1 อินพุตเกท

เกท (gate) หรือประตูนี้จะทำหน้าที่ตัดสินใจว่าส่วนไหนของข้อมูลปัจจุบันที่ x_t ควรถูกอัปเดตหรือเพิ่มเข้าไปในเซลล์สเตตหรือไม่ ถ้าอัปเดตจะใช้มอดดิวเลชันเกท (Modulation Gate) เป็นเกทตัดสินใจที่จะคอยตัดสินใจว่าส่วนไหนของข้อมูลใหม่ควรถูกเพิ่มเข้าสู่สถานะ cell ปัจจุบันและควบคุมว่าควรจะเพิ่มข้อมูลใหม่ลงไปใน cell state หรือไม่ประกอบไปด้วยสมการ

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.4}$$

โดยที่

i_t คือผลลัพธ์จากอินพุตเกท ณ เวลา t ผลลัพธ์นี้บ่งบอกถึงส่วนที่ควรเพิ่มเข้าไปในสถานะ cell ณ เวลา t

σ เป็น Sigmoid activation function ที่จะคืนค่าระหว่าง 0 และ 1 สำหรับ แอลเอสทีเอ็ม, มันใช้เพื่อตัดสินใจว่าจะรับข้อมูลใดบ้างเข้าไปในสถานะ cell หรือ output

W_i เป็นเมทริกซ์น้ำหนักสำหรับอินพุตเกท

h_{t-1} ข้อมูล output จาก timestep ก่อนหน้า

x_t ข้อมูลอินพุต ณ เวลา t

b_i เป็น bias term สำหรับ Input Gate

2.3.2.2 เซลล์สเตต

เป็นเซลล์สำหรับอัปเดตสถานะของเซลล์จากการรวมกันของ อินพุตเกท ฟอร์เกตเกท และมอดดิวเลชัน เพื่อให้แอลเอสทีเอ็มสามารถจดจำข้อมูลระยะไกลขึ้นโดยการพิจารณาว่าอับเดทเซลล์หรือไม่ ตามสมการต่อไปนี้

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{2.5}$$

โดยที่

C_t คือสถานะเซลล์ ณ เวลา t ซึ่งเป็นสิ่งที่ต้องการอัปเดต

f_t คือ output จาก Forget Gate บ่งบอกถึงส่วนของ C_{t-1} ที่ควรถูกลืม

C_{t-1} คือสถานะ cell ณ เวลา $t-1$ หรือสถานะ cell ก่อนหน้านี้

i_t คือ output จาก Input Gate บ่งบอกถึงส่วนของ $\tilde{C}_t$ ที่ควรถูกเพิ่มเข้าไป

$\tilde{C}_t$ คือสถานะ modulation cell state ที่ได้รับการอัปเดตโดยข้อมูลปัจจุบัน

2.3.2.3 มอดดิวเลชันเซลล์สเตต

มอดดิวเลชันเซลล์สเตต $(\tilde{C}_t)$ คือสถานะเซลล์ที่ถูกคำนวณใหม่จากข้อมูลปัจจุบัน (x_t) และสถานะ hidden ก่อนหน้านี้ (h_{t-1}) สถานะมอดดิวเลชันเซลล์สเตตนี้จะถูกคำนวณใหม่ทุกครั้งที่ได้รับข้อมูลปัจจุบัน (x_t) ค่าของ $\tilde{C}_t$ จะถูกนำไปเติมเข้าในสถานะเซลล์ปัจจุบันจากการคำนวณที่อธิบายไปแล้ว นั้นทำให้แอลเอสทีเอ็มสามารถจำข้อมูลใหม่ได้พร้อมทั้งยังรักษาข้อมูลระยะยาวได้ โดยใช้สมการ

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.6}$$

โดยที่

$\tilde{C}_t$ เป็นสถานะมอดดิวเลชันเซลล์สเตตที่คำนวณใหม่

W_c เป็นเมทริกซ์น้ำหนักสำหรับการคำนวณ $\tilde{C}_t$

h_{t-1} คือการรวมข้อมูล output จาก timestep ก่อนหน้า h_{t-1}

x_t ข้อมูลปัจจุบัน x_t

b_c เป็น bias term

$\tanh$ เป็น Hyperbolic Tangent Function ที่ใช้เพื่อให้ $\tilde{C}_t$ อยู่ในช่วง -1 ถึง 1

2.3.2.4 ฟอร์เกตเกท

มีหน้าที่ควบคุมข้อมูลในสถานะเซลล์ (C_t) ที่ควรถูกลืมหรือทิ้งไปในช่วงเวลาถัดไป โดยปกติจะใช้ฟังก์ชันสิกมอยด์ (sigmoid) เพื่อทำให้เอาต์พุทอยู่ในช่วง 0 ถึง 1 การคำนวณสามารถแสดงได้ดังสมการต่อไปนี้

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2.7}$$

โดยที่

f_t คือเอาต์พุทจากฟอร์เกตเกท ณ เวลา t

σ คือฟังก์ชันสิกมอยด์ (sigmoid) ที่จะทำให้เอาต์พุทอยู่ในช่วง 0 ถึง 1

W_f เป็นเมทริกซ์น้ำหนักสำหรับฟอร์เกตเกท

h_{t-1} คือ ผลลัพธ์ ณ เวลาก่อนหน้า และ

x_t ข้อมูลปัจจุบัน

b_f เป็น bias term

ค่า f_t ที่ได้จากฟอร์เกตเกทจะถูกใช้ในการอัปเดตสถานะเซลล์ (C_t) ดังสมการที่ 2.6 $f_t \times C_{t-1}$ คือการลืมข้อมูลที่ไม่จำเป็นออกจากสถานะเซลล์ก่อนหน้านี้ตามที่ฟอร์เกตเกทสั่งให้ลืม คือค่า f_t ที่อยู่ระหว่าง 0 ถึง 1 จะบอกถึงสัดส่วนของเซลล์ก่อนหน้าที่ควรถูกลืม ถ้า f_t มีค่าเป็น 0 จะหมายถึงลืมทั้งหมด ถ้า f_t มีค่าเป็น 1 จะหมายถึงเก็บข้อมูลทั้งหมด

2.3.2.5 เอาต์พุทเกท

ทำหน้าที่ควบคุมข้อมูลที่จะถูกส่งออกจากสถานะเซลล์ C_t ไปยังสถานะ hidden h_t เอาต์พุทเกทจะใช้ฟังก์ชันสิกมอยด์ (sigmoid) เพื่อกำหนดว่าส่วนไหนของข้อมูลในสถานะเซลล์ควรถูกส่งออก โดยมีสมการคำนวณดังนี้

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.8}$$

โดยที่

o_t คือ output จาก Output Gate ณ timestep t

σ คือฟังก์ชันสิกมอยด์ (sigmoid) ที่จะทำให้ output อยู่ในช่วง 0 ถึง 1

W_o เป็นเมทริกซ์น้ำหนักสำหรับ Output Gate

h_{t-1} ข้อมูล ณ เวลาก่อนหน้า

x_t ข้อมูลปัจจุบัน

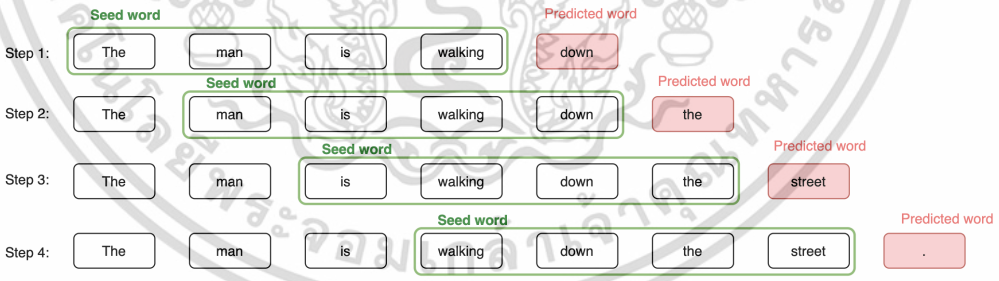
b_o เป็น bias term

2.3.2.6 hidden state

Output ที่ได้จะถูกใช้ในการคำนวณสถานะ hidden ปัจจุบันในโมเดลที่มีลักษณะเป็น recurrent หรือลูป hidden state จะถูกอัปเดตในแต่ละ timestep โดยใช้ข้อมูลปัจจุบันและสถานะซ่อนใน timestep ก่อนหน้าในโมเดล แอลเอสทีเอ็มทุกครั้งที่จะอัปเดตสถานะซ่อนโมเดลจะใช้ข้อมูลจากสถานะ cell state และค่าจาก Output Gate เพื่อคำนวณหา h_t ดังสมการ

$$h_t = o_t \times \tanh(C_t) \tag{2.9}$$

ตามสมการทั้งหมด แอลเอสทีเอ็มจะใช้การควบคุม การจดจำ และการลืมข้อมูลด้วยประตูที่แตกต่างกัน และใช้สมการทางคณิตศาสตร์เพื่อควบคุมการไหลของข้อมูลเข้าและออกจาก cell state ดังนั้นเมื่อใช้วิธีการนี้จะสามารถทำนายคำที่เกิดขึ้นต่อไปด้วยคำก่อนหน้าของประโยคด้วยการนำคำเริ่มต้นมาเป็นข้อความเริ่มต้นที่ให้กับโมเดลเพื่อให้โมเดลสามารถสร้างต่อจากข้อความนั้นไปเพื่อช่วยกำหนด context หรือแนวทางการสร้างข้อความ เมื่อป้อนคำเริ่มต้นเข้าไปในโมเดล แอลเอสทีเอ็มโมเดลจะทำนายข้อความต่อไป ผลลัพธ์ที่ได้จากการทำนายคำที่เกิดขึ้นคือประโยคที่ยาวขึ้นตามจำนวนคำที่ผู้วิจัยกำหนดไว้



รูปที่ 2.6 แนวคิดของการสร้างข้อความประกอบด้วยคำตั้งต้นแล้วต่อไปด้วยการสร้างคำต่อไป

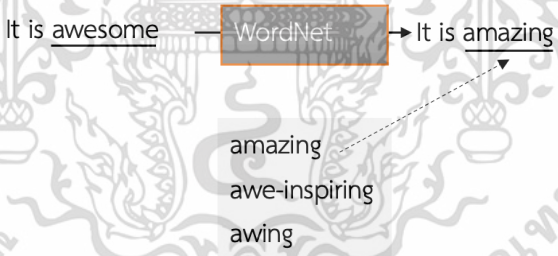
2.3.3 การเสริมข้อความด้วยวิธีออกเมนเทชัน

การเสริมข้อความด้วยวิธีออกเมนเทชันเป็นวิธีที่ใช้ในการขยายข้อความโดยการทำให้เกิดความหลากหลายในข้อมูลที่มีอยู่ โดยการแปลงและปรับเปลี่ยนคำในประโยคให้เป็นรูปแบบใหม่เมื่อเป็นคำใหม่

ดังนั้นประโยคจึงเป็นประโยคใหม่ด้วย วิธีการนี้จึงเป็นการเพิ่มปริมาณข้อมูลไปโดยปริยาย วิธีการออกเมนเทชันสามารถนำมาใช้กับข้อความในหลาย ๆ แนวทาง ดังนี้

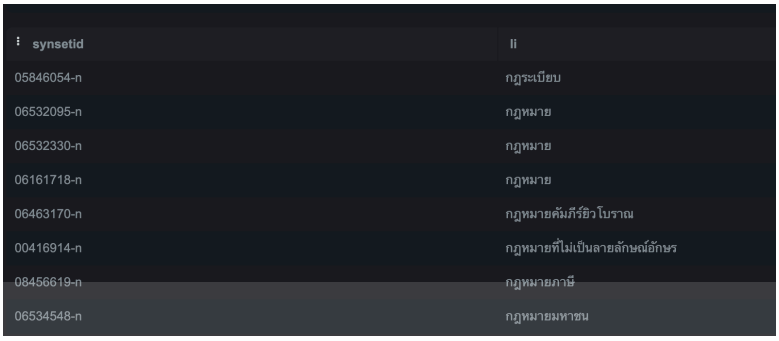
2.3.3.1 การเสริมข้อความด้วยวิธีเวิร์ดเน็ต (Wordnet augmentation)

เวิร์ดเน็ตคือฐานข้อมูลภาษาศาสตร์ (lexical database) ที่สร้างขึ้นโดยโครงการที่มีสำนักงานอยู่ที่ Princeton University ในสหรัฐอเมริกา โดยเฉพาะภาษาอังกฤษ แต่มีการสร้างเวิร์ดเน็ตสำหรับภาษาอื่น ๆ ด้วย ฐานข้อมูลนี้ได้เก็บรวบรวมคำศัพท์ที่มีความหมายเหมือนกันเข้าไปอยู่ในกลุ่มที่เรียกว่าซินเซ็ต (synsets) หรือคำพ้อง คือกลุ่มของคำที่มีความหมายที่ใกล้เคียงหรือเหมือนกัน ซึ่งเป็นหนึ่งในคุณสมบัติหลักของเวิร์ดเน็ต คำที่มีความหมายเหมือนกันหรือใกล้เคียงจะถูกรวมเข้าด้วยกัน และมักจะมีคำอธิบายหรือคำนิยามของกลุ่มคำนั้น ในภาษาอังกฤษซินเซ็ตของคำว่า “car” อาจประกอบด้วยคำอื่น ๆ ที่มีความหมายคล้ายคลึง เช่น “auto”, “automobile”, “machine”, “motorcar” ซินเซ็ตใช้สำหรับแสดงความสัมพันธ์ที่ซับซ้อนของคำที่มีความหมายใกล้เคียง และทำให้สามารถทำงานที่ต้องการความเข้าใจในความหมายและความสัมพันธ์ของคำได้ แต่ไม่จำเป็นต้องรู้ลำดับหรือโครงสร้างของประโยคที่ใช้คำเหล่านั้น ปัจจุบันเวิร์ดเน็ตเป็นฐานข้อมูลคำศัพท์เกี่ยวกับความสัมพันธ์ทางความหมายระหว่างคำมากกว่า 200 ภาษา



รูปที่ 2.7 การเพิ่มข้อความด้วยวิธีเวิร์ดเน็ต

สำหรับภาษาไทยมีการทำฐานข้อมูลภาษาไว้ด้วยเช่นกันโดยเป็นฐานข้อมูล SQLite โดยภายในจะมีการเก็บรวบรวมคำไว้ประมาณ 90k คำ และจัดเป็นกลุ่มๆ ด้วยหมายเลข synsetsid และคำต่างๆ เก็บไว้ในคอลัม li ดังรูปที่ 2.8



synsetid	li
05846054-n	กฎระเบียบ
06532095-n	กฎหมาย
06532330-n	กฎหมาย
06161718-n	กฎหมาย
06463170-n	กฎหมายคัมภีร์ยิวโบราณ
00416914-n	กฎหมายที่ไม่เป็นลายลักษณ์อักษร
08456619-n	กฎหมายภาษี
06534548-n	กฎหมายมหาชน

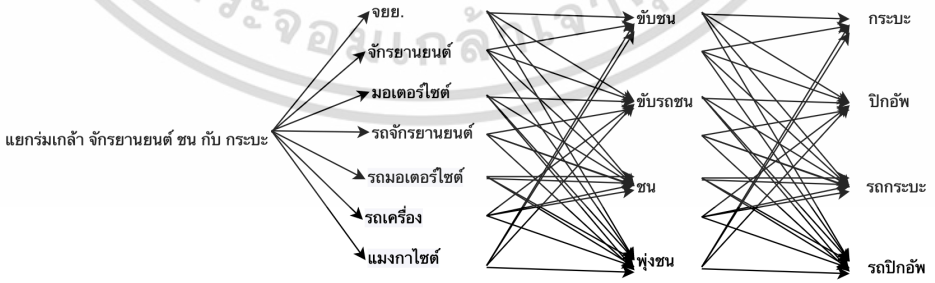
รูปที่ 2.8 โครงสร้างการเก็บข้อมูลของไทยเวิร์ดเน็ต

ดังที่กล่าวว่าเวิร์ดเน็ตจะเก็บข้อมูลเป็นกลุ่มด้วย synsetsid ตัวอย่างกลุ่มคำของคำว่า “รถยนต์” คำที่มีในฐานข้อมูลของ Thai wordnet จะเป็น synsetsid ที่ “02958343-n” และมีคำ 2 คำในกลุ่มนั้นคือคำว่า รถ และรถยนต์ดังรูปที่ 2.9

synsetid	li
02958343-n	รถ
02958343-n	รถยนต์

รูปที่ 2.9 กลุ่มคำว่ารถยนต์ในไทยเวิร์ดเน็ต

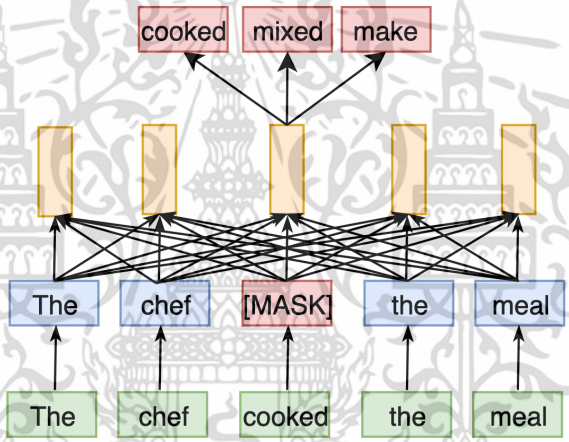
ดังนั้นเมื่อต้องการให้ข้อความเกิดขึ้นใหม่ด้วยการการปรับเปลี่ยนคำเพียงบางคำก็สามารถนำคำจากเวิร์ดเน็ตไปเปลี่ยนจากคำเดิม ดังนั้นข้อความก็จะเปลี่ยนไปนิดหน่อยแต่ความหมายยังคงเดิม



รูปที่ 2.10 การปรับเปลี่ยนบางคำเพื่อให้ได้ประโยคใหม่

2.3.3.2 การเสริมคำด้วยวิธีไทยทูทรานส์ฟอร์เมอร์ (Thai2transformers augmentation)

สำหรับงานด้านเอ็นแอลพีในทุกวันนี้มีวิธีที่ทำให้ประหยัดทรัพยากรได้มากคือวิธีการนำโมเดลที่มีการเทรนไว้ล่วงหน้าหรือพรีเทรนนำมาปรับใช้ให้เข้ากับงานที่ต้องการเรียกว่าการไฟน์ทูน (Fine-tune) โดยเรียกโมเดลนี้ว่าทรานส์ฟอร์เมอร์ สำหรับภาษาอังกฤษมักจะมีการทำโมเดลภาษาไว้มากมายมาใช้และสำหรับภาษาไทยมีการสร้างทรานส์ฟอร์เมอร์ขึ้นมาใช้งานใน WangchanBERTa ซึ่งมีการนำไทยทูทรานส์ฟอร์เมอร์มาใช้งานในการเสริมคำ โดยเป็นการนำเทคนิคที่เรียกว่าการเติมคำที่หายไป (Masked Language Model : MLM หรือเอ็มแอลเอ็ม) โดยเอ็มแอลเอ็มจะเป็นโมเดลทางภาษาอีกประเภทที่ถูกออกแบบมาให้พยายามหาคำที่หายไป (Masked words) จากบริบทคำรอบข้าง ด้วยคุณสมบัตินี้จึงมีแนวคิดนำมาทำการเสริมคำเพื่อสร้างประโยคให้เพิ่มมากขึ้น



รูปที่ 2.11 การเสริมคำด้วยวิธีไทยทูทรานส์ฟอร์เมอร์

2.3.3.3 การเสริมคำด้วยวิธีการเข้ารหัสแบบเอ็มเบ็ดดิงไบต์คู่ (BPEmb augmentation)

การเข้ารหัสแบบไบต์คู่ (Byte-Pair Encoding: BPE) คือรูปแบบของอัลกอริทึมการบีบอัดข้อมูล ซึ่งคู่ข้อมูลไบต์ต่อเนื่องกันที่พบบ่อยที่สุดจะถูกแทนที่ด้วยไบต์ที่ไม่ปรากฏในข้อมูลนั้น มีไว้เพื่อจัดการกับปัญหาคำที่ไม่อยู่ในคลังข้อมูล (Out-of-Vocabulary, OOV) โดยการแบ่งคำออกเป็นคำย่อย (subwords) โดยการเข้ารหัสตามรูปแบบคู่คำ คือ สมมติว่ามีข้อมูล aaabdaaabac จะทำการเข้ารหัส (บีบอัด) คู่ไบต์ aa ซึ่งเกิดขึ้นบ่อยที่สุด ดังนั้นเราจะแทนที่มันด้วย Z เนื่องจาก Z ไม่ได้เกิดขึ้นในข้อมูลนี้ ตอนนี้คำจะเป็น ZabdZabac โดยที่ Z = aa คู่ไบต์ทั่วไปถัดไปคือ ab ดังนั้นลองแทนที่มันด้วย Y ตอนนี้จะเป็น ZYdZYac โดยที่ Z = aa และ Y = ab เหลือคู่ไบต์เพียงคู่เดียวคือ ac ซึ่งปรากฏเป็นคู่เดียว ดังนั้นจะไม่ต้องเข้ารหัส จะสามารถใช้การเข้ารหัสคู่ไบต์แบบเรียกซ้ำเพื่อเข้ารหัส ZY เป็น X ได้ ขณะนี้ข้อมูลได้

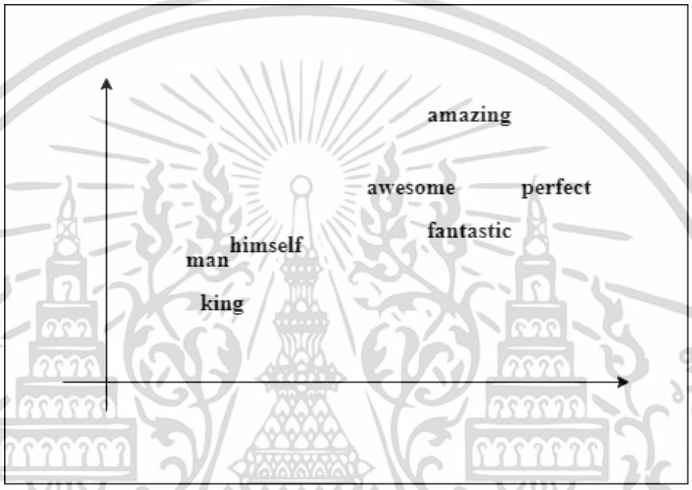
แปลงเป็น XdXac โดยที่ X = ZY, Y = ab และ Z = aa ไม่สามารถบีบอัดเพิ่มเติมได้เนื่องจากไม่มีคู่ไบต์ปรากฏมากกว่าหนึ่งครั้ง ถ้าต้องการขยายคืนค่าข้อมูลจะทำได้โดยดำเนินการแทนที่ในลำดับย้อนกลับ ดังนั้นวิธีการนี้สามารถนำมาใช้กับการเสริมคำ คือหลังจากที่เข้ารหัวคำแล้ว จะได้เซ็ตของคำย่อยที่ใช้บ่อย สามารถแทนที่ย่อยในข้อความด้วยคำย่อยอื่น ๆ ที่มีความหมายใกล้เคียงหรือคล้ายคลึงกันด้วยการหาคำคล้ายคลึงด้วยการฝังคำหรือเอ็มเบ็ดดิง ดังนั้นเมื่อผสานสองวิธีนี้เข้าด้วยกันจึงเรียกว่าแบบเอ็มเบ็ดดิงไบต์คู่ สำหรับการนำวิธีการเอ็มเบ็ดดิงไบต์คู่มาทำการเสริมคำ (BPEmb augmentation หรือบีพีอีเอ็มบี ออกเมนเทชัน) สามารถทำได้โดยใช้ข้อมูลของการฝังคำของกลุ่มคำย่อย (Subword Embeddings) คือการทำพรีเทรนของกลุ่มคำย่อยที่มีการทำไว้ล่วงหน้าสำหรับบีพีอีเอ็มบีมีทำไว้ทั้งสิ้น 275 ภาษารวมทั้งภาษาไทยมีการทำเอ็มเบ็ดดิงไว้ด้วยเช่นกัน โดยคำที่นำมาสร้างคำย่อยนำมาจากวิกีพีเดียแล้วนำมาแยกคำออกเป็นคู่ต่อมาจึงค่อยสร้างการฝังคำด้วยวิธีเวิร์ดทูเว็ค ดังนั้น อาจจะสรุปได้ว่าวิธีบีพีอีเอ็มบีจะเป็นการนำเวิร์ดทูเว็คมาแยกเป็นคู่ไบต์เพื่อให้มีความละเอียดขึ้นจะได้วิธีการฝังคำของกลุ่มคำย่อยออกมา และเมื่อต้องการสร้างการเสริมคำ สามารถนำคำจากข้อความที่ต้องการมาหาค่าน้ำหนักที่ใกล้กันโดยเรียกว่าการหาคำพ้องของคำนั้น ๆ เช่นต้องการหาคำพ้องของคำว่า “รถชน” จะได้ลิส (List) ของคำพ้องมาดังนี้

ตารางที่ 2.1 ตารางตัวอย่างคำพ้องของคำในบีพีอีเอ็มบี

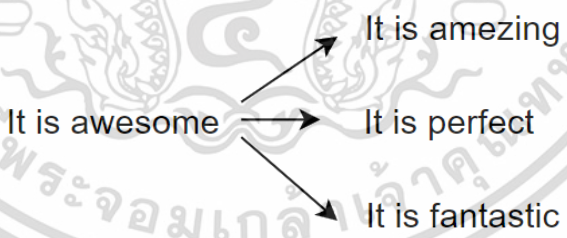
ลำดับที่	**คำในคลังข้อความ**	**ค่าคำพ้องกับคำว่า “รถชน”**
1	ประสบอุบัติเหตุ	0.5103998184204102
2	จากการถูก	0.41056373715400696
3	อุบัติเหตุ	0.39634835720062256
4	ขับรถชน	0.39246866106987
5	จนเสียชีวิต	0.35236817598342896
6	เกิดอุบัติเหตุ	0.3378407061100006
7	▁ประสบอุบัติเหตุ	0.33779633045196533
8	ทางรถยนต์	0.33655062317848206
9	ไฟไหม้	0.3347289562225342
10	และถูก	0.3316971957683563

เวิร์ดทูเว็คเป็นวิธีการที่ใช้ในการสร้างการแทนค่าด้วยเวกเตอร์ (vector representation) ของคำในภาษาธรรมชาติ คุณสมบัติหลักของเวิร์ดทูเว็ค คือ สามารถจับความสัมพันธ์ระหว่างคำต่าง ๆ ได้ ทำให้สามารถใช้การเรียนรู้ทางเครื่องเพื่อเรียนรู้ในการทำเอ็นแอลพีได้ ซึ่งเวิร์ดทูเว็คถูกสร้างขึ้นโดยใช้โครงข่าย

ประสาทเทียมในการประมวลผลข้อความ จากนั้นจะได้เว็คเตอร์ที่แทนคำแต่ละคำออกมา คุณสมบัติของเว็คเตอร์ที่ได้จะถูกปรับให้เหมาะสมตามความสัมพันธ์กับคำอื่น ๆ ในชุดข้อมูลเวิร์ดทูเว็คที่นำมาใช้ในการเสริมคำนี้ คือ การแทนที่ด้วยคำที่มีความหมายคล้ายคลึงกันเนื่องจากปัจจุบันเป็นเทคนิคที่สะดวกสามารถทำงานได้อย่างรวดเร็วเนื่องจากเป็นส่วนหนึ่งของไพไทยเอ็นแอลพีเป็นวิธีการที่ทำงานได้กับภาษาไทยโดยใช้คำในพจนานุกรมหรือกลุ่มคำในคลังคำศัพท์ที่เตรียมไว้เพื่อหาคำที่มีความหมายที่คล้ายคลึงกันและใช้คำนั้นแทนที่คำเดิมในข้อความ



รูปที่ 2.12 แสดงความสัมพันธ์ของคำในรูปแบบเว็คเตอร์



รูปที่ 2.13 แสดงการเปลี่ยนข้อความด้วยวิธีเวิร์ดทูเว็ค

การแทนที่คำด้วยคำที่มีความหมายคล้ายคลึงกันเป็นวิธีการที่ใช้เพื่อขยายข้อมูลข้อความโดยไม่เปลี่ยนแปลงความหมายของข้อความ วิธีนี้สามารถช่วยปรับปรุงแนวโน้มที่ดีของการเทรนของโมเดลและเพิ่มประสิทธิภาพการเรียนรู้ของโมเดลได้ โดยวิธีการเสริมคำ โดยใช้ความคล้ายคลึง สามารถทำได้ดังนี้

1) เลือกคำที่จะแทนที่ ควรเลือกคำที่ไม่ใช่คำวิเศษ (เช่น คำนาม คำกริยา) เพื่อไม่ทำให้ความหมายของประโยคเปลี่ยนไป

2) ค้นหาคำที่มีความหมายคล้ายคลึง สามารถใช้ฐานข้อมูลเช่นเวิร์ดเน็ตหรือบริการออนไลน์เพื่อค้นหาคำที่มีความหมายคล้ายคลึงกับคำที่ต้องการแทนที่

3) แทนที่คำ หลังจากที่ได้คำที่มีความหมายคล้ายคลึงแล้ว ให้แทนที่คำเดิมในประโยคด้วยคำใหม่

4) ตรวจสอบความหมาย หลังจากการแทนที่คำแล้วควรตรวจสอบว่าความหมายของประโยคยังคงเดิมหรือไม่

การใช้วิธีการเสริมคำจะช่วยในการสร้างข้อความได้อย่างมีประสิทธิภาพและมีความหลากหลาย ซึ่งสามารถนำไปใช้ในงานที่ต้องการข้อมูลที่มีความหลากหลายมากยิ่งขึ้น หรือนำไปเติมข้อความบางกลุ่มที่มีขนาดจำนวนน้อยให้มีเพิ่มขึ้นเท่ากับกลุ่มที่มีจำนวนมาก

2.3.4 การวัดค่าความเหมือนของคำด้วยวิธีเบลอสกอร์

การศึกษาแบบประเมินสองภาษา (Bilingual Evaluation Understudy : BLEU หรือเบลอ) เป็นวิธีการวัดความเหมือนกันระหว่างข้อความที่ถูกสร้างขึ้นเพื่อวัดค่าของภาษาที่แปลมาจากระบบกับข้อความต้นฉบับที่มนุษย์เป็นผู้แปล โดยจะใช้การนับจำนวนคำที่ตรงกันในข้อความเบลอสกอร์เน้นการเปรียบเทียบเอ็นแกรม (n-gram) ของประโยคที่ระบบสร้างขึ้น กับประโยคที่มนุษย์เป็นผู้แปลโดยจะให้น้ำหนักกับการตรงกันของเอ็นแกรมระหว่างสองประโยค ผลลัพธ์ของการวัดจะอยู่ในช่วง 0 ถึง 1 โดยที่ค่าที่ใกล้เคียง 1 แสดงว่าประโยคที่ระบบสร้างมีความเหมือนกันกับประโยคอ้างอิงที่มนุษย์เป็นผู้แปลมากขึ้น แต่เบลอสกอร์ที่คะแนน 0.6 หรือ 0.7 ถือว่าดีที่สุดที่ทำได้ แม้แต่มนุษย์สองคนยังมีแนวโน้มที่จะใช้รูปแบบประโยคที่แตกต่างกันสำหรับการแปลข้อความ และแทบจะไม่สามารถจับคู่ได้อย่างสมบูรณ์แบบได้ด้วยเหตุนี้ คะแนนที่เข้าใกล้ 1 จึงไม่สมจริงในทางปฏิบัติ และอาจจะสรุปได้ว่าแบบจำลองนั้นไม่เหมาะสมเกินไป การคำนวณนั้นสามารถคำนวณได้โดยนับคำที่แปลได้เหมือนกันว่ามีกี่คำ (N-gram Matches) กับคำแปลที่มาจากมนุษย์ (reference translations) สำหรับทุกเอ็นแกรม (เริ่มจาก 1-gram ถึง n-gram) โดยใช้สมการที่ 2.10

$$precision_n = \frac{Number\ of\ N-gram\ Matches}{Total\ Number\ of\ N-grams\ in\ Candidate} \quad (2.10)$$

เช่น จำนวนคำทั้งหมดในการแปลคือ 4 คำ และคำที่แปลได้ถูกทั้งหมดมี 3 คำ จะได้เป็น 3/4 = 0.75

2.3.5 การวัดค่าความคล้ายคลึงของความหมายด้วยเบิร์ตสกอร์

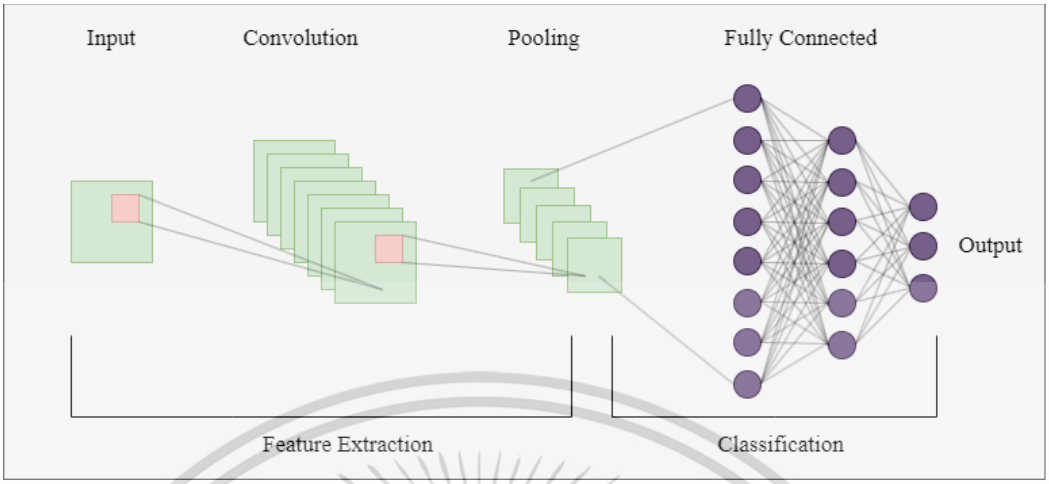
เบิร์ตสกอร์คือวิธีการวัดความคล้ายคลึงระหว่างประโยคหรือข้อความโดยใช้ระบบตัวแบบที่ถูกฝึกสอนด้วยหรือเป็นตัวชี้วัดความคล้ายคลึงของข้อความที่ถูกสร้างขึ้นโดยระบบโดยจะมุ้งเน้นไปที่การประเมินความคล้ายคลึงทางความหมายมากกว่าความเหมือนของคำ โดยเบิร์ตจะมีการฝึกฝนไว้ล่วงหน้าเมื่อมีคำหรือข้อความใหม่เข้ามาจะนำข้อความใหม่นั้นมาหาเวคเตอร์เพื่อนำค่าเวคเตอร์จากข้อความจริงและหาเวคเตอร์ของข้อความที่ถูกแปลขึ้นมานำมาเปรียบเทียบกันหากค่าเวคเตอร์ใกล้กันจะส่งผลให้คะแนนมีค่าสูงโดยอาศัยหลักการของความคล้ายคลึงแบบโคส์ไซน์ (Cosine Similarity) โดยการคำนวณค่าเบิร์ตสกอร์จะใช้ค่าความคล้ายคลึงที่คำนวณได้มาเปรียบเทียบกันเพื่อหาความห่างของคำในปริภูมิแบบยุคลิด โดยที่ค่าเบิร์ตสกอร์จะอยู่ในช่วง 0 ถึง 1 โดยที่ค่า 1 แสดงถึงความคล้ายคลึงที่สูงสุด

2.4 การสร้างแบบจำลองการจำแนกข้อความทวิตเตอร์

การจำแนกข้อความทวิตเตอร์มักเป็นกระบวนการที่ต้องการความรอบคอบและการเข้าใจทั้งในด้านข้อมูลและเทคนิคการสร้างแบบจำลองโดยรวม ยิ่งไปกว่านั้นผู้วิจัยยังต้องปรับเปลี่ยนและทดสอบเพื่อให้ได้ผลลัพธ์ที่ดีที่สุด โดยการผสมผสานเทคนิคต่างๆ หรือปรับปรุงพารามิเตอร์ให้เหมาะสมกับกลุ่มข้อมูลที่ต้องการจำแนก สำหรับงานวิจัยนี้ได้ศึกษาและทำความเข้าใจวิธีการที่เป็นที่นิยม ดังต่อไปนี้

2.4.1 การจำแนกข้อความด้วยซีเอ็นเอ็น

ซีเอ็นเอ็นเป็นกลุ่มของโครงข่ายประสาทเทียมที่ถูกออกแบบมาเพื่อสามารถใช้งานได้ดีกับข้อมูลที่มีโครงสร้างเชิงพื้นที่ (spatial structure) เช่น รูปภาพ ซีเอ็นเอ็นทำงานโดยการผ่านข้อมูลที่ชั้นคอนโวลูชันเนล (convolutional) โดยมีตัวกรอง (filters) หรือเคอเนล (kernels) ที่สามารถเลื่อนไปตามภาพ เพื่อเรียนรู้และสร้างแผนที่ลักษณะ (feature maps หรือฟีเจอร์แมพ) ที่บ่งบอกถึงลักษณะหรือคุณสมบัติที่สำคัญของข้อมูล ชั้นที่สำคัญอื่น ๆ ของซีเอ็นเอ็นคือชั้นการรวมสรุป (pooling layers หรือพูลลิงเลเยอร์) และชั้นที่เชื่อมต่อกันทั้งหมด (fully connected layers) ดังรูปที่ 2.14



รูปที่ 2.14 โครงสร้างของซีเอ็นเอ็นแสดงชั้นต่าง ๆ

ชั้นพูลลิง (pooling) ช่วยในการลดขนาดของข้อมูลและแยกแยะคุณสมบัติที่สำคัญ ในขณะที่ชั้นเชื่อมเต็ม (fully connected) มักถูกใช้เพื่อสร้างการทำนายหรือการจำแนกประเภทซีเอ็นเอ็นถูกใช้ในหลากหลายในหลายบริบท เช่น การรู้จำรูปภาพ การรู้จำเสียง การตรวจจับวัตถุในวิดีโอ และอีกงานที่กำลังเป็นที่นิยมคือการประมวลผลภาษาธรรมชาติ

การประมวลผลภาษาธรรมชาติหรือเอ็นแอลพีด้วยซีเอ็นเอ็นหมายถึงการใช้โมเดลซีเอ็นเอ็นในการแยกแยะและเรียนรู้คุณลักษณะที่สำคัญของข้อความ เพื่อรับรู้ความหมาย ความรู้สึก หรือข้อมูลอื่น ๆ ที่สำคัญซึ่งอยู่ภายในข้อความ ดังที่ทราบกันว่า ซีเอ็นเอ็น ถูกออกแบบมาสำหรับการจัดการกับข้อมูลที่มีโครงสร้างเชิงพื้นที่ เช่น ภาพ แต่อย่างไรก็ตาม ซีเอ็นเอ็น สามารถนำมาใช้กับข้อความได้ เนื่องจากข้อความนั้นมีโครงสร้างลำดับที่สำคัญ (คำต่อมาอาจขึ้นอยู่กับคำก่อนหน้า) โดยทั่วไปแล้ว ซีเอ็นเอ็นสามารถใช้สำหรับงานด้าน เอ็นแอลพี ได้ดังนี้ การวิเคราะห์ความรู้สึก (Sentiment Analysis) คือซีเอ็นเอ็นสามารถใช้สำหรับการวิเคราะห์ความรู้สึก โดยอาจจะวิเคราะห์เรื่องการรีวิวสินค้าเป็นบวกหรือลบ หรือวิเคราะห์ว่าข้อความในโซเชียลมีเดียเป็นความรู้สึกเชิงบวกหรือเชิงลบ หรือการสกัดข้อมูล (Information Extraction) ซีเอ็นเอ็นสามารถใช้ในการสกัดข้อมูลหรือแยกแยะข้อมูลที่สำคัญออกมาจากข้อความ และใช้เพื่อการจำแนกประเภทข้อความ (Text Classification) ซีเอ็นเอ็นสามารถใช้ในการจำแนกประเภทข้อความ เช่น การจำแนกข้อความว่าเป็นข่าวที่จริงหรือข่าวปลอม

การสร้างโมเดล (Model Building) ในส่วนนี้โมเดลซีเอ็นเอ็นจะถูกสร้างขึ้น โดยมีชั้น คอนโวลูชันเลเยอส์ (Convolutional layers) และชั้นหลัก ๆ จะใช้พูลลิงเลเยอส์ (pooling layers) และฟูลลีคอนเน็ตเต็ดเลเยอส์ (fully connected layers) ในโมเดลมักถูกใช้ร่วมกับ activation function เป็นชั้นสุดท้ายเพื่อจำแนกประเภท โดยมีรายละเอียดดังนี้

2.4.1.1 คอนโวลูชันเลเยอร์ (Convolutional layers)

คือการเลื่อนฟิลเตอร์ (filter) ตามข้อมูลป้อนเข้า เพื่อสกัดคุณสมบัติที่ต่างกันในงานเอ็นแอลพีนั้น คอนโวลูชันเลเยอร์ จะมีลักษณะคล้ายกับการทำคอนโวลูชันกับภาพ แต่ข้อมูลป้อนเข้าในกรณีของเอ็นแอลพีจะเป็นเมทริกซ์ของเวคเตอร์ฝังคำ (embedding vectors) แทนที่จะเป็นพิกเซลของภาพ เนื่องจากประโยคข้อความเป็นลำดับของคำ ประโยคที่สมบูรณ์จะถูกแทนด้วยแถวที่ถูกต่อเข้าด้วยกัน แล้วใช้ฟิลเตอร์ที่มีความกว้างเท่ากับขนาดของเวคเตอร์ของการฝังคำ (word embedding) และฟิลเตอร์จะตรงกับขนาดที่แตกต่างกันของแถวเดียวกัน ซึ่งเป็น $L = 2,3,4, \dots$ ตัวกรองขนาดต่างๆ ช่วยให้ตัวแยกประเภท ซีเอ็นเอ็นเรียนรู้คุณลักษณะมากมายของภาษาธรรมชาติ คือความสูงของตัวกรอง ซึ่งก็คือจำนวนแถวที่อยู่ติดกัน สมมติว่าตัวกรองแต่ละตัวถูกกำหนดพารามิเตอร์โดย $w \in R^{L \times D}$ เพื่อดำเนินการ Convolution กับทุกคำ (L-gram) ของ L ที่เริ่มต้นด้วยแถว$x_{t:t+L-1}$ ในเวลาเดียวกันในแต่ละย่อยประโยค การคูณตามองค์ประกอบจะถูกคำนวณแล้วจึงสรุปผลผลลัพธ์จากการคอนโวลูชันถูกใช้โดยฟังก์ชันที่ไม่เป็นเชิงเส้นเพื่อสร้างคุณลักษณะ (feature maps) ที่มีขนาดที่แตกต่างตามฟิลเตอร์ : $H_i = [h_1, h_2, \dots, h_{T+L-1}]$ โดยที่แต่ละองค์ประกอบถูกกำหนดโดยสมการ 2.11

$$h_t = f(W \cdot X_{t:t+L-1} + b_t) \tag{2.11}$$

โดยที่

$x_{t:t+L-1}$ คือการต่อด้วยเวคเตอร์อินพุทที่ L เข้าด้วยกัน

$b_t \in R$ เป็นเทอม bias

f คือฟังก์ชันสิกมอยด์ (sigmoid)

สมการข้างต้นเป็นการเอาฟิลเตอร์มาคูณกับข้อมูลป้อนเข้าที่ตำแหน่งที่สัมพันธ์กันและรวมผลลัพธ์ทั้งหมดเข้าด้วยกันเพื่อได้ผลลัพธ์สำหรับตำแหน่งนั้น ๆ

ฟังก์ชันกระตุ้น Activation function ฟังก์ชันกระตุ้น ReLU (Rectified Linear Unit) ในงานเอ็นแอลพี ไม่ได้แตกต่างจากการใช้ในงานประเภทอื่น ๆ ของ โครงข่ายประสาทเทียม ซึ่งฟังก์ชัน ReLU นิยมใช้กับซีเอ็นเอ็นและอื่น ๆ เพราะความง่ายและประสิทธิภาพในการเทรนแบบจำลอง เนื่องจากการคำนวณ ReLU เป็นไปอย่างรวดเร็วและง่าย เมื่อเทียบกับ functions อื่น ๆ เช่น sigmoid หรือ tanh ทั้งยังช่วยป้องกันปัญหา Vanishing Gradient ใน โครงข่ายประสาทเทียม ที่มีความลึกมาก ค่า gradient ของ activation functions บางชนิดจะทำให้มีแนวโน้มที่จะยืนยันค่าไปเรื่อยๆ หากคำนวณได้ค่าทางลบก็ยิ่งลบไปทวีคูณจนความชันเข้าใกล้ o จน Gradient หายไปหมดทำให้โมเดล Train ไม่ไปไหน ทำให้การ

ฝึกฝนสิ้นเปลืองเวลา แต่ ReLU สามารถช่วยลดปัญหานี้ หากนำมาแสดงเป็นกราฟของ ReLU จะเป็นแนวตรงเมื่อ $x > 0$ และเป็นแนวนอน (ค่าเท่ากับศูนย์) เมื่อ $x \leq 0$ สมการของ ReLU โดยสมการ Activation function มีดังนี้

$$f(x) = \max(0, x) \tag{2.12}$$

โดยที่

ถ้า x มีค่ามากกว่า 0, $f(x)$ จะเท่ากับ x

ถ้า x มีค่าน้อยกว่าหรือเท่ากับ 0, $f(x)$ จะเท่ากับ 0

2.4.1.2 การรวม (Pooling)

ใช้ลดขนาดของข้อมูล โดยการเลือกค่าสูงสุด (max pooling) ของกลุ่มข้อมูล (เช่น ค่าผลลัพธ์จาก Convolution Operation) ภายในหน้าต่างที่กำหนด มักถูกใช้ในเอ็นแอลพีเพื่อเลือกค่าสูงสุดจากผลลัพธ์ของการคอนโวลูชัน เพื่อให้ได้รับคุณสมบัติที่สำคัญที่สุดจากหน้าต่างคำที่คอนโวลูชันที่ชั้นนั้นๆ ในการประยุกต์ใช้กับงานเอ็นแอลพี พูลลิงมักใช้กับผลลัพธ์หลังจากการดำเนินการคอนโวลูชัน เพื่อลดขนาดคุณสมบัติสำคัญ (feature maps) และลดขนาดข้อมูลไปในตัวเพื่อทำให้แบบจำลองมีความซับซ้อนน้อยลงและทำงานได้เร็วขึ้น

$$P(i,j) = max_{m,n \in window} I(i+m, j+n) \tag{2.13}$$

โดยที่

$P(i,j)$ คือค่าที่ได้จากการดำเนินการ max pooling ที่ตำแหน่ง (i,j) ใน pooled feature map

$I(i+m, j+n)$ คือค่าใน input feature map ที่ตำแหน่ง $(i+m, j+n)$ Window หมายถึงขนาดของหน้าต่าง pooling ที่กำหนด (เช่น 2x2, 3x3)

ดังนั้น การทำ max pooling ใน ซีเอ็นเอ็น คือการนำค่าสูงสุดจากแต่ละหน้าต่างใน feature map มาใช้เป็นค่าแทนสำหรับหน้าต่างนั้น ทำให้ได้ feature map ที่มีขนาดเล็กลงแต่ยังรักษาคุณสมบัติสำคัญๆ ของข้อมูลเดิม

2.4.1.3 ชั้นเชื่อมเต็ม (Fully Connected Layer หรือ Dense Layer)

หลังจากผ่านการคอนโวลูชันและการรวมมาแล้ว ผลลัพธ์จะถูกส่งผ่านชั้นเชื่อมเต็มเพื่อประมวลผลส่วนท้ายและให้ผลลัพธ์สุดท้าย ชั้นนี้เชื่อมต่อแต่ละ node (หรือ neuron) ในชั้นนี้ไปยังทุก node ในชั้นก่อนหน้า และทุก node ในชั้นถัดไปมีหน้าที่สำคัญในการรวมข้อมูลหรือ features ที่ได้จากชั้นก่อนหน้านี้เพื่อทำนายผลลัพธ์ในชั้นสุดท้าย ในชั้นสุดท้ายของ Fully Connected Layer มักจะมี units หรือ neurons ที่มีจำนวนเท่ากับจำนวนคลาสที่ต้องการจำแนก (เช่น 2 หรือ 10 หรือมากกว่า) และใช้ฟังก์ชัน activation เช่น SoftMax ในการคำนวณความน่าจะเป็นของแต่ละคลาส Fully Connected Layer สามารถมีหลายชั้น โดยชั้นสุดท้ายของมันจะเชื่อมต่อกับชั้น output ที่ใช้ฟังก์ชัน activation เพื่อให้ผลลัพธ์ออกมาในรูปแบบที่ต้องการ, เช่น การทำนายคลาสในงานการจำแนกประเภท โดยสรุป Fully Connected Layer ทำหน้าที่รวม features หรือข้อมูลที่ได้รับจากชั้นก่อนหน้าและใช้ข้อมูลเหล่านี้เพื่อทำนายผลลัพธ์ในชั้นสุดท้ายของแบบจำลอง

$$y = W_x + b \tag{2.14}$$

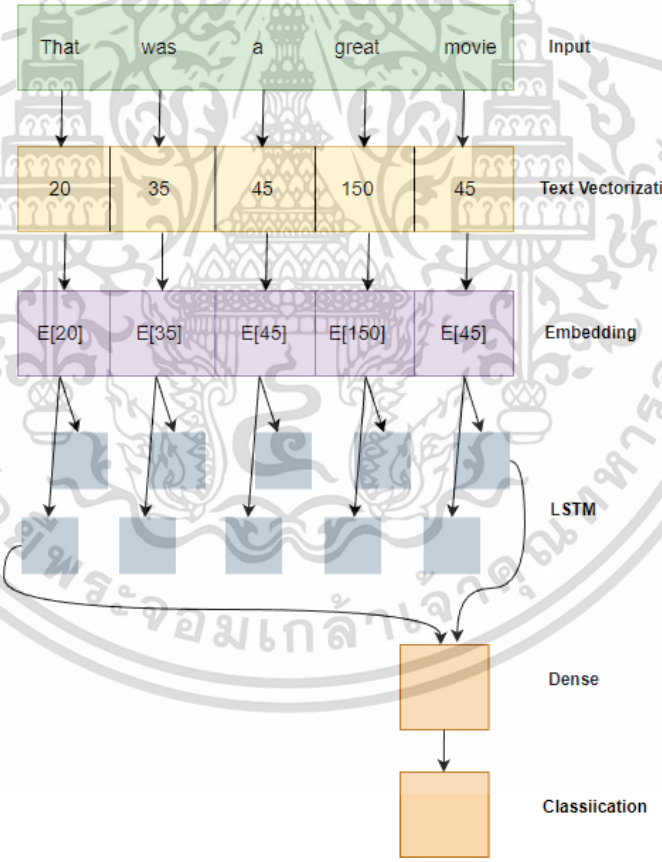
โดยที่

y คือเวคเตอร์ผลลัพธ์หลังจากทำการคูณเมทริกซ์และบวกด้วยเวคเตอร์ใบแอส
W คือเมทริกซ์น้ำหนักของเส้นเชื่อมระหว่างชั้นของ neurons
x คือเวคเตอร์ป้อนเข้าของชั้น
b คือ เวคเตอร์ใบแอสเทอม

และถ้าโมเดลทำงานได้ไม่ดีนักอาจจะต้องมีการกำหนดพารามิเตอร์ (Parameter Definition) ในการสร้างโมเดล Neural Network เช่นขนาดของ Filter ใน convolutional layers ซึ่งจะเป็นการกำหนดขนาดของการเลื่อน ที่ ซีเอ็นเอ็นจะใช้ในการดูข้อมูล ขนาดที่เหมาะสมจะขึ้นอยู่กับระดับที่ต้องการจากข้อมูล จำนวนของ Filters (Number of Filters) ในแต่ละ convolutional layer จะเป็นการกำหนดจำนวนของ feature maps ที่ ซีเอ็นเอ็นจะสร้าง จำนวนที่มากขึ้นจะทำให้โมเดลสามารถจับพฤติกรรมของข้อมูลที่มากขึ้นได้ แต่อาจทำให้โมเดลมีความซับซ้อนมากขึ้นและต้องการเวลาในการเทรนที่มากขึ้น ชนิดของ Activation ซึ่งจะทำหน้าที่ตัดสินใจว่าเซลล์ประสาท (neuron) จะส่งสัญญาณต่อหรือไม่ ซึ่งฟังก์ชันที่นิยมมี ReLU (Rectified Linear Unit) Sigmoid และ Tanh ดังนั้น หลักการในการกำหนดพารามิเตอร์ทั้งหมดนี้สามารถถูกปรับปรุงและหาค่าที่เหมาะสมได้ด้วยการทดลอง การใช้งานเทคนิค Grid Search หรือ Random Search สำหรับ Hyperparameter Optimization

2.4.2 การจำแนกข้อความด้วยการเรียนรู้เชิงลึกแอลเอสทีเอ็ม

แอลเอสทีเอ็มเป็นแบบจำลองโครงข่ายประสาทเทียมชนิดหนึ่ง ถูกพัฒนาต่อยอดมาจากอาร์เอ็นเอ็น) เพื่อแก้ไขปัญหาที่เกิดขึ้นกับแบบจำลองประสาทเทียมแบบพื้นฐาน (vanilla neural networks) ในการเรียนรู้ของข้อมูลที่มีระยะทางยาวนาน ที่ทำให้เกิดปัญหาสูญเสียความสามารถในการจำแนกและให้ความสำคัญกับข้อมูลย้อนหลังไม่เพียงพอ (vanishing gradient problem) และปัญหาการเกิดโอเวอร์ฟิตติง (overfitting) ในข้อมูลที่ยาวนาน แอลเอสทีเอ็มมีหลักการทำงานที่ถูกออกแบบมาเพื่อเก็บและลืมข้อมูลได้ในขั้นตอนการทำงาน โดยมีส่วนสำคัญแต่ละ cell ของ แอลเอสทีเอ็มประกอบด้วย ส่วนประกอบหลัก ๆ คือ input gate forget gate output gate และ cell state ดังที่ได้อธิบายมาแล้วในข้อที่ 2.3.2 แต่สำหรับการจำแนกข้อความด้วย แอลเอสทีเอ็มจะมีความแตกต่างกันตรงส่วนข้อมูลเข้าและผลลัพธ์ คือ ข้อความเข้าจะเป็นข้อความที่มีความยาวเท่ากันด้วยการทำ padding



รูปที่ 2.15 การจำแนกข้อความด้วยวิธีแอลเอสทีเอ็ม

การนำแอลเอสทีเอ็มมาใช้สำหรับการจำแนกข้อความซึ่งเป็นโครงข่ายประสาทเทียมได้นั้นควรมีรูปแบบของข้อมูลที่เข้ามาเป็นลำดับของข้อความ เช่น ประโยคหรือข้อความที่แยกแยะกันด้วยเครื่องหมายตัวอักษรหรือมีการแบ่งคำไว้แล้ว นอกจากนี้ ในการประมวลผลข้อมูลที่เป็นข้อความแบบภาษาธรรมชาติหรือเอ็นแอลพีอาจจำเป็นต้องทำการเตรียมข้อมูลเพิ่มเติม เช่น การแปลงคำเป็นตัวเลข (Word Embedding) เพื่อให้สามารถนำเข้าข้อมูลลงในโครงข่ายประสาทเทียมได้ การทำงานแต่ละ Layer ใน แอลเอสทีเอ็มประกอบด้วยชั้นการทำงานหลายๆ ชั้น (layers) โดยแต่ละชั้นประกอบด้วยเซลล์หลายๆ ตัว ในแต่ละชั้นเซลล์ทำงานร่วมกันเพื่อเรียนรู้และแยกแยะลักษณะต่างๆ ของข้อมูล ซึ่งชั้นแรกทำหน้าที่รับข้อมูลนำเข้า และในแต่ละชั้นถูกส่งต่อไปยังชั้นถัดไปซึ่งทำหน้าที่เพิ่มข้อมูลลงในเซลล์ในช่วงเวลาถัดไป

2.4.3 การวัดผลการจำแนกข้อความ

การทำนายข้อความแบบไม่สมดุลในชุดข้อมูลคือเหตุการณ์ที่ข้อมูลในบางกลุ่มมีจำนวนน้อยมากเมื่อเทียบกับกลุ่มอื่น ๆ ในชุดข้อมูลเดียวกัน ตัวอย่างเช่น ในงานการตรวจจับการฉ้อโกงบัตรเครดิต การฉ้อโกงจะเกิดขึ้นไม่บ่อยนัก (เช่น 0.1% ของทั้งหมด) ขณะที่การใช้บัตรอย่างปกติจะเกิดขึ้นบ่อย (99.9% ของทั้งหมด) หากใช้ ความถูกแม่นยำ (Accuracy) ดังนั้น ในกรณีของข้อมูลไม่สมดุล การใช้ความถูกต้องเป็นตัววัดผลอาจนำไปสู่ความเข้าใจที่ผิดเพี้ยน จากตัวอย่างข้างต้น ถ้ามีโมเดลที่ทำนายว่าทุกการใช้บัตรเครดิตเป็น "ไม่ฉ้อโกง" 100% ตลอดเวลา ความถูกต้องจะยังคงสูงมาก ณ 99.9% แม้ว่าโมเดลจะไม่สามารถตรวจจับการฉ้อโกงได้เลย ดังนั้น ในสถานการณ์ที่ข้อมูลไม่สมดุล ความถูกต้องไม่ใช่ตัววัดที่ดีสำหรับการประเมินประสิทธิภาพของโมเดล ค่าเอฟวันซึ่งเป็นการรวมการประเมินทั้งค่าพรีซิชัน (Precision) และค่ารีคอล (Recall) เข้าด้วยกัน ทั้งสองตัวนี้ล้วนสำคัญสำหรับข้อมูลที่ไม่สมดุล โดยมี Precision จากการทำนายที่ว่าเป็นการฉ้อโกง มีกี่เปอร์เซ็นต์ที่เป็นจริง และค่ารีคอลจากการฉ้อโกงจริง ๆ มีกี่เปอร์เซนต์ที่โมเดลสามารถตรวจจับได้ ค่าเอฟวันจะให้ค่าสูงเมื่อทั้งค่าพรีซิชันและค่ารีคอลสูง ซึ่งเป็นสิ่งที่ต้องการในการทำนายข้อมูลที่ไม่สมดุล โดยค่าพรีซิชันและค่ารีคอลได้มาจากการคำนวณค่าของ True Positive (TP) คือ สิ่งคำตอบที่โมเดลทำนายว่า “จริง” และ มีค่าเป็น “ จริง ”, True Negative (TN) คือ สิ่งที่โมเดลทำนายว่า “ไม่จริง” และ มีค่า “ ไม่จริง ”, False Positive (FP) คือ สิ่งที่โมเดลทำนายว่า “จริง” แต่ มีค่าเป็น “ไม่จริง” และสุดท้าย False Negative (FN) คือ สิ่งที่โมเดลทำนายว่า “ไม่จริง” แต่ มีค่าเป็น “จริง”

การวัดผลด้วย ค่าเอฟวัน (F1) เป็นหนึ่งในวิธีการวัดประสิทธิภาพของการจำแนกหรือการแยกประเภท (classification) ที่มักนิยมใช้ในงานที่ค่าประสิทธิภาพของการจำแนกข้อมูล โดยที่ ค่าเอฟวันคำนวณได้จากสมการ 2.15 - 2.17

$$precision = \frac{TP}{TP+FP} \quad (2.15)$$

$$recall = \frac{TP}{TP+FN} \quad (2.16)$$

$$F1 = \frac{2 \times precision \times recall}{precision+recall} \quad (2.17)$$

เมื่อ :

Precision คือค่าความแม่นยำ ได้จากอัตราส่วนของข้อมูลที่ทำนายว่าเป็นจริงได้ถูกต้องจากข้อมูลทั้งหมดที่ทำนายว่าเป็นจริง

Recall คือค่าความครอบคลุม ได้จากอัตราส่วนของข้อมูลที่ทำนายว่าเป็นจริงได้ถูกต้องจากข้อมูลที่ทำนายแล้วไม่ถูกต้องเมื่อเทียบกับเฉลย

ค่าเอฟวันมีค่าระหว่าง 0 ถึง 1 โดยค่าที่มากที่สุดคือ 1 ซึ่งหมายถึงความแม่นยำและความครอบคลุมที่ดีที่สุด แสดงให้เห็นว่าโมเดลมีความสามารถในการจำแนกหรือการแยกประเภทของข้อมูลอย่างครอบคลุมและแม่นยำ ซึ่งเป็นที่นิยมในงานที่ความสำคัญของความแม่นยำและความครอบคลุมเท่ากัน

ความสำคัญของค่าเอฟวันในงานที่เกี่ยวข้องกับการจำแนกหรือการแยกประเภท (classification) ถูกนำมาใช้งานอย่างกว้างขวางในหลากหลายงานด้านการประมวลผลภาษาธรรมชาติ (Natural Language Processing) โดยเฉพาะเช่นการจำแนกข้อความ (Text Classification) การตรวจจับข้อความ (Text Detection) และงานที่เกี่ยวข้องกับความคล้ายคลึงของข้อความ (Text Similarity) เป็นต้น ความสำคัญของ ค่าเอฟวันมีหลายเหตุผลดังนี้

- ความสมดุลของความแม่นยำและความครอบคลุม ค่าเอฟวันนำมาคำนวณค่าเฉลี่ยความแม่นยำและความครอบคลุม ทำให้สามารถให้ความสำคัญในทั้งสองค่า หากค่าตัวใดตัวหนึ่งมีค่าต่ำกว่าอีกตัวอย่างนั้นมาก จะทำให้ค่าเอฟวันมีค่าลดลง ดังนั้นโมเดลที่ให้ ค่าเอฟวันสูงแสดงถึงความสามารถในการทำนายที่ครอบคลุมและแม่นยำในเวลาเดียวกัน
- ในกรณีที่ชุดข้อมูลมีความไม่สมดุลกัน คือมีประเภทหนึ่งที่มีจำนวนข้อมูลน้อยกว่าประเภทอื่นๆ การใช้แอคคิวเรซี (accuracy) เพียงอย่างเดียวอาจทำให้โมเดลตัดสินใจเพียงแค่ทำนายประเภทที่มีจำนวนมากที่สุดเสมอแต่ค่าเอฟวันนำความแม่นยำและความครอบคลุมมาคำนวณในครั้งเดียวกัน ทำให้การวัดประสิทธิภาพของโมเดลนั้นเหมาะสมกับกรณีที่ข้อมูลไม่สมดุลกัน

2.5 งานวิจัยที่เกี่ยวข้อง

Angelica Salas, Panagiotis Georgakis และ Yannis Petalas [12] มีการวิจัยเกี่ยวกับการพัฒนาระบบการตรวจจับการเกิดอุบัติการณ์ทางการจราจรด้วยข้อความจากทวิตเตอร์แบบเรียลไทม์ในสหราชอาณาจักร (UK) โดยมีการดึงข้อมูลมาประมวลผลและจำแนกประเภทข้อความออกเป็นข้อความจราจร (Traffic) และข้อความที่ไม่เกี่ยวกับจราจร (Non-Traffic) ด้วยวิธี Support Vector Machine (SVM) แล้วใช้ข้อความสำหรับการเทรน (Train) โมเดลที่เท่ากันที่ 871 ข้อความและใช้ข้อความสำหรับการทดสอบ (Test) ที่ 290 ข้อความ โดยงานนี้มีการใช้ข้อความที่เท่ากันทั้ง 2 Class โดยมีการทดสอบการจำแนกประเภทด้วยการทดสอบแบบ Unigrams ผลลัพธ์ของค่า Accuracy ที่ 88.28% และมีค่าเอฟวันที่ 88.67 และ 87.86 จะเห็นว่าค่าเอฟวันมีความใกล้เคียงกันทั้ง 2 ประเภท

Salas และคณะ [13] เสนอกรอบการตรวจจับเหตุการณ์จราจรบนโซเชียลมีเดีย โดยเป็นการรวบรวมข้อความจากทวีตเตอร์และได้จัดประเภทเป็นข้อความทวีตที่เกี่ยวกับการรายงานสภาพจราจรและข้อความทวีตที่ไม่เกี่ยวข้องกับสภาพจราจร จากนั้นได้มีการแบ่งหมวดหมู่ออกเป็น 4 หมวดหมู่ได้แก่ งานซ่อมถนน อุบัติเหตุ ภัยพิบัติทางธรรมชาติ และการจราจร แล้วนำมาวิเคราะห์ความรู้สึกที่มีต่อเหตุการณ์จราจรทั้ง 4 รูปแบบเพื่อให้ทราบว่าผู้ใช้ถนนบริเวณนั้นรู้สึกอย่างไรต่อเหตุการณ์โดยมีการแบ่งเป็นความเครียดและความสบายใจโดยใช้การวิเคราะห์โดยใช้คำศัพท์เพื่อตรวจจับและบ่งชี้ความรู้สึกที่อยู่ในข้อความและใช้เทคนิค SentiStrength เพื่อสกัดหาคุณสมบัติของคำ

Yuanyuan Chen และ Yisheng Lv [5] นำเสนอการจำแนกประเภทข้อความที่เกี่ยวข้องกับการรายงายสภาพจราจรจากโซเชียลมีเดีย Weibo ซึ่งเป็นแพลตฟอร์มไมโครบล็อกของจีน โดยใช้เทคนิค bag of word โมเดล เพื่อค้นหาคำหลัก แล้วใช้การเรียนรู้เชิงลึกเพื่อทำการทดลองโดยใช้ซีเอ็นเอ็นและแอลเอสทีเอ็มเพื่อเรียนรู้ที่จะจดจำคำและเหตุการณ์ต่างๆ และตรวจจับเหตุการณ์การจราจร โดยในงานนี้ได้อธิบายถึงกระบวนการสร้างระบบตรวจจับข้อความด้วยการเรียนรู้เชิงลึก เริ่มจากการรวบรวมข้อความด้วยวิธีการอ่านค่าจาก HTML แล้วนำมาเก็บไว้ในฐานข้อมูล และนำข้อความนั้นมาแยกออกเป็นคำด้วยการใช้เครื่องมือวิเคราะห์คำศัพท์ภาษาจีน ต่อมาคือการเข้ารหัสข้อความด้วยการฝังคำ (word embedding) ในงานนี้ได้ใช้วิธีการ Continuous Bag of Words (CBOW) เพื่อให้ได้ข้อความที่เข้ารหัสและนำไปใช้สำหรับการสร้างโมเดลการจำแนกประเภทข้อความซึ่งได้แบ่งข้อความออกเป็น 2 ประเภท คือข้อความเกี่ยวข้องกับจราจรและข้อความที่ไม่เกี่ยวข้องกับจราจรจากข้อความทั้งหมดประมาณ 3 ล้านข้อความมีค่าเอฟวันสำหรับการแยกประเภทอยู่ที่ 0.91

Jorge Cristhian Chamby-Diaz และ Ana Bazzan [14] นำเสนอการทดลองเพื่อการจำแนกข้อความสภาพจราจรในเมือง Porto Alegre ประเทศบราซิล ได้มีการรวบรวมข้อมูลการรับส่งข้อความจากทวิตเตอร์เนื่องจากข้อความบนทวิตเตอร์นั้นเป็นข้อความสั้นที่มีความกระชับและได้ใจความจึงเหมาะ

กับการนำมาเป็นข้อมูลเพื่อตรวจจับสภาพจราจรบนถนนแต่ทวิตเตอร์ก็ยังมีข้อจำกัดหลายๆ ด้าน เช่นเดียวกับสื่อสังคมออนไลน์อื่นๆ คือ มีคำศัพท์เยอะ และมีหลายหมวดหมู่ โดยในงานนี้ได้มีการนำเสนอแนวทางการจำแนกข้อความจากทวิตเตอร์ด้วยวิธีการทำเหมืองข้อมูล (text mining) ด้วยวิธีการที่แตกต่างกันคือ Naive Bayes และ Decision Tree โดยงานนี้ได้มีการแบ่งกลุ่มข้อความออกเป็น 6 กลุ่ม แต่ละกลุ่มมีจำนวนไม่เท่ากันทำให้ผลการทดลองที่ได้มีความแม่นยำที่กระจายออกไปในทั้ง 2 วิธีด้วยความแม่นยำเฉลี่ยประมาณ 83% แต่เมื่อดูอย่างใกล้ชิดการทำนายในกลุ่มของ Traffic lights และ Weather มีความแม่นยำถึง 99% แต่หากศึกษาถึงจำนวนข้อมูลที่นำมาใช้มีจำนวนต่ำกว่ากลุ่มอื่นๆ อย่างเห็นได้ชัด สิ่งนี้บ่งบอกให้เป็นว่าการที่กลุ่มข้อความไม่เท่ากันหรือห่างกันมากๆ จะทำให้ผลการทดลองคลาดเคลื่อน

Sina Dabiria และ Kevin Heaslip [2] นำเสนอการจำแนกหมวดหมู่ของข้อความการตรวจจับอุบัติการณ์บนท้องถนนจากข้อความทวิตเตอร์ โดยแบ่งกลุ่มข้อมูลออกเป็น 3 กลุ่ม ประมาณห้าหมื่นข้อความ ประกอบไปด้วย 1) ไม่ใช่การจราจร 2) เหตุการณ์การจราจร และ 3) ข้อมูลและสภาพการจราจร โดยมีวิธีการดำเนินงานโดยย่อดังนี้ เริ่มต้นการรวบรวมข้อมูลจากบัญชีทวิตเตอร์ที่มีความน่าเชื่อถือ เช่น บัญชีทวิตเตอร์ของส่วนงานราชการและดำเนินการสร้างป้ายกำกับ (label) ให้กับข้อมูลด้วยผู้เชี่ยวชาญเฉพาะด้าน จากนั้นนำข้อความที่ได้มาผ่านกระบวนการสร้างรูปแบบการฝังคำเพื่อให้เป็นการเข้ารหัสในรูปแบบมิติต่ำ แล้วใช้รูปแบบการจำแนกข้อความด้วยวิธีการเรียนรู้เชิงลึก convolutional neural network ซีเอ็นเอ็นและ Long Short-Term Memory แอลเอสทีเอ็มโดยการทดสอบด้วยการหาค่าไฮเปอร์พารามิเตอร์ของ ซีเอ็นเอ็นและค่าไฮเปอร์พารามิเตอร์ของ แอลเอสทีเอ็มด้วยวิธี Grid Search เมื่อได้ค่าไฮเปอร์พารามิเตอร์ที่เหมาะสมแล้วจึงเริ่มสร้างโมเดลเพื่อจำแนกข้อความด้วยการแบ่งข้อมูลเพื่อเทรนและเทสเป็น 5 Fold ในงานนี้มีการสร้างโมเดลเพื่อจำแนกข้อความจากทวิตเตอร์ได้ผลลัพธ์การทำนายออกมาได้ความแม่นยำอยู่ที่ 0.986 และค่า F1 อยู่ที่ 0.974 โดยวิธีการที่ดีที่สุดจากงานนี้คือการจำแนกข้อความด้วยวิธี ซีเอ็นเอ็นที่ได้การฝังคำจากวิธีการ word2vec และเมื่อสังเกตุถึงการทดสอบการจำแนกข้อความด้วยข้อมูลแต่ละคลาสจะเห็นว่าการมีข้อมูลที่เท่ากันในทุกกลุ่มจะทำให้ประสิทธิภาพการทำนายสูงขึ้น

จากการศึกษางานวิจัยที่เป็นการนำข้อความมาจำแนกพบว่างานที่มีการใช้งานข้อมูลแต่ละกลุ่มไม่เท่ากันนั้นจะส่งผลให้ประสิทธิภาพการจำแนกกลุ่มของข้อความลดลงต่างกันกับงานที่มีการใช้งานข้อความแต่ละกลุ่มเท่ากันจะให้ประสิทธิภาพที่ดีและทำนายได้ครอบคลุมทุกกลุ่ม ดังนั้นการจำแนกข้อความจากทวิตเตอร์มักจะพบกับปัญหาความไม่สมดุลของข้อมูลอยู่เสมอ การจัดการกับความไม่สมดุลนี้มีทางเลือกหลากหลายไม่ว่าจะเป็นการจัดการที่ธรรมดาที่สุดคือ การเก็บข้อมูลกลุ่มที่มีน้อยเพิ่มแต่ข้อเสียคือต้องใช้จ่ายเวลาเพิ่มขึ้นมาก หากต้องการให้ได้ข้อมูลที่รวดเร็วจึงต้องใช้กระบวนการเรียนรู้ทางเครื่อง (machine learninng) เข้ามาช่วยจัดการกับข้อมูลที่ไม่สมดุลนั้น มีวิธีที่ใช้งานกันอันดับแรกคือการสุ่ม

เลือกไม่ว่าจะเป็น undersampling หรือ Oversamplinng หรือจะเป็นการสร้างข้อความเพิ่มเติมให้เพิ่มขึ้นด้วยวิธีการต่าง ๆ ดังเช่นการศึกษาดังนี้

Shaikh และคณะ [8] นำเสนอการสร้างข้อความด้วยวิธีการเรียนรู้เชิงลึกชื่อแอลเอสทีเอ็มมีกระบวนการโดยใช้วิธีการเรียนรู้เชิงลึกเข้ามาทำนายคำต่อไป (Next Sentence Prediction) โดยจะต้องเริ่มต้นจากคำตั้งต้นแล้วทำการสร้างคำต่อไปจนครบตามจำนวนคำที่ต้องการในประโยค จากการสำรวจพบว่าวิธีการนี้เป็นการสร้างข้อความขึ้นมากจากกลุ่มข้อความที่มีอยู่เดิมและเพิ่มเติมเข้ามาด้วยก้อนข้อมูลจากภายนอก โดยในงานนี้เป็นการนำข้อมูลจาก GPT-2 และนำมา finetune ด้วยกลุ่มข้อมูลของผู้วิจัยเอง การประเมินกระบวนการสร้างลำดับข้อความควบคู่ไปกับการตรวจสอบความถูกต้องทางไวยากรณ์ของชุดข้อมูลที่ไม่สมดุล การทดลองในบทความนี้เกี่ยวกับชุดข้อมูลที่ไม่สมดุลสูงสามชุดจากโดเมนที่แตกต่างกันแสดงให้เห็นว่าประสิทธิภาพของโมเดลโครงข่ายประสาทเทียมจะมีประสิทธิภาพดีขึ้นถึง 17% เมื่อชุดข้อมูลมีความสมดุลโดยใช้ข้อความที่สร้างขึ้น

Heinzerling และ Strube [10] นำเสนอ BPEmb ซึ่งเป็นรูปแบบการเก็บข้อมูลเป็นหน่วยคำย่อยที่ผ่านการฝึกอบรมมา 275 ภาษาที่ได้จากวิกิพีเดียรวมทั้งภาษาไทยด้วย การเข้ารหัสคู่ไบต์ (Byte Pair Encoding) เป็นการเข้ารหัสที่มีความยาวผันแปรได้ ซึ่งจะดูข้อความเป็นลำดับของสัญลักษณ์ และผสานคู่สัญลักษณ์ที่ใช้บ่อยที่สุดซ้ำๆ ให้เป็นสัญลักษณ์ใหม่ เช่น การเข้ารหัสข้อความภาษาอังกฤษอาจประกอบด้วยการรวมคู่สัญลักษณ์ที่พบบ่อยที่สุดเข้ากับสัญลักษณ์ใหม่ก่อน จากนั้นจึงรวมคู่สัญลักษณ์ดังกล่าวเข้าด้วยกันในการวนซ้ำครั้งต่อไป เป็นต้น เนื่องจากอัลกอริทึม BPE ทำงานร่วมกับลำดับสัญลักษณ์ใดๆ ก็ตาม โดยไม่ต้องมีโทเค็นไว้ก่อน การฝังคำแบบ Byte-Pair มีขนาดเล็กมากเมื่อเทียบกับงานอื่นเนื่องจากไม่ต้องทำ tokenization

Lowphansirikul และคณะ [15] ได้พัฒนา Pre-trained Thai Language โมเดล ภายใต้ชื่อ WangchanBERTa ที่ถูกเทรนด้วยข้อมูลภาษาไทยขนาดใหญ่มีการนำไปใช้งานหลายรูปแบบด้วยกัน สำหรับงาน text augmentation ใช้เทคนิค Masked Language โมเดล (MLM) คือการหาคำเสริมจากการพิจารณาบริบทได้จากทุกคำที่อยู่รอบๆ ที่มีอยู่ในโมดูล Thai2transformersAug

บทที่ 3
งานวิจัยที่นำเสนอ

การจำแนกข้อความนับเป็นงานที่มีความท้าทายโดยมีการพัฒนาตัวแยกประเภทในรูปแบบต่าง ๆ ไม่ว่าจะเป็นการสร้างตัวแยกประเภทด้วยวิธีการค้นหาคำที่ต้องการ (text search) แต่เมื่อข้อความมีมากขึ้นคำมีมากขึ้นการใช้วิธีการค้นหาคำอาจจะทำได้ลำบากจึงได้มีวิธีการทางเครื่องเข้ามาช่วยเนื่องจากคอมพิวเตอร์ปัจจุบันเอื้อต่อการสร้างตัวจำแนกเหล่านี้ และเมื่อมีการวิจัยแพร่หลายมากขึ้นการสร้างตัวจำแนกข้อความจึงพัฒนาอย่างรวดเร็วจนมาถึงการนำการเรียนรู้เชิงลึกมาใช้ในการสร้างตัวจำแนกข้อความ ทั้งนี้ไม่ว่าจะเป็นการ การเรียนู้ทางเครื่อง หรือการเรียนรู้เชิงลึก ก็ตามล้วนแล้วแต่ต้องการข้อมูลเพื่อมาสอนตัวจำแนก (นำมาเทรนโมเดล) แต่เมื่อมีการสร้างตัวจำแนกที่หลากหลายขึ้นกลับพบว่าการสร้างตัวจำแนกแบบหลายกลุ่มที่มาจากกลุ่มข้อมูลมูลจำนวนไม่เท่ากันยังมีจุดบกพร่อง คือ โมเดลจำแนกข้อความของกลุ่มที่มีจำนวนน้อยผิดพลาด โดยดูได้จากค่าเอฟวันของกลุ่มที่มีจำนวนน้อยจะมีค่าต่ำ ดังนั้นงานวิจัยนี้นำเสนอวิธีการที่จะนำมาปรับปรุงข้อด้อยนี้ด้วยการสร้างกลุ่มข้อมูลให้มีจำนวนน้อยให้เพิ่มมากขึ้นเท่ากับกลุ่มที่มีจำนวนข้อมูลมาก และเมื่อสร้างข้อความเสร็จแล้วจะมีการทดสอบว่าข้อความสร้างขึ้นมามีความหมายที่อ่านแล้วเข้าใจหรือไม่ ทั้งยังมีการสร้างโมเดลขึ้นมาเพื่อทดสอบการจำแนกข้อความเพื่อดูการเปลี่ยนแปลงของค่าเอฟวันว่ามีการเปลี่ยนแปลงไปหรือไม่และสุดท้ายจะสรุปได้ว่ากระบวนการที่นำเสนอนั้นสามารถให้ประสิทธิภาพในการจำแนกข้อความที่สูงขึ้นได้

3.1 ขั้นตอนการวิจัย

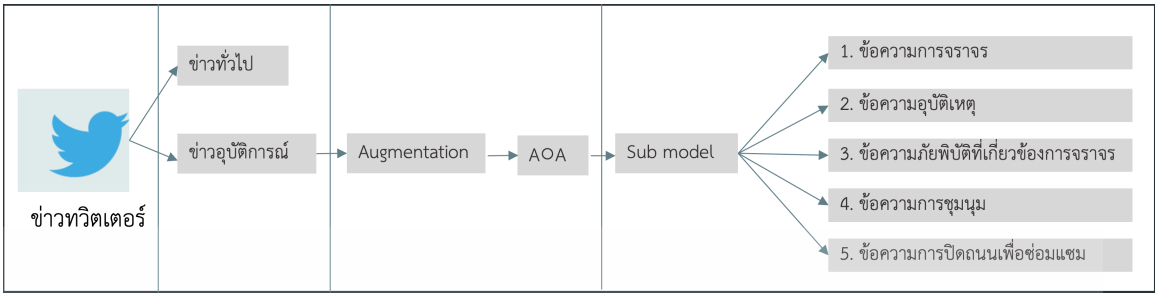
การนำเทคนิคการเรียนรู้เชิงลึกมาใช้ในการจำแนกข้อความเกี่ยวกับสภาพจราจรเป็นวิธีที่มีศักยภาพในการปรับปรุงการบริหารจัดการทางการจราจรและนำเสนอข้อมูลที่มีคุณค่าให้กับสาธารณชนและผู้บริหาร ดังนั้น งานวิจัยในเรื่องนี้มีความสำคัญอย่างมาก โดยนำเสนอขั้นตอนการดำเนินงานวิจัยดังนี้

ขั้นตอนที่ 1 รวบรวมข้อมูลข้อความจากทวิตเตอร์

ขั้นตอนที่ 2 การเตรียมข้อมูลให้พร้อมสำหรับการนำมาใช้งาน เช่น การทำความสะอาดข้อมูล และการระบุของข้อความข่าวทั่วไปและข่าวสภาพจราจร

ขั้นตอนที่ 3 การนำข้อมูลมาผ่านกระบวนการเพื่อให้ข้อมูลแต่ละกลุ่มมีปริมาณข้อมูลที่เท่ากัน โดยใช้วิธีการต่าง ๆ

ขั้นตอนที่ 4 การสร้างโมเดลเพื่อการจำแนกข้อความด้วยเทคนิกการเรียนรู้เชิงลึกซีเอ็น



รูปที่ 3.1 ขั้นตอนการดำเนินงาน

3.2 การรวบรวมข้อความจากทวิตเตอร์และจัดแบ่งกลุ่มข้อความ

ส่วนนี้จะเป็นการนำเสนอวิธีการได้รับข้อความจากทวิตเตอร์ด้วยทวิตเตอร์เอพีไอ (API) รวบรวมข้อความมาจากหน่วยงานที่เกี่ยวข้องกับการประชาสัมพันธ์การจราจรทางการของประเทศไทยประกอบไปด้วย js100radio fm91trafficpro Traffic_1197 และ motorway_th ทั้งสิ้น 20,000 ข้อความโดยเป็นการเก็บย้อนหลังไปในทุกบัญชี และทำป้ายกำกับด้วยพนักงานที่เชี่ยวชาญ แบ่งออกเป็นข้อมูลสองระดับ ระดับแรกการจำแนกข้อความที่เกี่ยวกับการรายงานสภาพจราจรและข้อความที่ไม่เกี่ยวกับการรายงานสภาพจราจรระดับที่สองคือการจำแนกประเภทของข้องความที่เกี่ยวกับการรายงานสภาพจราจรแบ่งออกเป็น 5 กลุ่ม

3.2.1 การรวบรวมข้อมูลจากทวิตเตอร์

ทวิตเตอร์มีการเปิดให้นักพัฒนาสามารถเข้าถึงข้อมูลการทวิต (tweet) ได้จากเครื่องมือที่เรียกว่าทวิตเตอร์เอพีไอ (Twitter API) ซึ่งได้เปิดตัวอย่างเป็นทางการและเปิดให้นักพัฒนาใช้งานมาตั้งแต่ปี 2006 โดยเวอร์ชัน (version) แรกที่เปิดใช้งานคือทวิตเตอร์เอพีไอ v1 ในปี 2012 และในปี 2020 ทวิตเตอร์ได้เปิดตัวทวิตเตอร์เอพีไอ เวอร์ชันล่าสุดคือทวิตเตอร์เอพีไอ v2 ซึ่งเป็นเวอร์ชันปัจจุบัน สำหรับเวอร์ชันปัจจุบันมีการเพิ่มคุณสมบัติต่าง ๆ เข้ามาเพื่อจัดการการเข้าถึงของนักพัฒนา โดยมีการแบ่งเป็น 3 ระดับดังนี้

- ระดับมาตรฐาน (Standard) สำหรับระดับมาตรฐานเป็นระดับแรกสุดที่จะให้นักพัฒนาเข้าใช้งาน API ได้เหมาะสำหรับการเริ่มต้นการพัฒนาประกอบไปกรณีศึกษาที่ดี ซึ่งเหมาะสำหรับการเรียนรู้หรือสอน

- ระดับงานวิจัยทางวิชาการ (Academic Research) สำหรับระดับงานวิจัยทางวิชาการ นักวิจัยสามารถใช้ Twitter API เพื่อทำความเข้าใจข้อความหรือบทสนทนาที่เปิดเผยสู่สาธารณะและในอนาคตจะมีการคัดเลือกนักวิจัยที่มีแนวทางในการสร้างสรรค์หรือปรับปรุงระบบต่าง ๆ และได้เตรียมเครื่องมือหรือคำแนะนำเพื่อให้ง่ายต่อการดำเนินงานในระดับงานวิจัยทางวิชาการ
- ระดับธุรกิจ (Business) นักพัฒนาที่กำลังสร้างสรรค์ธุรกิจบน Twitter API ได้มีการเพิ่มพันธมิตรทางธุรกิจและข้อมูลลูกค้าเข้ามาในข้อมูล API นี้

ในส่วนของหน้ารวมนักพัฒนา (Developer portal) มีการปรับปรุงรูปแบบใหม่เพื่อเป็นเครื่องมือให้นักพัฒนาสามารถสร้างโครงการ (Project) และสารมารถเข้าถึงส่วนสนับสนุนได้จากส่วนการจัดการนี้อีกทาง รูปแบบของการเรียกข้อมูลจาก Twitter API สามารถทำได้โดยการระบุรายละเอียดของ Project ที่ถูกสร้างขึ้นภายใน Portal ประกอบไป

- API Key คือ คีย์เพื่อการอนุญาตให้เข้าถึง App ที่สร้างไว้ใน portal
- API Key Secret คือ คีย์เพื่อบ่งบอกรหัสผ่านและการอนุญาตการเข้าถึง App ที่สร้างไว้ใน portal
- Access Token คือ โทเคนเพื่อแสดงถึงบัญชีของเจ้าของ App ที่สร้างไว้ใน Portal และช่วยในการเข้าถึงบัญชี Twitter ได้
- Access Token Secret คือ รหัสลับของโทเคนเพื่อแสดงถึงบัญชีของเจ้าของ App ที่สร้างไว้ใน Portal และช่วยในการเข้าถึงบัญชี twitter
- Bearer Token คือโทเคนที่ช่วยให้ app สามารถตรวจสอบคำขอร้องเพื่อการตรวจสอบสิทธิ์ความปลอดภัยในระดับ OAuth2.0

เมื่อดำเนินการเกี่ยวกับคีย์ (Key) ต่าง ๆ ถูกต้องแล้วจะได้รับข้อมูลการตอบกลับมาจากเอพีไอ โดยนักพัฒนาระดับมาตรฐาน (Standard) จะสามารถเรียกข้อความได้ 300 ครั้งต่อ 15 นาที ต่อหนึ่ง app ที่สร้างไว้และสามารถเรียกข้อความทวิตได้ 500,000 ครั้งต่อเดือน โดยข้อมูลที่ได้จากเอพีไอจะมีรูปแบบเป็นเจซัน (JSON) ซึ่งข้อมูลที่ได้มาประกอบเป็นออฟเจ็ค (Object) ที่ถูกบรรจุข้อมูลการทวิตของผู้ใช้งานที่นำมาใช้ในงานวิจัย ดังนี้ created_at หมายถึง วันที่ เดือน ปี ของข้อความที่ถูกทวิต, id หมายถึง เลขไอดีของข้อความนี้, full_text หมายถึง ข้อความทั้งหมดที่ถูกทวิต, entities.hashtags หมายถึง แฮสแท็กของทวิตนี้, user.screen_name หมายถึง ชื่อบัญชีที่โพสทวิตนี้

```
_json={
  'created_at': 'Wed Mar 03 14: 27: 06 +0000 2021',
  'id': 1367119345948614658,
  'id_str': '1367119345948614658',
  'full_text': '21.26น. ถ.สาทรใต้  มุ่งหน้า แยกสาทร-สุรศักดิ์  ห้ายอยู่หน้าโรงพยาบาล เซนต์หลุยส์ #รายงานจราจร #รถติด #FM91',
  'truncated': False,
  'display_text_range': [
    0,
    106
  ],
  'entities': {
    'hashtags': [
      {
        'text': 'รายงานจราจร',
      },
    ],
  },
  'user': {
    'screen_name': 'fm91trafficpro',
    'location': '',
    'description': 'สถานีวิทยุข่าวจราจรและความปลอดภัย\nกสทช.91 เราพึ่งได้ 1644 โทรฟรีทั่วประเทศ ตลอด 24 ชั่วโมง',
```


รูปที่ 3.2 รายละเอียดข้อมูลเจสันที่ได้จากทวิตเตอร์เอพีไอ

สำหรับการเรียกข้อมูลจากทวิตเตอร์อัตโนมัติได้ใช้โปรแกรมภาษาไพธอน (Python) สำหรับการพัฒนาโดยได้นำเอาไลบรารี Tweepy หรือทวีพี เพื่ออำนวยความสะดวกในการทำงานโดยมีคำสั่งในการเรียกข้อมูลจากทวิตเตอร์เอพีไอคือฟังก์ชัน “tweepy.Cursor()” และมีการส่งพารามิเตอร์ที่ต้องการคือ “api.user_timeline” หมายถึงประเภทที่ต้องการข้อมูลจากทวิตเตอร์สำหรับงานวิจัยครั้งนี้ต้องการ Timeline ของผู้ใช้งานย้อนหลัง ต่อมาคือพารามิเตอร์ “screen_name” หมายถึงชื่อผู้ใช้งานเป้าหมายที่ต้องการข้อมูล “tweet_mode” หมายถึงรูปแบบในการเรียกข้อมูลให้ตั้งค่าเป็น “extended” คือการตั้งค่าให้เรียกข้อมูลมาทั้งหมด และสามารถตั้งค่าจำนวนข้อความที่ต้องการเรียกในแต่ละรอบคือการตั้งค่าฟังก์ชัน “item()”

```
screen_name =  js100radio
ID =  1434142955095445520
Geo =  None
Message =  คนขับมีอาการชักเกร็ง ทารถบรรทุกปีนข้ามเกาะกลางชนโกดังเก็บของ บาดเจ็บ 2 คน

https://t.co/D32h2RNnuf #JS100 https://t.co/nmXt1Rw1v9
วันที่ =  2021-09-04 20:15:00
--------------------------------
```

รูปที่ 3.3 ผลลัพธ์การเรียกข้อมูลจากทวิตเตอร์เอพีไอ-ทวีพี

ข้อความที่ได้นำมาศึกษาคือข้อความที่ได้มาจากทวิตเตอร์ โดยภายในข้อความของทวิตเตอร์นั้นมีองค์ประกอบของข้อมูลหลายองค์ประกอบ ซึ่งสามารถนำข้อมูลเหล่านั้นมาใช้งานจำแนกและวิเคราะห์

ความหมายได้ โดยส่วนประกอบจากแพลตฟอร์มดังกล่าวจะประกอบไปด้วยองค์ประกอบหลักดังนี้ คือ ชื่อของผู้โพสต์ข้อความ (Name) ข้อความ (Content) เวลา (Date-Time) และแฮชแท็ก (Hashtag) ดังแสดงในรูป 3.3

ทวิตเตอร์ที่มีในปัจจุบันนั้นมีผู้ใช้งานหลากหลายทั้งที่เป็นทวิตเตอร์รายบุคคล หรือทวิตเตอร์ระดับองค์กร โดยในงานวิจัยนี้พิจารณาเลือกใช้ทวิตเตอร์ในระดับองค์กรที่เกี่ยวข้องกับการรายงานสภาพจราจรสำหรับประเทศไทย จำนวน 4 หน่วยงาน ได้แก่

1. @js100radio เป็นบัญชีทวิตเตอร์ที่มีรากฐานมาจากการรายงานข่าวทางสื่อวิทยุต่อมาได้พัฒนาเพิ่มเติมส่วนของการรายงานข่าวผ่านสื่อสังคมออนไลน์ภายใต้การดูแลของ บริษัท แปซิฟิค คอร์ปอเรชั่น จำกัด มีผู้ติดตามกว่า 3 ล้านผู้ใช้งาน

2. @Traffic_1197 เป็นบัญชีทวิตเตอร์ที่อยู่ภายใต้การดูแลของกองบังคับการตำรวจจราจร (บก. 02) ซึ่งเป็นหน่วยงานของราชการ มีจำนวนผู้ติดตามกว่า 2 หมื่นผู้ใช้งาน

3. @fm91trafficpro เป็นบัญชีทวิตเตอร์ของหน่วยงานราชการ คือ สถานีวิทยุพิทักษ์สันติราษฎร์ สวพ. FM91 อยู่ภายใต้การดูแลของกองตำรวจสื่อสาร สำนักงานตำรวจแห่งชาติ มีจำนวนผู้ติดตามกว่า 2 ล้านผู้ใช้งาน

4. @motorway_th เป็นบัญชีทวิตเตอร์ของหน่วยงานราชการ คือ ฝ่ายควบคุมและสั่งการจราจรกองทางหลวงพิเศษระหว่างเมือง กรมทางหลวง ผู้ติดตามกว่า 9 หมื่นผู้ใช้งาน

รูปที่ 3.4 หน้าจอแสดงผลและองค์ประกอบการแสดงข้อมูลข่าวสาร

3.2.2 การแบ่งกลุ่มข้อความ

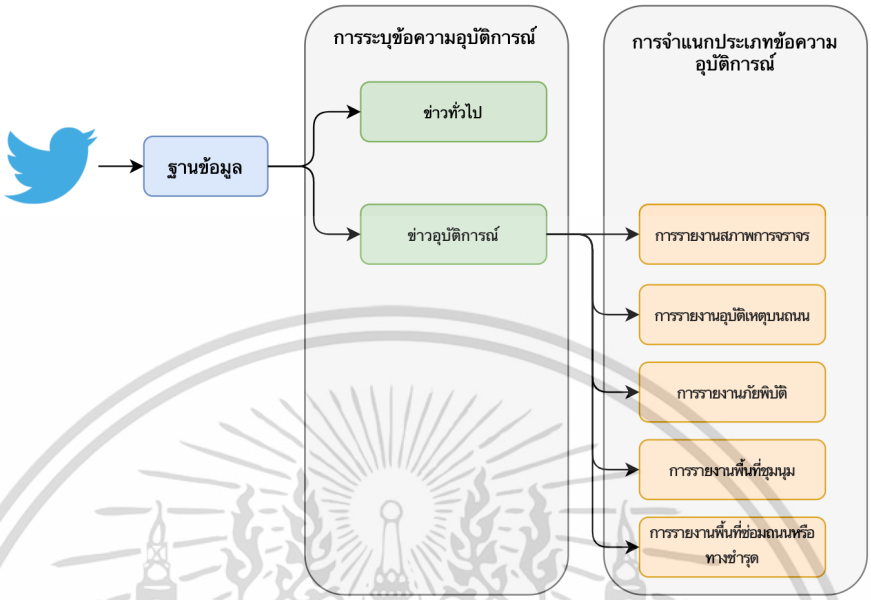
ข้อความสภาพจราจรที่ได้รวบรวมจากทวิตเตอร์เป็นการรวบรวมข้อความมาทั้งหมด โดยข้อความนั้นจะประกอบไปด้วยข้อความที่เป็นการรายงานสภาพจราจรซึ่งเป็นข้อความที่ต้องการนำมาวิเคราะห์เพื่อจำแนกประเภทของข้อความและนอกจากนั้นยังมีข้อความอื่น ๆ ที่ไม่เกี่ยวกับการรายงานสภาพจราจร ดังนั้นการจำแนกข้อความสำหรับงานวิจัยในครั้งนี้จะมีการแบ่งข้อความเป็นสองระดับ ดังรูปที่ 3.5 คือการจำแนกข้อความที่ไม่ใช่การรายงานสภาพจราจรกับข้อความที่เกี่ยวกับการรายงานสภาพจราจร ดังตารางที่ 3.1 และการจำแนกประเภทของข้อความสภาพจราจรถูกแบ่งออกเป็น 5 ประเภท ดังตารางที่ 3.2

ตารางที่ 3.1 การระบุข้อความอุบัติการณ์

อ้างอิง	ความหมาย	คำอธิบาย
1	ข้อความที่ไม่เกี่ยวกับการรายงานสภาพการจราจรหรือข้อความทั่วไป	คือการรายงานเหตุการณ์ทั่ว ๆ ไป เช่น การประกาศประชาสัมพันธ์ ๆ ข่าวการเมือง กีฬา เป็นต้น
2	การรายงานข้อความที่เกี่ยวกับสภาพจราจรหรือข้อความอุบัติการณ์	คือ การรายงานเหตุการณ์ที่จะส่งผลกระทบต่อสภาพจราจรบนถนนทั้งหมด

ตารางที่ 3.2 การจำแนกประเภทข้อความอุบัติการณ์

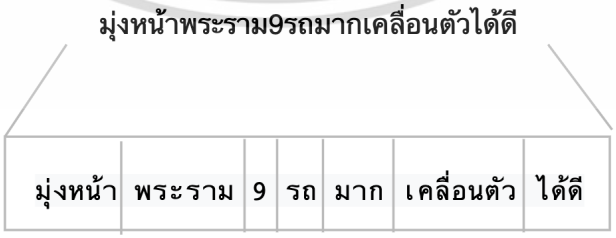
อ้างอิง	ความหมาย	คำอธิบาย
1	การรายงานสภาพการจราจร	คือการรายงานเหตุการณ์จราจรที่กำลังเกิดขึ้น ณ ขณะเวลานั้น ๆ หรือการแจ้งสถานการณ์ สิ้นสุดการจัดการเหตุการณ์ในพื้นที่ เป็นต้น
2	การรายงานอุบัติเหตุบนถนน	คือ เหตุการณ์ไม่พึงประสงค์ที่ก่อให้เกิดการกีดขวางการจราจรในช่วงระยะเวลาหนึ่ง เช่น รถชน รถเสีย
3	การรายงานภัยพิบัติ	คือ เหตุการณ์ที่เกี่ยวกับภัยทางธรรมชาติที่มีผลต่อการจราจร เช่น น้ำท่วม น้ำขัง ดินถล่ม ต้นไม้ล้ม
4	การรายงานพื้นที่ชุมนุม	คือ เหตุการณ์ที่มีผู้คนอยู่บนถนนหรืออยู่ข้างถนนแล้วล้นออกมาบนถนนจนให้การจราจรไม่ปกติ เช่น การชุมนุมประท้วง งานกิจกรรม
5	การรายงานพื้นที่ซ่อมถนนหรือทางชำรุด	เหตุการณ์ที่มีผลกระทบต่อการจราจรบนผิวจราจรที่ทำให้ผู้ขับขี่ต้องปรับเปลี่ยนพฤติกรรมหรือความเร็ว เช่น หลุมบนถนน หรือ การปิดกั้นช่องจราจรเพื่อการดำเนินงานบำรุงรักษา เป็นต้น



รูปที่ 3.5 การแบ่งกลุ่มข้อความ

3.3 การเตรียมข้อมูลสำหรับการทดลอง

ในการประมวลภาษาธรรมชาติหรือเอ็นแอลพีจากงานวิจัยที่มีการพิสูจน์มาแล้วมีวิธีการที่จำเป็นอย่างมากคือการแบ่งประโยคหรือข้อความออกเป็นส่วน ๆ กล่าวคือแบ่งประโยคออกเป็นคำ ๆ ซึ่งมีเทคนิคที่เรียกว่าการตัดคำ (word segmentation) โดยเป็นขั้นตอนแรกสุดในการประมวลผลคำทางภาษา ซึ่งสำหรับงานวิจัยนี้ได้ใช้ไพธอนในการพัฒนาและได้นำเอาไลบรารีไพไทยเอ็นแอลพีที่มีฟังก์ชัน “word tokenize” สามารถใช้งานได้ทันที โดยในงานวิจัยนี้ได้ใช้วิธีการ (engine) ตัดคำที่ชื่อ “newmm”

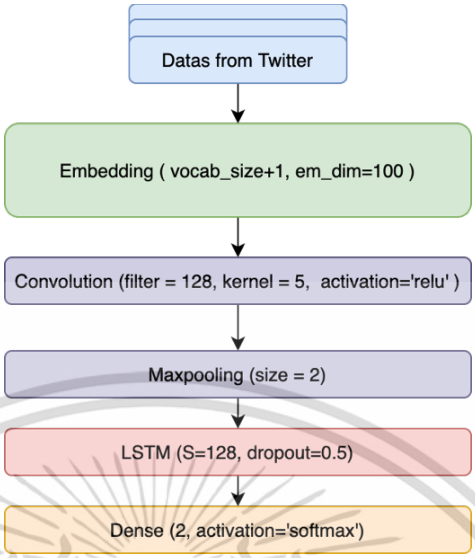


รูปที่ 3.6 การตัดคำจากข้อความ

เนื่องจากในข้อความที่นำมาประมวลผลยังมีอักษรหรืออักขระที่ไม่มีความหมายในทางการประมวลผลตัวอย่างเช่นมีเครื่องหมาย ‘#’ หรือเครื่องหมาย ‘/’ ซึ่งในทางการประมวลผลแบบภาษาธรรมชาตินั้นไม่มีความหมายเลยต้องดำเนินการคัดกรองคำ (Text Filter) ซึ่งในงานวิจัยนี้ได้ใช้วิธีการค้นหานิพจน์ปรกติ (Regular expression) จากข้อความที่ผ่านการตัดคำมาแล้วข้อความเหลือเพียงแค่ข้อความที่เป็นตัวเลข 0-9 และ ก-๙ เท่านั้น

3.4 การแบ่งประเภทข่าวทั่วไปกับข่าวอุบัติการณ์

จากข้อความที่รวบรวมได้จากทวิตเตอร์ที่มีจำนวน 20,000 เพื่อเป็นการกรองข้อความในขั้นแรกจะเป็นการนำข้อความมาเทรนโมเดลด้วยการแยกข้อความออกเป็น 2 กลุ่ม คือกลุ่มข้อความที่ไม่เกี่ยวกับการรายงานสภาพการจราจรหรือข้อความทั่วไปและการรายงานข้อความที่เกี่ยวกับสภาพจราจรหรือข้อความอุบัติการณ์ ดังรูปที่ 3.7 ในส่วนของการจำแนกข้อความอุบัติการณ์นั้นได้ออกแบบให้มีการนำวิธีการเรียนรู้เชิงลึกซีเอ็นเอ็นผสานแอลเอสทีเอ็มเข้ามาเป็นตัวแยกประเภท โดยเริ่มจากการนำข้อความที่รวบรวมมาจากทวิตเตอร์แล้วที่ผ่านการบวนการเตรียมข้อมูลในขั้นตอนก่อนหน้ามาเข้าสู่การสร้างโมเดลโดยในการสร้างโมเดลจะนำข้อความมาป้อนเข้าสู่กระบวนการในแต่ละชั้น โดยในชั้นแรกเป็นการสร้างคุณลักษณะให้กับข้อความด้วยการเข้ารหัสแบบฝังคำโดยมีการปรับแต่งไฮเปอร์พารามิเตอร์ (hyper parameter) ขนาด $V \times N$ โดยที่ V คือขนาดของคำศัพท์ (Vocabulary) ที่มีทั้งหมดคูณกับขนาดของมิติ N จากนั้นนำข้อความที่ผ่านการเข้ารหัสนั้นสู่ชั้นของคอนโวลูชันเอ็นเอ็นที่มีฟิวเตอร์เป็น 128 ความกว้างมิติ (kernel) ที่ 5 คำต่อครั้ง ชั้นต่อมาเป็นชั้นของการลดขนาดข้อความเพื่อลดความซับซ้อนด้วยการทำให้เป็นค่าเดียวด้วยวิธี Maxpooling ขนาดเป็น 2 คำ คือการนำค่ามากสุดมา และในชั้นต่อมาเป็นการใช้แอลเอสทีเอ็มโดยการตั้งค่าพารามิเตอร์คือจะทำการตั้งค่าเป็น S=128 คือค่าชั้นของ hidden layer และชั้นสุดท้ายเป็นชั้นรวมโครงข่ายเพื่อตัดสินใจออกมาเป็นคำตอบโดยให้ค่าพารามิเตอร์เป็นจำนวนเท่ากับค่าของกลุ่มทั้งหมด ในที่นี้คือ 2 กลุ่ม



รูปที่ 3.7 โมเดลการแบ่งข้อความข่าวทั่วไปออกจากข่าวอุบัติการณ์

3.5 การสร้างข้อความเพิ่ม

3.5.1 การสร้างข้อความด้วยวิธีลูกโซ่มาร์คอฟ

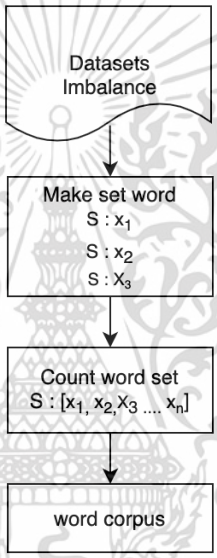
การสร้างข้อความใหม่ขึ้นมาจากกลุ่มข้อมูลที่ได้รวบรวมมาเป็นวิธีการที่มีความสำคัญในการพัฒนาและปรับปรุงระบบเสนอข้อมูลแบบอัตโนมัติ เช่น แชทบอท (chatbot) ระบบแนะนำเนื้อหา หรือการสร้างคำค้นหา โดยในงานนี้จะเป็นการนำลูกโซ่มาร์คอฟมาใช้งานในส่วนของการสร้างข้อความในกลุ่มที่มีจำนวนน้อยให้มีจำนวนเท่ากับกลุ่มข้อมูลที่มีจำนวนมาก ในบทความนี้ผู้วิจัยจะนำเสนอวิธีการใช้ลูกโซ่มาร์คอฟในการสร้างข้อความใหม่ขึ้นมา

ลูกโซ่มาร์คอฟเป็นโมเดลสถิติที่สามารถใช้ในการจำลองการเคลื่อนไหวของข้อมูลแบบลำดับ โดยวิธีการทำงานคือหากมีชุดข้อมูลเริ่มต้น (state) และตารางความน่าจะเป็นการเปลี่ยนสถานะ (transition probabilities) ระหว่างสถานะต่าง ๆ จะสามารถใช้ลูกโซ่มาร์คอฟเพื่อสร้างลำดับข้อมูลใหม่ได้ โดยการเลือกสถานะถัดไปที่ขึ้นอยู่กับสถานะปัจจุบัน และคำนวณความน่าจะเป็นที่จะเปลี่ยนสถานะไปยังสถานะถัดไป สามารถยกตัวอย่างการใช้ลูกโซ่มาร์คอฟในการสร้างข้อความใหม่ ดังนี้

สถานะ (States) หมายถึงสิ่งที่แทนคำในลำดับข้อความที่กำลังสร้างด้วยลูกโซ่มาร์คอฟ สถานะเป็นสิ่งที่มีความสำคัญในข้อความที่กำลังสร้างขึ้น เช่นข้อความ “มุ่ง หน้า พระราม9 รถ มาก เคลื่อน ตัว ได้ ดี” โดยในแต่ละขั้นตอนของลูกโซ่มาร์คอฟจะต้องเลือกสถานะปัจจุบันจากลำดับข้อมูลและใช้ความน่าจะเป็นในการเลือกสถานะถัดไป การสร้างข้อความจะเริ่มต้นจากสถานะเริ่มต้นและจะเลือกสถานะถัดไปตาม

ความน่าจะเป็นที่กำหนดในตารางความน่าจะเป็นของลูกโซ่มาร์คอฟ ทำแบบนี้วนไปจนกว่าจะได้ข้อความที่กำหนด

ตารางความน่าจะเป็นการเปลี่ยนสถานะ (transition probabilities) ตารางนี้ระบุความน่าจะเป็นที่จะเปลี่ยนไปจากสถานะหนึ่งไปสู่อีกสถานะหนึ่งในช่วงเวลาหนึ่งหนึ่ง โดยการเปลี่ยนสถานะบอกโดยค่าความน่าจะเป็นที่แต่ละคู่จะเปลี่ยนสถานะ การสร้างตารางความน่าจะเป็นสามารถสร้างได้จากการนับความถี่ของคำทั้งหมดที่มีแล้วหาอัตราคำต่อไปที่จะเกิดขึ้นมาใส่ในตาราง การสร้างเซ็ตของความน่าจะเป็นที่จะเกิดคำต่อไปมีกระบวนการดังรูปที่ 3.8

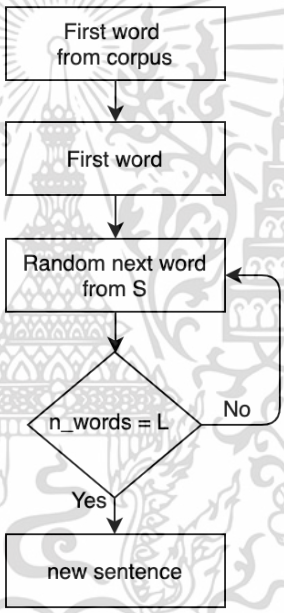


รูปที่ 3.8 กระบวนการสร้างเซ็ตของคำเพื่อระบุความน่าจะเป็น

จากรูปที่ 3.8 การสร้างคู่คำกระทำได้โดยเริ่มจากการอ่านจากข้อมูลเดต้าเซ็ต (Datasets) อาจจะอยู่ในรูปแบบซีเอสวี (CSV) หรืออื่น ๆ จากนั้นวนลูปเพื่อรับค่าคำเริ่มจากคำแรกแล้วสร้างคู่คำ (corpus[i] และ corpus[i+1]) ออกมาแล้วเพิ่มคู่คำนั้นไปในตัวแปร S จะได้คู่คำและนำคู่คำที่ได้มานับแล้วนำไปหารกับคำทั้งหมดเพื่อหาค่าความน่าจะเป็นตามสมการที่ 2.3

ตารางที่ 3.3 ตัวอย่างตารางความน่าจะเป็น

คำปัจจุบัน	คำถัดไป	ความน่าจะเป็น
มุ่ง	หน้า	0.4
	พระราม9	0.3
	รถ	0.2
	มาก	0.1
หน้า	พระราม9	0.5
	รถ	0.3
	มาก	0.1

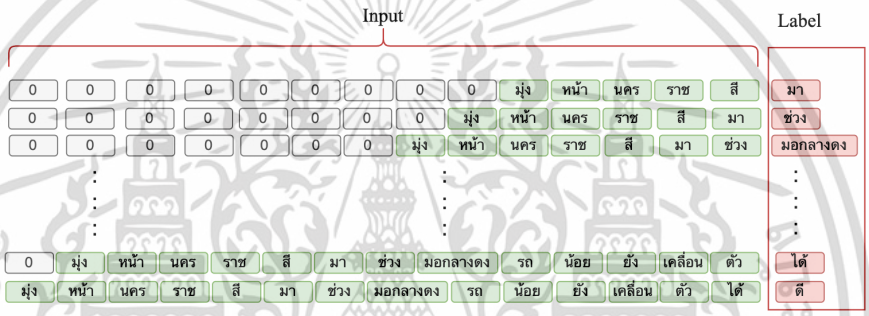


รูปที่ 3.9 การสร้างข้อความด้วยวิธีลูกโซ่มาร์คอฟ

จากรูปที่ 3.9 นำข้อมูลที่ได้อธิบายในตารางที่ 3.3 จะสามารถเลือกคำถัดไปในข้อความ “มุ่ง หน้า พระราม 9 รถ มาก เคลื่อน ตัว ได้ ดี” โดยสุ่มตามความน่าจะเป็นที่กำหนดในตาราง เช่น หากเริ่มที่คำว่า “มุ่ง” จะมีโอกาสที่ 0.4 ที่จะเลือก “หน้า” เป็นคำถัดไป หรือ 0.3 ที่จะเลือก “พระราม9” เป็นคำถัดไป และสามารถดำเนินการเช่นนี้ต่อไปเรื่อย ๆ เพื่อสร้างข้อความให้ได้เท่ากับความยาวของจำนวนคำที่ต้องการ

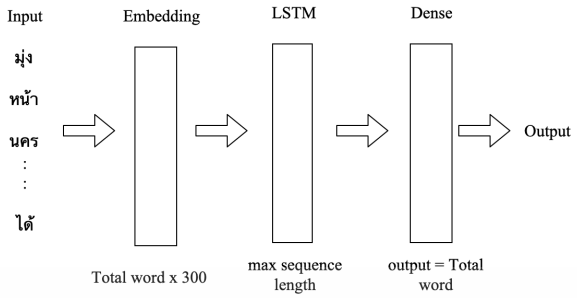
3.5.2 การสร้างข้อความด้วยวิธีการเรียนรู้เชิงลึกด้วยแอลเอสทีเอ็ม

วิธีการแอลเอสทีเอ็มเป็นกระบวนการสร้างข้อความเพิ่มอีกวิธีการที่ได้ศึกษาเริ่มจากการทำความสะอาดข้อมูลก่อนเช่นเดียวกับกระบวนการอื่น ๆ คือ การตัดคำ การกรองคำ การลบคำหยุด การแปลงคำเป็นตัวเลข และการทำให้ข้อความยาวเท่ากันด้วยวิธีเติมข้อความ (padding หรือแพดดิง) ถึงตรงนี้จะมีความแตกต่างจากเดิมคือการแพดดิงจะต้องเติมคำไปข้างหน้าเท่านั้น (pre-padding หรือพรี-แพดดิง) เนื่องจากการสร้างข้อความด้วยกระบวนการนี้จะต้องใช้คำสุดท้ายของแต่ละลำดับไปสร้างเป็นคำตอบของชุดลำดับย่อย โดยลำดับย่อยแรกประกอบไปด้วย 1 คำแรกของประโยคที่ได้จากการสุ่มมาจากคำแรกของทุกประโยคในข้อมูล และลำดับย่อยถัดไปคือ 3 คำแรกของประโยค เป็นต้น ดังรูปที่ 3.10



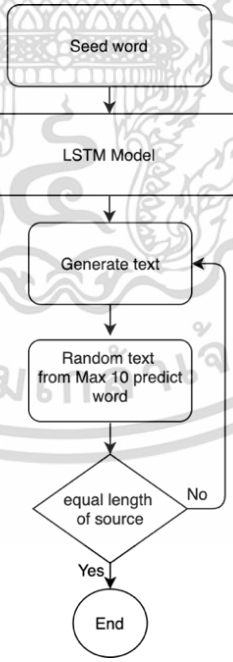
รูปที่ 3.10 ประโยคย่อยหลังทำพรี-แพดดิง

เทคนิคแอลเอสทีเอ็มเป็นการเรียนรู้เชิงลึกที่ถูกพัฒนามามากจากอาร์เอ็นเอ็นโดยเพิ่มการตัดสินใจเลือกข้อมูลข้างหลังหรือข้างหน้าข้อมูลใดเพื่อมาประกอบในการทำนายผลในคำต่อไปโดยการสร้างโมเดลของเทคนิคแอลเอสทีเอ็มมีส่วนประกอบหลักคือชั้นฝังคำ (embedding layer) ตามด้วยชั้นแอลเอสทีเอ็มและชั้นรวมโหนด (dense layer) เพื่อรวมคำทำนายเป็นค่าสุดท้ายการกำหนดชั้นฝังคำจะมีพารามิเตอร์เป็นจำนวนคำทั้งหมดตามจำนวนคำของข้อความต้นฉบับ และถัดมาเป็นแอลเอสทีเอ็มโดยจะกำหนดความยาวของแอลเอสทีเอ็มด้วยค่าความยาวของประโยค และสุดท้ายเป็นชั้นรวมโหนดจะกำหนดพารามิเตอร์เป็นจำนวนคำทั้งหมดและใช้ซอฟต์แม็กซ์ (Softmax) เป็นตัวเลือกคำตอบสุดท้ายเพื่อทำนายคำถัดไป ดังรูปที่ 3.11



รูปที่ 3.11 การสร้างโมเดลเพื่อสร้างข้อความใหม่ด้วยวิธีแอลเอสทีเอ็ม

หลังจากได้โมเดลเพื่อการสร้างข้อความใหม่แล้ว จากนั้นนำโมเดลนั้นมาดำเนินการสร้างข้อความเพิ่มในกลุ่มที่มีข้อความน้อยให้มีจำนวนเพิ่มมากขึ้นเท่ากับกลุ่มที่มีข้อความจำนวนมาก โดยการสร้างข้อความจะเริ่มต้นด้วยการสุ่มคำจากกลุ่มคำต้นประโยคของข้อความมาใช้เป็นคำตั้งต้น (Seed word) แล้วนำมาทำนายคำต่อไปโดยเริ่มจากการคำแรกวนรอบไปจนกว่าจะครบตามจำนวนคำที่ต้องการในประโยค ดังรูปที่ 3.12 เพื่อป้องกันไม่ให้แอลเอสทีเอ็มมีการนำคำที่ทำนายได้ไปสร้างข้อความที่ซ้ำกัน จึงมีการออกแบบให้นำคำที่ทำนายได้สูงสุด 10 คำมาสุ่มเอา 1 คำเพื่อไปสร้างเป็นคำต่อไปเพื่อให้ได้ตามจำนวนคำเท่ากับประโยคต้นฉบับ

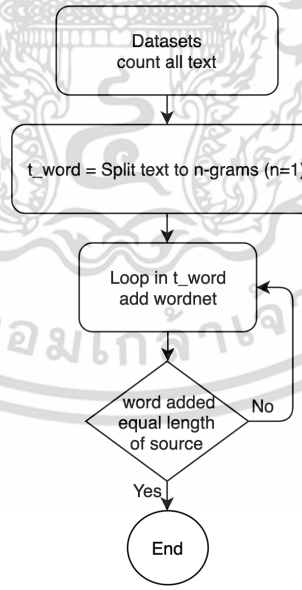


รูปที่ 3.12 ขั้นตอนการสร้างประโยคด้วยวิธีแอลเอสทีเอ็ม

หลังจากนี้จะเป็นการเสริมข้อความ (Sentence augmentation) หมายถึงการทำข้อความที่มีอยู่เดิมให้มีจำนวนมากขึ้นด้วยการเปลี่ยนคำบางคำในประโยคอาจจะเป็นคำนามหรือคำกริยาให้เปลี่ยนแปลงไปนิดหน่อยแต่มีความหมายคงเดิมการกระทำข้างต้นย่อมส่งผลให้เกิดประโยคใหม่แต่ความหมายยังคงเดิมเช่นกัน ตัวอย่าง เช่น การเปลี่ยนคำว่า “รถชน” เป็น “อุบัติเหตุ” หรือคำว่า “รถยนต์” เป็นคำว่า “ยานพาหนะ” โดยคำที่นำมาเปลี่ยนแทนจะมาจากคำที่มีความหมายใกล้เคียงกันหรือความหมายตรงข้ามกันในการทำงานด้านเอ็นแอลพีมีวิธีการหาคำที่คล้ายกันหลายวิธี ดังนั้น ผู้วิจัยขอนำเสนอวิธีการเสริมข้อความด้วยวิธีการต่าง ๆ ดังต่อไปนี้

3.5.3 การเสริมคำในข้อความด้วยเวิร์ดเน็ต

จากที่ได้อธิบายมาแล้วในหัวข้อก่อนหน้าถึงวิธีการของเวิร์ดเน็ตคือการนำคำที่อยู่ในกลุ่มเดียวกันมาแทนที่บางคำในประโยคทำให้ได้ประโยคใหม่โดยใช้กลุ่มคำที่มีอยู่ในฐานข้อมูลนั้นสำหรับงานนี้ได้มีการนำเวิร์ดเน็ตมาใช้งานเพื่อสร้างข้อความในกลุ่มที่มีจำนวนน้อยให้มีจำนวนเพิ่มมากขึ้นด้วยวิธีการ แบ่งประโยคออกเป็นคำเพื่อให้ได้การกระจายตัวของคำที่เปลี่ยนไปในส่วนต่าง ๆ ของประโยค โดยการสุ่มคำจากกลุ่มเดียวกันในเวิร์ดเน็ตแล้วนำคำที่คล้ายกันนั้นมาแทนที่คำเดิมจากนั้นทำเช่นเดียวกันกับคำอื่น ๆ ในประโยคจนครบทุกคำวิธีนี้ยิ่งประโยคมีความยาวก็จะสามารถสร้างการเปลี่ยนคำได้หลายจุดส่งผลให้ได้ประโยคใหม่หลายประโยคตามมานั้นเอง ดังแสดงการทำงานได้ดังรูปที่ 3.13



รูปที่ 3.13 กระบวนการทำงานการเปลี่ยนคำด้วยเวิร์ดเน็ต

การเปลี่ยนคำในบางคำสามารถแสดงให้เห็นตัวอย่างได้เช่น ข้อความ “มุ่งหน้าพระราม9รถมากเคลื่อนตัวได้ดี” สามารถแบ่งออกได้เป็นชุด ดังนั้นเมื่อใช้เวิร์ดเน็ตมาเปลี่ยนคำสุดท้ายจะได้ดังตารางที่ 3.4

ตารางที่ 3.4 ตัวอย่างการเสริมคำด้วยวิธีเวิร์ดเน็ต

คำจากข้อความเดิม	**คำจากเวิร์ดเน็ต**				
มุ่งหน้า	มุ่งหน้า	กำหนดทิศทาง	นำทาง		
พระราม	พระราม	พระราเมศ	พระรามราฆพ		
รถ	รถ	รถยนต์	ยานยนต์	พาหนะ	ยานพาหะนะ
มาก	มาก	มากมาย	จำนวนมาก		
เคลื่อนตัว	เคลื่อนตัว				
ได้ดี	ได้ดี	อย่างดี	ที่ดี		

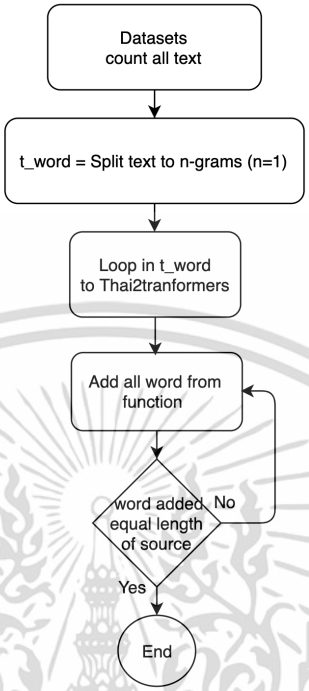
จากนั้นนำข้อความที่ได้ทั้งหมดมาต่อกันให้ครบประโยคจะได้ประโยคดังตัวอย่างเช่น

มุ่งหน้าพระรามรถมากเคลื่อนตัวได้ดี
มุ่งหน้าพระราเมศรถมากเคลื่อนตัวได้ดี
มุ่งหน้าพระราเมศรถยนต์มากเคลื่อนตัวอย่างดี
มุ่งหน้าพระรามรถมากเคลื่อนตัวที่ดี

จากตารางที่ 3.4 จะเห็นว่าข้อความ 1 ข้อความสามารถใช้วิธีการเสริมข้อมูลให้แตกต่างกันได้ข้อความใหม่เกิดมากขึ้น ดังนั้นวิธีการเวิร์ดเน็ตสามารถทำให้ข้อความมีความเท่ากันกับกลุ่มที่มีข้อความมากได้

3.5.4 การเสริมข้อความด้วยวิธีการไทยทูทรานส์ฟอร์เมอร์

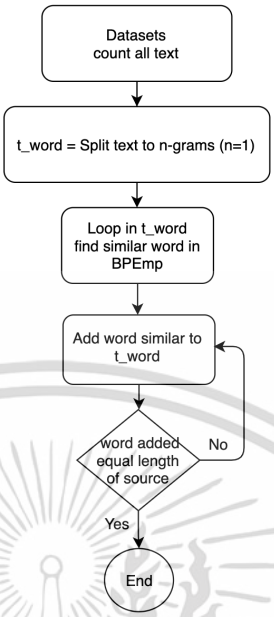
วิธีการเสริมข้อความด้วยไทยทูทรานส์ฟอร์เมอร์ (Thai2transformers) เป็นวิธีการสร้างข้อความจากโมเดลที่ถูกเทรนไว้ก่อนล่วงหน้าของภาษาไทยที่ถูกพัฒนาโดยสถาบันวิทยสิริเมธีหรือ VISTEC ซึ่งพัฒนาระบบขึ้นมาด้วยการเทรนจากชุดข้อมูลขนาด 78.5GB ที่ใช้โมเดลชื่อ “airesearch/wangchanBERTa-base-att-spm-uncased” โดยมีไพไทยเอ็นแอลพีนำวิธีการนี้มารวมไว้ในฟังก์ชัน augment ซึ่งสามารถนำฟังก์ชันนี้มาเสริมคำได้ไม่ยากด้วยการแบ่งประโยคออกเป็นคำเช่นเดียวกับขั้นตอนก่อนหน้าแล้วจึงนำคำย่อยเข้าสู่ฟังก์ชันที่ชื่อไทยทูทรานส์ฟอร์เมอร์ออก โดยฟังก์ชันจะตอบกลับมาเป็นคำที่เปลี่ยนไปหากต้องการให้ประโยคเพิ่มมากขึ้นสามารถใช้พารามิเตอร์ “num_replace_tokens” หลังจากได้คำที่ผ่านการเสริมคำแล้วจะนำคำนั้นมารวมกันเป็นประโยครวมเป็นขั้นสิ้นสุด สามารถอธิบายได้ดังรูปที่ 3.14



รูปที่ 3.14 ขั้นตอนการเสริมคำด้วยวิธีการไทยทูทรานส์ฟอร์เมอร์

3.5.5 การเสริมคำด้วยวิธีการเข้ารหัสแบบเอ็มเบ็ดดิงไบต์คู่

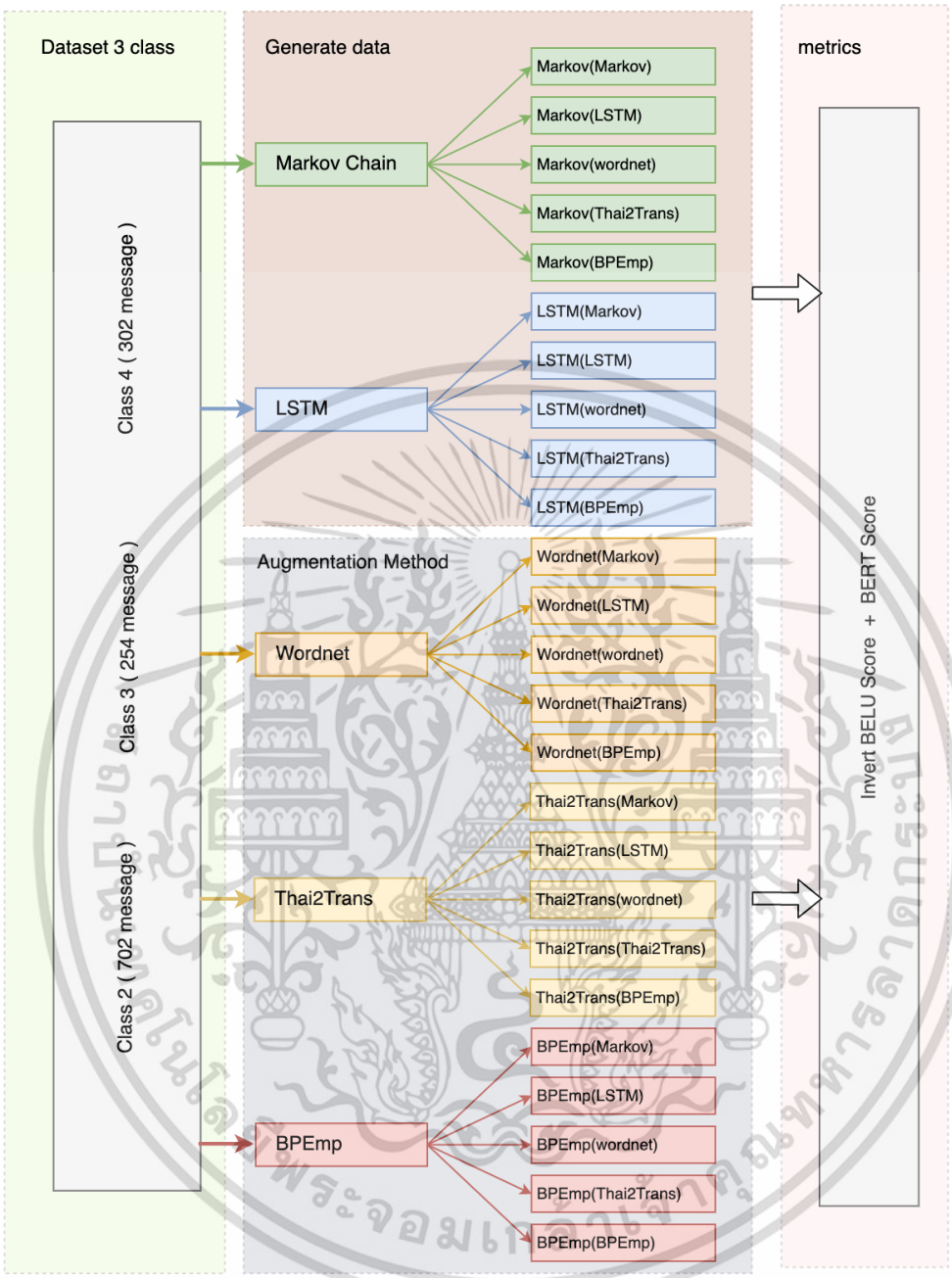
วิธีนี้จะเป็นการพัฒนามาจากการทำเวิร์ดทูเว็คหรือการทำคำเป็นเวคเตอร์แต่วิธีบีพีอีเอ็มบีจะเป็นการนำคำมาแยกออกเป็นไบต์คู่ก่อนที่จะนำไปสร้างเวคเตอร์คำ ดังนั้นคำที่จะเกิดขึ้นจะมีจำนวนมากและยังมีความละเอียดอีกด้วย โดยวิธีบีพีอีเอ็มบีจะมีการฝึกล่วงหน้าไว้อย่างพรีเทรนในรูปแบบของโมเดลคำในชื่อบีพีอีเอ็มบีเป็นไลบรารีที่เป็นลักษณะชุดข้อมูลเปิด (open source) สามารถติดตั้งและใช้งานได้ผ่านทางภาษาไพธอนโดยภายในมีการเตรียมฟังก์ชันให้สามารถใช้งานได้หลายฟังก์ชัน สำหรับงานวิจัยนี้ได้นำเอาฟังก์ชัน most_similar คือการหาค่าความคล้ายของคำซึ่งจะตอบกลับมาเป็นลิส (List) ของคำที่คล้ายกัน (word similar) แล้วนำคำที่ได้จากลิสมาเปลี่ยนแทนที่คำในประโยค ดังรายละเอียดขั้นตอนการเสริมคำด้วยวิธีบีพีอีเอ็มบีมีรายละเอียดดังรูปที่ 3.15



รูปที่ 3.15 ขั้นตอนการเสริมคำด้วยวิธีบีพีอีเอ็มบี

3.5.6 การเสริมข้อความด้วยการซ้อนวิธีการ

การออกแบบการทดลองสำหรับการทดลองนี้ได้ศึกษาวิธีการเสริมข้อความของข้อความในกลุ่มที่มีจำนวนน้อยด้วยกัน 3 กลุ่มคือ กลุ่มของการรายงานภัยพิบัติ การรายงานพื้นที่ชุมนุม และการรายงานพื้นที่ซ่อมถนนหรือทางชำรุด มาทำการเสริมข้อความมาด้วยวิธีการทั้งหมด 5 วิธีแล้วนำข้อความที่ผ่านการเสริมข้อความในรอบแรกเข้าสู่การเสริมข้อความซ้ำ กล่าวคือข้อความที่ได้จากวิธีที่ 1 จะถูกป้อนเข้าสู่วิธีที่ 1 วิธีที่ 2 วิธีที่ 3 วิธีที่ 4 และวิธีที่ 5 ในส่วนของข้อความที่ได้จากวิธีที่ 2 เช่นเดียวกันจะถูกป้อนเข้าสู่วิธีการเสริมข้อความวิธีที่ 1 วิธีที่ 2 วิธีที่ 3 วิธีที่ 4 และวิธีที่ 5 เช่นเดียวกันข้อความที่ได้จากการเสริมข้อความวิธีที่ 3 และ 4 และ 5 จะถูกป้อนเข้าสู่วิธีการเสริมข้อความวิธีที่ 1 วิธีที่ 2 วิธีที่ 3 วิธีที่ 4 และวิธีที่ 5 จนกระทั่งถึง ข้อความที่ได้จากการเสริมข้อความวิธีที่ 5 ก็จะถูกป้อนเข้าสู่วิธีการเสริมข้อความวิธีที่ 1 ถึงวิธีที่ 5 ตามลำดับ อาจจะกล่าวได้อย่างง่าย คือการนำข้อความที่ได้จากการเสริมข้อความแล้วนั้นนำกลับเข้าไปทำซ้ำกระบวนการส่งข้อความอีกครั้งโดยครั้งที่สองนี้จะเป็นการสร้างข้อความที่มีวิธีการเพิ่มขึ้นถึง 25 วิธีแสดงได้ดังรูปที่ 3.16 จากนั้น นำข้อความที่ได้จากวิธีการเดิม 5 วิธีและวิธีการใหม่อีก 25 วิธี รวมเป็น 30 วิธี มาหาค่าความคล้ายคลึงคำด้วยค่าคะแนนแบบเบิร์ตและหาค่าความต่างของคำด้วยค่าคะแนนเบลอในลำดับถัดไป



รูปที่ 3.16 ขั้นตอนการเสริมคำด้วยการซ้อนวิธี

3.5.7 การประเมินข้อความด้วยวิธีเบลอสกอร์ร่วมกับเบิร์ตสกอร์

การใช้เบลอสกอร์และเบิร์ตสกอร์ในการวัดผลการสร้างข้อความ ซึ่งวิธีการทั้งสองวิธีมีคุณสมบัติของการวัดที่แตกต่างกันในส่วนของเบลอสกอร์นั้นจะเป็นการวัดค่าความเหมือนของคำแต่ในส่วนของเบิร์ตสกอร์นั้นจะเป็นการวัดความหมายที่คล้ายกันของคำหรือข้อความ การใช้เบลอสกอร์ร่วมกับเบิร์ตสกอร์ใน

การวัดผลการทำออกเมนเทชันจะขึ้นอยู่กับวัตถุประสงค์ของการทดลองและลักษณะของข้อมูล โดยทั่วไปแล้วเบลอสกอร์มักถูกใช้ในการวัดความเหมือนของประโยคที่แปลจากระบบแปลภาษา กับประโยคที่ถูกแปลโดยมนุษย์ โดยการนับจำนวนคำหรือวลีที่ตรงกันระหว่างการแปลจากระบบกับการแปลของมนุษย์และคำนวณค่าออกมาดังเช่นการอธิบายในบทที่ 2 มาร่วมกับเบิร์ตสกอร์ซึ่งเป็นวิธีการที่ใช้คะแนนความคล้ายคลึงของคำในข้อความ ซึ่งเป็นวิธีที่เหมาะสำหรับการวัดผลการทำออกเมนเทชันในงานเอ็นแอลพีได้ แต่อย่างไรก็ตาม การใช้เบลอสกอร์หรือร่วมกับเบิร์ตสกอร์ในการวัดผลการทำออกเมนเทชันมีการพิจารณาตามลักษณะของข้อมูลและวัตถุประสงค์ของการทดลองคือ ต้องการหาว่าข้อความที่เสริมมานั้นเหมือนหรือแตกต่างจากข้อความเดิมมากเท่าไร และมีความหมายคล้ายกับของเดิมมากเท่าไร แต่ด้วยการใช้เบลอสกอร์และร่วมกับเบิร์ตสกอร์เพื่อวัดผลการทำออกเมนเทชันจึงต้องมีการรวมค่าจากทั้งสองสกอร์เข้าด้วยกันเป็นค่าเดียวได้ ด้วยวิธีการรวมค่าดังสมการ 3.1

$$Combined\ score = BERT\ score + BLEU\ Score \tag{3.1}$$

โดยที่:

$BERT\ score$ คือค่าคะแนนที่ได้จากการวัดผลด้วยวิธีเบิร์ตสกอร์

$BLEU\ Score$ คือค่าคะแนนที่ได้จากการวัดผลด้วยวิธีเบลอสกอร์

แต่ด้วยความสำคัญของแต่ละสกอร์ไม่เท่ากันโดยในที่นี้จะให้ความสำคัญไปที่ความหมายของคำมากกว่าจึงกำหนดน้ำหนักเข้าไปในสมการทั้งสองตัวแปรกำหนด W_{BERT} และ W_{BLEU} และเมื่อวิเคราะห์ค่าเบลอสกอร์พบว่าเดิมทีจะเป็นการวัดค่าความเหมือนของคำที่ต้นฉบับแปลกับคำที่มนุษย์แปลแต่ในงานวิจัยนี้ต้องการให้ข้อความเกิดความต่างกันจึงต้องกลับด้านค่าการวัดค่าด้วยการนำ $1 - BLEU\ Score$ เมื่อแทนค่าในสมการที่ 3.1 จะได้ดังสมการที่ 3.2

$$\begin{aligned} Combined\ score = {} & (w_{BERT} \times BERT\ score) \\ & +(w_{BLEU} \times (1 - BLEU\ Score)) \end{aligned} \tag{3.2}$$

โดยที่:

W_{BERT} คือค่าน้ำหนักถ่วงของเบิร์ตสกอร์

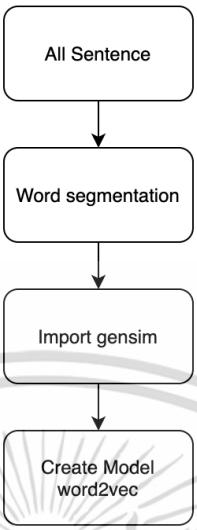
W_{BLEU} คือค่าน้ำหนักถ่วงของเบลอสกอร์

3.6 การจำแนกข้อความข่าวประเภทของอุบัติการณ์

ข้อความสภาพจราจรที่ได้รวบรวมจากทวิตเตอร์จะถูกแบ่งข้อความออกเป็น ข้อความที่ไม่เกี่ยวการรายงานสภาพจราจร และข้อความที่เกี่ยวกับสภาพจราจร ซึ่งการจำแนกข้อความส่วนแรกนี้ไม่มีปัญหาเรื่องข้อความไม่สมดุลกัน และในส่วนของข้อความสภาพจราจรหรือข้อความอุบัติการณ์จะถูกแบ่งออกเป็นห้าประเภทนั้นจะมีความไม่สมดุลกันของข้อความมาก ดังนั้น งานวิจัยฉบับนี้จะมุ่งเน้นไปที่การปรับปรุงข้อความเพื่อให้มีประสิทธิในการจำแนกข้อความได้แม่นยำขึ้นด้วยวิธีการที่เป็นการปรับปรุงข้อมูลให้มีจำนวนเท่ากันในทุกกลุ่มข้อมูล เมื่อข้อมูลในทุกกลุ่มเท่ากันแล้วขั้นตอนต่อไปคือการนำข้อมูลนั้นมาสอนโมเดลเพื่อสร้างโมเดลเพื่อการจำแนกข้อความจากทวิตเตอร์ซึ่งในงานวิจัยนี้จะมีการแบ่งกลุ่มข้อมูลออกเป็นสองขั้น ขั้นแรกแบ่งข้อความออกเป็น 2 กลุ่มคือ 1. ข้อความที่ไม่เกี่ยวกับการรายงานสภาพจราจร 2. ข้อความที่เป็นการรายงานสภาพจราจร ในขั้นต่อมาจะแบ่งข้อความสภาพจราจรออกเป็น 5 กลุ่ม คือ การจราจร อุบัติเหตุ ภัยพิบัติ การชุมนุม งานซ่อมถนน เมื่อมีการแบ่งกลุ่มออกมามากกว่า 2 กลุ่มจะถูกเรียกว่าการจำข้อมูลแบบการจัดประเภทหลายป้ายกำกับ (Multiple label) การสร้างโมเดลจึงควรสร้างให้รองรับข้อมูลแบบนี้ด้วยเช่นกัน การเรียนรู้เชิงลึกที่ใช้สำหรับการจำแนกข้อมูลหรือข้อความมีหลายวิธี ซึ่งหนึ่งในนั้นคือจากการจำแนกข้อความด้วยวิธีการเรียนรู้เชิงลึกคือวิธีซีเอ็นเอ็นผสานกับวิธีการแอลเอสทีเอ็ม

3.6.1 การเข้ารหัสข้อความด้วยวิธีเวิร์ดเอ็มเบ็ดดิง

การทำงานด้วยการประมวลผลข้อความหรือเอ็นเอลพีนั้นคือการดำเนินการกับข้อความซึ่งตรงตัวว่าข้อความไม่ใช่ตัวเลข แต่การประมวลผลด้วยคอมพิวเตอร์จำเป็นต้องใช้ตัวเลขดังนั้นนักวิจัยพยายามที่จะค้นหาตัวแทนอะไรสักอย่างเพื่อมาแทน (Represent) ข้อความให้คอมพิวเตอร์สามารถเข้าใจได้ในขั้นตอนแรกมีการสร้างตัวแทนข้อความหรือคำด้วยวิธีการเข้ารหัสที่เรียกว่าการเข้ารหัสแบบวัน-ฮอท (One-hot encoding) คือการแทนคำด้วยตัวเลข 1 และ 0 แต่วิธีการนี้จะทำให้เกิดเมทริกซ์เบาบาง (Sparse Matrix) คือการที่ข้อมูลมีค่าเลข 0 เป็นจำนวนมากดังนั้นจะทำให้การประมวลผลเป็นไปอย่างช้าเนื่องจากข้อมูลเยอะเกินไป จึงเป็นที่มาของการหาทางลดขนาดของเลข 0 ด้วยวิธีการเข้ารหัสแบบการฝังคำ หรือเวิร์ดเอ็มเบ็ดดิง (word embedding) คือการพยายามค้นหาค่าเวคเตอร์ด้วยวิธีการเวิร์ดทูเว็ค (Word2vec) สำหรับงานวิจัยนี้ได้ใช้วิธีการเข้ารหัสคำด้วยวิธีเวิร์ดทูเว็คด้วยเช่นกันเนื่องจากจะต้องนำค่าเวคเตอร์นี้ไปใช้งานในหลายส่วนของการสร้างโมเดลการจำแนกกลุ่มข้อความ ซึ่งผู้วิจัยได้สร้างโมเดลเวิร์ดทูเว็คขึ้นมาจากตัวช่วยอย่างเจ็นซิม และนำข้อมูลกลุ่มคำที่มีอยู่มาสร้างโมเดล



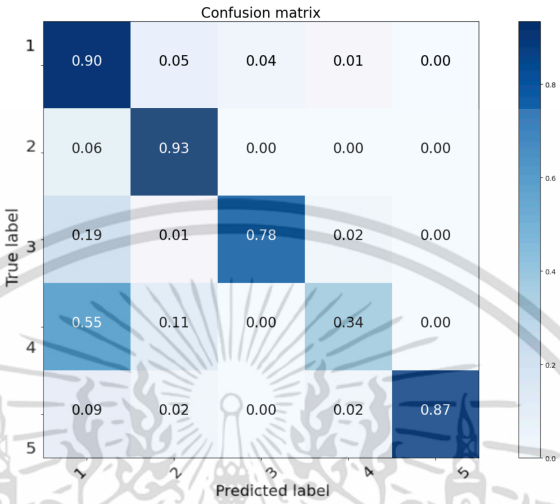
รูปที่ 3.17 กระบวนการสร้างเวิร์ดทูเว็คจากข้อความทั้งหมด

จากรูปที่ 3.17 คือขั้นตอนการสร้างเวิร์ดเอ็มเบ็ดดิงสำหรับคำที่มีในเดต้าเซ็ตทั้งหมด โดยในขั้นตอนแรกเป็นการเรียกใช้เจ็นซิมจากการเขียนด้วยภาษาไพธอนขั้นตอนต่อมาเป็นการนำเข้าข้อความจากเดต้าเซ็ตที่เตรียมไว้แล้วสำหรับขั้นตอนนี้จะใช้ข้อความทั้งหมดมาทำอาจจะมีคำทั้งหมดประมาณสี่พันคำ จากนั้นเรียกใช้ฟังก์ชันเวิร์ดทูเว็คและกำหนดค่าพารามิเตอร์ที่ต้องการ เช่น ค่า vector_size จะเป็นค่า 100 คือ ค่าขนาดของมิติเวคเตอร์ที่ต้องการหากกำหนดค่าสูงก็จำเป็นจะต้องใช้การประมวลผลที่นาน ดังนั้นค่าที่เหมาะสมควรเป็น 10 หรือ 100 หรือ 200 ต่อมาเป็น ค่า sg คือต้องการให้โมเดลใช้วิธีการใดในการสร้างเวคเตอร์โดย sg=0 จะเป็นสคิป-แกรมและ sg=1 จะซีบีโอดับเบิ้ลยู window=3 คือ จำนวนคำที่ใช้ในการทำนายคำถัดไป min_count=1 คือ จำนวนครั้งขั้นต่ำที่คำต้องปรากฏในข้อความ และ workers = multiprocessing.cpu_count()) คือการนับจำนวน CPU ในเครื่องเพื่อประโยชน์ในการเทรน

3.6.2 การจำแนกข้อความประเภทอุบัติการณ์

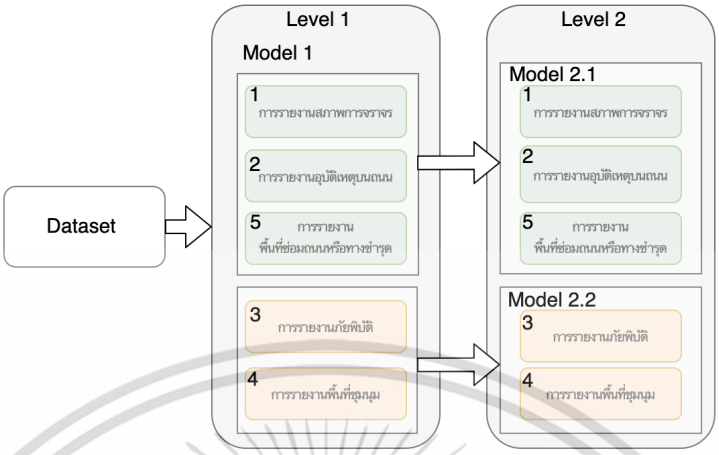
การจำแนกข้อความสำหรับงานวิจัยในครั้งนี้หากดูตามรูปที่ 3.5 นั้นสามารถทราบได้ว่าเป็นการจำแนกข้อความออกเป็น 2 ระดับคือครั้งแรกเมื่อได้ข้อความมาจากทวิตเตอร์จะทำการแยกข้อความครั้งแรกก่อน คือการแยกข้อความอุบัติการณ์ออกจากข้อความข่าวประสัมพันธ์ ซึ่งในส่วนการทำงานส่วนนี้ไม่มีความซับซ้อนมากนักเนื่องจากเป็นการจัดกลุ่มข้อมูลออกเป็น 2 กลุ่มจึงได้ค่าความแม่นยำที่ 0.97 แต่เมื่อแยกข้อความอุบัติการณ์ออกมาได้เรียบร้อยแล้วจากนั้นจะทำการจำแนกข้อความออกเป็นข้อความที่เกี่ยวกับอุบัติการณ์หรือข้อความสภาพจราจรมีด้วยกัน 5 ประเภทนั้นเมื่อมีทำการแยกประเภทพบว่าเกิด

ความแม่นยำมีค่าไม่สูงมากพอและมีผลลัพธ์ความเอนเอียงไปทางกลุ่มที่มีจำนวนมาก ถึงแม้หลังจากได้สร้างข้อความจนข้อความแต่ละกลุ่มมีจำนวนเท่ากันในทุกกลุ่มแล้วก็ตาม ดังรูปที่ 3.18



รูปที่ 3.18 ตารางประเมินประสิทธิภาพการจำแนกประเภทข้อความอุบัติการณ์

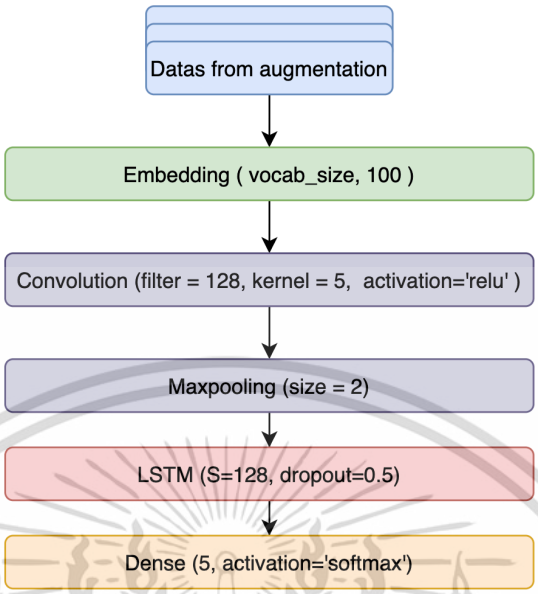
จึงมีแนวความคิดออกแบบการสร้างโมเดลสำหรับการจำแนกข้อความออกเป็น 5 กลุ่ม สำหรับการทดลองนี้จะมีการสร้างโมเดลออกเป็นสองระดับโดยในระดับแรกจะเป็นการสร้างโมเดลเพื่อแยกข้อความของกลุ่มที่ 1 กลุ่มที่ 2 และกลุ่มที่ 5 (การจราจร อุบัติเหตุ และงานซ่อมถนน) ออกจากข้อความกลุ่มที่ 3 และกลุ่มที่ 4 (ภัยพิบัติและการชุมนุม) ในระดับที่สองเป็นการสร้างโมเดลออกมาสองโมเดลเพื่อแยกข้อความการจราจร อุบัติเหตุ และงานซ่อมถนน ออกจากกัน และอีกโมเดลเป็นการแยกข้อความภัยพิบัติและการชุมนุมออกจากกัน ดังนั้นเมื่อสร้างโมเดลสำเร็จจะได้ข้อความที่ถูกคัดแยกแบบการกรองสองชั้นเพื่อให้ได้ข้อความแบ่งออกเป็น 5 กลุ่มอย่างชัดเจน ดังรูปที่ 3.19



รูปที่ 3.19 การสร้างโมเดลเพื่อระบุคลาสแบบ 2 ระดับ

ในการสร้างโมเดลจะใช้วิธีการซีเอ็นเอ็นผสานกับวิธีการแอลเอสทีเอ็ม ซึ่งซีเอ็นเอ็นเป็นโมเดลที่ใช้ในการจำแนกข้อมูลเชิงพื้นที่ เช่น รูปภาพที่มีขนาด 2 มิติ แต่ในบริบทของการจำแนกข้อความ ซีเอ็นเอ็นก็สามารถทำงานได้ดีเช่นกัน และอีกวิธีคือแอลเอสทีเอ็มเป็นการพัฒนาต่อมาจากวิธีการอาร์เอ็นเอ็นเพื่อมาแก้ปัญหาการเรียนรู้ระยะยาว โดยแอลเอสทีเอ็มจะมีหน่วยความจำย่อยเพื่อจดจำคำก่อนหน้าไปไกล ๆ ได้ ว่าคำก่อนหน้าในประโยคพูดถึงเรื่องอะไรแล้วค่อยตรวจว่าจะลืมหรือจะจำคำไหนบ้างด้วยคุณสมบัตินี้ทำให้แอลเอสทีเอ็มจึงเหมาะกับการจำแนกข้อความที่เป็นประโยคดังเช่นข้อความจากทวิตเตอร์ กระบวนการจำแนกข้อความด้วยวิธีแอลเอสทีเอ็มนั้นจะทำตามขั้นตอนที่คล้ายกับวิธีการจำแนกแบบทั่ว ๆ ไป คือ ต้องมีข้อความที่มีการทำป้ายกำกับไว้เรียบร้อยเพื่อนำมาเทรนให้กับโมเดลโดยเป็นวิธีการเรียนรู้แบบมีผู้สอน (Supervised Learning)

จากการศึกษางานวิจัยในหลายงานพบว่าโมเดลการจำแนกข้อความที่เป็นที่นิยมมากที่สุดคือการใช้วิธีการเรียนรู้เชิงลึกซีเอ็นเอ็นผสานกับแอลเอสทีเอ็มให้ทำงานร่วมกันซึ่งวิธีการนี้จะเป็นการนำวิธีทั้งสองมาซ้อนกันโดยนำผลลัพธ์จากซีเอ็นเอ็นไปต่อเข้ากับแอลเอสทีเอ็มเพื่อให้สามารถจำแนกข้อความได้มีประสิทธิภาพเพิ่มขึ้น โดยมีรายละเอียดการสร้างโมเดลดังนี้

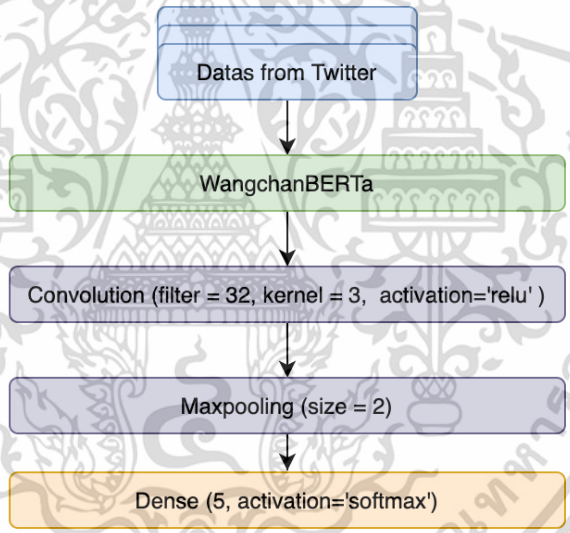


รูปที่ 3.20 ขั้นตอนการจำแนกประเภทข้อความสภาพจราจรด้วยวิธีซีเอ็นเอ็นผสานแอลเอสทีเอ็ม

จากรูปที่ 3.20 คือขั้นตอนการสร้างโมเดลการจำแนกข้อความด้วยวิธีการซีเอ็นเอ็นผสานแอลเอสทีเอ็มรายละเอียดเริ่มจากการนำข้อมูลจากข้อความที่ได้มีการสร้างให้มีจำนวนเท่ากันทุกกลุ่มแล้วนำเข้าการหารูปแบบการเวคเตอร์ที่มีขนาด $V \times N$ โดยที่ V คือขนาดของคำศัพท์ (Vocabulary) ที่มีทั้งหมดคูณกับขนาดของมิติ N ที่ต้องการในงานนี้ตั้งค่าที่ 100 ชั้นต่อมาเป็นชั้นคอนโวลูชันหรือก็คือชั้นของ ซีเอ็นเอ็นที่มีฟิวเตอร์เป็น 128 ความกว้างมิติ (kernel) ที่ 5 คำต่อครั้ง ชั้นต่อมาเป็นชั้นการลดขนาดมิติหรือพูลลิงสำหรับงานนี้ได้ใช้เป็นแม็กซ์-พูลลิ่ง (max-pooling) ซึ่งจะเป็นการลดขนาดโดยการนำค่ามากสุดมาเพื่อลดขนาดมิติลงให้เหลือครึ่งนึงโดยการสกัดเอาเฉพาะค่าสูงสุดของคำเก็บไว้ เพื่อเพิ่มประสิทธิภาพการประมวลผลให้รวดเร็วยิ่งขึ้นสำหรับซีเอ็นเอ็นจะจบลงที่ชั้นพูลลิง ต่อมาจะนำค่าที่ได้เข้าสู่แอลเอสทีเอ็มโดยการตั้งค่าพารามิเตอร์คือจะทำการตั้งค่าเป็น S=128 คือค่าชั้นซ่อน (hidden layer หรือฮิตเดน-เลเยอร์) ในส่วนของการป้องกันเหตุการณ์การเรียนรู้มากเกินไป (Overfitting หรือโอเวอร์ฟิตติง) คืออาการแม่นยำเกินไปของโครงข่ายตั้งค่าไว้ที่ดรอปเอาท์ (dropout) ไว้ที่ 0.5 ชั้นสุดท้ายเป็นชั้นรวมโครงข่ายเพื่อตัดสินใจออกมาเป็นคำตอบโดยให้ค่าพารามิเตอร์เป็นจำนวนเท่ากับค่าของกลุ่มทั้งหมด ในที่นี้คือ 5 กลุ่ม

3.7 การจำแนกข้อความด้วยวิธีเบิร์ตร่วมกับซีเอ็นเอ็น

ทุกวันนี้มีแนวคิดเรื่องการดำเนินการเกี่ยวกับเอ็นเอลพีที่มีการนำกลุ่มข้อมูลขนาดใหญ่มาสอนไว้ล่วงหน้าซึ่งมีการทำไว้หลายโมเดลหลากหลายภาษามาใช้งานด้วยการส่งต่อการสอน (Transfer Learning) ที่นำเอาความสามารถของการทำ (feature extration หรือฟีเจอร์เอกซ์แทร็กชัน) มาใช้ร่วมกับงานการจำแนกข้อความ โดยมีงานวิจัยของ Gregorius Aria Neruda และ Edi Winarko มีการสร้างระบบเพื่อจำแนกข้อความจากทวิตเตอร์ที่มีการรายงานสภาพจราจรในเมืองจาการ์ตา ประเทศอินโดนีเซีย โดยงานนี้มีการนำเบิร์ตมาทำงานร่วมกับซีเอ็นเอ็น ซึ่งมีผลการทดลองที่น่าสนใจ ซึ่งตรงกับงานที่ผู้วิจัยกำลังทำและเนื่องจากเบิร์ตที่นำมาใช้เป็นอินโด-เบิร์ต (Indo-BERT) ซึ่งเป็นโมเดลภาษาอินโดนีเซีย ดังนั้นสำหรับงานวิจัยการจำแนกภาษาไทยจึงได้นำเอา WangchanBERTa มาทำงานร่วมกับวิธีการซีเอ็นเอ็นโดยมีการออกแบบการทดลองดังรูปที่ 3.21



รูปที่ 3.21 ขั้นตอนการสร้างโมเดลด้วยวิธีเบิร์ตผสานซีเอ็นเอ็น

สำหรับงานวิจัยที่นำเสนอในบทนี้ได้นำเสนอกระบวนการในการเพิ่มข้อความในกลุ่มข้อมูลที่น้อยเพื่อให้มีจำนวนกลุ่มข้อมูลที่มีจำนวนมากสำหรับการจัดการกับกลุ่มข้อมูลที่ไม่สมดุลซึ่งมีวิธีการดำเนินการในหลายวิธีทั้งในส่วนของการจัดการข้อมูลด้วยวิธีการสร้างข้อความขึ้นมาด้วยวิธีการเรียนรู้เชิงลึกหรือการหากลุ่มคำมาแทนด้วยวิธีการเวิร์ดเน็ตรวมถึงการนำคำจากคลังข้อความมาเปลี่ยนแทนในคำที่มีความใกล้เคียงกันทางเวคเตอร์ดังเช่นวิธีบีพีอีเอ็มบีและเมื่อข้อความมีความสมดุลกันในทุกกลุ่มแล้วจะนำข้อความนั้นมาเทรนโมเดลเพื่อจำแนกประเภทข้อความพร้อมทั้งนำเสนอวิธีการของงานวิจัยที่เป็น baseline ของงานวิจัยนี้อีกด้วย สำหรับขั้นตอนต่อไปจะนำการออกแบบนี้ไปพัฒนาเพื่อทดสอบหาผลลัพธ์ต่อไป

บทที่ 4
ผลการวิจัยและการอภิปราย

หลังจากที่ได้ดำเนินการส่วนของการนำเสนอวิธีการทำงานวิจัยสำหรับในบทนี้เป็นการนำวิธีการนั้นมาพัฒนาเพื่อให้เป็นไปตามที่ได้ออกแบบไว้และนำเสนอผลของการพัฒนาตามแบบ ได้แก่ ผลของการออกแบบในส่วนของการเก็บรวบรวมข้อมูล การจัดรูปแบบข้อมูลออกเป็นกลุ่มพร้อมทั้งทำเลเบลให้กับข้อความแต่ละข้อความ ผลของการเตรียมข้อความ ผลของการสร้างข้อความด้วยวิธีการลูกโซ่มาร์คอฟ และวิธีแอลเอสทีเอ็มผลของการเสริมข้อความด้วยวิธีการเวิร์ดเน็ตวิธีไทยทูทรานส์ฟอร์เมอร์และวิธีบีพีอีเอ็มบี เมื่อเป็นที่ทราบแล้วว่าวิธีการไหนคือวิธีการจัดการกับข้อความไม่เท่ากันได้ดีที่สุด ส่วนต่อมาคือการสร้างระบบการจำแนกข้อความด้วยข้อความผ่านการจัดการเรื่องจำนวนให้เท่ากันหมดแล้วมาเทรนโมเดลเพื่อสร้างขึ้นมาด้วยวิธีการซีเอ็นเอ็นผสานแอลเอสทีเอ็มจากนั้นทดสอบว่าโมเดลที่สร้างขึ้นมาทำงานได้ดีกว่าวิธีการเดิมหรือไม่ ในข้างต้นนี้คือการสรุปเนื้อหาของบทนี้ให้ทราบถึงรายละเอียดที่จะอธิบายในรายละเอียดดังต่อไปนี้

4.1 ผลการรวบรวมและจัดกลุ่มข้อความจากทวิตเตอร์

จากการศึกษาข้อมูลการเผยแพร่ข้อความทวิตเตอร์พบว่ามีการรายงานข่าวหรือข้อความทั้งที่เกี่ยวและไม่เกี่ยวข้องสภาพจราจรเป็นจำนวนมากทำให้ต้องพัฒนาระบบเพื่อทำการเก็บข้อมูลแบบอัตโนมัติเข้าไปเก็บในฐานข้อมูลจากการใช้เอพีไอของทวิตเตอร์ในการรวบรวมข้อมูลโดยผลของการรวบรวมข้อมูลแสดงดังรูปที่ 4.1

	id [PK] bigint	massage text	datetime timestamp without time zone	id_tweet bigint	screen_name text
1	94029	ระดมสมอง 15 ปี พ.ร.บ.ควบคุมเครื่องดื่มแอลกอฮอล์ ...	2023-02-23 16:04:17	1628682146356559872	js100radio
2	88733	อิหร่าน ส่งหนังสือฟ้อง UN กล่าวหาอิสราเอล โจมตี...	2023-02-03 09:41:36	1621338083546501120	js100radio
3	88736	3 - 11 ก.พ.66 งานเกษตรแฟร์ ณ ม.เกษตรศาสตร์ (บ...	2023-02-03 09:11:13	1621330435195408384	js100radio
4	88779	19.46 เพลิงไหม้หญ้า ใกล้เคียงตลาดเคหะชุมชนราม...	2023-02-02 19:46:33	1621127934823256065	js100radio
5	94030	MEA เขตสามเสน แจ้งย้ายที่ตั้งงานบริการชั่วคราว ตั้...	2023-02-23 16:04:16	1628682140841025536	js100radio
6	94031	RT @warinnpp: @js100radio #อุบัติเหตุ รถกระบะช...	2023-02-23 15:59:24	1628680917593255936	js100radio
7	88781	หน้าเหมือนกันก็อันตราย! ตำรวจเยอรมันจับกุมหญิงวั...	2023-02-02 19:07:56	1621118216469905409	js100radio
8	88782	18.55 ล่าสุดเพลิงสงบ รายละเอียดเพิ่มเติมอยู่ระหว่า...	2023-02-02 18:55:52	1621115181538607104	js100radio
9	94035	15.13 #อุบัติเหตุ #ถนนสุโขทัย ขาออก ที่ก่อนถึงแย...	2023-02-23 15:13:42	1628669417390477313	js100radio
10	94036	15.09 เพลิงไหม้ภายในโกดังบริษัท ไดกิ (ประเทศไท...	2023-02-23 15:09:12	1628668284877828097	js100radio
11	94038	องคมนตรี พิจารณาโครงการอันเนื่องมาจากพระราชดำ...	2023-02-23 13:55:55	1628649838719156224	js100radio

รูปที่ 4.1 การเก็บข้อมูลในฐานข้อมูล

จากรูปมีการเก็บข้อมูลไว้ในฐานข้อมูลมีการเก็บแบ่งตามคอลัมน์และกำหนดให้เก็บเฉพาะข้อมูลที่สำคัญ คือ ข้อความ บัญชีทางการที่โพสต์หรือรีทวิตส์มา เวลาที่มีการโพสต์ เมื่อได้รวมข้อมูลมาได้แล้วขั้นตอนต่อมาคือการกำหนดกลุ่มให้กับข้อความทุก ๆ ข้อความ ด้วยการช่วยเหลือของพนักงานในศูนย์บริหารจราจรทางพิเศษในการอ่านและวิเคราะห์เพื่อให้ทราบถึงกลุ่มของข้อความว่าข้อความนั้นควรอยู่ในกลุ่มใด ซึ่งในขั้นแรกจะแบ่งข้อความออกเป็น 2 กลุ่มคือข้อความทั่วไปและข้อความสภาพจราจรดังตารางที่ 4.1 และในขั้นที่สองมีการแบ่งออกเป็น 5 กลุ่มโดยเมื่อจัดกลุ่มแล้วเสร็จจะแสดงผลการจัดกลุ่มจากข้อความทั้งหมดได้ดังตารางที่ 4.2

ตารางที่ 4.1 จำนวนข้อความที่เกี่ยวข้องกับสภาพจราจรและไม่เกี่ยวข้องกับสภาพจราจร

อ้างอิง	ความหมาย	จำนวน
1	ข้อความที่ไม่เกี่ยวกับการรายงานสภาพการจราจร	10,640
2	การรายงานข้อความที่เกี่ยวกับสภาพจราจร	9,360

ตารางที่ 4.2 จำนวนข้อความแบ่งตามประเภทของข้อความในแต่ละกลุ่ม

อ้างอิง	ความหมาย	จำนวน
1	การรายงานสภาพการจราจร	3,743
2	การรายงานอุบัติเหตุบนถนน	4,321
3	การรายงานภัยพิบัติ	702
4	การรายงานพื้นที่ชุมนุม	252
5	การรายงานพื้นที่ซ่อมถนนหรือทางชำรุด	342

4.2 ผลการเตรียมข้อมูล

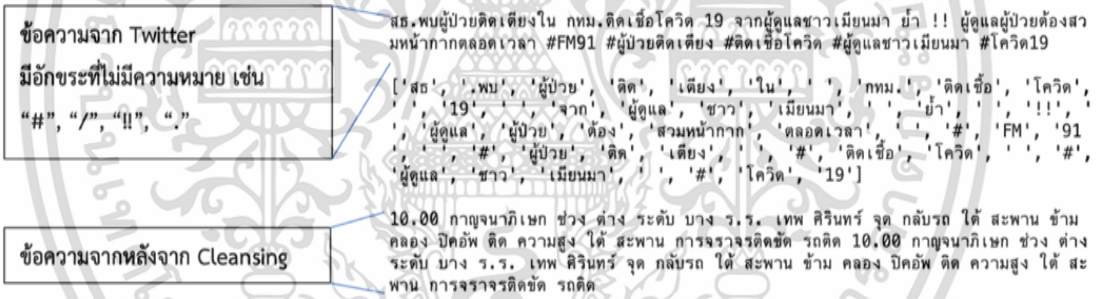
เมื่อแบ่งกลุ่มข้อมูลเป็นที่เรียบร้อยแล้วขั้นตอนต่อมาจะเป็นการนำข้อมูลที่ได้รวบรวมมาทำการเตรียมข้อมูลเพื่อให้พร้อมสำหรับการดำเนินการขั้นต่อไปคือการตัดประโยคออกเป็นคำ (word segmentation) ในขั้นตอนนี้โดยใช้ฟังก์ชัน word_tokenize ที่มีอยู่ในไพไทยเอ็นแอลพี ดังรูปที่ 4.2

```
txt = excel_data_df.tweet[5]
txt_c = word_tokenize(txt)
print(txt,'\n')
print(txt_c)
```

```
วันศุกร์แห่งชาติ ! เย็นวันนี้สภาพการจราจรจะเป็นอย่างไร ติดตามได้กับ : เจาะ จุด แจม : 5
ก.พ.64

['วัน', 'ศุกร์', 'แห่งชาติ', ' ', '!', ' ', 'เย็น', 'วันนี้', 'สภาพ', 'การจราจ
ร', 'จะ', 'เป็น', 'อย่างไร', ' ', 'ติดตาม', 'ได้', 'กับ', ' ', ':', ' ', 'เจ
าะ', ' ', 'จุด', ' ', 'แจม', ' ', ':', ' ', '5', ' ', 'ก.พ.', '64']
```

รูปที่ 4.2 แสดงตัวอย่างผลการตัดประโยคออกเป็นคำ

หลังจากการตัดคำยังคงมีเครื่องหมายหรือสัญลักษณ์ที่ไม่มีความหมาย เช่นเครื่องหมายอัศเจรีย์ (!) หรือเครื่องหมายทวิภาค (:) ดังนั้นจึงต้องมีการกรองสัญลักษณ์พวกนี้ออกไปโดยใช้วิธีการสร้างเงื่อนไขแบบนิพจปกติ (Regular expression) โดยเงื่อนไขที่สร้างคือการตั้งเงื่อนไขให้คงไว้แค่ อักษร ก-ฮ และตัวเลข 0-9 และภาษาอังกฤษคือ a-z ทั้งพิมพ์ใหญ่และพิมพ์เล็ก



รูปที่ 4.3 ตัวอย่างการกรองข้อมูลสัญลักษณ์ออก

และเพื่อให้ข้อความมีความกระชับขึ้นจึงได้นำเทคนิคการตัดคำฟุ่มเฟือยออกโดยนำเอาฟังก์ชันลบคำฟุ่มเฟือย มาใช้งานซึ่งหลังจากข้อความเดิมผ่านฟังก์ชันลบคำฟุ่มเฟือยแล้วนั้นจะได้ข้อความที่กระชับและได้ใจความมากขึ้น เมื่อนำข้อความขั้นสุดท้ายนี้ไปสร้างโมเดลจะช่วยลดภาระเวลาการสร้างโมเดลลง เนื่องจากคำที่ต้องประมวลผลน้อยลง

```
textCut = ['กาญจนาภิเษก','พบ','ช่วง','ต่าง','ระดับ','บาง','ร.ร.','เทพ','ศิรินทร์', 'จัด','กลับ','รถ',
           'ใต้','สะพาน','ข้าม','คลอง','ปิคอัพ','ติด' ,'ความสูง','ใต้','สะพาน','การจราจร','ติดขัด']

list_word_not_stopwords = [i for i in textCut if i not in stopwords]
```

```
print(list_word_not_stopwords)
```

```
['กาญจนาภิเษก', 'ระดับ', 'ร.ร.', 'เทพ', 'ศิรินทร์', 'รถ', 'สะพาน', 'ข้าม', 'คลอง', 'ปิคอัพ', 'ติด', 'ความสูง', 'สะพาน', 'การจร
าจร', 'ติดขัด']
```

รูปที่ 4.4 ข้อความหลังจากทำลบคำฟุ่มเฟือย

4.3 ผลการสร้างข้อความเพิ่ม

จากข้อความที่รวบรวมมาและทำการเตรียมข้อมูลสำหรับการทำงานไม่ว่าจะเป็นการตัดประโยคออกเป็นคำต่อด้วยตัดคำที่ไม่ต้องการออก เช่น สัญลักษณ์ต่าง ๆ หรือแม้แต่คำฟุ่มเฟือยที่ไม่ค่อยให้ความหมาย เมื่อข้อความในทุกกลุ่มดำเนินการข้างต้นเรียบร้อยแล้วสำหรับขั้นตอนนี้จะเป็นการดำเนินการเพื่อสร้างข้อความในกลุ่มน้อยให้มีจำนวนเท่ากับจำนวนในกลุ่มข้อความที่มีมากด้วยวิธีการต่าง ๆ ดังนี้

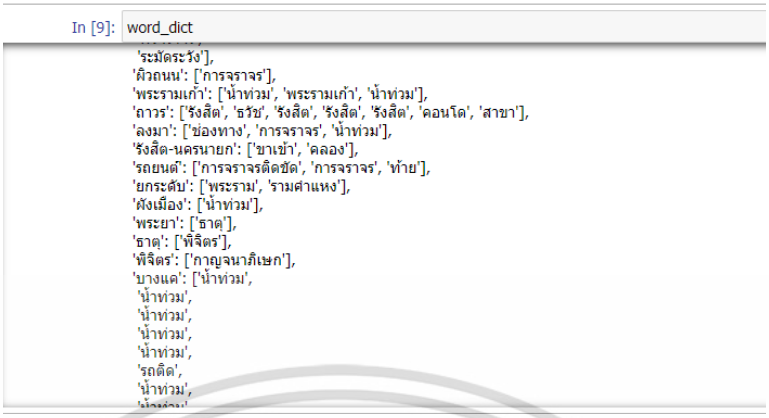
4.3.1 ผลการสร้างข้อความด้วยวิธีลูกโซ่มาร์คอฟ

วิธีการลูกโซ่มาร์คอฟ คือการสร้างข้อความใหม่จากข้อความเดิมที่นำคำมาต่อกันจากความน่าจะเป็นของคู่คำที่เกิดขึ้นบ่อย ดังนั้นจึงต้องค้นหาคู่คำที่เกิดขึ้นในกลุ่มประโยคทั้งหมดที่มีโดยใช้ประโยคที่ผ่านการเตรียมข้อมูลมาเรียบร้อยแล้ว โดยผลของการดำเนินการนี้จะมีขั้นตอนการค้นหาดังรูปที่ 4.5

```
def make_pairs_2(data_from_csv):
    for i in range(len(data_from_csv)):
        for j in range(len(data_from_csv[i])-1):
            yield (data_from_csv[i][j], data_from_csv[i][j+1])

pairs_2 = make_pairs_2(data_from_csv)
```

```
# จับคู่คำที่เกิดขึ้นต่อกัน
word_dict = {}
for word_1, word_2 in pairs_2:
    if word_1 in word_dict.keys():

        word_dict[word_1].append(word_2)

    else:
        word_dict[word_1] = [word_2]
```

รูปที่ 4.5 ขั้นตอนการค้นหาคู่คำในประโยค



รูปที่ 4.6 ผลการค้นหาคู่คำในประโยค

จะเห็นว่ามีบางคำที่มีคู่คำเพียง 1 คำเท่านั้นส่วนบางคำจะมีคำที่ต่อมาจากคำนั้นหลายคำโดยขั้นตอนนี้จะเก็บคำที่เกิดต่อไว้ในตัวแปรของคำต้นและตามด้วยลิสของคำที่ต่อท้ายมาเก็บไว้ในตัวแปรใด ๆ และทำแบบนี้กับคำในข้อมูลที่มีอยู่ในคลังทั้งหมด และในขั้นตอนต่อมาเป็นการนำคู่คำนั้นมาหาค่าความน่าจะเป็นที่จะเกิดคำต่อไป

ตารางที่ 4.3 ตัวอย่างผลการสร้างตารางความน่าจะเป็น

คำปัจจุบัน	คำถัดไป	ความน่าจะเป็น
กรมทางหลวง	รายงาน	0.0062
	อำนวย	0.0021
รายงาน	สถานการณ์	0.0062
	ภัยพิบัติ	0.0041
	จราจร	0.0041
	การจราจร	0.0041
	ขาออก	0.0021
	ขาเข้า	0.0021
สถานการณ์	น้ำท่วม	0.0062
	ตกหนัก	0.0021
	คลอง	0.0021
น้ำท่วม	พื้นที่	0.0021
	บริเวณ	0.0021

จากที่ได้คิดค่าความน่าจะเป็นของคำต่อไปที่จะเกิดขึ้นต่อไปเป็นการสร้างประโยคจากตารางความน่าจะเป็นโดนการสุ่มคำจากตารางในกลุ่มของคำต้นเป็นหลักและสุ่มคำต่อไปด้วยคำที่อยู่ในลิสของคำต้นนั้น (สาเหตุที่ใช้การสุ่มเนื่องจากตารางความน่าจะเป็นมีค่าเท่ากับในหลาย ๆ คำจึงใช้วิธีการสุ่มเอาคำในลิสออกมา) ด้วยการสร้างฟังก์ชันเพื่อสร้างข้อความโดยรับคำต้นของข้อความเดิมแล้วนำมาค้นหาคำต่อไปในตัวแปร word_dict ทำไปจนครบจำนวนทุกคำในข้อความจึงรวมคำทุกคำเข้าเป็นประโยคและคืนค่ากลับไปเพื่อเก็บรวบรวมไว้และทำแบบนี้กับประโยคต่อไปจนครบจำนวนที่ต้องการ

```python
#ฟังก์ชันสร้างข้อความด้วยวิธี Markov chain
def gen_Text_3to8word(first_word,amount_word):
    chain = [first_word]
    for i in range(amount_word):
        try:
            chain.append(np.random.choice(word_dict[chain[-1]]))
        except KeyError as e:
            continue

    return ' '.join(chain)


# หาคำแรกและหาจำนวนคำของแต่ละข้อความ
# ส่งคำแรกของประโยคไปให้ function สร้างข้อความ
list_message_gen = []
list_t_detail = []
for i in range(len(df_csv_Sourcetext)):
    for j in range(5):
        sequence = df_csv_Sourcetext[i][j]

        wordSplit = sequence.split()
        print((len(wordSplit)-1))
        print(wordSplit[0])


        message_gen = gen_Text_3to8word(wordSplit[0],(len(wordSplit)-1))
        list_message_gen.append(message_gen)
        list_t_detail.append(3)
```

รูป 4.7 ขั้นตอนการสร้างประโยคจากคำในตารางความน่าจะเป็น

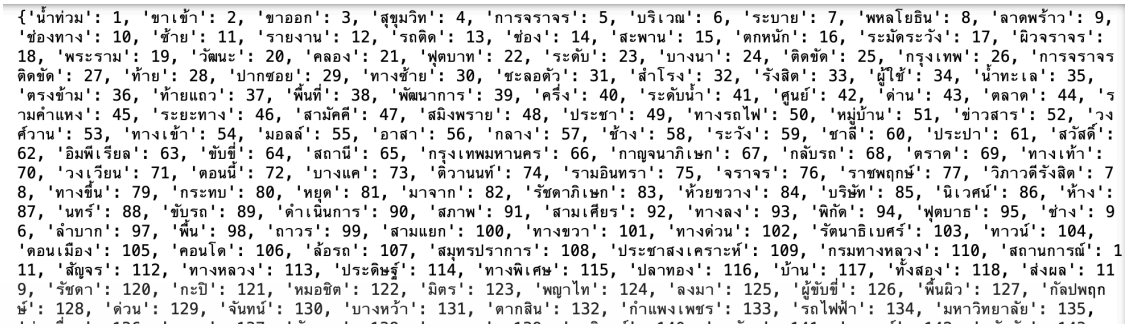
จากขั้นตอนการสร้างข้อความที่ได้อธิบายไปในขั้นต้นจะได้ข้อความที่สร้างขึ้นมาในจำนวนที่ต้องการจากอัตราส่วนที่ต่างกัน สำหรับกลุ่มภัยพิบัติต้องการข้อความอีก 6 เท่า กลุ่มข้อความพื้นที่ชุมนุมต้องการข้อความเพิ่มขึ้น 17 เท่า และกลุ่มข้อความพื้นที่ซ่อมถนนหรือทางชำรุดต้องการข้อความเพิ่มขึ้น 14 เท่า เมื่อเท่ากันทุกกลุ่มแล้วจึงนำข้อความที่สร้างได้เก็บไว้ใช้งานต่อไป

```
{'massage': ['กวาง ระดับน้ำ ทางเท้า ขับรถ ระวัง การจราจร ชะลอตัว รถติด ขัดหนัก',
  'คอนเนค คอนโด กรุงเทพ พหลโยธิน ขาเข้า น้ำท่วม ช่อง ทางด่วน ระยะทาง',
  'วงเวียน ราชภัฏ พระนคร น้ำท่วม การประปา ดำเนินการ การจราจร น้ำท่วม ผิวจราจร',
  'คลอง สำโรง น้ำทะเล ช่อง ซ้าย น้ำท่วม มิตร ห้วยขวาง บริเวณ',
  'ทางขึ้น สะพาน พื้น ตกหนัก น้ำท่วม ทั้งสอง การจราจร ส่งผล ชะลอตัว',
  'ร่มเกล้า ขาออก ศูนย์ การค้า พลาซา บางนา ช่องทาง ประปา ช่อง',
  'หัวโค้ง ตำรวจ น้ำท่วม ผิวจราจร ลงมา การจราจร บอนด์ สตรีท ทางเข้า',
  'ราชประสงค์ น้ำท่วม ฟุตบาท ลาดพร้าว วังทองหลาง วังทองหลาง กรุงเทพมหานคร น้ำท่วม ฟุตบาท',
  'ประชาสงเคราะห์ ประชาสงเคราะห์ น้ำท่วม ระมัดระวัง น้ำท่วม ลาดพร้าว น้ำท่วม ระบาย หมู่บ้าน',
  'ทีนี่ น้ำท่วม ช่อง ทางซ้าย ระดับน้ำ ทางเท้า สมิงพราย คลอง สำโรง',
  'พัฒน น้ำท่วม ฟุตบาธ ซ้าย น้ำท่วม ระดับ ตกหนัก บริเวณ พระราม',
  'กระทิง พื้นที่ น้ำท่วม การจราจร ติดขัด หยุดนิ่ง ท้ายแถว ศูนย์การค้า น้ำท่วม',
  'ขับขี่ ระมัดระวัง ผู้ใช้ ระมัดระวัง รายงาน สถานการณ์ คลอง น้ำท่วม ผิวจราจร',
  'สถานีขนส่ง หมอชิต ทางเข้า อสมท. ขับขี่ ระมัดระวัง รายงาน สถานการณ์ ตกหนัก',
  'สมบัติ ทัวร์ ผู้ใช้ ขับขี่ ระมัดระวัง รายงาน สถานการณ์ น้ำท่วม ซ้าย',
  'ภัยพิบัติ พื้นที่ กระทะ น้ำท่วม รถติด พญาไท น้ำท่วม ช่องทาง ซ้าย',
  'ค้าง ระดับ กระทบ ทางพิเศษ กาญจนาภิเษก บริเวณ สมิงพราย คลอง คลอง',
  'ทรวงอก น้ำท่วม ตลอดทาง การจราจร ชะลอตัว น้ำทะเล รายงาน จราจร รัตนาธิเบศร์',
  'มหาวิทยาลัย กรุงเทพ การจราจร ชะลอ ท้าย ขาออก ตกหนัก ท้าย ศึกษาภัณฑ์',
  'อาสา มิตรภาพ ช่อง ซ้าย ชะลอตัว สายด่วน มอเตอร์เวย์ ขาเข้า สุขุมวิท',
  'วัสดุ ถาวร รังสิต ปลาทอง บ้าน กลาง สมิติเวช สุขุมวิท ขาออก',
  'สามัคคี หมู่บ้าน สถาพร รังสิต-นครนายก คลอง สำโรง น้ำท่วม พหลโยธิน ขาออก',
  'ตรงข้าม ด่าน น้ำท่วม การจราจร รถติด สะพาน กลับรถ สะพาน สำโรง',
  'ระมัดระวัง รายงาน ขาออก ตรงข้าม ศูนย์ รับทราบ',
  'ประสาน มิตร น้ำท่วม สถานี รถไฟฟ้า บางหว้า น้ำท่วม หมู่บ้าน ลีวลี',
  'กงสุล วัฒนะ ตรวจสอบ การจราจร พื้นที่ ลำบาก รูปภาพ',
  'ถังขยะ ระมัดระวัง รายงาน จราจร ชะลอตัว ระบาย นะคะ',
  'ขาออก สะพาน ควาย น้ำท่วม ผิวจราจร ซ้าย กลาง ระยะทาง ระดับน้ำ',
  'สะพานลอย น้ำท่วม ตลาด ช่องทาง น้ำท่วม ช่อง ทางขวา ระบาย หมู่บ้าน',
  'บางนา ตอนนี้ น้ำท่วม พหลโยธิน ขาเข้า กรมทหารราบ ตกหนัก น้ำท่วม ทางเข้า',
  'คอสะพาน สุขุมวิท ตกหนัก ต้นไม้ ทางจราจร การจราจร ติดขัด ท้ายแถว ศูนย์การค้า',
  'ประชาชน จำนวนมาก ช่อง ทางซ้าย รถติด ท้าย วัฒนะ บริเวณ ศูนย์การค้า',
  'ออกจาก ด่วน ระดับน้ำ ระบาย รถติด คลอง ประปา น้ำท่วม ลาดพร้าว',
  'กงสุล วัฒนะ สะพาน กลับรถ สะพาน คลอง ตกหนัก อาสา มิตรภาพ',
  'ราษฎร์ บำเพ็ญ ห้วยขวาง กรุงเทพมหานคร น้ำท่วม พื้นผิว การจราจร รถติด รังสิต',
  'ทางขึ้น สะพาน รัตนาธิเบศร์ รายงาน จราจร ติดขัด ช่อง น้ำท่วม ทางเข้า',
```


รูปที่ 4.8 ผลลัพธ์การสร้างข้อความจากวิธีลูกโซ่มาร์คอฟ

4.3.2 ผลการสร้างข้อความด้วยวิธีแอลเอสทีเอ็ม

การสร้างข้อความจากวิธีการเรียนรู้เชิงลึกด้วยเทคนิค แอลเอสทีเอ็มคือการสร้างข้อความที่ใช้กลุ่มข้อความเดิมมาเทรนโมเดลเพื่อสร้างระบบโดยมีแนวความคิด คือ การทำนายคำถัดไปที่จะเกิดขึ้นด้วยการประมวลผลจากคำหรือประโยคที่อยู่ก่อนหน้าเนื่องจากเทคนิค แอลเอสทีเอ็มเป็นเทคนิคแบบอนุกรมเวลา (time series) ทำให้เหมาะกับการประมวลผลในเชิงการสร้างประโยคเป็นอย่างมาก และจากการออกแบบไว้ในหัวข้อก่อนหน้าทำให้ทราบถึงกระบวนการที่จะสร้าง โดยในหัวข้อนี้เป็นการนำเสนอผลลัพธ์จากการสร้างโมเดลตามที่ได้ออกแบบไว้ เริ่มจากการดำเนินการทางกระบวนการสร้างโมเดลทั่วไป คือการเตรียมข้อมูลในที่นี้เป็นจัดการกับประโยคซึ่งได้ดำเนินการไปแล้วจึงได้นำเอาส่วนข้อมูลนั้นมาใช้ได้เลยโดยการนำคำที่ผ่านการ ตัดคำ กรองสัญลักษณ์และลบคำฟุ่มเฟือยออกแล้วจึงนำมาเข้ากระบวนการเพื่อเข้ารหัสในขั้นแรกคือการทำให้คำเป็นตัวเลขลำดับคำ (text to sequence) เช่น ประโยค “สุขุมวิท น้ำท่วม ระบาย” เมื่อเปลี่ยนเป็นตัวเลขจะกลายเป็น “4, 1, 7” ซึ่งเลขที่ได้นี้นำมาจากการใส่ลำดับที่ให้กับคำโดยการใช้วิธีการดัชนีคำ (word index) จะได้เลขดัชนีดังรูปที่ 4.9



รูปที่ 4.9 ผลลัพธ์การสร้างดัชนีคำ

จากนั้นนำเลขดัชนีที่ได้มาแทนที่คำในประโยคจะได้ผลลัพธ์ดังรูปที่ 4.10

```
['ดินแดง', 'พระรามเก้า', 'น้ำท่วม', 'ขึ้น']
[164, 217, 1, 43]
['พหลโยธิน', 'ขาออก', 'ช่วง', 'ถาวร', 'รังสิต', 'น้ำท่วม', 'การจราจร']
[17, 8, 3, 28, 23, 1, 11]
```

รูปที่ 4.10 การแทนเลขดัชนีคำในประโยค

จากนั้นนำค่าตัวเลขที่ได้มาเพิ่มเลข 0 ไปข้างหน้า (pre-padding) เพื่อให้ความยาวของทุกประโยคมีความยาวเท่ากัน จะได้ผลลัพธ์ดังรูปที่ 4.11

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 164 217]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 164 217   1]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 164 217   1  43]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17   8]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17   8   3]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17   8   3  28]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17   8   3  28  23]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17   8   3  28  23   1]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17   8   3  28  23   1  11]]
```

รูปที่ 4.11 ผลลัพธ์จากการเติม 0 ไปข้างหน้าอาร์เรย์

เมื่อได้ข้อมูลพร้อมแล้วจึงดำเนินการตั้งค่าการสร้างโมเดลดังรูปที่ 4.12 โดยใช้ตัวช่วยสร้างอย่าง TensorFlow ดำเนินการโดยการเพิ่มชั้นต่าง ๆ เข้าไป จากรูปมีการเพิ่มชั้นของการฟังคำที่มีมิติเป็นจำนวนคำคูณกับความกว้างในขั้นนี้ใช้ความกว้างที่ 150 และขั้นต่อมาเป็นชั้นแอลเอสทีเอ็มโดยให้ค่า S

เป็นความยาวของประโยคสูงสุดและชั้นสุดท้ายเป็นชั้นรวมโหนดของนิวรอน (Dense) เพื่อการทำนายผลขั้นสุดท้าย

```
model = Sequential()
model.add(Embedding(total_words, 150))
model.add(Bidirectional(LSTM(max_sequence_len)))
model.add(Dense(total_words, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

```

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, None, 150)         72600

 bidirectional_2 (Bidirecti  (None, 48)                33600
 onal)

 dense_2 (Dense)             (None, 484)               23716

=================================================================
Total params: 129916 (507.48 KB)
Trainable params: 129916 (507.48 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

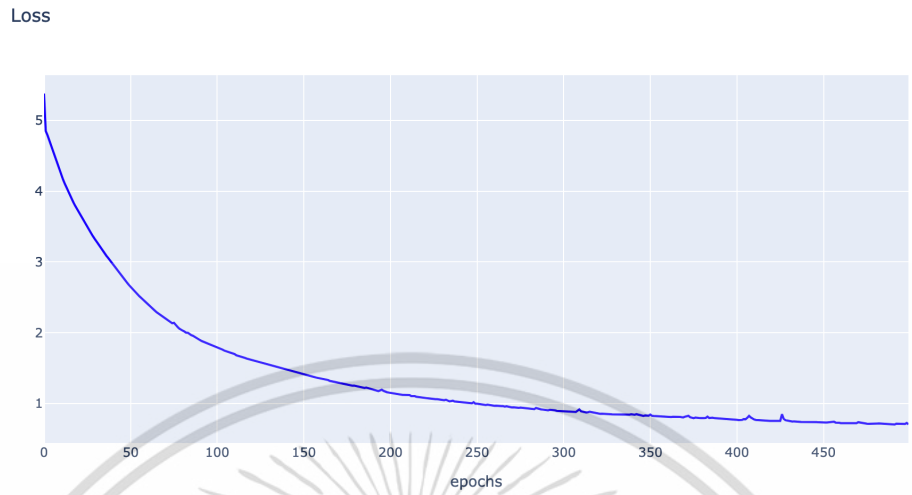
รูปที่ 4.12 การตั้งค่าเพื่อสร้างโมเดลแอลเอสทีเอ็มสำหรับการสร้างข้อความ

และขั้นตอนสุดท้ายคือกระบวนการเทรนโมเดล ซึ่งเป็นขั้นตอนที่ต้องใช้เวลาในการเทรนขึ้นอยู่กับจำนวนรอบในการเทรน (epochs) สำหรับงานนี้ใช้จำนวนรอบคือ 500 รอบ แล้วทำการเทรนโมเดลใช้เวลาประมาณ 7 นาที ดังรูปที่ 4.13

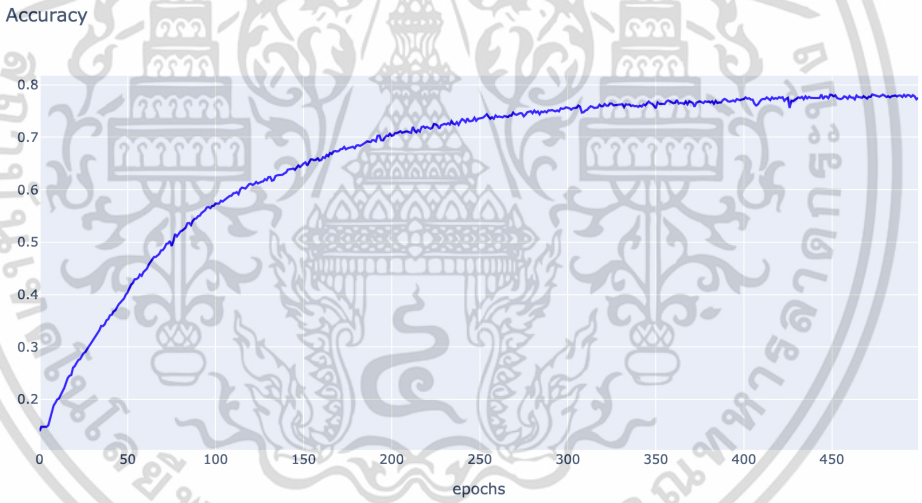
```
96/96 [==============================] - 1s 9ms/step - loss: 0.7049 - accuracy: 0.7810
Epoch 493/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7156 - accuracy: 0.7775
Epoch 494/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7146 - accuracy: 0.7807
Epoch 495/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7077 - accuracy: 0.7801
Epoch 496/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7071 - accuracy: 0.7771
Epoch 497/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7077 - accuracy: 0.7804
Epoch 498/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7130 - accuracy: 0.7781
Epoch 499/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7263 - accuracy: 0.7726
Epoch 500/500
96/96 [==============================] - 1s 9ms/step - loss: 0.7079 - accuracy: 0.7752
```

รูปที่ 4.13 ผลการเทรนโมเดลแอลเอสทีเอ็มสำหนับการสร้างข้อความเพิ่ม

สำหรับผลการเทรนของขั้นตอนนี้จะเห็นว่าความแม่นยำเพิ่มเรื่อย ๆ จนอยู่ที่ ประมาณ 0.77



รูปที่ 4.14 กราฟผลลัพธ์ความผิดพลาดในการเทรนโมเดลเพื่อสร้างข้อความเพิ่ม



รูปที่ 4.15 กราฟผลลัพธ์ความแม่นยำในการเทรนโมเดลเพื่อสร้างข้อความเพิ่ม

เมื่อสร้างโมเดลเสร็จจากนั้นนำโมเดลที่ได้มาทำนาย (predict) เพื่อหาคำถัดไปจากคำขึ้นต้นหรือคำก่อนหน้า โดยกำหนดความยาวของข้อความที่สร้างขึ้นอยู่กับความยาวประโยคต้นฉบับ และสร้างข้อความตามจำนวนที่ต้องการแต่ในขั้นตอนการทำนายนั้นโมเดลจะทำนายออกมาเป็นตัวเลขดังนั้นจึงต้องมีการแปลตัวเลขกลับมาเป็นข้อความโดยการเทียบค่าตัวเลขจากเลขดัชนีคำ

```
def gen_Text_3to8word(seed_text,amount_word):
    next_words= amount_word
    token_list = tokenizer.texts_to_sequences([seed_text])[0]
    token_list = pad_sequences([token_list], maxlen=max_sequence_len-1, padding='pre')
    for _ in range(next_words):
        token_list = tokenizer.texts_to_sequences([seed_text])[0]
        token_list = pad_sequences([token_list], maxlen=max_sequence_len-1, padding='pre')
        predicted = np.argmax(model.predict(token_list, verbose=0),axis=-1)
        output_word = ""
        for word, index in tokenizer.word_index.items():
            if index == predicted:
                output_word = word
                break
        text_gen += " " + output_word
    return text_gen
```

รูปที่ 4.16 ขั้นตอนการสร้างข้อความด้วยโมเดลแอลเอสทีเอ็ม

```
{'massage': ['การซ่อมแซม คอนโด พระราม คะนอง ขาเข้า บริเวณ ปากซอย สวัสดิ์ ช่องทาง รถติด',
 'ขับรถ ป้าย พหลโยธิน ขาเข้า ขาออก บริเวณ รามอินทรา วันที่ มิถุนายน ช่วงเวลา',
 'ทหารบก ราชพฤกษ์ ก่อสร้าง ขับรถ ระวัง สำนัก การจราจร พื้นที่ ขออภัย สายด่วน',
 'บางปะกง ร้อย การจราจร สะพาน กาญจนาภิเษก ตะวันตก การจราจร ซ้าย พหลโยธิน ขาออก',
 'บริเวณ ราชพฤกษ์ วงเวียน ห้าง กัลป์ ฤกษ์ การจราจร สะพาน ก่อสร้าง การจราจร',
 'ทางขึ้น พื้น ผิวจราจร การจราจร กลาง สะพาน คลอง สิงหาคม วันที่ มิถุนายน',
 'สัมพันธ์ อ่าน รถไฟฟ้า ดำเนินการ ปากซอย สวัสดิ์ ท้าย ปากซอย สวัสดิ์ รายงาน',
 'ทางแยก ทางด่วน สะพาน การจราจร ช่องทาง ช่อง ทางซ้าย ผิวจราจร รถติด รถติด',
 'รัตนาธิเบศร์ จรัญ การจราจร ผิวจราจร ช่อง ทางขวา ประปา ดำเนินการ ชะลอตัว ท้ายแถว',
 'พงษ์ ดำเนิน สำโรง บริษัท มิชลิน ทำงาน ขาเข้า ผิวจราจร ช่อง ทางขวา',
 'มอลล์ ชาน ผิวจราจร ช่องทาง การจราจร สะพาน รถยนต์ ขาเข้า ขาออก ช่องทาง',
 'อมรินทร์ ลำปาง ขาออก บริเวณ ทางพิเศษ กาญจนาภิเษก บริเวณ บริเวณ ทำงาน จราจร',
 'ช้าง อมรินทร์ รังสิต ผิวจราจร กุมภาพันธ์ วันที่ มีนาคม ทำการ การจราจร กลับรถ',
 'อ่าน มาจาก รถไฟฟ้า การจราจร ขาออก ช่อง ทางซ้าย ทางด่วน รถติด ท้าย',
 'ขาเข้า กู้ภัย ทางขึ้น สะพาน มาจาก วงเวียน ปรับปรุง ผิวจราจร ช่องทาง การจราจรติดขัด',
 'สะดวก บรรเทา ทำการ รถไฟฟ้า วันที่ กรกฎาคม วันที่ กรกฎาคม วันที่ กรกฎาคม',
 'ระมัดระวัง จราจร ช่องทาง ศูนย์ การแพทย์ กาญจนาภิเษก ปฏิบัติ การจราจร พหลโยธิน ขาออก',
 'จรัญสนิทวงศ์ ทางพิเศษ สวัสดิ์ ปากซอย สวัสดิ์ ช่องทาง รถติด ปากซอย สวัสดิ์ ดำเนิน',
```

รูปที่ 4.17 ผลการสร้างข้อความด้วยวิธีแอลเอสทีเอ็ม

ในข้างต้นเป็นการนำเสนอเพียงการสร้างข้อความเฉพาะบางส่วนเท่านั้น แต่ในการทดลองจริงได้มีการสร้างข้อความสำหรับทุกกลุ่มที่มีจำนวนน้อยให้มีปริมาณเท่าหรือใกล้เคียงกับกลุ่มที่มีจำนวนมาก เมื่อดำเนินการสร้างข้อความของทั้งสองวิธีในทุกกลุ่มเป็นที่เรียบร้อยได้ทำการเก็บข้อความนั้นเป็นไฟล์เพื่อรอการนำไปใช้ในการทดสอบความคล้ายคลึงกันของข้อความที่สร้างกับข้อความต้นทางเพื่อให้ทราบว่ามีความหมายที่เป็นไปในแนวทางเดียวกันหรือไม่

การเสริมข้อความคือการเปลี่ยนคำบางคำในประโยคให้เป็นคำอื่น ๆ จะทำให้ประโยคเปลี่ยนแปลงไปเล็กน้อยแต่ไม่เปลี่ยนความหมายดังนั้นจะทำให้ได้ประโยคใหม่แต่ความหมายยังคงเดิม จากที่ได้ออกแบบไว้ในหัวข้อก่อนหน้าถึงวิธีการเสริมข้อความสำหรับข้อความที่มีจำนวนน้อยให้เท่ากับกลุ่มข้อความที่มีจำนวนมากนั้นเป็นการนำเอาเทคนิคที่เป็นการสร้างไว้ล่วงหน้า เช่น เทคนิคเวิร์ดเน็ตคือการ

สร้างฐานข้อมูลกลุ่มคำไว้ หรือวิธีการไทยทูทรานส์ฟอร์เมอร์เป็นวิธีการเทรนโมเดลภาษาไว้ก่อนแล้วค่อยนำมาใช้งานต่อ หรือวิธีบีพีอีเอ็มบีที่ใช้ค่าเวคเตอร์ของการเข้ารหัสคำแบบไบต์คู่ โดยวิธีการทั้งหมดผู้วิจัยได้ทำการทดลองเพื่อหาวิธีการที่ดีที่สุดในการเสริมข้อความ ดังต่อไปนี้

4.3.3 ผลการเสริมข้อความด้วยวิธีเวิร์ดเน็ต

การเสริมข้อความด้วยวิธีเวิร์ดเน็ตคือการนำกลุ่มคำจากฐานข้อมูลที่มีมาแทนที่คำในประโยคต้นทางเพื่อให้ได้ประโยคใหม่ ซึ่งมีวิธีการคือ เริ่มจากการนำข้อความที่ได้รวบรวมมาผ่านขั้นตอนกระบวนการเตรียมข้อมูลและตัดคำฟุ่มเฟือยพร้อมทั้งกรองสัญลักษณ์ออก จากนั้นใช้ฟังก์ชัน WordNetAug ที่มีในไพไทยเอ็นแอลพีโดยคำสั่งจะเป็นการส่งคำหรือประโยคที่ต้องการเสริมเข้าไปในฟังก์ชันจะตอบกลับมาเป็นข้อความที่มีการเปลี่ยนคำที่มีความหมายในกลุ่มเดียวกันทั้งหมด

```
['ขาเข้า ก่อสร้าง รถไฟฟ้า หลอด ทางขวา',
 'กาญจนาภิเษก กลัว ย่าน สะพาน ลำคลอง พิมล ผิวจราจร ช่อง ทีวี การจราจร รถติด',
 'จราจร ช่องทาง ติวานนท์ วัฒนะ ขาเข้า ขาออก ติวานนท์ บริเวณ คาลเท็กซ์']
```

รูปที่ 4.18 การเตรียมข้อความก่อนการใช้วิธีเวิร์ดเน็ต

```
list_gen_five_gram = []
list_t_detail = []
for round_number in range(0, 1):
    for message in data_from_csv:
        print("ข้อความต้นฉบับ : ",message)
        sentences = []
        wordSplit = message.split()
        print("wordSplit :: ",wordSplit)
        str1= ''
        for i in range(len(wordSplit)):
            try:
                subAug = WNAug.augment(wordSplit[i])
                result_array = [item[0] for item in subAug]
                str1 = str1+" "+random.choice(result_array)
            except Exception as e:
                print(f"เกิดข้อผิดพลาด: {e}")
                str1 = str1+" "
                continue
        try:
            print("Wordnetaug :: ",str1)
            list_gen_five_gram.append(str1)
            list_t_detail.append(3)
            print("---------------")
            pass
        except Exception as e:
            print(f"เกิดข้อผิดพลาด: {e}")
            continue
```

รูปที่ 4.19 ขั้นตอนการเสริมข้อความด้วยวิธีเวิร์ดเน็ต

จะได้ผลลัพธ์ในรูปแบบลิสของคำที่เสริมทั้งหมดดังรูปที่ 4.20

```
ข้อความต้นฉบับ :  ขาเข้า ก่อสร้าง รถไฟฟ้า หลอด ทางขวา
wordSplit ::  ['ขาเข้า', 'ก่อสร้าง', 'รถไฟฟ้า', 'หลอด', 'ทางขวา']
Wordnetaug ::  ขาเข้าสร้างรถไฟฟ้าหลอดดูดทางขวามือ
--------------
ข้อความต้นฉบับ :  กาญจนาภิเษก กลัว ย่าน สะพาน ลำคลอง พิมล ผิวจราจร ช่อง ทีวี การจราจร รถติด
wordSplit ::  ['กาญจนาภิเษก', 'กลัว', 'ย่าน', 'สะพาน', 'ลำคลอง', 'พิมล', 'ผิวจราจร', 'ช่อง', 'ทีวี', 'การจราจร', 'รถติด']
Wordnetaug ::  กาญจนาภิเษกกลัวบริเวณสะพานลำคลองพิมลผิวจราจรรูโทรทัศน์การจราจรรถติด
--------------
ข้อความต้นฉบับ :  จราจร ช่องทาง ติวานนท์ วัฒนะ ขาเข้า ขาออก ติวานนท์ บริเวณ คาลเท็กซ์
wordSplit ::  ['จราจร', 'ช่องทาง', 'ติวานนท์', 'วัฒนะ', 'ขาเข้า', 'ขาออก', 'ติวานนท์', 'บริเวณ', 'คาลเท็กซ์']
Wordnetaug ::  จราจรช่องทางติวานนท์วัฒนะขาเข้าขาออกติวานนท์โซนคาลเท็กซ์
--------------
```

รูปที่ 4.20 ผลการเสริมคำด้วยวิธีเวิร์ดเน็ต

และสุดท้ายคือการนำคำทั้งหมดมารวมกันเป็นข้อมูลทั้งหมดจะได้ผลดังรูปที่ 4.21

	massage	t_detail
0	ทางพิเศษพระรามคะนองจราจรพระรามขาเข้าขาออกช่องทางย่านสิงหาคมติดตั้ง	3
1	สะพานสะพานการจราจรพื้นราบพระรามขาออก	3
2	ช่องทางรถติดขาออกการแพทย์กาญจนาภิเษกดีขึ้นพื้นดินผิวจราจรหลอดดูดซุปเปอร์ไฮเวย์	3
3	กรมทางหลวงจราจรพหลโยธินขาเข้าพระนครศรีอยุธยาบ่ายติดตั้งโปสเตอร์	3
4	ติดขัดสวัสดีขาเข้าหลวงลุ่ลวงการจราจรซ้ายรถติดท้ายแถวปากซอยสวัสดีรายงานจราจรรถติด	3
...	...	...
3291	การจราจรสะพานสาธุประดิษฐ์รัชดาสาธุขาเข้าทำงานปรับปรุง	3
3292	จราจรทางพิเศษมหานครขาออกทางลงติดตั้งผืนดินสะพานพฤหัสบดีสิงหาคม	3
3293	ประชาประชาหลอดโทรทัศน์ทางขวาสร้าง	3
3294	สร้างประชาฤทัยทางอ้อมดีขึ้นประชาชาลีกรุงเทพชาลีกรุงเทพ	3
3295	รถติดพหลโยธินขาเข้าสะพานควา่ยทำงานกีดขวางซ้ายรถติดท้ายแถวกำแพงเพชร	3

รูปที่ 4.21 ผลลัพธ์การเสริมคำด้วยวิธีเวิร์ดเน็ตที่เป็นประโยคตามต้องการ

4.3.4 การเสริมข้อความด้วยการใช้วิธีการไทยทูทรานส์ฟอร์เมอร์

วิธีการเสริมข้อความอีกวิธีที่นำมาใช้เนื่องจากเป็นวิธีที่แตกต่างจากวิธีก่อนหน้าวิธีการนี้เป็นการนำฟังก์ชัน Thai2transformersAug ที่มีในไพไทยเอ็นแอลพีซึ่งเป็นการต่อยอดมาจากฟังก์ชัน สำหรับการทำโจทย์ Masked Language โมเดล (MLM) ของ WangchaBERTa โดยเริ่มจากการแบ่งประโยคออกเป็นคำแล้วนำแต่ละคำไปผ่านฟังก์ชันเพื่อหา <Masked> โดยสามารถตั้งค่าได้ว่าจะให้ผลลัพธ์ตอบกลับมากี่ครั้งโดยระบุไปในพารามิเตอร์ “num_replace_tokens” ในที่นี้ตั้งค่าเป็น 5 หมายถึงให้ตอบกลับมาคำ

ละ 5 ผลลัพธ์ ดังนั้นในมุมของการเสริมข้อความของกลุ่มข้อมูลทั้งหมดข้อมูลที่มีจำนวนน้อยนั้นต้องการให้เพิ่มขึ้นกี่เท่าก็สามารถระบุไปที่พารามิเตอร์นี้ได้เลย

```
ข้อความต้นฉบับ :  ขาเข้า ก่อสร้าง รถไฟฟ้า ช่อง ทางขวา
wordSplit ::  ['ขาเข้า', 'ก่อสร้าง', 'รถไฟฟ้า', 'ช่อง', 'ทางขวา']
wordAugmented ::  ขาเข้าแม่ก่อสร้าง.รถไฟฟ้า "ช่องนมทางขวาก่อน
---------------
ข้อความต้นฉบับ :  กาญจนาภิเษก กลัว กรวย บริเวณ สะพาน คลอง พิมล ผิวจราจร ช่อง ทางขวา การจราจร รถติด
wordSplit ::  ['กาญจนาภิเษก', 'กลัว', 'กรวย', 'บริเวณ', 'สะพาน', 'คลอง', 'พิมล', 'ผิวจราจร', 'ช่อง', 'ทางขวา', 'การจราจร',
'รถติด']
wordAugmented ::  กาญจนาภิเษกแม่กลัว ก่อนที่กรวย5บริเวณลสะพาน พี่คลอง พี่พิมลกรรมผิวจราจรสช่อง "ทางขวาฟการจราจรลมรถติดมาก
---------------
ข้อความต้นฉบับ :  จราจร ช่องทาง ติวานนท์ วัฒนะ ขาเข้า ขาออก บริเวณ ติวานนท์ บริเวณ คาลเท็กซ์ วันที่
wordSplit ::  ['จราจร', 'ช่องทาง', 'ติวานนท์', 'วัฒนะ', 'ขาเข้า', 'ขาออก', 'บริเวณ', 'ติวานนท์', 'บริเวณ', 'คาลเท็กซ์', 'วันที่']
wordAugmented ::  จราจรช่องทางครับติวานนท์โคตรวัฒนะ...ขาเข้า)ขาออกยบริเวณดำติวานนท์โคตรบริเวณลคาลเท็กซ์าวันที่
---------------
```

รูปที่ 4.22 ตัวอย่างการเสริมคำด้วยวิธีไทยทูทรานส์ฟอร์เมอร์

```python
for round_number in range(0, 1):
    for message in data_from_csv:
        print("ข้อความต้นฉบับ : ",message)
        sentences = []
        wordSplit = message.split()
        print("wordSplit :: ",wordSplit)
        str1= ''
        for i in range(len(wordSplit)):
            try:
                subAug = Thai2Tran_aug.augment(wordSplit[i],num_replace_tokens= 5)
                str1 = str1+""+random.choice(subAug)
            except Exception as e:
                print(f"เกิดข้อผิดพลาด: {e}")
                str1 = str1+"error"
                continue  # ใช้ continue เพื่อให้ลูปกลับไปเริ่มใหม่
        try:
            print("wordAugmented :: ",str1)
            list_gen_five_gram.append(str1)
            list_t_detail.append(3)
            print("---------------")
            pass  # คำสั่งที่คุณต้องการให้ลูปทำงาน
```

รูปที่ 4.23 ขั้นตอนการเสริมคำด้วยวิธีไทยทูทรานส์ฟอร์เมอร์

จากขั้นตอนการเสริมคำที่ได้พัฒนามาจึงได้ผลลัพธ์ของการเสริมคำดังรูปที่ 4.24

	massage	t_detail
5	ขาเข้าแม่ก่อสร้าง.รถไฟฟ้า "ช่องนมทางขวาก่อน	3
6	กาญจนาภิเษกแม่กลัว ก่อนที่กรวย5บริเวณลสะพาน พี่คลอง พี่พิมลกรรมผิวจราจรสช่อง "ทางขวาฟการจราจรลมรถติดมาก	3
7	จราจรช่องทางครับติวานนท์โคตรวัฒนะ...ขาเข้า)ขาออกยบริเวณดำติวานนท์โคตรบริเวณลคาลเท็กซ์าวันที่	3

รูปที่ 4.24 ผลลัพธ์การเสริมคำด้วยวิธีการไทยทูทรานส์ฟอร์เมอร์

4.3.5 การเสริมคำด้วยวิธีการบีพีอีเอ็มบี

การเสริมคำวิธีสุดท้ายที่เลือกมาทดสอบในงานวิจัยในครั้งนี้เป็นการนำคำศัพท์ที่ถูกพรีเทรนเอ็มเบ็ดดิงมาจากคำย่อยของคำที่ได้มาจากข้อความหลายภาษารวมทั้งภาษาไทยที่ได้มีการทำคำย่อยด้วยวิธีการแยกไบต์คู่ วิธีการนี้ได้นำมาใช้ในหลายงานเช่นการหาคำผิด การทำ Masked Language สำหรับในงานนี้ได้นำเอาบีพีอีเอ็มบีมาเสริมข้อความคล้ายกับวิธีอื่นแต่เป็นเพียงกลุ่มข้อมูลคนละแหล่งโดยคำที่นำมาเสริมจะเลือกคำที่มีค่าน้ำหนัก (weight) ใกล้เคียงกับคำที่ต้องการ ซึ่งวิธีการนี้ไพไทยเอ็นแอลพีได้นำมารวมเข้ากับฟังก์ชันอื่น ๆ โดยชื่อ BPEmbAug ซึ่งวิธีการเสริมคำด้วยวิธีนี้จะมีกระบวนการทำงานคล้ายกับวิธีการอื่น ๆ คือเริ่มจากจัดการกับข้อความด้วยการ ตัดคำ กรองคำ จากนั้นส่งแต่ละคำเข้าไปในฟังก์ชันเพื่อเสริมคำโดยตั้งค่าให้ได้ผลลัพธ์ที่ 10 ผลลัพธ์ ดังรูปที่ 4.25

```
1 BPEaug.augment("รถชน", n_sent=10, p=0.1)
```

```
['รถชน',
 'รถชาว',
 'รถของชน',
 'รถยารา',
 'รถผิวดำ',
 'รถพิมพ์ใจ',
 'รถพี',
 'รถย์',
 'รถยิว',
 'รถขับรถ']
```

รูปที่ 4.25 ตัวอย่างคำที่ผ่านการเสริมคำด้วยวิธีบีพีอีเอ็มบี

เมื่อสามารถทดสอบการเสริมคำได้แล้วจากนั้นนำฟังชันก์นี้มาพัฒนาต่อเพื่อนำไปใช้กับข้อความทั้งหมดโดยวิธีการจะคล้ายกับวิธีก่อนหน้าคือมีการแบ่งประโยคออกเป็นคำแล้วนำคำที่ผ่านการเสริมแล้วมารวมกันอีกที

```
for message in data_from_csv:
    print("ข้อความต้นฉบับ : ",message)
    sentences = []
    wordSplit = message.split()
    print("wordSplit :: ",wordSplit)
    str1= ''
    for i in range(len(wordSplit)):
        try:
            subAug = BPEaug.augment(wordSplit[i], n_sent=10, p=0.1)
            str1 = str1+""+random.choice(subAug)
        except Exception as e:
            print(f"เกิดข้อผิดพลาด: {e}")
            str1 = str1+" "
```

รูปที่ 4.26 ขั้นตอนการเสริมคำด้วยวิธีบีพีอีเอ็มบี

```
ข้อความต้นฉบับ :  ทิงอ ลักษมณ์ ขาออก บริเวณ ธันวาคม กันยายน
wordSplit ::  ['ทิงอ', 'ลักษมณ์', 'ขาออก', 'บริเวณ', 'ธันวาคม', 'กันยายน']
wordAugmented ::  เหม่ยือนิมิตรองรับขาจะใกล้กับตุลาคมพฤษภาคม
---------------
ข้อความต้นฉบับ :  สะพาน น้ำ พื้น พื้นที่ ลักษมณ์
wordSplit ::  ['สะพาน', 'น้ำ', 'พื้น', 'พื้นที่', 'ลักษมณ์']
wordAugmented ::  แม่น้ำเจ้าพระยาและน้ำสีขาวบริเวณนิมิตรอเรนซ์
---------------
ข้อความต้นฉบับ :  กลาง รถติด ศูนย์ การแพทย์ สร้าง พื้น ติดขัด
wordSplit ::  ['กลาง', 'รถติด', 'ศูนย์', 'การแพทย์', 'สร้าง', 'พื้น', 'ติดขัด']
wordAugmented ::  แถบสีขาวรถไว้สถาบันฟิสิกส์ที่สร้างที่ตั้งวัดติดต่อความ
---------------
```

รูปที่ 4.27 ข้อความที่มีการเสริมคำด้วยวิธีการบีพีอีเอ็มบี

จากการดำเนินการที่ผ่านมาได้มีการทดลองการสร้างและเสริมข้อความด้วยวิธีต่าง ๆ และเก็บบันทึกข้อความที่สร้างขึ้นมาได้ในรูปแบบของไฟล์ซีเอสวีเพื่อสามารถนำไปใช้ในการทดลองอื่น ๆ ได้โดยง่าย สำหรับขั้นตอนต่อไปคือการทดสอบเพื่อหาความคล้ายหรือการหาความเหมือนของประโยคต้นและประโยคที่สร้างขึ้นว่ามีความคล้ายกันเท่าไร

4.3.6 ผลการวัดค่าด้วยเบลอสกอร์ร่วมกับเบิร์ตสกอร์

จากที่ได้ออกแบบไว้ในหัวข้อก่อนหน้าการวัดผลเบลอสกอร์และเบิร์ตสกอร์จะกระทำเพื่อค้นหาวิธีการเสริมและสร้างข้อความที่ดีที่สุดด้วยการวัดค่าความคล้ายหรือการหาความเหมือน โดยการทดลองจะทำทั้งหมด 30 วิธี ซึ่งมาจากการคอมไบน์กันของวิธีการทั้ง 5 วิธี และวิธีดั้งเดิมอีก 5 วิธี ดังแสดงในตารางที่ 4.4

ตารางที่ 4.4 ผลเบลอสกอร์และเบิร์ตสกอร์ทั้ง 30 วิธี

No.	Aug#1	Aug#2	BLEU	BERT	Augmentation Score
1	G4(Thai2trans)	G4(Thai2trans)	0.27087	0.86780	0.82620
2	G5(BPEmb)	G5(BPEmb)	0.01937	0.72677	0.80293
3	G4(Thai2trans)	G2(LSTM)	0.15050	0.77893	0.80010
4	G2(LSTM)	G1 (Markov)	0.12683	0.76517	0.79757
5	G4(Thai2trans)	G3(WNAug)	0.15827	0.77725	0.79660
6	G5(BPEmb)	G4(Thai2trans)	0.05117	0.72677	0.79339
7	G4(Thai2trans)	G1 (Markov)	0.07707	0.73277	0.78982
8	G4(Thai2trans)	G5(BPEmb)	0.03573	0.71417	0.78920

ตารางที่ 4.4 ผลเบลอสกอร์และเบิร์ตสกอร์ทั้ง 30 วิธี (ต่อ)

No.	Aug#1	Aug#2	BLEU	BERT	Augmentation Score
9	G5(BPEmb)	G3(WNAug)	0.04250	0.71647	0.78878
10	G5(BPEmb)	G2(LSTM)	0.02403	0.70197	0.78417
11	G2(LSTM)	G3(WNAug)	0.27370	0.80870	0.78398
12	G3(WNAug)	G5(BPEmb)	0.06783	0.71077	0.77719
13	G1 (Markov)	G2(LSTM)	0.20247	0.76137	0.77222
14	G1 (Markov)	G5(BPEmb)	0.05547	0.69813	0.77205
15	G3(WNAug)	G1 (Markov)	0.16920	0.74607	0.77149
16	G1 (Markov)		0.22173	0.76480	0.76884
17	G1 (Markov)	G4(Thai2trans)	0.20537	0.75680	0.76815
18	G5(BPEmb)	G1 (Markov)	0.04250	0.68653	0.76782
19	G1 (Markov)	G3(WNAug)	0.22400	0.75783	0.76328
20	G3(WNAug)	G2(LSTM)	0.31947	0.79793	0.76271
21	G2(LSTM)	G2(LSTM)	0.37170	0.81873	0.76160
22	G1 (Markov)	G1 (Markov)	0.21550	0.74972	0.76015
23	G2(LSTM)		0.40677	0.82940	0.75855
24	G2(LSTM)	G4(Thai2trans)	0.37243	0.81033	0.75550
25	G2(LSTM)	G5(BPEmb)	0.08360	0.68513	0.75451
26	G4(Thai2trans)		0.32777	0.78427	0.75066
27	G5(BPEmb)		0.04930	0.65580	0.74427
28	G3(WNAug)	G3(WNAug)	0.65963	0.89977	0.73195
29	G3(WNAug)	G4(Thai2trans)	0.62167	0.87897	0.72878
30	G3(WNAug)		0.70540	0.81673	0.66009

จากตารางที่ 4.4 ผลการสร้างข้อความจากวิธีการออกเมนเตชั่นซึ่งซ้อนกันจากวิธีการทั้งห้าวิธีเมื่อทำการหาค่าคะแนนทั้งเบลอสกอร์และเบิร์ตสกอร์จะได้ค่าคะแนนที่ต่างกันจึงจำเป็นต้องมีการหาค่าเพื่อเป็นตัวแทนหรือค่ากลางสำหรับการแสดงผลโดยแสดงในช่องขวาจะเห็นคอลัมน์ Augmentation Score ที่มีผลมาจากการคำนวณในสมการที่ 3.2 และผู้วิจัยได้คัดเลือกข้อความ ที่ได้จากการทำออกเมนเทชันที่มี

คะแนน Augmentation Score สูงสุดห้าอันดับมาทำการเทรนโมเดลเพื่อทดสอบการจำแนกข้อความเพื่อจำแนกครอบคลุมข้อความออกเป็นห้าประเภทเพื่อสรุปว่าจะเอา วิธีการไหนมาใช้ ให้ใส่ตารางการทดสอบ 5 วิธี ดังตารางที่ 4.5

ตารางที่ 4.5 ผลการทดสอบการจำแนกประเภทข้อความอุบัติการณ์จากการเสริมข้อความ

No	Method	Precision	Recall	f1-score	accuracy
1	$Thai2trans \circ Thai2trans$	0.60	0.58	0.58	0.78
2	$BPEmb \circ BPEmb$	0.43	0.46	0.44	0.73
3	$Thai2trans \circ LSTM$	0.67	0.52	0.54	0.79
4	$LSTM \circ Markov$	**0.87**	**0.83**	**0.85**	**0.91**
5	$Thai2trans \circ WNAug$	0.60	0.65	0.62	0.81

ตารางที่ 4.6 ตัวอย่างผลการสร้างข้อความด้วยการซ้อนวิธี

No	Method	Messages
	ข้อความต้นฉบับ	1. ดินแดง พระรามเก้า น้ำท่วม 2. น้ำท่วม ระดับ ฟุตบาท สุขุมวิท ขาเข้า 3. ขาเข้า ก่อสร้าง รถไฟฟ้า ช่อง ทางขวา 4. สมรภูมิ กรณี ชุมนุม ผลกระทบ การจราจร 5. บริเวณ กรณี ชุมนุม ผลกระทบ การจราจร
1	$Thai2trans \circ Thai2trans$	1. ดินแดง. พระรามเก้าา น้ำ 2. น้ำนม ระดับกับ ฟุตบาทดำ สุขุมวิท5 ขาเข้าแม่ง 3. ขาเข้า) ก่อสร้าง. รถไฟฟ้า ช่อง ทางขวาฮอ 4. สมรภูมิจาก กรณีนี้ ชุมนุมกอ ผลกระทบ. การจราจรda 5. บริเวณเป็น กรณีมี ชุมนุมส ผลกระทบ. การจราจร
2	$BPEmb \circ BPEmb$	1. และน้ำ เอ็กซ์ตร้า 2. ชั้นมัธยมศึกษา พัน ซอย 3. กองทัพบกช่อง 4. เซาะทราย ผู้คน 5. บริเวณ neol พงศธร ติดผิว

ตารางที่ 4.6 ตัวอย่างผลการสร้างข้อความด้วยการซ้อนวิธี (ต่อ)

No	Method	Messages
3	$Thai2trans \circ LSTM$	1. ดินแดงนม พระรามเก้านม น้ำท่วมment 2. " น้ำท่วมดี สุขุมวิทs ช้าง สามเศียรา รายงาน "" ข่าวสาร " 3. ขาเข้าก่อน ปรับพื้นme ผิวจราจรแดง กีดขวาง 4. สมรภูมิแม่ กรณีล ชุมนุม ผลกระทบแดง การจราจรกอ วันที่ 5. บริเวณส กรณีแดง ชุมนุมเลยค่ะ ผลกระทบแดง การจราจร กอ ประชาธิปไตย.
4	$LSTM \circ Markov$	1. ดินแดง ตอนนี้ ระบาย ฟุตบาท ซ้าย 2. น้ำท่วม ระบาย รายงาน ผู้ใช้งาน 3. ขาเข้า บริเวณ อาคาร ช่อง ทางขวา การจราจรติดขัด 4. สมรภูมิ ชุมนุม ผลกระทบ การจราจร ดินแดง ประชาสงเคราะห์ รวมถึง มิตร 5. บริเวณ ลาดพร้าว รัชโยธิน กลุ่ม ชุมนุม ผลกระทบ การจราจร
5	$Thai2trans \circ WNAug$	1. ดินแดงแม่ พระรามเก้า 2. ดีกรีค่ะ ฟุตบาทก สุขุมวิทs ขาเข้า) 3. " ขาเข้า) ก่อสร้าง รถไฟฟ้า ช่อง "" ทางขวาต่อ" 4. ผลกระทบแดง การจราจรกอ 5. บริเวณดำ กรณีล ชุมนุม5 ผลกระทบส การจราจรล

4.4 ผลการจำแนกข้อความด้วยวิธีการเรียนรู้เชิงลึก

การจำแนกข้อความสำหรับงานนี้ได้ออกแบบไว้สองระดับคือขั้นแรกจะเป็นการจำแนกข้อความที่เกี่ยวกับสภาพจราจรออกจากข้อความทั่วไปหรือข้อความที่ไม่เกี่ยวข้องกับการรายงานสภาพจราจรเมื่อแยกข้อความที่เกี่ยวกับสภาพจจราจรออกมาได้แล้วจะนำเอาสู่โมเดลเพื่อจำแนกประเภทของข่าวที่เกี่ยวข้องกับการรายงานสภาพจราจรอีกครั้งโดยจะแยกออกเป็น 5 ประเภท และในการแยกประเภทของข้อความสภาพจราจรนี้เองที่จำเป็นต้องมีการสร้างข้อความให้มีจำนวนเท่ากันทุกกลุ่มเนื่องจากข้อความบางกลุ่มมีจำนวนน้อยมาก เช่น ข้อความการซ่อมถนนหรือข้อความผู้ชุมนุม โดยหลังจากดำเนินการสร้างข้อความเพิ่มในทุกกลุ่มจนมีจำนวนเท่ากันเป็นที่เรียบร้อยแล้ว ในขั้นตอนต่อมาเป็นการนำข้อความนั้นมาสร้างโมเดลเพื่อการจำแนกกลุ่มข้อความ โดยกระบวนการสร้างโมเดลนั้นจะดำเนินการตามที่ได้ออกแบบ

ไว้ ซึ่งเป็นการสร้างโมเดลเพื่อการจำแนกข้อความด้วยวิธีการเรียนรู้เชิงลึกด้วยเทคนิคซีเอ็นเอ็นร่วมกับแอลเอสทีเอ็ม สำหรับหัวข้อนี้เป็นการนำเสนอผลดำเนินการในขั้นตอนต่าง ๆ ดังนี้

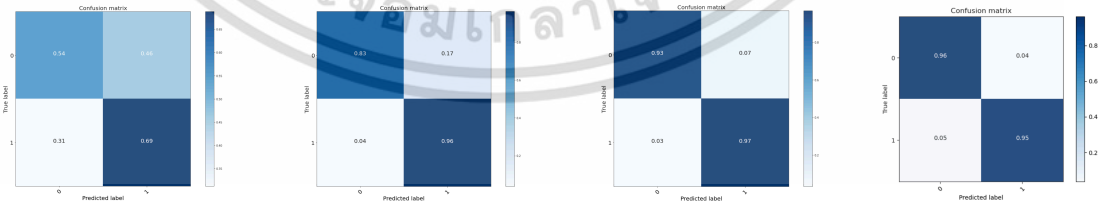
4.4.1 ผลการทดสอบเพื่อคัดเลือกวิธีการจำแนกข้อความด้วยการเรียนรู้เชิงลึก

จากที่ได้ศึกษาในหลาย ๆ งานวิจัยพบว่าในปัจจุบันมีการจำแนกข้อความด้วยวิธีการเรียนรู้เชิงลึกมากขึ้น ซึ่งนับว่าเป็นวิธีการที่ทันสมัยที่สุด แต่ด้วยเทคนิกหรือวิธีการนั้นมีหลากหลายวิธีอีกทั้งแต่ละวิธีมีความเฉพาะของแต่ละ เช่น งานทางด้านคอมพิวเตอร์วิชัน (Computer vision) หรืองานทางด้านการประมวลผลภาษาธรรมชาติหรือเอ็นแอลพี ซึ่งในส่วนของเอ็นแอลพีนั้นมีการนำวิธีการเรียนรู้เชิงลึกมาทดลองหลายวิธีด้วยการ ตัวอย่างเช่น วิธีการโครงข่ายประสาทเทียมหลายชั้น (multi-layer perceptron หรือ เพอร์เซ็ปตรอนหลายชั้น) วิธีการซีเอ็นเอ็น และสุดท้ายวิธีการแอลเอสทีเอ็ม โดยในวิทยานิพนธ์ในนี้มีการทดสอบเพื่อให้ทราบว่าวิธีการเรียนรู้เชิงลึกไหนเหมาะกับการจำแนกข้อความสภาพจราจรจากทวิตเตอร์ออกเป็นสองกลุ่ม คือ ข้อความที่ไม่เกี่ยวกับการรายงานสภาพการจราจรและข้อความที่เกี่ยวกับการรายงานสภาพจราจร โดยมีการทดลองนำข้อมูลตัวอย่างมาประมาณ 1 ใน 3 ของข้อมูลทั้งหมดเพื่อให้ใช้เวลาไม่นานมากนักต่อการเทรนโมเดล ผลการทดลองในตารางที่ 4.7 และรูปที่ 4.28

ตารางที่ 4.7 รายละเอียดผลการทดสอบเปรียบเทียบวิธีการระบุข้อความอุบัติการณ์

	MLP	CNN	LSTM	CNN+LSTM
accuracy	0.62	0.94	0.95	0.96
macro avg	0.62	0.94	0.95	0.96

จากการทดลองพบว่าวิธีการซีเอ็นเอ็นผสานแอลเอสทีเอ็มในค่าความแม่นยำสูงสุดผู้วิจัยจึงยึดเอาวิธีการดังกล่าวมาทดลองในขั้นต่อไป



(MLP) (CNN) (LSTM) (CNN+LSTM)

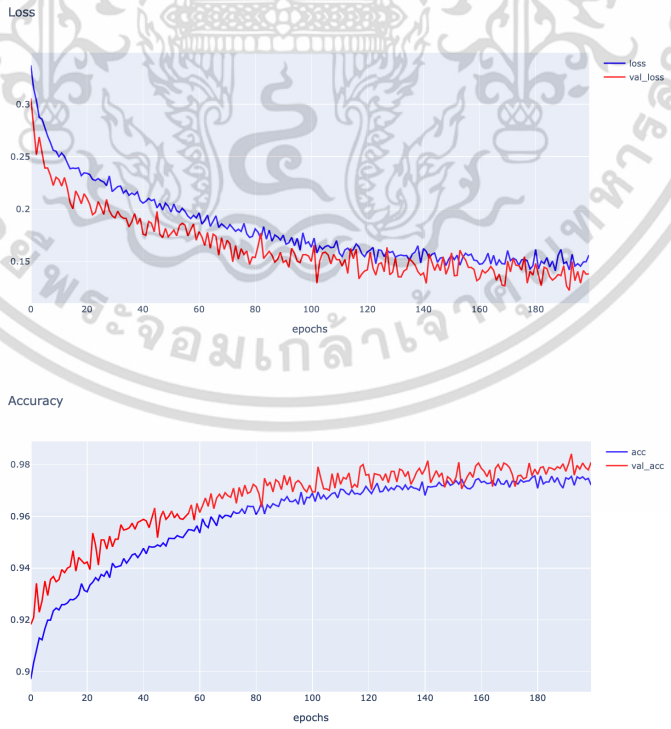
รูปที่ 4.28 ผลการเปรียบเทียบวิธีการระบุข้อความอุบัติการณ์

4.4.2 ผลการระบุข้อความอุบัติการณ์

เริ่มต้นด้วยผลของการจำแนกข้อความทั่วไปกับข้อความการรายงานสภาพจราจร สำหรับขั้นตอนนี้เป็นขั้นตอนที่ไม่ค่อยมีปัญหามากนัก เนื่องเป็นการแบ่งกลุ่มข้อความออกเป็น 2 กลุ่มมีการตั้งค่าการเทรนด้วยวิธี 5-Folds ดังรูปที่ 4.29 และได้ผลการเทรนโมเดลแสดงดังรูปที่ 4.30 ทำการเทรนที่เวลาประมาณ 3 ชม ได้ผลลัพธ์ดังตารางที่ 4.8 และรูปที่ 4.31

```python
num_folds = 5

kf = KFold(n_splits=num_folds, shuffle=True, random_state=10)

# Split ข้อมูลสำหรับ trin และ validate
for train_index, test_index in kf.split(X_train):
    X_train_fold, X_test_fold = X_train[train_index], X_train[test_index]
    y_train_fold, y_test_fold = y_train[train_index], y_train[test_index]

    # เรียกฟังชันร้างโมเดล
    model = create_model()

    # หาจุดยุติเมื่อได้โมเดลที่ดีที่สุด
    checkpoint_callback = ModelCheckpoint("best_model.h5", monitor='val_accuracy',
                                          save_best_only=True, mode='max', verbose=1)

    # เทรนโมเดล
    num_epochs = 150
    history = model.fit(X_train_fold, y_train_fold, batch_size=batch_size,
                        epochs=num_epochs,
                        validation_data=(X_test_fold, y_test_fold),
                        callbacks=[EarlyStopping(monitor='val_loss', patience=10, min_delta=0.0001),
                                   checkpoint_callback])

    # ทดสอบ Fold
    test_loss, test_accuracy = model.evaluate(X_test_fold, y_test_fold)
    fold_scores.append(test_accuracy)

    # หาโมเดลที่ดีที่สุด
    if test_accuracy > best_accuracy:
        best_accuracy = test_accuracy
        best_model = model
```

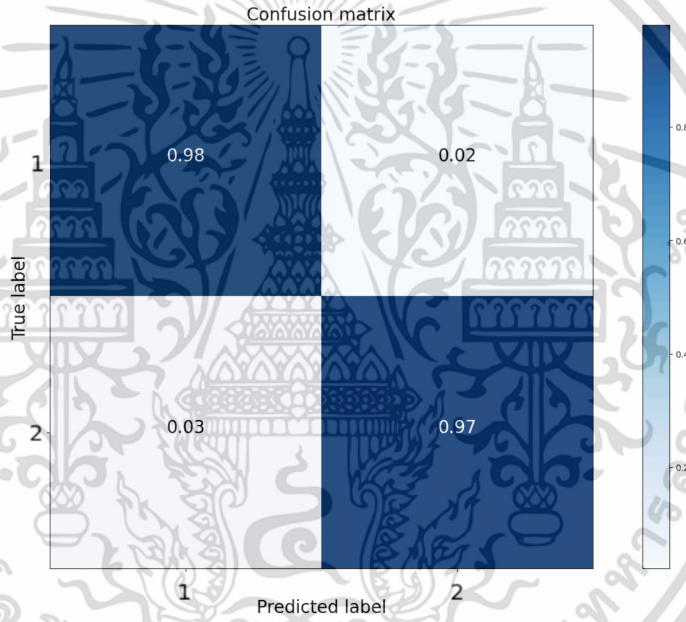
รูปที่ 4.29 ขั้นตอนการเทรนด้วยวิธี 5-Fold



รูปที่ 4.30 ผลการเทรนโมเดลเพื่อระบุข่าวอุบัติการณ์

ตารางที่ 4.8 รายละเอียดผลการทดสอบการระบุข้อความอุบัติการณ์

	CNN+LSTM		
ประเภทข้อความ	Precision	Recall	F1-score
ข้อความที่ไม่เกี่ยวกับการรายงานสภาพการจราจร	0.97	0.98	0.98
ข้อความที่เกี่ยวกับการรายงานสภาพจราจร	0.98	0.97	0.97
accuracy			0.97
macro avg	0.97	0.97	0.97



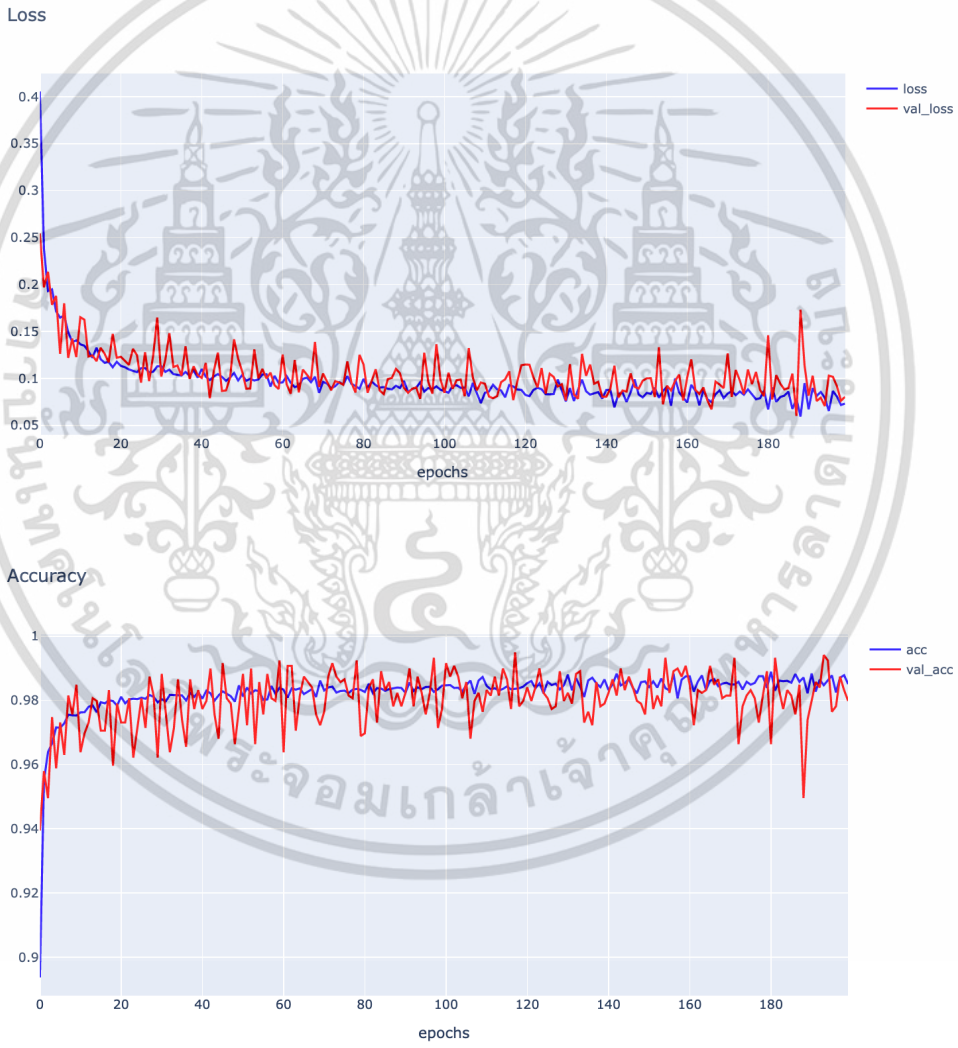
รูปที่ 4.31 ผลการทดสอบการระบุข้อความข่าวทั่วไปและข่าวอุบัติการณ์

4.4.3 ผลการจำแนกประเภทข้อความอุบัติการณ์ระดับแรก

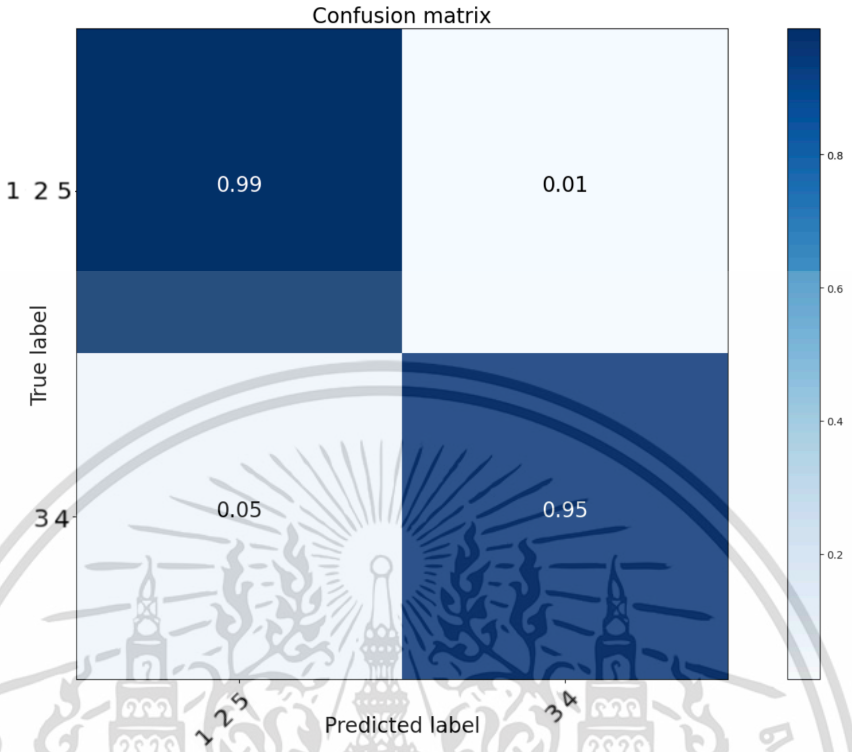
จากที่ได้ออกแบบการทดลองในหัวข้อก่อนหน้าแล้วนั้น สำหรับหัวข้อนี้จะเป็นการทดสอบโมเดลเพื่อแยกประเภทของข้อความอุบัติการณ์โดยในชั้นแรกนี้จะเป็นการแยกข้อความออกเป็นสองกลุ่มโดยกลุ่มที่ศูนย์จะแทนกลุ่ม ข้อความอุบัติเหตุ ข้อความจราจร และข้อความปิดถนนซ่อมบำรุง และในกลุ่มที่มีลาเบลหมายเลขหนึ่งจะแทน กลุ่มข้อความของการชุมนุมปิดถนนและข้อความภัยพิบัติน้ำท่วมถนน ซึ่งการเทรนโมเดลสำหรับหัวข้อนี้จะใช้เทคนิคการเรียนรู้เชิงลึกซีเอ็นเอ็นและแอลเอสทีเอ็มในการสร้างโมเดลดังที่ได้ออกแบบไว้ในหัวข้อก่อนหน้า โดยผลการเทรนโมเดลแสดงได้ดังต่อไปนี้

```python
# Create CNN+LSTM model
model = Sequential()
model.add(Embedding(len(embedding_model.wv.key_to_index)+1,
                    embedding_dim, embeddings_initializer=Constant(embedding_matrix),
                    input_length=max_length, trainable=False))
model.add(Conv1D(128, 5, activation='relu',kernel_regularizer=regularizers.l2(0.01)))
model.add(MaxPooling1D(2))
model.add(LSTM(128))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))
```

รูปที่ 4.32 ขั้นตอนการเทรนโมเดลเพื่อจำแนกข้อความอุบัติการณ์



รูปที่ 4.33 ผลการเทรนโมเดลเพื่อจำแนกประเภทข้อความจราจรระดับแรก



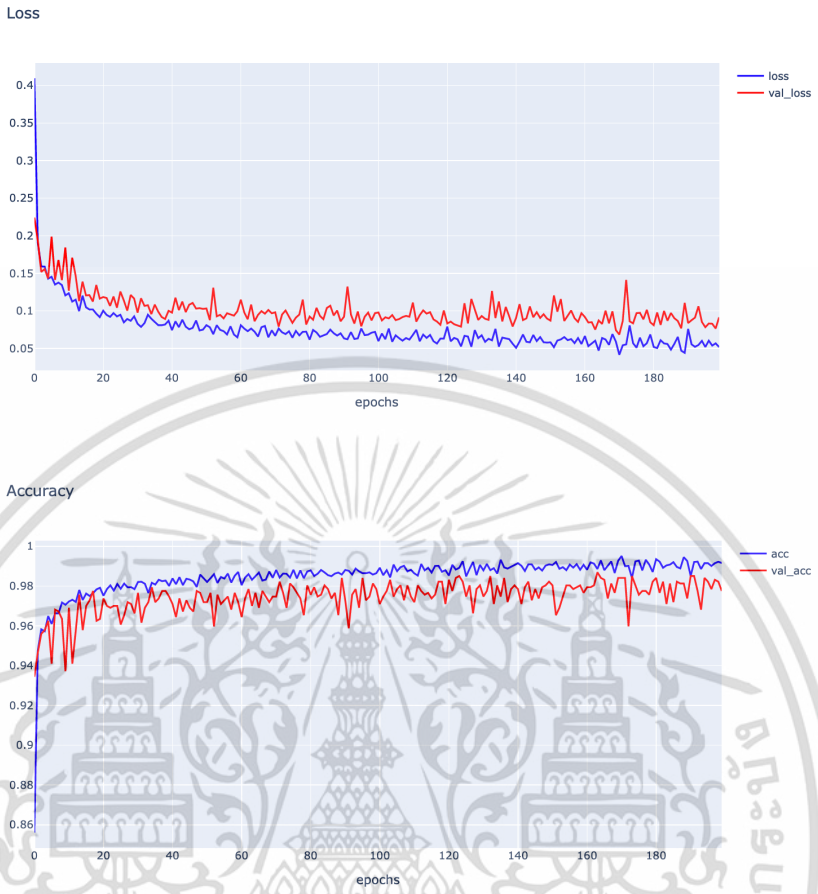
รูปที่ 4.34 ผลการทดสอบการระบุข้อความระดับแรก

4.4.4 ผลการจำแนกประเภทข้อความอุบัติการณ์ระดับสอง

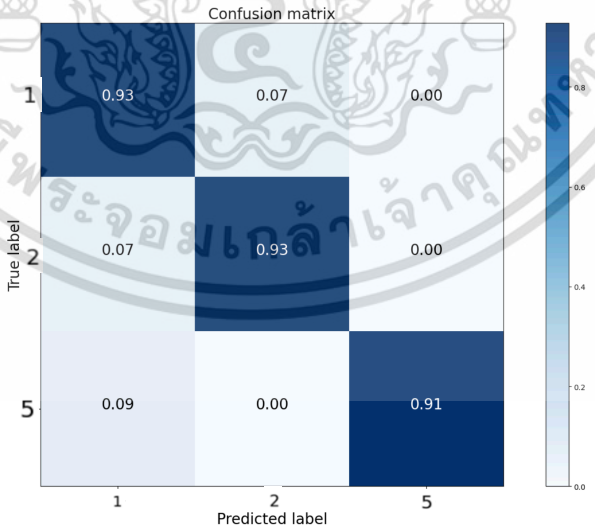
ในขั้นตอนต่อมาเป็นการแยกประเภทข้อความอุบัติการณ์ในระดับที่สองซึ่งเป็นระดับที่จะต้องระบุประเภทของข้อความออกเป็นห้าประเภทแต่เนื่องจากเป็นการทำงานในระดับที่สองซึ่งต่อจากระดับที่หนึ่งที่มีการแยกเป็นสองกลุ่ม คือคลาสศูนย์และคลาสหนึ่งซึ่งคลาสศูนย์นั้นเป็นข้อความของกลุ่มข้อความจราจร ข้อความอุบัติเหตุ และข้อความซ่อมถนน ทั้งสามข้อความนี้นำมาเทรนโมเดลเพื่อแยกประเภททั้งสามออก ส่วนคลาสหนึ่งเป็นข้อความสองกลุ่มคือกลุ่มข้อความภัยพิบัติ และข้อความชุมนุมปิดถนน มีรายละเอียดผลดังต่อไปนี้

4.4.4.1 ผลการจำแนกประเภทข้อความอุบัติการณ์ 3 กลุ่ม

จากที่ได้อธิบายไว้ในการเทรนโมเดลระดับที่สองจะเป็นการสร้างโมเดลออกเป็นสองโมเดล โมเดลแรกนี้จะเป็นการจำแนกประเภทออกเป็นสามกลุ่มซึ่งก็คือกลุ่ม ข้อความจราจร ข้อความอุบัติเหตุ และข้อความซ่อมถนนมีผลการเทรนโมเดลและผลการจำแนกข้อความดังต่อไปนี้



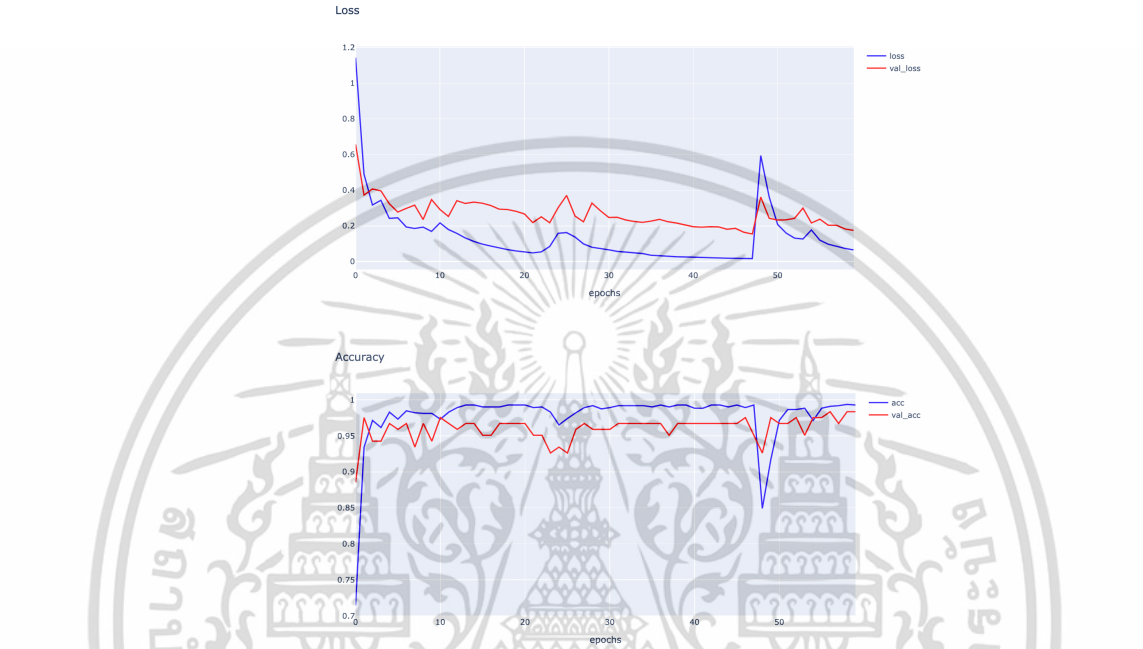
รูปที่ 4.35 ผลการเทรนโมเดลเพื่อจำแนกข้อความระดับที่สองโมเดลแยกสามกลุ่ม



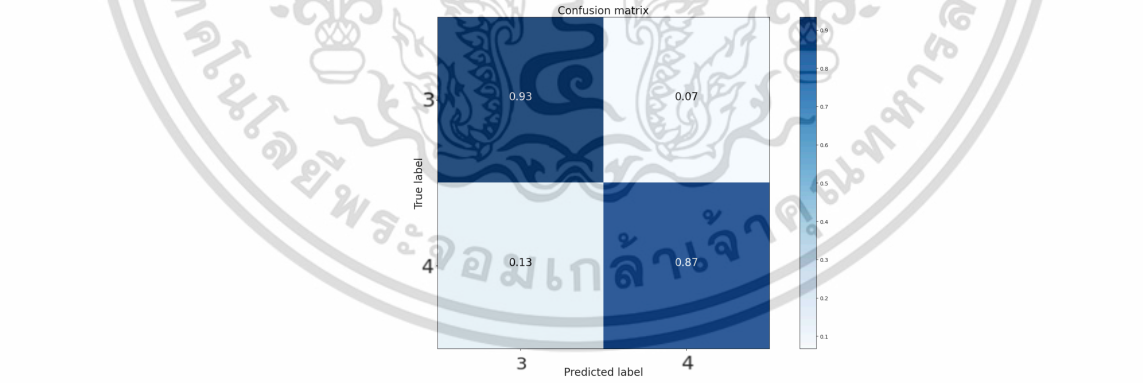
รูปที่ 4.36 ผลการทดสอบจำแนกข้อความระดับที่สองโมเดลแยกสามกลุ่ม

4.4.4.2 ผลการจำแนกประเภทข้อความอุบัติการณ์ 2 กลุ่ม

โมเดลในระดับที่สองสำหรับการจำแนกประเภทข้อความออกเป็นสองกลุ่มซึ่งโมเดลนี้จะเป็นการแยกประเภทกลุ่มของ ข้อความภัยพิบัติ และข้อความชุมนุมปิดถนน ซึ่งมีผลการเทรนโมเดลและผลการจำแนกข้อความดังต่อไปนี้



รูปที่ 4.37 ผลการเทรนโมเดลเพื่อจำแนกข้อความระดับที่สองโมเดลแยกสองกลุ่ม



รูปที่ 4.38 ผลการทดสอบจำแนกข้อความระดับที่สองโมเดลแยกสองกลุ่ม

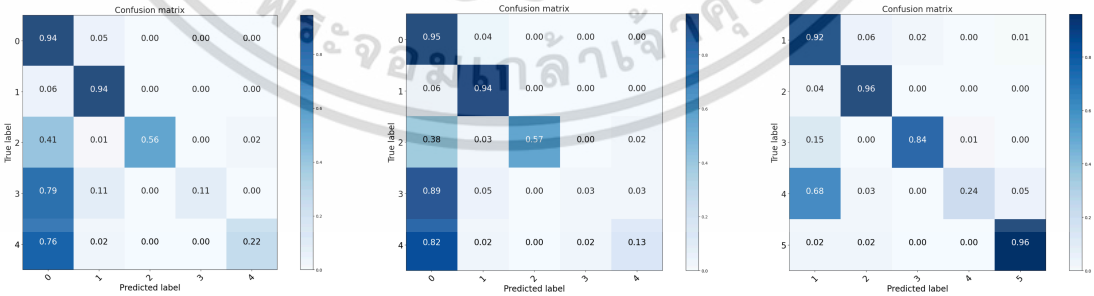
4.4.5 ผลการจำแนกประเภทข้อความอุบัติการณ์สรุป

ในการพัฒนาโมเดลเพื่อจำแนกข้อความอุบัติการณ์ในงานวิจัยในครั้งนี้จากที่ได้ออกแบบโดยมีการสร้างโมเดลเพื่อจำแนกข้อความอุบัติการณ์ออกเป็นสองระดับด้วยกัน ซึ่งเป็นผลจากการทดลองเพื่อหา

วิธีการที่ทำให้การจำแนกประเภทแม่นยำขึ้นประกอบไปด้วยวิธีการที่ไม่มีการแยกออกเป็นสองระดับและเปรียบเทียบกับวิธีที่มีการแยกกลุ่ม ดังผลลัพธ์ในตารางที่ 4.9 และรูปที่ 4.39 จะเห็นว่าผลการจำแนกข้อความด้วยวิธีการคลาสย่อยจะให้ความแม่นยำสูงขึ้น ดังนั้นจึงใช้วิธีการนี้ทำการสร้างโมเดลจริง โดยโมเดลในระดับแรกจะเป็นการแยกข้อความออกเป็นสองกลุ่มโดยโมเดลแรกเป็นการจำแนกข้อความสามกลุ่มคือ ข้อความจราจร ข้อความอุบัติเหตุ และข้อความซ่อมถนน และในโมเดลที่สองจะเป็น ข้อความภัยพิบัติและข้อความชุมนุมปิดถนนโดยโมเดลทั้งสองจะเป็นการแยกข้อมูลออกเป็นห้ากลุ่ม และสุดท้ายแล้วจะนำโมเดลในระดับที่สองมารวมกันเพื่อแยกข้อความออกเป็นห้าประเภทตามข้อมูลที่ได้รวบรวมมา โดยผลการจำแนกประเภทข้อความอุบัติการณ์ที่สร้างโมเดลมาจากข้อความที่ผ่านการเสริมข้อมูลทำให้การเทรนโมเดลและการจำแนกข้อความมีประสิทธิภาพมากยิ่งขึ้นดังแสดงได้ในตารางที่ 4.10 ซึ่งเป็นการเปรียบเทียบกับวิธีการอ้างอิง

ตารางที่ 4.9 ผลการเปรียบเทียบการจำแนกประเภทข้อความอุบัติการณ์จากสามวิธี

ประเภทข้อความ	$F1_{CNN\text{-}LSTM}$	$F1_{Duble\ aug\ CNN\text{-}LSTM}$	$F1_{Sub\ model\ CNN\text{-}LSTM}$
การรายงานสภาพการจราจร	0.86	0.90	0.86
การรายงานอุบัติเหตุบนถนน	0.95	0.95	0.94
การรายงานภัยพิบัติ	0.72	0.85	0.72
การรายงานพื้นที่ชุมนุม	0.18	0.38	0.05
การรายงานพื้นที่ซ่อมถนนหรือทางชำรุด	0.34	0.92	0.22
ACC	0.87	0.91	0.86
F1-Score Avg.	0.61	0.80	0.56

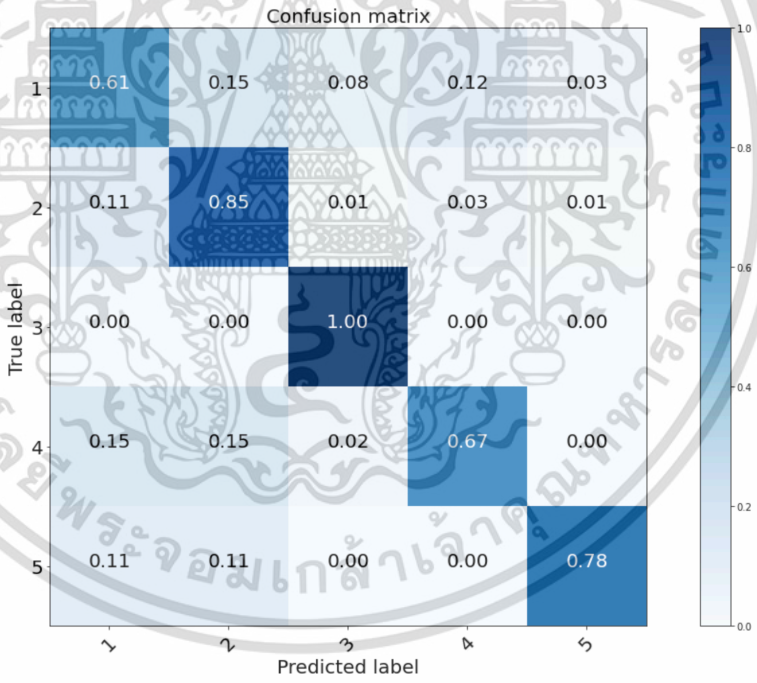


(ก) (ข) (ค)

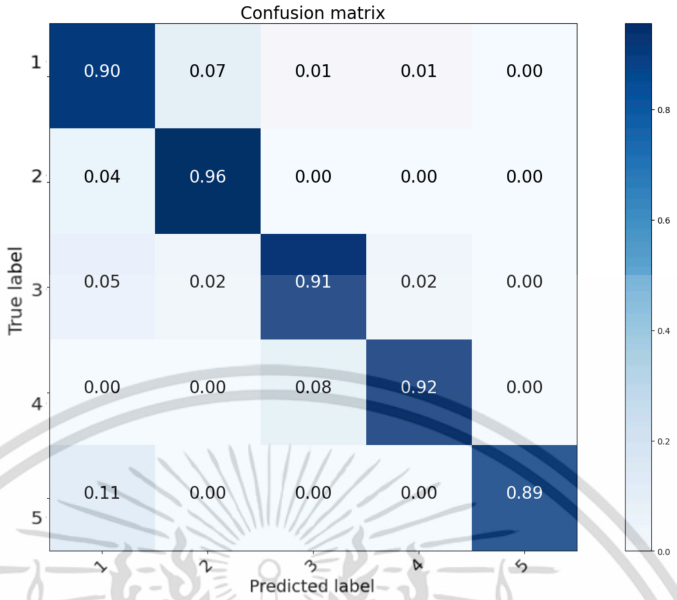
รูปที่ 4.39 ผลการเปรียบเทียบการจำแนกประเภทข้อความอุบัติการณ์จากสามวิธี

ตารางที่ 4.10 ผลการจำแนกประเภทข้อความอุบัติการณ์เปรียบเทียบกับวิธีการ Baseline

	BERT + CNN (Baseline)			Augmentation+CNN+LSTM		
ประเภทข้อความ	Precision	Recall	F1-score	Precision	Recall	F1-score
การรายงานสภาพการจราจร	0.92	0.90	0.91	0.93	0.90	0.91
การรายงานอุบัติเหตุบนถนน	0.88	0.88	0.88	0.94	0.96	0.95
การรายงานภัยพิบัติ	0.85	0.83	0.84	0.90	0.91	0.91
การรายงานพื้นที่ชุมนุม	0.37	0.50	0.43	0.81	0.92	0.86
การรายงานพื้นที่ซ่อมถนนหรือทางชำรุด	0.78	0.87	0.82	0.98	0.89	0.93
ACC			0.76			0.93
F1-Score Avg.			0.77			0.91



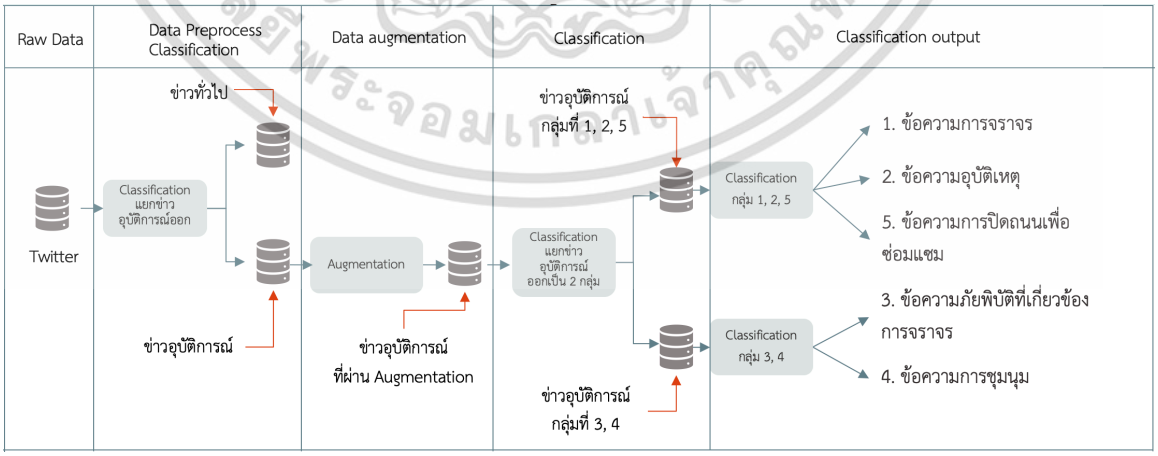
รูปที่ 4.40 ผลการจำแนกข้อความอุบัติการณ์ด้วยวิธีเบิร์ตร่วมกับซีเอ็นเอ็น (Baseline)



รูปที่ 4.41 ผลการจำแนกข้อความอุบัติการณ์ด้วยวิธีการเสริมคำร่วมกับซีเอ็นเอ็นผสานแอลเอสทีเอ็ม

4.5 สรุปผลการทดลอง

จากการทดลองที่ได้ดำเนินการมาทั้งหมดสามารถสรุปผลการทดลองให้อยู่ในเฟรมเวิร์คของการจำแนกข้อความอุบัติการณ์จากทวิตเตอร์ด้วยการปรับปรุงข้อมูลที่ไม่สมดุลด้วยการเสริมข้อความด้วยวิธีแอลเอสทีเอ็มและลูกโซ่มาร์คอฟ เพื่อนำมาสร้างโมเดลในขั้นสุดท้ายสำหรับการจำแนกข้อความอุบัติการณ์จากทวิตเตอร์ ดังรูปที่ 4.42



รูปที่ 4.42 ขั้นตอนการทำงานสรุปจากการทดลอง

จากรูปที่ 4.42 แสดงรายละเอียดสรุปการทดลองการระบุข้อความอุบัติการณ์และการจำแนกประเภทข้อความที่เกี่ยวกับสภาพจราจรได้ผลสามารถสรุป ดังนี้

ขั้นแรกเป็นการเทรนโมเดลด้วยซีเอ็นเอ็นผสานแอลเอสทีเอ็มเพื่อระบุข้อความอุบัติการณ์ซึ่งมีอยู่ 2 กลุ่มซึ่งข้อความนั้นมีจำนวนข้อมูลที่ใกล้เคียงกันทั้งสองกลุ่ม ดังนั้น จึงไม่เป็นต้องมีกระบวนการจัดการความไม่สมดุล ส่งผลให้การเทรนโมเดลเป็นไปได้อย่างดีและมีผลลัพธ์ของการระบุข้อความอุบัติการณ์จากข้อมูลทดสอบที่มีค่าความแม่นยำอยู่ที่ 0.97 ดังตารางที่ 4.8 และจากการสำรวจประสิทธิภาพจากตารางเมทริกซ์ความสับสนในรูปที่ 4.31 จะเห็นว่ามีการทำนายได้ผลที่มีความแม่นยำเป็นที่น่าพอใจ

ขั้นตอนที่สองเป็นการเทรนโมเดลเพื่อจำแนกประเภทของข้อความอุบัติการณ์ในส่วนนี้จะแบ่งออกเป็น 5 กลุ่มซึ่งมีบางกลุ่มที่มีจำนวนน้อยกว่าข้อความกลุ่มอื่นหลายเท่าจึงจำเป็นต้องนำมาปรับปรุงเพื่อให้ข้อความนั้นสมดุลเสียก่อน ซึ่งวิธีการที่นำมาปรับปรุงข้อมูลนั้นมีการทดลองด้วยกัน 30 วิธี ซึ่งวิธีที่ให้ผลลัพธ์มีประสิทธิภาพสุด คือ การเพิ่มข้อมูลคือวิธีการแอลเอสทีเอ็มผสานกับวิธีลูกโซ่มาร์คอฟที่ให้ค่าคะแนนการเสริมอยู่ที่ 0.78 ดังตารางที่ 4.5 และเมื่อนำข้อมูลมาเทรนโมเดลเพื่อจำแนกข้อความที่เกี่ยวกับสภาพจราจรนั้นสามารถจำแนกได้ดีกว่าวิธีการ baseline ที่ใช้วิธีการรวมข้อมูลจากเบิร์ตมาร่วมกับวิธีซีเอ็นเอ็นซึ่งการทดลองมีผลลัพธ์ดังตารางที่ 4.10 ดังนั้นสรุปได้ว่าวิธีการที่ผู้วิจัยนำเสนอนั้นสามารถจำแนกประเภทข้อความได้ดียิ่งขึ้น 18.18% จากวิธีการของ baseline อย่างไรก็ตามจากงานวิจัยครั้งนี้พบว่าการสร้างข้อความที่มากขึ้นจะทำให้การเทรนโมเดลใช้เวลาการเทรนมากตามไปด้วย หากต้องการสร้างข้อความมากขึ้นไปอีกจะต้องมีการปรับเปลี่ยนเครื่องมือหรือคอมพิวเตอร์สำหรับการประมวลผลที่ดีเพื่อการเทรนโมเดลการเรียนรู้เชิงลึกโดยเฉพาะทำให้การเทรนโมเดลจะทำได้เร็วยิ่งขึ้น

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

การใช้ทวิตเตอร์เพื่อเป็นช่องทางในการรับรู้ข่าวสารอุบัติการณ์ต่าง ๆ บนถนนเป็นช่องทางที่มีประสิทธิภาพและรวดเร็ว แต่ข้อความที่ได้รวบรวมมาจากทวิตเตอร์มีความไม่สมดุลโดยเลี่ยงไม่ได้ดังนั้น การนำข้อความที่รวบรวมได้มาเทรนโมเดลทันทีเลยนั้นจะทำให้โมเดลที่ได้เกิดการเอนเอียง (bias) ซึ่งงานวิจัยฉบับนี้มุ่งเน้นแก้ปัญหาความไม่สมดุลของข้อความและได้ผลลัพธ์การทดลองเป็นที่น่าพอใจและมีความสอดคล้องกับสมมุติฐานในขั้นต้นที่ว่าการสร้างข้อความให้สมดุลก่อนแล้วจึงนำข้อมูลนั้นมาเทรนโมเดลด้วยวิธีการเรียนรู้เชิงลึกที่มีการนำซีเอ็นเอ็นมาทำงานร่วมกับแอลเอสทีเอ็มจะทำให้โมเดลที่สร้างขึ้นมาสามารถจำแนกข้อความที่เกี่ยวกับสภาพจราจรได้อย่างมีประสิทธิภาพ อย่างไรก็ตามผู้วิจัยได้สรุปขอบเขตและข้อจำกัดของงาน ปัญหาและอุปสรรครวมถึงข้อเสนอแนะอื่น ๆ ดังนี้

5.1 ขอบเขตและข้อจำกัด

การเก็บข้อมูลมาทดสอบสำหรับงานนี้ เดิมทีเป็นการเก็บข้อมูลมาทดลองเพียงน้อยจึงพบว่าข้อมูลน้อยไม่สามารถเทรนโมเดลได้อย่างมีประสิทธิภาพมาก นักวิจัยจึงเก็บข้อมูลเพิ่มจนได้ข้อมูลตามที่ได้นำเสนอไป โดยเป็นการรวบรวมข้อมูลมาจากบัญชีที่เป็นหน่วยงานที่ดำเนินการงานเกี่ยวกับรายงานสภาพการจราจรเป็นหลักทำให้ได้ข้อความที่เป็นการพิมพ์ในรูปแบบภาษาที่เป็นทางการ ส่งผลให้การเทรนโมเดลเพื่อจำแนกข้อความจากสื่อสังคมออนไลน์ยังจำกัดอยู่เพียงแค่ข้อความที่เป็นทางการเท่านั้น

5.2 ปัญหาและอุปสรรค

ในส่วนของการดำเนินงานตลอดการวิจัยมีปัญหาและอุปสรรคในการเก็บข้อมูลเนื่องจากผู้วิจัยได้จำกัดขอบเขตที่จะศึกษาเฉพาะข้อความที่เป็นบัญชีทางการเท่านั้นข้อมูลที่ได้จึงมีน้อยจนทำให้ข้อมูลในบางกลุ่มมีน้อยมาก ๆ เช่น กลุ่มข้อความที่เป็นการรายงานการปิดการจราจรเพื่อซ่อมแซมถนนมีการรายงานแจ้งข่าวการทำงานไม่ครอบคลุมการทำงานจริง

ปัญหาอีกส่วนที่ผู้วิจัยพบเจอ คือการจัดหมวดหมู่ให้กับข้อความเนื่องจากข้อความบางข้อความมีการรายงานที่เป็นลักษณะเหตุการณ์ต่อเนื่องจึงยากต่อการแบ่งกลุ่ม เช่น การรายงานสภาพจราจรติดขัดจากเหตุการณ์อุบัติเหตุ นั้นคือการรายงานข่าวเดียวแต่มีความสับสนว่าจะแบ่งกลุ่มไปเป็นกลุ่มใด หรือข้อความที่เป็นการรายงานท่อน้ำประปาแตกน้ำท่วมถนนรถไม่สามารถผ่านได้ ข้อความนี้อาจจะทำให้ทั้งผู้

ที่ดำเนินการจัดกลุ่มสับสนว่าจะเป็นการระบุให้เป็นข้อความซ่อมถนนหรือข้อความภัยพิบัติ ปัญหาเหล่านี้ทำให้การทำงานในการระบุหมวดหมู่ข้อความเป็นไปอย่าล่าช้า

5.3 ข้อเสนอแนะ

จากการพัฒนาโมเดลเพื่อการระบุข้อความอุบัติการณ์และการจำแนกประเภทของข้อความสภาพจราจรนั้นทำให้ทราบถึงขั้นตอนกระบวนเทรนโมเดลในส่วนของกระบวนการต่าง ๆ และสามารถสร้างระบบเพื่อการตรวจจับและระบุข้อความที่เกิดขึ้นบนทวิตเตอร์ได้อย่างมีประสิทธิภาพ ในการพัฒนาต่อในส่วนของการสร้างระบบเพื่อจำแนกข้อความจากสื่อสังคมออนไลน์นั้น ผู้วิจัยขอเสนอแนะให้มีการนำข้อมูลที่หลากหลายขึ้น ไม่จำกัดเพียงแค่บัญชีทางการเท่านั้นอาจจะเป็นข้อความที่มาจากแหล่งอื่น ๆ ทั่วไปไม่จำกัดเพียงแค่ทวิตเตอร์แต่จะเป็นในส่วนของเฟสบุ๊ค หรือกระทู้ในเว็บไซต์ต่าง ๆ ซึ่งหากมีการนำข้อมูลเพิ่มเติมเข้ามามากขึ้นนั้น เลี่ยงไม่ได้ที่จะต้องมีการจัดหมวดหมู่ที่มากขึ้นตามไปด้วย ซึ่งจะเป็นการสิ้นเปลืองทรัพยากรบุคคลไปโดยเปล่าประโยชน์ ดังนั้นในส่วนของขั้นตอนการจัดหมวดหมู่อาจจะเป็นการนำวิธีการเรียนรู้ทางเครื่องเข้ามาช่วยไม่ว่าจะเป็น การใช้ต้นแบบการจัดกลุ่ม (Clustering Model) ซึ่งเป็นการจัดกลุ่มข้อมูลแบบไม่มีผู้สอน (Unsupervised Learning) จะสามารถช่วยลดภาระการทำงานของบุคลากรได้เป็นอย่างดี

เอกสารอ้างอิง

[1] X. Wan, M. C. Lucic, H. Ghazzai, and Y. Massoud, "Empowering real-time traffic reporting systems with NLP-processed social media data," *in IEEE Open Journal of Intelligent Transportation Systems*, vol. 1, pp. 159-175, 2020.

[2] S. Dabiri and K. Heaslip, "Developing a Twitter-based traffic event detection model using deep learning architectures," *Expert Systems with Applications*, vol. 118, pp. 425-439, 2019.

[3] V. Maslej-Krešňáková, M. Sarnovský, P. Butka, and K. Machová, "Comparison of deep learning models and various text pre-processing techniques for the toxic comments classification" *Applied Sciences*, vol. 10, no. 23, pp. 8631, 2020.

[4] G. A. Neruda and E. Winarko, "Traffic event detection from Twitter using a combination of CNN and BERT," *in 2021 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, Depok, Indonesia, 2021.

[5] Y. Chen, Y. Lv, X. Wang, L. Li, and F.-Y. Wang, "Detecting traffic information from social media texts with deep learning approaches," *in IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 3049-3058, Aug. 2019.

[6] C. F. Moreno-García, C. Jayne, and E. Elyan, "Class-decomposition and augmentation for imbalanced data sentiment analysis," *in 2021 International Joint Conference on Neural Networks (IJCNN)*, Shenzhen, China, 2021.

[7] S. Akkaradamrongrat, P. Kachamas, and S. Sinthupinyo, "Text generation for imbalanced text classification," *in 2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Chonburi, Thailand, 2019, pp. 181-186.

[8] S. Shaikh, SM. Daudpota, AS. Imran, and Z. Kastrati, "Towards improved classification accuracy on highly imbalanced text dataset using deep neural language models," *Applied Sciences*, vol. 11, no. 2, pp. 869, 2021.

[9] P. Prakrankamanant and E. Chuangsuwanich, "Tokenization-based data augmentation for text classification," *in 2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Bangkok, Thailand, 2022.

[10] B. Heinzerling and M. Strube, “BPEmb: Tokenization-free pre-trained subword embeddings in 275 languages,” in Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 2018.

[11] W. Phatthiyaphaibun, K. Chaovavanich, C. Polpanumas et al., “PyThaiNLP: Thai Natural Language Processing in Python,” June 2016.

[12] A. Salas, P. Georgakis and Y. Petalas, “Incident detection using data from social media,” *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, 2017, pp. 751-755.

[13] A. Salas, P. Georgakis, C. Nwagboso, A. Ammari and I. Petalas, "Traffic event detection framework using social media,” 2017 IEEE International Conference on Smart Grid and Smart Cities (ICSGSC), Singapore, 2017, pp. 303-307.

[14] J. C. Chamby-Diaz and A. Bazzan, “Identifying Traffic Event Types from Twitter by Multi-Label Classification,” 2019 8th Brazilian Conference on Intelligent Systems (BRACIS), Salvador, Brazil, 2019, pp. 806-811.

[15] L. Lowphansirikul, C. Polpanumas, N. Jantrakulchai, et al., “Wangchanberta: Pretraining transformer-based Thai language models,” *arXiv*, 2021.

[16] S. Lai and D. Lei, “Calculation of sentence vector similarity based on fasttext model of weighted fusion,” in 2022 4th International Conference on Advances in Computer Technology, Information Science and Communications (CTISC), Suzhou, China, 2022, pp. 1-6

[17] Bishop, Christopher M., and Nasser M. Nasrabadi, “Pattern recognition and machine learning,” Vol. 4. No. 4. New York: springer, 2006.

[18] Rong, Xin. “word2vec parameter learning explained,” arXiv, 2014.

[19] Papineni, Kishore et al, “Bleu: a Method for Automatic Evaluation of Machine Translation,” Annual Meeting of the Association for Computational Linguistics, 2002.

[20] Zhang, Tianyi, et al, “Bertscore: Evaluating text generation with bert,” arXiv, 2019

[21] Thoongsup Sareewan, et al, “Thai wordnet construction,” Proceedings of the 7th Workshop on Asian Language Resources, 2009.

ประวัติผู้เขียน

ชื่อ-นามสกุล	นายธวัชชัย รักษาชาติ
วัน เดือน ปีเกิด	4 เมษายน 2528 ที่นครศรีธรรมราช
ที่อยู่	แฟรตสวัสดิการกรมทางหลวง ถ. หลวงแพ่ง
	แขวงทับยาว เขตลาดกระบัง กรุงเทพฯ 10520 โทร.096-070-8103
ประวัติการศึกษา	2556 วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์
	มหาวิทยาลัยราชมงคลพระนคร วิทยาเขตประนครเหนือ
ความชำนาญเฉพาะด้าน	1.) ระบบงานการบริหารจัดการจราจร
	2.) ควบคุมงานโครงการ
	3.) พัฒนาเว็บไซต์ด้วย PHP
	4.) พัฒนาการเรียนรู้เชิงลึกด้าน NLP ด้วย Python
ประสบการณ์การทำงานและผลงานวิจัย	
พ.ศ.2556-2557	ตำแหน่งช่างเทคนิคติดตั้งระบบเรียกพยาบาลในโรงพยาบาล
	บริษัท อินเซนอินเทลลิเจนซ์ จำกัด
พ.ศ.2557-2560	ตำแหน่งที่ปรึกษาพนักงานบริหารจัดการจราจร
	บริษัท ฟาติมา อาร์.บี.ดี.เอส. อินเตอร์เนชั่นแนล จำกัด
พ.ศ.2557-ปัจจุบัน	ตำแหน่งวิศวกรคอมพิวเตอร์ กองวิจัยและนวัตกรรม การทางพิเศษแห่งประเทศไทย
	- ผลงานตีพิมพ์เรื่อง การพัฒนาการแสดงตำแหน่งของโทรศัพท์มือถือสำหรับ Mobile Application บนทางพิเศษ ในการประชุมวิชาการ งานวิจัยและพัฒนาเชิงประยุกต์ครั้งที่ 10 (10th ECTI-CARD 2018, PhitsanulokThailand)
	- ผลงานตีพิมพ์เรื่อง การพฒันาระบบระบุตำแหน่งอุปกรณ์ของทางพิเศษโดยใช้แอพพลิเคชันบนโทรศัพท์มือถือ ในการประชุมวิชาการ งานวิจัยและพัฒนาเชิงประยุกต์ครั้งที่ 11 (11th ECTI-CARD 2019, Ubon Ratchathani Thailand)
	- ผลงานตีพิมพ์เรื่อง การพัฒนาอัลกอริทึมสำหรับสรุปจุดติดขัดบนโครงข่ายทางพิเศษในการประชุมวิชาการ งานวิจัยและพัฒนาเชิงประยุกต์ครั้งที่ 12 (12th ECTI-CARD 2020, Nakhon sawan Thailand)