การจำแนกประเภทข้อความในภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร

นายธนภัทร์ คุ้มสุภา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2559

Text Categorization for Thai Corpus using Character-Level Convolutional Neural Network

Mr. Thanabhat Koomsubha

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2016

หัวข้อวิทยานิพนธ์	การจำแนกประเภทข้อความในภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร
โดย	นายธนภัทร์ คุ้มสุภา
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ดร. พีรพล เวทีกูล

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร. สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(ศาสตราจารย์ ดร. บุญเสริม กิจศิริกุล)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(ดร. พีรพล เวทีกูล)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. โชติรัตน์ รัตนามหัทธนะ)

.....กรรมการภายนอกมหาวิทยาลัย
(รองศาสตราจารย์ ดร. กฤษณะ ไวยมัย)

ธนภัทร์ คุ้มสุภา : การจำแนกประเภทข้อความในภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร (Text Categorization for Thai Corpus using Character-Level Convolutional Neural Network) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ดร. พีรพล เวทีกูล, 62 หน้า.

นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรเป็นวิธีการจำแนกประเภทข้อความที่มีประสิทธิภาพ วิธีการนี้ใช้การเรียนรู้ข้อความจากระดับตัวอักษร เมื่อนำมาใช้กับการจำแนกประเภทข้อความในภาษาไทยแล้ว จะทำให้สามารถจำแนกข้อความได้โดยไม่ต้องใช้ขั้นตอนการตัดคำ ทั้งนี้ นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรดั้งเดิมนั้นมีการจำกัดข้อความอยู่ที่ 1,014 ตัวอักษร ตัวอักษรส่วนเกินในข้อความตั้งต้นจะถูกตัดออกและไม่ถูกนำไปใช้ ในวิทยานิพนธ์ฉบับนี้จึงได้ทำการปรับปรุงโครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรให้สามารถรองรับข้อมูลความยาวใด ๆ โดยที่ยังคงใช้จำนวนพารามิเตอร์อื่น ๆ คงเดิม

ผลการทดลองกับข้อมูลข่าวภาษาไทยแสดงให้เห็นว่า วิธีการที่เสนอไปนั้นสามารถเพิ่มความแม่นยำให้กับการจำแนกได้เมื่อเปรียบเทียบกับวิธีการดั้งเดิม นอกจากนี้ นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่เสนอนั้นยังให้ความแม่นยำในการจำแนกที่สูงกว่าวิธีการที่ได้รับความนิยมอื่น ๆ เช่น นาอีฟเบย์ แมกซิมัมเอนโทรปี และซัพพอร์ตเวกเตอร์แมชชีน โดยมีเพียงวิธีนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำเท่านั้น ที่ให้ความแม่นยำมากกว่าประมาณ 0.5% ทั้งนี้ การใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรมีข้อดีคือประสิทธิภาพของของการจำแนกจะไม่ขึ้นอยู่กับประสิทธิภาพของการตัดคำ

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2559

5770925021 : MAJOR COMPUTER SCIENCE
KEYWORDS: CHARACTER-LEVEL CONVOLUTIONAL NEURAL NETWORK / DYNAMIC INPUT LENGTH / THAI TEXT CATEGORIZATION

THANABHAT KOOMSUBHA: Text Categorization for Thai Corpus using Character-Level Convolutional Neural Network. ADVISOR: DR. PEERAPON VATEEKUL, 62 pp.

A Character-level Convolutional Neural Network (Char-CNN) is an efficient method for text categorization. This method uses an input from characters, therefore, when applying it to categorize Thai text, a word segmentation step is not required. However, an original model of Char-CNN limits an input length to 1,014 characters. Any exceeding character is ignored. This thesis presents an improvement of Char-CNN which can accept any input length while it still uses the same number of parameters.

Experiments show that our proposed model can produce a better accuracy than an original model. Moreover, the proposed technique outperforms many classical techniques e.g. Naïve Bayes, Maximum Entropy and Support Vector Machine. Note that there is only one technique, a word-level Convolutional Neural Network, that it performs better than our model about 0.5%. However, a Char-CNN has an advantage because its accuracy does not depend on a performance of word segmentation.

Department:	Computer Engineering	Student's Signature	.....
Field of Study:	Computer Science	Advisor's Signature	.....
Academic Year:	2016		

กิตติกรรมประกาศ

การที่วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดีนั้น นอกจากการทำงานของตัวผู้วิจัยแล้ว ยังมีบุคคลท่านอื่นที่เป็นส่วนสำคัญที่ให้ความช่วยเหลือในการทำวิทยานิพนธ์ฉบับนี้ขึ้นมา ผู้จัดทำต้องขอขอบคุณบุคคลเหล่านี้ผู้ซึ่งทำให้เกิดผลสำเร็จนี้ขึ้นมาได้

ขอขอบคุณอาจารย์ที่ปรึกษา ดร. พีรพล เวทีกูล ผู้ที่คอยให้ความช่วยเหลือ คำแนะนำ และการกระตุ้นอยู่เสมอจนทำให้ผลงานฉบับนี้เกิดขึ้นมาได้

ขอขอบคุณ ศ. ดร. บุญเสริม กิจศิริกุล ผู้ให้คำแนะมากมาย ตั้งแต่การเริ่มทำโครงร่างวิทยานิพนธ์ รวมถึงการเป็นประธานในการสอบวิทยานิพนธ์

ขอขอบคุณกรรมการการสอบวิทยานิพนธ์ ผศ. ดร. โชติรัตน์ รัตนามหัทธนะ และ รศ. ดร. กฤษณะ ไวยมัย ที่ให้คำแนะนำและเสนอสิ่งที่ควรทำเพิ่มเติมในการทำงาน

ขอขอบคุณอาจารย์ทุกท่าน ที่ได้สั่งสอนเรื่องต่าง ๆ ในหลักสูตร ซึ่งแนวคิดและกระบวนการเหล่านั้น ล้วนประกอบกันจนทำให้ผลงานชิ้นนี้ลุล่วงไปได้

ขอขอบคุณเพื่อน ๆ พี่ ๆ น้อง ๆ ในห้องปฏิบัติการณ์ที่คอยช่วยเหลือสิ่งต่าง ๆ ทั้งคอยให้การสนับสนุน เป็นกำลังใจ จนวิทยานิพนธ์ฉบับนี้สำเร็จ และขอให้ผลเหล่านี้ย้อนกลับไปถึงตัวพวกท่านเอง

สุดท้าย ขอขอบคุณคุณพ่อ คุณแม่ และครอบครัว ที่ให้การสนับสนุน และส่งเสริมทั้งทางด้านการศึกษา และทางด้านการใช้ชีวิต จวบจนปัจจุบัน

สารบัญ

หน้า

หน้า

หน้า

หน้า

สารบัญภาพ

หน้า

สารบัญตาราง

หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

การจำแนกข้อความมีความสำคัญอย่างยิ่ง โดยเฉพาะในยุคของอินเทอร์เน็ต ที่ทุกคนสามารถเขียนข้อความต่าง ๆ ขึ้นมาและเผยแพร่ออกไปได้โดยง่าย ไม่ว่าจะเป็นการเขียนรีวิวสินค้า การเขียนบทความ การเขียนข่าว หรือการเขียนลงในเครือข่ายสังคมออนไลน์ ข้อความที่เกิดจากงานเขียนเหล่านี้เพิ่มจำนวนขึ้นมาอย่างรวดเร็วมากในแต่ละวัน การจำแนกข้อความจึงถือว่าเป็นสิ่งสำคัญที่จะทำให้การจัดเก็บข้อความเหล่านี้มีความเป็นระเบียบมากยิ่งขึ้น เพื่อความสะดวกรวดเร็วในการค้นหาและเข้าถึง

สำหรับงานวิจัยด้านการจำแนกข้อความในภาษาไทยนั้น วิธีมาตรฐานที่นำมาใช้โดยส่วนใหญ่มีพื้นฐานมาจากแบบจำลองแบบถุงคำ (bag-of-words) โดยมีการนำมาใช้งานในการหาอารมณ์หรือการวิเคราะห์ความคิดเห็นของข้อความภาษาไทย [1-3] และการจำแนกประเภทข้อความ [4, 5] ซึ่งวิธีการเหล่านี้เป็นการนำคำทั้งหมดในชุดข้อมูลมาสร้างเป็นพจนานุกรม แล้วจึงทำการหาเวกเตอร์แทนข้อความที่แสดงถึงการปรากฏของคำต่าง ๆ ในแต่ละเอกสาร จากนั้นจึงใช้ตัวจำแนกแบบต่าง ๆ เช่น ซัพพอร์ตเวกเตอร์แมชชีน (SVM) นาอีฟเบย์ (naïve bayes) หรือต้นไม้ตัดสินใจ (decision tree) มาทำการจำแนกข้อความ ทั้งนี้ การการแทนที่ข้อความด้วยถุงคำนั้นได้ละทิ้งลำดับของคำไป ซึ่งสามารถทำให้สูญเสียความหมายของข้อความได้ เช่น ในงานวิเคราะห์อารมณ์ของข้อความ คำว่า “โกงความตาย” ให้อารมณ์ในแง่บวก แต่หากไม่มีการใช้ลำดับของคำที่ต่อเนื่องกัน และแยกออกเป็นคำว่า “โกง” และคำว่า “ความตาย” จะทำให้การวิเคราะห์เกิดความผิดพลาดกลายเป็นอารมณ์ในแง่ลบแทนได้

ส่วนวิธีการอื่นที่แพร่หลายน้อยกว่าในงานวิจัยทางด้านการจำแนกข้อความภาษาไทย ได้แก่ การหาคำที่แสดงถึงขั้วของอารมณ์ในการทำเหมืองความคิดเห็น [6] การใช้คอนเซฟชวลกราฟในการจำแนกประเภทของประโยค [7] และการใช้รูปแบบของประโยคเพื่อชี้วัดอารมณ์ [8] ซึ่งวิธีการทางด้านภาษาข้างต้นนั้นจำเป็นต้องมีขั้นตอนในการตัดคำ ซึ่งถือได้ว่าเป็นอีกหนึ่งขั้นตอนที่ต้องมีการใช้เวลาในการดำเนินการอย่างรอบคอบและเหมาะสม โดยหากทำได้ไม่แม่นยำเพียงพอก็จะเป็นสาเหตุหนึ่งที่ทำให้การจำแนกข้อความผิดพลาดได้ เช่นคำว่า “กระจกตา” หมายถึงส่วนหนึ่งของอวัยวะในร่างกาย แต่หากตัดคำออกมาเหลือเพียง “กระจก” และ “ตา” อาจจะตีความเป็นสิ่งของชนิดหนึ่งและคำใช้เรียกแทนพ่อของแม่

นิวรอลเน็ตเวิร์กคอนโวลูชัน (convolutional neural network) เป็นนิวรอลเน็ตเวิร์กเชิงลึกที่มีจุดเด่นในชั้นคอนโวลูชันซึ่งเป็นการสกัดความรู้ออกมาจากฟีเจอร์ที่อยู่ใกล้เคียงกัน ในเริ่มแรกนั้น

วิธีการนี้ได้ถูกนำมาใช้ในงานทางด้านการเรียนรู้ลายมือภาษาอังกฤษและตัวเลข [9, 10] และจากความสามารถในการสกัดความรู้จากลำดับของฟีเจอร์ที่อยู่ใกล้เคียงกัน ทำให้วิธีการนี้ได้ถูกนำไปใช้ในงานทางด้านภาษาในการหาชนิดของคำ (parts of speech) การระบุคำที่เป็นชื่อเฉพาะ (named entity tags) และการหาคำที่มีความหมายใกล้เคียงกัน [11] หลังจากนั้น นิวรอลเน็ตเวิร์กคอนโวลูชันได้ถูกใช้มากยิ่งขึ้นในการจำแนกประเภทและการจำแนกอารมณ์ของข้อความ [12-14] โดยผลลัพธ์ที่ได้นั้นให้ผลที่ดีกว่าวิธีที่ใช้ถุงคำ รวมทั้งยังให้ผลลัพธ์ที่ดีกว่าวิธีนิวรอลเน็ตเวิร์กแบบอื่น ๆ ทั้งนี้นอกจากการใช้นิวรอลเน็ตเวิร์กคอนโวลูชันที่ใช้ข้อมูลรับเข้าในระดับคำแล้ว ยังมีงานวิจัยที่ใช้นิวรอลเน็ตเวิร์กคอนโวลูชันโดยใช้ข้อมูลรับเข้าเป็นลำดับของตัวอักษรที่มีความแม่นยำมากกว่านิวรอลเน็ตเวิร์กคอนโวลูชันแบบอื่น ๆ อีกด้วย [15]

งานวิจัยชิ้นนี้ เป็นการนำนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร มาประยุกต์ใช้ในการจำแนกประเภทของข้อความภาษาไทย โดยมีจุดมุ่งหมายเพื่อลดขั้นตอนในการจำแนกข้อความโดยการละทิ้งขั้นตอนการตัดคำ ในขณะที่ยังคงความแม่นยำในการจำแนกข้อความเอาไว้

1.2 วัตถุประสงค์

เพื่อพัฒนาวิธีการจำแนกข้อความในภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรโดยไม่จำเป็นต้องมีขั้นตอนการตัดคำ ให้มีความแม่นไม่ด้อยกว่าวิธีการจำแนกข้อความโดยใช้ซัพพอร์ตเวกเตอร์แมชชีน นาอีฟเบย์ แมกซิมัมเอนโทรปี และนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำ

และเพื่อทำการสร้างคลังของคำฝังตัวในภาษาไทยจากข้อมูลที่เก็บมาได้ ซึ่งสามารถนำไปใช้งานเกี่ยวกับข้อความภาษาไทยต่อไปได้ในอนาคต

1.3 ขอบเขตการดำเนินงาน

ข้อมูลที่นำมาใช้จะเป็นข้อความจากข่าวภาษาไทย โดยแต่ละข่าวจะถูกจัดให้อยู่ในประเภทใดประเภทหนึ่งเท่านั้น และแต่ละประเภทไม่มีความสัมพันธ์ใด ๆ ต่อกัน ทั้งนี้ จำนวนข้อความที่จะนำมาทดสอบจะมีไม่ต่ำกว่า 100,000 ข้อความ

1.4 ประโยชน์ที่คาดว่าจะได้รับ

ได้วิธีการจำแนกข้อความภาษาไทยด้วยการใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่ไม่ต้องมีการตัดคำและมีความแม่นไม่ด้อยกว่าวิธีการซัพพอร์ตเวกเตอร์แมชชีน นาอีฟเบย์ แมกซิมัมเอนโทรปี และนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำ และได้คลังข้อมูลของคำฝังตัวในภาษาไทยที่จะสามารถนำไปใช้งานต่อไปได้

1.5 วิธีดำเนินการวิจัย

วิธีการดำเนินการวิจัย สามารถแบ่งออกได้เป็นขั้นตอนดังนี้

1. ศึกษางานวิจัยที่เกี่ยวข้อง
2. เก็บรวบรวมข้อมูลจากทวิตเตอร์ภาษาไทยเพื่อจะนำมาใช้ในการทดลองเบื้องต้นโดยการจำแนกอารมณ์
3. ทดลองจำแนกอารมณ์จากทวิตเตอร์ภาษาไทยโดยวิธีมาตรฐาน และนิวรอลเน็ตเวิร์กแบบต่าง ๆ ในระดับคำ
4. ตีพิมพ์ผลงานทางวิชาการ
5. เก็บรวบรวมข้อมูลภาษาไทยจากข่าวที่เป็นหมวดหมู่เพื่อที่จะนำมาใช้ในงานวิจัย
6. ทำการทดลองเบื้องต้นในการจำแนกข้อความข่าวภาษาไทยโดยใช้วิธีมาตรฐานที่ใช้วิธีการตัดคำแบบต่าง ๆ และนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร
7. วิเคราะห์ผลลัพธ์ และทำการปรับปรุงนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรให้ดีมากยิ่งขึ้น
8. ตีพิมพ์ผลงานทางวิชาการ
9. สรุปผลการทดลองและจัดทำวิทยานิพนธ์
10. สอบวิทยานิพนธ์

1.6 ผลงานตีพิมพ์จากงานวิจัย

ส่วนหนึ่งของการศึกษาเบื้องต้นของงานวิจัยชิ้นนี้ ได้รับการตีพิมพ์ดังรายละเอียดต่อไปนี้

- A Study of Sentiment Analysis using Deep Learning Techniques on Thai Twitter Data โดย พีรพล เวทีกูล และธนภัทร์ คุ้มสุภา ในงานประชุมวิชาการ “The 13th International Joint Conference on Computer Science and Software Engineering (JCSSE 2016)” ซึ่งจัดขึ้น ณ จังหวัดขอนแก่น ประเทศไทย ระหว่างวันที่ 13 ถึง 15 กรกฎาคม 2559 แสดงในภาคผนวก ก

- A Character-level Convolutional Neural Network with Dynamic Input Length for Thai Text Categorization โดยธนภัทร์ คุ้มสุภา และ พีรพล เวทีกูล ในงานประชุมวิชาการ “The 9th International Conference on Knowledge and Smart Technology (KST 2017)” ซึ่งจัดขึ้น ณ พัทยา ประเทศไทย ระหว่างวันที่ 1 ถึง 4 กุมภาพันธ์ 2560 แสดงในภาคผนวก ข

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 การแทนข้อความ (Text Representation)

ในการจำแนกประเภทของข้อความนั้น สิ่งหนึ่งที่ต้องทำคือการแทนที่ข้อความด้วยฟีเจอร์เพื่อจะนำไปสู่กระบวนการจำแนกต่อไป วิธีการในการแทนข้อความนั้นมีดังต่อไปนี้ ทั้งนี้ ในแต่ละหัวข้อจะแสดงถึงการแทนข้อความ 2 ข้อความ ได้แก่ 1) “ฉันไปโรงเรียน โรงเรียนฉันสวย” และ 2) “โรงเรียนของฉันน่าอยู่”

2.1.1 ถุงคำ (Bag-of-words หรือ BoW)

เป็นการแสดงข้อความในรูปของเวกเตอร์ขนาดเท่ากับจำนวนคำทั้งหมดในพจนานุกรมของชุดข้อมูลนั้น ๆ โดยไม่มีการใช้ไวยากรณ์และลำดับของคำ จากตัวอย่างข้อความ สามารถสร้างเป็นพจนานุกรมของคำได้ดังนี้ [“ฉัน”,“ไป”,“โรงเรียน”,“สวย”,“ของ”,“น่าอยู่”] และในการแทนข้อความด้วยความถี่ของคำที่ปรากฏ สามารถทำได้ดังนี้

1) “ฉันไปโรงเรียน โรงเรียนฉันสวย” แทนด้วย [2 1 2 1 0 0]
2) “โรงเรียนของฉันน่าอยู่” แทนด้วย [1 0 1 0 1 1]

ทั้งนี้ การแทนค่าในแต่ละตำแหน่ง นอกจากการใช้ความถี่ของคำแล้ว ยังสามารถใช้ค่าทางสถิติอื่น ๆ ในการแทนค่าได้ เช่น การแทนที่แบบฐานสอง กล่าวคือ ในแต่ละช่องของเวกเตอร์ จะมีค่าเพียง 1 หรือ 0 แสดงถึงการมีอยู่ของคำนั้นในข้อความ

2.1.2 ทีเอฟไอดีเอฟ (Term Frequency-Inverse Document Frequency หรือ TF-IDF)

การแทนข้อความด้วยทีเอฟไอดีเอฟ เป็นวิธีการใช้ถุงคำอย่างหนึ่ง แต่มีการแทนค่าในแต่ละช่องของเวกเตอร์ด้วยความถี่ของคำในข้อความคูณด้วยค่าผกผันของความถี่ของคำนั้น ๆ จากทั้งชุดข้อมูล กำหนดให้ tf คือ ความถี่ของคำ N คือจำนวนของข้อความทั้งหมดในชุดข้อมูล และ n_t คือจำนวนของข้อความในชุดข้อมูลที่มีคำนั้น ๆ สามารถคำนวณ $tfidf$ ได้ดังสมการที่ (1) โดยที่คำนวณ idf ได้ดังสมการที่ (2)

$$tfidf = tf \times idf \quad (1)$$

$$idf = log(\frac{N}{n_t}) \quad (2)$$

จากข้อความตัวอย่าง จะคำนวณค่าของ idf ของแต่ละคำได้เป็น [0 0.3 0 0.3 0.3 0.3] และสามารถแทนข้อความได้ดังนี้

1) “ฉันไปโรงเรียน โรงเรียนฉันสวย” แทนด้วยเวกเตอร์ [0 0.3 0 0.3 0 0]

2) “โรงเรียนของฉันน่าอยู่” แทนด้วยเวกเตอร์ [0 0 0 0 0.3 0.3]

2.1.3 เวกเตอร์วันฮอท (One-hot Vector)

เป็นการแทนที่ข้อความด้วยกลุ่มของเวกเตอร์ที่เรียงลำดับกันจำนวนเท่ากับความยาวของข้อความ โดยที่แต่ละเวกเตอร์จะมีขนาดเท่ากับจำนวนคำทั้งหมดที่ปรากฏในชุดข้อมูล และจะแสดงถึงคำที่ปรากฏในลำดับนั้น ๆ โดยที่ค่าภายในเวกเตอร์ จะมีค่าที่เป็น 1 เพียงช่องเดียว ส่วนช่องอื่น ๆ ในเวกเตอร์จะมีค่าเป็น 0 ทั้งนี้ จากตัวอย่าง จะได้ลำดับของคำที่มีทั้งหมดคือ “ฉัน”, “ไป”, “โรงเรียน”, “สวย”, “ของ”, “น่าอยู่” และจะสามารถแสดงการแทนคำด้วยเวกเตอร์วันฮอทได้ เช่น “ฉัน” แทนด้วย $\begin{bmatrix}1\\0\\0\\0\\0\\0\end{bmatrix}$, “ไป” แทนด้วย $\begin{bmatrix}0\\1\\0\\0\\0\\0\end{bmatrix}$ เป็นต้น และในการแสดงข้อความ จะนำเวกเตอร์วันฮอทของแต่ละคำมารวมกันตามลำดับได้ดังนี้

1) “ฉันไปโรงเรียน โรงเรียนฉันสวย” แทนด้วย $\begin{bmatrix}1&0&0&0&1&0\\0&1&0&0&0&0\\0&0&1&1&0&0\\0&0&0&0&0&1\\0&0&0&0&0&0\\0&0&0&0&0&0\end{bmatrix}$

2) “โรงเรียนของฉันน่าอยู่” แทนด้วย $\begin{bmatrix}0&0&1&0\\0&0&0&0\\1&0&0&0\\0&0&0&0\\0&1&0&0\\0&0&0&1\end{bmatrix}$

2.1.4 คำฝังตัว (Word Embedding)

คำฝังตัวเป็นการแทนที่ข้อความด้วยเวกเตอร์ขนาดต่ำ ๆ เช่นการใช้เวกเตอร์คำ (word vector) โดยในการแทนที่ข้อความจะใช้จำนวนของเวกเตอร์เท่ากับความยาวของข้อความ และขนาดของเวกเตอร์จะสามารถกำหนดเองได้ ทั้งนี้ การสร้างเวกเตอร์คำนิยมสร้างโดยการวิเคราะห์ข้อความจากชุดข้อมูลทั้งหมดก่อน แล้วจึงสร้างเวกเตอร์คำโดยให้คู่ของคำที่มีความหมายใกล้เคียงกัน มีระยะห่างของเวกเตอร์คำใกล้เคียงกันด้วย วิธีการสร้างเวกเตอร์คำที่นิยมได้แก่ เวิร์ดทูเวก (word2vec) [16] และโกลฟ (GloVe) [17] กำหนดให้ใช้เวกเตอร์ขนาด 3 ในการแทนที่แต่ละคำเป็น “ฉัน” แทนด้วย $\begin{bmatrix}0.23\\0.31\\0.85\end{bmatrix}$, “ไป” แทนด้วย $\begin{bmatrix}0.04\\0.36\\0.78\end{bmatrix}$, “โรงเรียน” แทนด้วย $\begin{bmatrix}0.11\\0.56\\0.86\end{bmatrix}$, “สวย” แทนด้วย

$\begin{bmatrix}0.28\\0.83\\0.98\end{bmatrix}$, “ของ” แทนด้วย $\begin{bmatrix}0.66\\0.78\\0.79\end{bmatrix}$, “น่าอยู่” แทนด้วย $\begin{bmatrix}0.41\\0.27\\0.81\end{bmatrix}$ เมื่อนำเวกเตอร์ของคำมารวมกัน จะแทนที่ข้อความตัวอย่างได้ดังนี้

1) “ฉันไปโรงเรียน โรงเรียนฉันสวย” แทนด้วย

$$\begin{bmatrix}0.23 & 0.04 & 0.11 & 0.11 & 0.23 & 0.28\\0.31 & 0.36 & 0.56 & 0.56 & 0.31 & 0.83\\0.85 & 0.78 & 0.86 & 0.86 & 0.85 & 0.98\end{bmatrix}$$

2) “โรงเรียนของฉันน่าอยู่” แทนด้วย

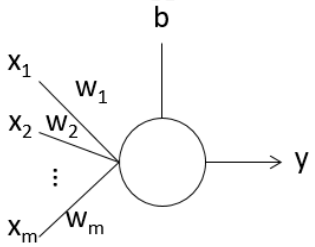
$$\begin{bmatrix}0.11 & 0.66 & 0.23 & 0.41\\0.56 & 0.78 & 0.31 & 0.27\\0.86 & 0.79 & 0.85 & 0.81\end{bmatrix}$$

2.2 นิวรอลเน็ตเวิร์ก (Neural Network)

นิวรอลเน็ตเวิร์กเป็นแบบจำลองที่ได้รับแรงบันดาลใจมาจากสมองของมนุษย์ โดยสามารถเปรียบเทียบการทำหน้าที่ได้คือการเรียนรู้จากข้อมูลที่มีอยู่แล้ว เพื่อใช้ทำนายข้อมูลในลักษณะเดียวกัน โดยในหัวข้อนี้จะมีหัวข้อย่อยเรียงตามลำดับดังต่อไปนี้ หน่วยพื้นฐานของนิวรอลเน็ตเวิกร์ที่เรียกว่าเพอร์เซ็ปตรอน นิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้า และนิวรอลเน็ตเวิร์กเชิงลึก

2.2.1 เพอร์เซ็ปตรอน (Perceptron)

เพอร์เซ็ปตรอนคือส่วนประกอบที่เล็กที่สุดของนิวรอลเน็ตเวิร์ก เปรียบได้กับเซลล์ประสาทหนึ่งเซลล์ที่เรียกว่านิวรอล ลักษณะของเพอร์เซ็ปตรอนแสดงได้ดังรูปที่ 2.1



รูปที่ 2.1 โครงสร้างของเพอร์เซ็ปตรอน ข้อมูลรับเข้าและส่งออก

ทั้งนี้ เพอร์เซ็ปตรอนเป็นขั้นตอนวิธีที่ใช้ในจำแนกผลลัพธ์เป็นสองกลุ่ม กำหนดให้ฟังก์ชันของเพอร์เซ็ปตรอนแทนด้วย $f(x)$ โดยมีข้อมูลรับเข้าคือ x และข้อมูลส่งออกคือ $\hat{y}$ โดยแสดงการคำนวณข้อมูลส่งออกได้ดังสมการที่ (3)

$$\hat{y} = f(x) = \begin{cases}1 & if\ \sum_{i=1}^{m} w_i x_i + b > 0\\0 & otherwise\end{cases} \quad (3)$$

โดยที่ w คือ เวกเตอร์ของน้ำหนัก (weights) b คือค่าไบแอส (bias) และ m คือขนาดของข้อมูลรับเข้า สำหรับกระบวนการเรียนรู้ของเพอร์เซ็ปตรอน กำหนดให้ชุดข้อมูลตัวอย่างแทนด้วย x และผลลัพธ์จริงของตัวอย่างนั้น แทนด้วย y สมการในการเรียนรู้แสดงได้โดย (4) และ (5)

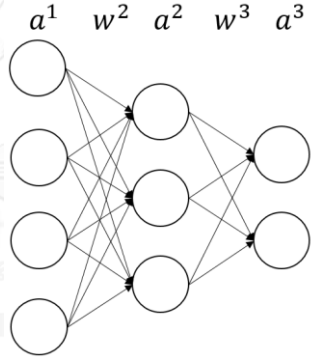
$$w_i \leftarrow w_i + \Delta w_i \tag{4}$$

โดยที่ $$\Delta w_i = \alpha(\hat{y} - y)x_i \tag{5}$$

ทั้งนี้ α คืออัตราการเรียนรู้ (learning rate) เป็นค่าที่บ่งบอกว่าในแต่ละรอบของการเรียนรู้จะมีการเปลี่ยนแปลงน้ำหนักเทียบกับอัตราส่วนของผลต่างของผลลัพธ์ไปมากเท่าใด

2.2.2 นิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้า (Feedforward Neural Network)

นิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้ามีลำดับในการคำนวณและส่งผ่านของข้อมูลไปในทิศทางเดียว โดยโครงสร้างจะแบ่งออกเป็นลำดับชั้น ในแต่ละลำดับชั้น จะมีเพอร์เซ็ปตรอนจำนวนหนึ่งซึ่งไม่มีเส้นเชื่อมถึงกันภายในชั้นเดียวกัน แต่จะมีเส้นเชื่อมกับเพอร์เซ็ปตรอนตัวอื่นที่อยู่ในลำดับชั้นที่ติดกันทั้งหมด โดยข้อมูลส่งออกของเพอร์เซ็ปตรอนในชั้นก่อนหน้า จะเป็นข้อมูลรับเข้าของเพอร์เซ็ปตรอนในชั้นปัจจุบัน โครงสร้างของนิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้าแสดงได้ดังรูปที่ 2.2



รูปที่ 2.2 โครงสร้างของนิวรอลเน็ตเวิร์กแบบป้อนไปข้างหน้า

กำหนดสัญลักษณ์แทนการคำนวณไปข้างหน้า (feedforward) โดยให้ a_k^{l-1} แทนผลลัพธ์ของเพอร์เซ็ปตรอนตัวที่ k ในลำดับชั้น $l-1$ และ w_{jk}^l แทนน้ำหนักสำหรับเพอร์เซ็ปตรอนตัวที่ j ในลำดับชั้น l ที่มีเส้นเชื่อมมาจากเพอร์เซ็ปตรอนตัวที่ k ในลำดับชั้นก่อนหน้า และ b_j^l คือไบแอส นอกจากนี้ ให้ g แทนฟังก์ชันกระตุ้น และให้ n แทนจำนวนเพอร์เซ็ปตรอนในลำดับชั้นที่ $l-1$ จะสามารถแสดงการคำนวณ a_j^l ได้โดยสมการที่ (6) และ (7)

$$z_j^l = \sum_{k=1}^{n} w_{jk}^l a_k^{l-1} + b_j^l \tag{6}$$

$$a_j^l = g(z_j^l) \tag{7}$$

2.2.2.1 ฟังก์ชันกระตุ้น (Activation Function)

ในส่วนของข้อมูลส่งออกของแต่ละเพอร์เซ็ปตรอน จะมีการใช้ฟังก์ชันกระตุ้น $g(z)$ เพื่อทำให้นิวรอลเน็ตเวิร์กสามารถแก้ปัญหาได้หลากหลายมากขึ้น ฟังก์ชันกระตุ้นมีหลากหลายรูปแบบ โดยแบบที่นิยมกันมีดังต่อไปนี้

1) ฟังก์ชันซิกมอยด์ (Sigmoid Function)

เป็นฟังก์ชันที่ให้ค่าผลลัพธ์ออกมาอยู่ในช่วง 0 ถึง 1 สมการของฟังก์ชันซิกมอยด์ แทนด้วย σ แสดงได้โดย (8)

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{8}$$

2) ฟังก์ชันแทนเจนต์ไฮเพอร์โบลิก (Hyperbolic Tangent Function)

เป็นฟังก์ชันที่ให้ค่าผลลัพธ์ออกมาอยู่ในช่วง -1 ถึง 1 สมการของฟังก์ชันแทนเจนต์ไฮเพอร์โบลิก แทนด้วย $tanh$ คำนวณได้จากสมการที่ (9)

$$tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{9}$$

3) ฟังก์ชันค่าสูงสุดอย่างอ่อน (Softmax Function)

เป็นฟังก์ชันที่ให้ค่าผลลัพธ์ออกมาอยู่ในช่วง 0 ถึง 1 กำหนดให้ในชั้นของเน็ตเวิร์กที่ต้องการคำนวณมีผลลัพธ์ทั้งหมด K ตัว ค่าของผลลัพธ์นั้นแทนด้วยสัญลักษณ์ z จะได้ว่า ฟังก์ชันค่าสูงสุดอย่างอ่อนของผลลัพธ์ตัวที่ j หรือแทนด้วยสัญญัลักษณ์ f_j คำนวณได้จากสมการที่ (10)

$$f(z)_j = \frac{e^{z_j}}{\sum_{i=1}^{K} e^{z_i}} \tag{10}$$

4) ฟังก์ชันเรคติไฟต์เชิงเส้น (Rectified Linear Unit Function หรือ ReLU)

เป็นฟังก์ชันที่ให้ผลลัพธ์ออกมาเป็นจำนวนบวกหรือเป็นศูนย์เสมอ สมการของฟังก์ชันเรคติไฟต์เชิงเส้น f สามารถคำนวณได้จากสมการที่ (11)

$$f(z) = \begin{cases} 0 & if\ z < 0 \\ z & if\ z \geq 0 \end{cases} \tag{11}$$

5) ฟังก์ชันขีดแบ่ง (Threshold Function)

เป็นฟังก์ชันในรูปทั่วไปของฟังก์ชันเรคติไฟต์เชิง โดยจะมีค่าขีดแบ่ง t ตามที่กำหนด สมการของฟังก์ชันขีดแบ่ง f สามารถคำนวณได้จากสมการที่ (12)

$$f(z,t) = \begin{cases} 0 & if\ z < t \\ z & if\ z \geq t \end{cases} \tag{12}$$

2.2.2.2 ฟังก์ชันต้นทุน (Cost Function หรือ Lost Function หรือ Objective Function)

ฟังก์ชันต้นทุน เป็นฟังก์ชันของนิวรอลเน็ตเวิร์กที่แสดงถึงต้นทุนของเน็ตเวิร์ก กล่าวคือ ในการเรียนรู้ของเน็ตเวิร์กนั้น จะทำการปรับค่าน้ำหนักเพื่อที่จะลดค่าผลลัพธ์ของฟังก์ชันต้นทุนนี้ ฟังก์ชันต้นทุนที่เป็นที่นิยมเป็นไปตามรายละเอียดต่อไปนี้ ทั้งนี้ สัญลักษณ์ของสมการในแต่ละข้อจะใช้ J แทนฟังก์ชันต้นทุน n คือจำนวนข้อมูลทั้งหมดที่ใช้ในการเรียนรู้ y_i แทนผลลัพธ์จริงที่ต้องการของข้อมูลชุดที่ i และ $\hat{y}_i$ แทนผลลัพธ์ที่ทำนายได้ของข้อมูลชุดที่ i

1) ค่าเฉลี่ยความผิดพลาดกำลังสอง (Mean Squared Error หรือ MSE) คำนวณได้จากสมการที่ (13)

$$J = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{13}$$

2) ค่าเฉลี่ยครอสเอนโทรปีแบบทวิภาค (Binary Cross-entropy) แสดงโดยสมการที่ (14)

$$J = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{14}$$

3) ค่าลบลอการิทึมของความเป็นไปได้ (Negative Log-Likelihood หรือ NLL) แสดงได้โดยสมการที่ (15)

$$J = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\hat{y}_i) \tag{15}$$

2.2.2.3 การหาค่าเหมาะที่สุด (Optimization)

การเรียนรู้ของนิวรอลเน็ตเวิร์กเป็นการเรียนรู้เพื่อที่จะลดค่าของฟังก์ชันต้นทุนให้น้อยที่สุด โดยใช้การปรับปรุงน้ำหนักของเส้นเชื่อมในเน็ตเวิร์ก สำหรับวิธีการที่ใช้ในการปรับปรุงน้ำหนักของเส้นเชื่อมที่ได้รับความนิยมมีดังต่อไปนี้

1) สโตแคสติกเกรเดียนเดสเซนท์ (Stochastic Gradient Descent หรือ SGD)

กำหนดให้ w แทนพารามิเตอร์ซึ่งเป็นน้ำหนักที่ต้องการจะปรับค่า α คืออัตราการเรียนรู้ $\frac{\partial J}{\partial w}$ คือเกรเดียนของฟังก์ชันต้นทุนเทียบกับ w การเรียนรู้โดยสโตแคสติกเกรเดียนเดสเซนท์จะมีการปรับค่าของ w ดังสมการที่ (16)

$$\mathrm{w}_t = \mathrm{w}_{t-1} - \alpha\frac{\partial J_t}{\partial w} \tag{16}$$

นอกจากนี้ยังมีการใช้โมเมนตัม (Momentum) โดยมีจุดประสงค์เพื่อทำให้การเรียนรู้มีการลู่เข้าที่ดีขึ้นจากการหลีกเลี่ยงการติดอยู่ที่โลคอลออพติมา (Local Optima) ให้ v แทนค่าความเร็วซึ่งมีการปรับค่าพร้อมกับ w และ γ แทนค่าสัมประสิทธ์ของโมเมนตัม (Momentum Coefficient) สามารถแสดงสมการในการเรียนรู้ได้จาก (17) และ (18)

$$v_t = \gamma v_{t-1} + \alpha \frac{\partial J_t}{\partial w} \quad (17)$$

$$\mathrm{w}_t = \mathrm{w}_{t-1} - v_t \quad (18)$$

2) วิธีเกรเดียนที่ปรับตัวได้ (Adaptive Gradient Method หรือ AdaGrad)

เป็นวิธีที่จะมีการปรับอัตราการเรียนรู้ได้ด้วยตนเองจากค่าเริ่มต้นที่กำหนด โดยในการปรับค่าของอัตราการเรียนรู้นั้นจะมีการใช้ค่าเกรเดียนในอดีตมาใช้ กำหนดให้ g_t แทนเกรเดียนที่เวลา t สมการของการเรียนรู้แสดงโดย (19) และ (20)

$$g_t = \frac{\partial J_t}{\partial w} \quad (19)$$

$$\mathrm{w}_t = \mathrm{w}_{t-1} - \frac{\alpha}{\sqrt{\sum_{k=1}^{t} {g_i}^2}} g_t \quad (20)$$

3) อาร์เอ็มเอสพรอพ (RMSProp)

เป็นวิธีที่มีการเก็บค่าเกรเดียนของครั้งก่อนหน้าไว้เพื่อที่จะนำมาใช้ในรอบของการเรียนรู้ปัจจุบันโดยการนำไปปรับปรุงอัตราส่วนของอัตราการเรียนรู้ โดยนอกเหนือจากการใช้ g_t แล้วยังมีการใช้ $MeanSquare_t$ สำหรับการเก็บค่าเฉลี่ยของเกรเดียน และให้ γ แทนอัตราการใช้เกรเดียนของอดีตในการเรียนรู้ซึ่งโดยปกติจะใช้ค่านี้ที่ 0.9 สามารถแสดงการคำนวณการเรียนรู้ด้วยวิธีอาร์เอ็มเอสพรอพได้โดยสมการที่ (21), (22) และ (23)

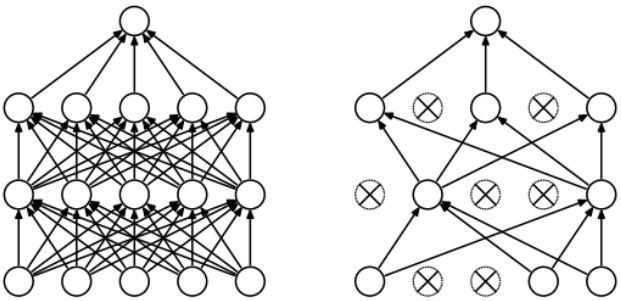
$$g_t = \frac{\partial J_t}{\partial w} \quad (21)$$

$$MeanSquare_t = \gamma MeanSquare_{t-1} + (1-\gamma){g_t}^2 \quad (22)$$

$$\mathrm{w}_t = \mathrm{w}_{t-1} - \frac{\alpha}{\sqrt{MeanSquare_t}} g_t \quad (23)$$

2.2.2.4 การดรอปเอาท์ (Dropout)

การดรอปเอาท์ [18] เป็นวิธีป้องกันการอิงกับข้อมูลเรียนรู้มากเกินไป (overfitting) ทำโดยการสุ่มตัดเส้นเชื่อมของเน็ตเวิร์กในระหว่างการเรียนรู้ ทั้งนี้จะทำการสุ่มใหม่ในทุก ๆ รอบของการเรียนรู้ของข้อมูลแต่ละรายการ ส่วนในระหว่างการทดสอบจะไม่ใช้การดรอปเอาท์ รูปที่ 2.3 แสดงการทำดรอปเอาท์ โดยรูปขวาแสดงเส้นเชื่อมที่เหลืออยู่



รูปที่ 2.3 รูปซ้ายแสดงนิวรอลเน็ตเวิร์กแบบปกติ รูปขวาแสดงการดรอปเอาท์
(อ้างอิงจาก Fig. 1 ใน [18])

2.2.3 การแพร่กระจายย้อนกลับและการเรียนรู้ (Backpropagation and Training)

เมื่อพิจารณาขั้นตอนของการป้อนไปข้างหน้า การหาค่าความผิดพลาดของเพอร์เซ็ปตรอนในลำดับชั้นสุดท้ายนั้นสามารถทำได้โดยง่ายจากการคำนวณเกรเดียนของฟังก์ชันต้นทุนเทียบกับค่าผลลัพธ์ในชั้นสุดท้าย แต่ในการหาค่าความผิดพลาดของเพอร์เซ็ปตรอนเพื่อใช้ในการเรียนรู้ของลำดับชั้นก่อนหน้านั้นไม่สามารถหาได้โดยตรง จึงต้องอาศัยวิธีการที่เรียกว่าการแพร่กระจายย้อนกลับ

กำหนดให้กำหนดให้ δ_j^l แทนค่าความผิดพลาดของเพอร์เซ็ปตรอนตัวที่ j ในลำดับชั้น l กำหนด j แทนฟังก์ชันต้นทุน กำหนด z เป็นค่าที่คำนวณได้ก่อนจะผ่านฟังก์ชันกระตุ้น g จะสามารถเขียนสมการของค่าความผิดพลาดได้ดังสมการที่ (24)

$$\delta_j^l = \frac{\partial J}{\partial z_j^l} = \frac{\partial J}{\partial a_j^l}\frac{\partial a_j^l}{\partial z_j^l} = \frac{\partial J}{\partial a_j^l} g'(z_j^l) \tag{24}$$

สำหรับการหาค่า $\frac{\partial J}{\partial a_j^l}$ นั้น ในลำดับชั้นสุดท้ายสามารถคำนวณหาได้โดยตรงจากฟังก์ชันต้นทุนที่เลือกใช้ ส่วนในลำดับชั้นก่อนหน้า จะต้องหาโดยวิธีการแพร่กระจายย้อนกลับ โดยจะทำคล้ายกับการป้อนไปข้างหน้า เพียงแต่กลับทิศกันเท่านั้น โดยคำนวณได้ดังสมการที่ (25)

$$\frac{\partial J}{\partial a_j^l} = \sum_{k=1}^{m} \frac{\partial J}{\partial z_k^{l+1}}\frac{\partial z_k^{l+1}}{\partial a_j^l} = \sum_{k=1}^{m} \delta_k^{l+1} w_{kj}^{l+1} \tag{25}$$

โดย m คือจำนวนเพอร์เซ็ปตรอนในลำดับชั้นที่ $l+1$ จากนั้น เมื่อคำนวณค่าความผิดพลาดของแต่ละระดับชั้นได้ ก็สามารถหาค่าผิดพลาดเทียบกับน้ำหนักและค่าไบแอสใด ๆ ได้จากสมการที่ (26) และ (27)

$$\frac{\partial J}{\partial w_{jk}^l} = \frac{\partial J}{\partial z_j^l}\frac{\partial z_j^l}{\partial w_{jk}^l} = \delta_j^l a_k^{l-1} \tag{26}$$

$$\frac{\partial J}{\partial b_j^l} = \frac{\partial J}{\partial z_j^l}\frac{\partial z_j^l}{\partial b_j^l} = \delta_j^l \tag{27}$$

ยกตัวอย่างเช่น ในกรณีที่ใช้สโตแคสติกเกรเดียนเดสเซนท์ การปรับปรุงค่าน้ำหนัก w_{jk}^{l} จะทำได้โดยสมการที่ (28)

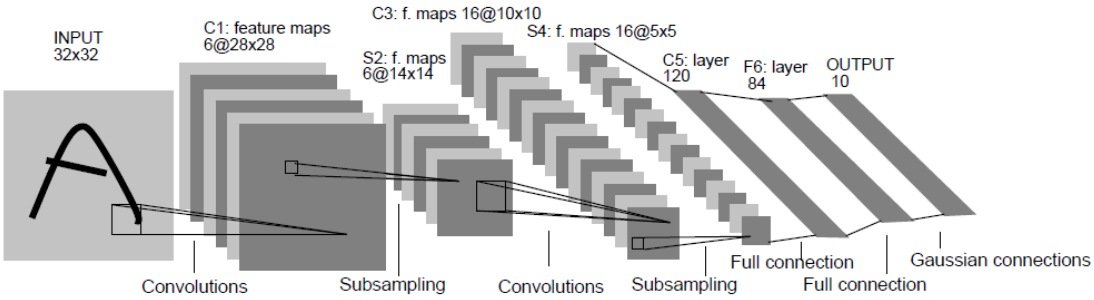
$$w_{jk,t}^{l} = w_{jk,t-1}^{l} - \alpha a_{k,t}^{l-1}\delta_{j,t}^{l} \tag{28}$$

2.2.4 นิวรอลเน็ตเวิร์กเชิงลึก (Deep Neural Network)

คือนิวรอลเน็ตเวิร์กที่มีจำนวนชั้นมาก ๆ มีอยู่หลากหลายรูปแบบ เช่น เน็ตเวิร์กความเชื่อเชิงลึก (Deep Belief Network หรือ DBN) เน็ตเวิร์กกองซ้อนของตัวเข้ารหัส (Stacked Auto-Encoders) นิวรอลเน็ตเวิร์กแบบวกกลับ (Recurrent Neural Network) หน่วยความจำระยะสั้นแบบยาว (Long-Short Term Memory หรือ LSTM) นิวรอลเน็ตเวิร์กคอนโวลูชัน (Convolutional Neural Network) โดยหัวข้อ 2.3 จะอธิบายถึงนิวรอลเน็ตเวิร์กคอนโวลูชัน

2.3 นิวรอลเน็ตเวิร์กคอนโวลูชัน (Convolutional Neural Network)

นิวรอลเน็ตเวิร์กคอนโวลูชันเป็นนิวรอลเน็ตเวิร์กเชิงลึกรูปแบบหนึ่ง มีจุดเริ่มต้นมาจากการงานวิจัยทางด้านการรู้จำภาพตัวอักษร โดยมักจะใช้ข้อมูลรับเข้าเป็นเมทริกซ์จากการแปลงมาจากรูปภาพ โครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันแสดงได้ดังรูปที่ 2.4 ซึ่งเน็ตเวิร์กทั้งหมดเกิดจากการนำชั้นหลาย ๆ ประเภทมาประกอบเข้าด้วยกัน ดังต่อไปนี้



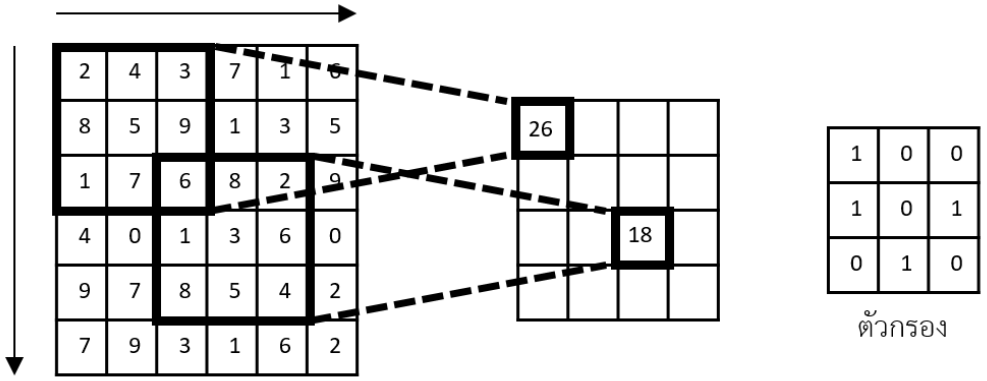
รูปที่ 2.4 นิวรอลเน็ตเวิร์กคอนโวลูชัน (อ้างอิงจาก Fig. 2 ใน [9])

2.3.1 ชั้นคอนโวลูชัน (Convolutional Layer)

เป็นชั้นที่ทำการหาฟีเจอร์จากกลุ่มของข้อมูลรับเข้าที่อยู่ใกล้ ๆ กัน โดยใช้วิธีการดอทเมทริกซ์กับตัวกรอง (filter) โดยนำหนักของตัวกรองนั้น จะเป็นน้ำหนักที่มีการใช้ร่วมกันในทุก ๆ การทำคอนโวลูชันของข้อมูลรับเข้า กำหนดให้ข้อมูลรับเข้าแทนด้วยเมทริกซ์ a^{l-1} ขนาด $N \times N$ และมีตัวกรองที่มีนำหนัก w ขนาด $m \times m$ ผลลัพธ์ a^{l} ของการทำคอนโวลูชันสามารถคำนวณได้ดังสมการที่ (29) และ (30)

$$z_{ij}^{l} = \sum_{a=0}^{m-1}\sum_{b=0}^{m-1} w_{a,b}^{l} a_{i+a,j+b}^{l-1} + b^{l} \tag{29}$$

$$a_{ij}^{l} = g(z_{ij}^{l}) \tag{30}$$



รูปที่ 2.5 ตัวอย่างการทำคอนโวลูชัน โดยมีขนาดของข้อมูลรับเข้าขนาด 6 x 6 และเมทริกตัวกรอง ขนาด 3 x 3

ในชั้นคอนโวลูชัน มีองค์ประกอบที่ต้องคำนึงถึงดังต่อไปนี้

2.3.1.1 ขนาดของตัวกรอง (Filter Size)

คือความกว้างและความสูงของตัวกรองที่จะนำมาใช้ในการทำคอนโวลูชัน

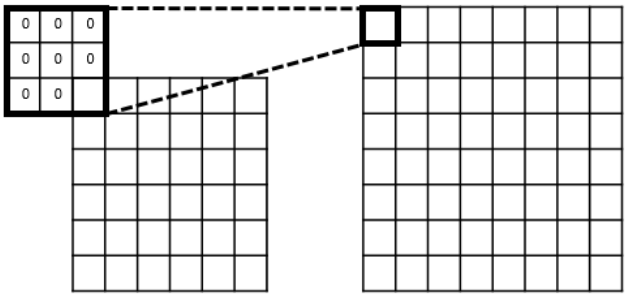
2.3.1.2 ชนิดของการทำคอนโวลูชัน (Convolution Type)

1) คอนโวลูชันแบบแคบ (Narrow Convolution)

การทำคอนโวลูชันโดยทั่วไป มักจะเป็นการทำคอนโวลูชันแบบแคบ กล่าวคือ ในการทำคอนโวลูชัน ตัวกรองที่นำไปทำการดอทเมทริกซ์นั้นจะไม่มีการกระทำเลยขอบของเมทริกซ์รับเข้า ส่งผลให้ผลลัพธ์ของการทำคอนโวลูชันที่มีข้อมูลรับเข้าขนาด $N \times N$ กับตัวกรองขนาด $m \times m$ จะได้เมทริกซ์ขนาด $(N - m + 1) \times (N - m + 1)$

2) คอนโวลูชันแบบกว้าง (Wide Convolution)

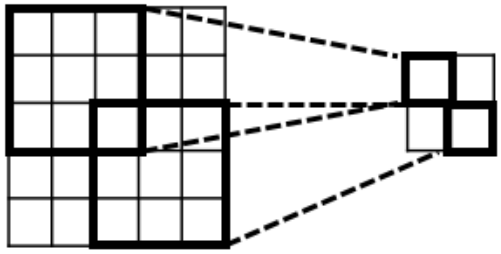
เป็นการทำคอนโวลูชันที่มีการกระทำเลยขอบของเมทริกซ์รับเข้าออกไป โดยพื้นที่ที่เกินออกไปนั้น จะมีการแทนค่าของข้อมูลช่องนั้น ๆ ด้วย 0 เรียกว่าการเสริมเติม (padding) ผลลัพธ์ของการทำคอนโวลูชันแบบกว้างที่มีข้อมูลรับเข้าขนาด $N \times N$ กับตัวกรองขนาด $m \times m$ จะได้เมทริกซ์ขนาด $(N + m - 1) \times (N + m - 1)$ ทั้งนี้การทำคอนโวลูชันแบบกว้างนี้มีขึ้นเพื่อป้องกันการสูญเสียข้อมูลตรงบริเวณขอบของข้อมูลรับเข้า รูปที่ 2.6 แสดงการทำคอนโวลูชันแบบกว้างและการเสริมเติม



รูปที่ 2.6 การทำคอนโวลูชันแบบกว้างและการเสริมเติม

2.3.1.3 ขนาดของการก้าวข้าม (Stride Size)

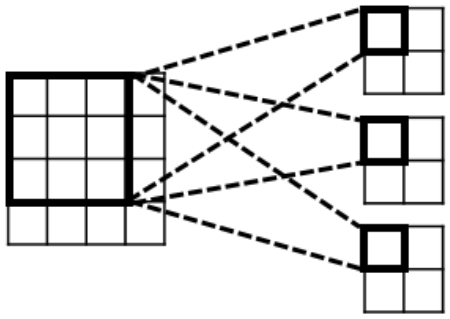
ขนาดของการก้าวข้ามคือจำนวนช่องของข้อมูลรับเข้า ที่จะทำการเลื่อนไปเมื่อทำการหาผลลัพธ์ของคอนโวลูชันในแต่ละช่อง โดยทั่วไปมักจะใช้ขนาดของการก้าวข้ามเป็น 1 รูปที่ 2.7 แสดงลักษณะของการทำคอนโวลูชันที่มีขนาดของการก้าวข้ามเป็น 2



รูปที่ 2.7 การทำคอนโวลูชันโดยมีข้อมูลรับเข้าขนาด 5 x 5 ตัวกรองขนาด 3 x 3 และมีขนาดของการก้าวข้ามเป็น 2

2.3.1.4 จำนวนตัวกรอง (Number of Filters)

ในการแต่ละชั้นคอนโวลูชันนั้น สามารถมีตัวกรองได้มากกว่าหนึ่ง โดยนำหนักของตัวกรองแต่ละตัวจะใช้แยกกัน โดยจำนวนตัวกรองในชั้นคอนโวลูชันใด ๆ จะเป็นการกำหนดจำนวนช่องสัญญาณ (Channel) ของของมูลรับเข้าในชั้นถัดไป รูปที่ 2.8 แสดงตัวอย่างการทำคอนโวลูชันโดยมีจำนวนตัวกรองเป็น 3



รูปที่ 2.8 การทำคอนโวลูชันโดยมีจำนวนตัวกรองเท่ากับ 3

2.3.1.5 จำนวนช่องสัญญาณ (Channel)

จำนวนช่องสัญญาณ หรือเรียกได้อีกอย่างหนึ่งว่าความลึกของข้อมูลรับเข้า อาจจะมีค่ามากกว่าหนึ่งได้ เช่นในงานวิจัยทางด้านการภาพ มีการใช้ช่องสัญญาณทั้งหมด 3 ช่องสัญญาณแทนค่าของแม่สี หรือสามารถเกิดจากจำนวนข้องตัวกรองในชั้นคอนโวลูชันก่อนหน้า กำหนดให้จำนวนช่องสัญญาณมีค่าเป็น k จะคำนวณผลลัพธ์ของชั้นคอนโวลูชันได้ดังสมการที่ (31) และ (32)

$$z_{ij}^{l} = \sum_{c=0}^{k-1}\sum_{a=0}^{m-1}\sum_{b=0}^{m-1} w_{a,b}^{l} a_{c,i+a,j+b}^{l-1} + b^{l} \tag{31}$$

$$a_{ij}^{l} = g(z_{ij}^{l}) \tag{32}$$

2.3.1.6 การแพร่กระจายย้อนกลับและการเรียนรู้ (Backpropagation and Training)

เป็นไปในลักษณะเดียวกับนิวรอลเน็ตเวิร์กโดยทั่วไป นั่นคือ ในการหาค่าความผิดพลาดเทียบกับค่า z_{ij}^{l} ในระดับชั้นใด ๆ จะทำได้จากสมการที่ (33)

$$\delta_{ij}^{l} = \frac{\partial J}{\partial z_{ij}^{l}} = \frac{\partial J}{\partial a_{ij}^{l}}\frac{\partial a_{ij}^{l}}{\partial z_{ij}^{l}} = \frac{\partial J}{\partial a_{ij}^{l}} g'(z_{ij}^{l}) \tag{33}$$

และในการหา $\frac{\partial J}{\partial a_{ij}^{l}}$ สามารถหาได้จากการแพร่กระจายย้อนกลับ แสดงโดยสมการที่ (34)

$$\frac{\partial J}{\partial a_{ij}^{l}} = \sum_{a=0}^{m-1}\sum_{b=0}^{m-1} \frac{\partial J}{\partial z_{i-a,j-b}^{l+1}} \frac{\partial z_{i-a,j-b}^{l+1}}{\partial a_{ij}^{l}} = \sum_{a=0}^{m-1}\sum_{b=0}^{m-1} \delta_{i-a,j-b}^{l+1} w_{a,b}^{l+1} \tag{34}$$

โดยที่ m คือขนาดของตัวกรอง จากนั้น เมื่อคำนวณค่าความผิดพลาดของแต่ละระดับชั้นได้ก็สามารถหาค่าผิดพลาดเทียบกับน้ำหนักและค่าไบแอสใด ๆ ได้จากสมการที่ (35) และ (36)

$$\frac{\partial J}{\partial w_{ab}^{l}} = \sum_{i=0}^{N-m}\sum_{j=0}^{N-m} \frac{\partial J}{\partial z_{ij}^{l}} \frac{\partial z_{ij}^{l}}{\partial w_{ab}^{l}} = \sum_{i=0}^{N-m}\sum_{j=0}^{N-m} \delta_{ij}^{l} a_{i+a,j+b}^{l-1} \tag{35}$$

$$\frac{\partial J}{\partial b^{l}} = \sum_{i=0}^{N-m}\sum_{j=0}^{N-m} \frac{\partial J}{\partial z_{ij}^{l}} \frac{\partial z_{ij}^{l}}{\partial b^{l}} = \sum_{i=0}^{N-m}\sum_{j=0}^{N-m} \delta_{ij}^{l} \tag{36}$$

ทั้งนี้ N คือขนาดของข้อมูลรับเข้าในชั้นที่ l

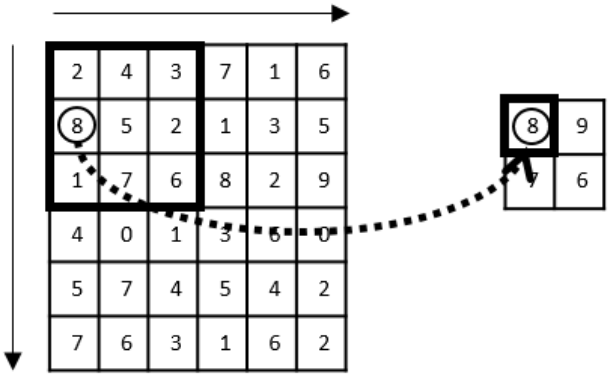
2.3.2 ชั้นการรวม (Pooling Layer)

ชั้นการรวมเป็นชั้นที่ทำหน้าที่ลดขนาดของข้อมูลลง เพื่อให้เหลือแค่เพียงข้อมูลที่สำคัญ ๆ เท่านั้น โดยทั่วไป มักจะทำการเลือกข้อมูลที่มีค่ามากที่สุดมาจากแต่ละช่วงของเมทริกซ์เพื่อสร้างเป็นเมทริกซ์ที่ขนาดเล็กลง ในงานวิจัยชิ้นนี้จะกล่าวถึงชั้นการรวมที่สำคัญดังต่อนี้

2.3.2.1 ชั้นการรวมโดยใช้ค่ามากสุด (Max pooling)

ชั้นการรวมโดยใช้ค่ามากสุดจะทำการเลือกเฉพาะค่ามากที่สุดจากกลุ่มของข้อมูลที่เราสนใจและนำไปใช้งานต่อไปในชั้นถัดไป จากรูปที่ 2.9 เป็นการทำการรวมโดยค่ามากสุดบนเมทริกซ์ขนาด 6

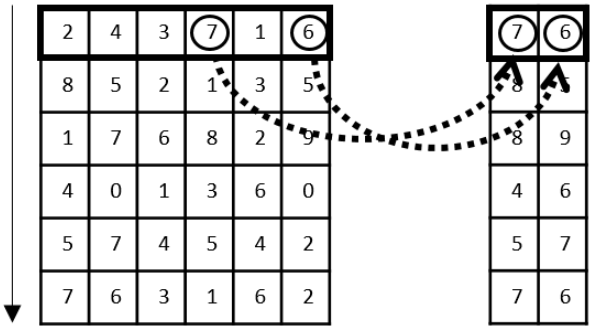
x 6 โดยกลุ่มที่สนใจจะมีขนาด 3 x 3 ซึ่งขอบเขตของกลุ่มที่สนใจจะมีการเลื่อนไปจนครอบคลุมเมทริกซ์ต้นฉบับทั้งหมด



รูปที่ 2.9 ชั้นการรวมโดยใช้ค่ามากสุดใน 2 มิติ

2.3.2.2 ชั้นการรวมโดยใช้เคค่ามากสุด (K-max pooling)

ชั้นการรวมโดยใช้เคค่ามากสุดจะทำการเลือก k ฟีเจอร์ที่มีค่ามากที่สุดจากข้อมูลทั้งหมดที่มีมักจะกระทำบนเวกเตอร์และนำไปใช้ในเน็ตเวิร์กที่กระทำกับข้อมูลที่เป็นข้อความ จากรูปที่ 2.10 เป็นการทำการรวมโดยใช้เคค่ามากสุดซึ่งกำหนดให้ k = 2 ทั้งนี้ในการทำการรวมแต่ละครั้ง จะสนใจในแนวเวกเตอร์ในแนวแกน x และการเลื่อนขอบเขตที่สนใจจะทำการเลื่อนเฉพาะแนวแกน y โดยจะสังเกตได้ว่าผลลัพธ์จากการทำการรวมโดยเคค่ามากสุดจะเรียงลำดับแบบเดิมเช่นเดียวกับเมทริกซ์ในชั้นก่อนหน้าในแนวแกน x



รูปที่ 2.10 ชั้นการรวมโดยใช้เคค่ามากสุดใน 1 มิติ โดยกำหนดให้ k = 2

2.3.2.3 ชั้นการรวมโดยใช้เคค่ามากสุดแบบพลวัต (Dynamic k-max pooling)

เป็นชั้นการรวมที่พัฒนาต่อจากชั้นการรวมโดยใช้เคค่ามากสุด โดยจะมีความแตกต่างอยู่ที่จะไม่มีการกำหนดค่า k ไว้ก่อน แต่จะทำการคำนวณค่า k ใหม่ให้เหมาะสมกับทุกขนาดของข้อมูลในชั้น

ก่อนหน้า วิธีการนี้ถูกเสนอโดย N. Kalchbrenner และคณะในปี ค.ศ. 2014 [12] เพื่อใช้ในนิวรอลเน็ตเวิร์กคอนโวลูชันพลวัตในการจำแนกข้อความภาษาอังกฤษ

2.3.2.4 การแพร่กระจายย้อนกลับ (Backpropagation)

สำหรับการแพร่กระจายย้อนกลับ เนื่องจากว่าในชั้นนี้เป็นการคัดลอกข้อมูลรับเข้าแต่ละช่องมาโดยตรง ดังนั้นจึงสามารถทำการแพร่กระจายค่าความผิดพลาดไปยังช่องของข้อมูลรับเข้าที่ถูกเลือกใช้ได้โดยตรง และในช่องข้อมูลอื่น ๆ ที่ไม่ได้ถูกเลือกใช้นั้น จะไม่มีการทำการแพร่กระจายย้อนกลับ

2.3.3 ชั้นการเชื่อมโยงเต็มรูปแบบ (Fully Connected Layer)

หลังจากการกระกอบกันของชั้นคอนโวลูชันและชั้นการรวมจำนวนหนึ่งแล้ว ในชั้นสุดท้ายของนิวรอลเน็ตเวิร์กคอนโวลูชัน จะเป็นการเชื่อมโยงเต็มรูปแบบ นั่นคือ ในชั้นนี้ประกอบด้วยชั้นย่อย ๆ ที่มีเพอร์เซ็ปตรอนอยู่จำนวนหนึ่ง โดยที่เพอร์เซ็ปตรอนแต่ละตัว จะมีเส้นเชื่อมกับเพอร์เซ็ปตรอนทุกตัวในชั้นก่อนหน้า และเพอร์เซ็ปตรอนทุกตัวในชั้นถัดไป ทั้งนี้ การคำนวณการป้อนไปข้างหน้าและการแพร่กระจายย้อนกลับสามารถทำได้ด้วยวิธีการปกติ

2.4 การวัดประสิทธิภาพ (Performance Evaluation)

การวัดประสิทธิภาพของการจำแนกแบบหลายคลาส (Multiclass classification) สามารถแสดงได้ดังนี้

2.4.1 คอนฟิวชันเมทริกซ์ (Confusion Matrix)

คอนฟิวชันเมทริกซ์คือเมทริกซ์ที่แสดงผลของการจำแนกโดยแสดงรายละเอียดแบ่งตามคลาส ตารางที่ 2.1 แสดงคอนฟิวชันเมทริกซ์ของการจำแนกแบบ 3 คลาส

ตารางที่ 2.1 คอนฟิวชันเมทริกซ์ของการจำแนกแบบ 3 คลาส

		คลาสที่ทำนาย		
		A	B	C
คลาสจริง	A	$M_{1,1}$ (TP_A)	$M_{1,2}$	$M_{1,3}$
	B	$M_{2,1}$	$M_{2,2}$ (TP_B)	$M_{2,3}$
	C	$M_{3,1}$	$M_{3,2}$	$M_{3,3}$ (TP_C)

สำหรับคอนฟิวชันเมทริกซ์ของการจำแนกจากข้อมูลทั้งหมด C คลาส ค่าในแต่ละแถวจะแสดงถึงจำนวนข้อมูลที่อยู่ในคลาสนั้นจริง ๆ ส่วนค่าในแต่ละสดมภ์จะหมายถึงจำนวนข้อมูลที่ทำนายได้คลาสนั้น กำหนดให้สำหรับแต่ละคลาสใด ๆ

- TP คือ จำนวนข้อมูลที่ทำนายได้คลาสนั้นและผลลัพธ์คือคลาสนั้น (True Positive) แสดงวิธีการคำนวณได้ดังสมการที่ (37)
- FP คือ จำนวนข้อมูลที่ทำนายได้คลาสนั้นแต่ผลลัพธ์คือคลาสอื่น (False Positive) แสดงวิธีการคำนวณได้ดังสมการที่ (38)
- TN คือ จำนวนข้อมูลที่ทำนายได้คลาสอื่นและผลลัพธ์คือคลาสอื่น (True Negative) แสดงวิธีการคำนวณได้ดังสมการที่ (39)
- FN คือ จำนวนข้อมูลที่ทำนายได้คลาสอื่นแต่ผลลัพธ์คือคลาสนั้น (False Negative) แสดงวิธีการคำนวณได้ดังสมการที่ (40)

$$TP_i = M_{i,i} \tag{37}$$

$$FP_i = \sum_{j=1}^{C} M_{j,i} \ (\text{ยกเว้น } j = i) \tag{38}$$

$$FN_i = \sum_{j=1}^{C} M_{i,j} \ (\text{ยกเว้น } j = i) \tag{39}$$

$$TN_i = \sum_{j=1}^{C} \sum_{k=1}^{C} M_{j,k} \ (\text{ยกเว้น } j = i \text{ หรือ } k = i) \tag{40}$$

2.4.2 ตัววัดประสิทธิภาพจำแนกตามคลาส

การคำนวณหาค่าความเที่ยง (Precision) ค่าความระลึก (Recall) และค่าเอฟวัน (F_1) นั้นสามารถคำนวณได้จาก (41), (42) และ (43)

$$Pr_i = \frac{TP_i}{TP_i + FP_i} \tag{41}$$

$$Re_i = \frac{TP_i}{TP_i + FN_i} \tag{42}$$

$$F_{1,i} = \frac{2 \times Pr_i \times Re_i}{Pr_i + Re_i} \tag{43}$$

2.4.3 ตัววัดประสิทธิภาพโดยรวม

การคำนวณประสิทธิภาพของการจำแนกโดยรวมจะใช้ค่าเฉลี่ยของตัววัดประสิทธิภาพในแต่ละคลาสมาคำนวณได้ดังสมการที่ (44), (45) และ (46)

$$Pr = \frac{\sum_{i=1}^{C} Pr_i}{C} \tag{44}$$

$$Re = \frac{\sum_{i=1}^{C} Re_i}{C} \tag{45}$$

$$F_1 = \frac{\sum_{i=1}^{C} F_{1,i}}{C} \tag{46}$$

และในการหาค่าความแม่น (Accuracy) กำหนดให้ N คือจำนวนข้อมูลทั้งหมด จะแสดงสมการของความแม่นได้โดย (47)

$$Accuracy = \frac{\sum_{i=1}^{C} TP_i}{N} \tag{47}$$

2.5 งานวิจัยที่เกี่ยวข้อง (Related Work)

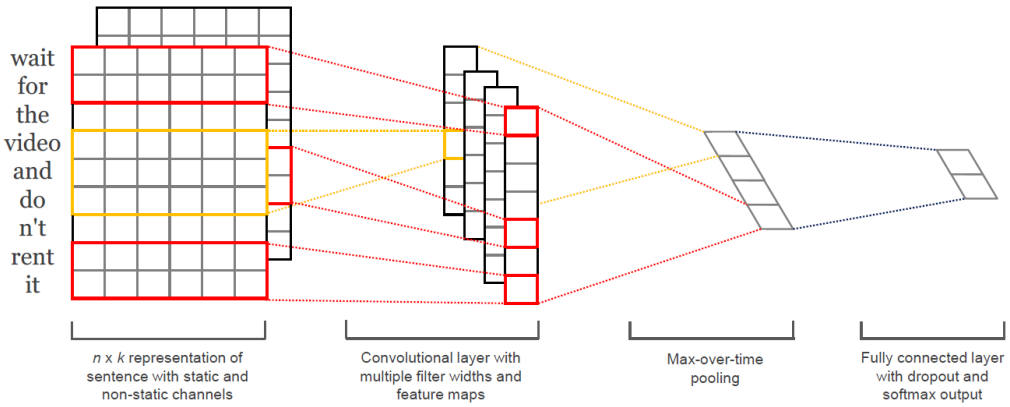
งานวิจัยที่เกี่ยวข้องกับวิทยานิพนธ์ฉบับนี้ เป็นงานวิจัยที่ใช้นิวรอลเน็ตเวิร์กคอนโวลูชันในการจำแนกประเภทของข้อความ โดยงานวิจัยแต่ละชิ้นจะทำการสร้างเน็ตเวิร์กที่นำไปใช้ในงานแบบต่าง ๆ โดยประกอบขึ้นด้วยหน่วยย่อยของเน็ตเวิร์กดังที่กล่าวไปข้างต้น ในหัวข้อนี้จะแบ่งงานวิจัยออกเป็นสองกลุ่มได้แก่ นิวรอลเน็ตเวิร์กคอนโวลูชันที่ใช้ข้อมูลรับเข้าระดับคำ และนิวรอลเน็ตเวิร์กคอนโวลูชันที่ใช้ข้อมูลรับเข้าระดับตัวอักษร

2.5.1 นิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำ

ในการใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับในการจำแนกข้อความ มีโครงสร้างของเน็ตเวิร์กอยู่สองรูปแบบที่ได้รับความนิยม ดังนี้

2.5.1.1 นิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำ โดย Y. Kim และคณะ

นิวรอลเน็ตเวิร์กคอนโวลูชันถูกนำมาใช้ในการจำแนกประเภทข้อความในปี 2014 ซึ่งตัวอย่างหนึ่งของโครงสร้างของเน็ตเวิร์กที่ได้รับความนิยม ได้แก่ นิวรอลเน็ตเวิร์กคอนโวลูชันที่เสนอโดย Y. Kim [13] แสดงได้ดังรูปที่ 2.11



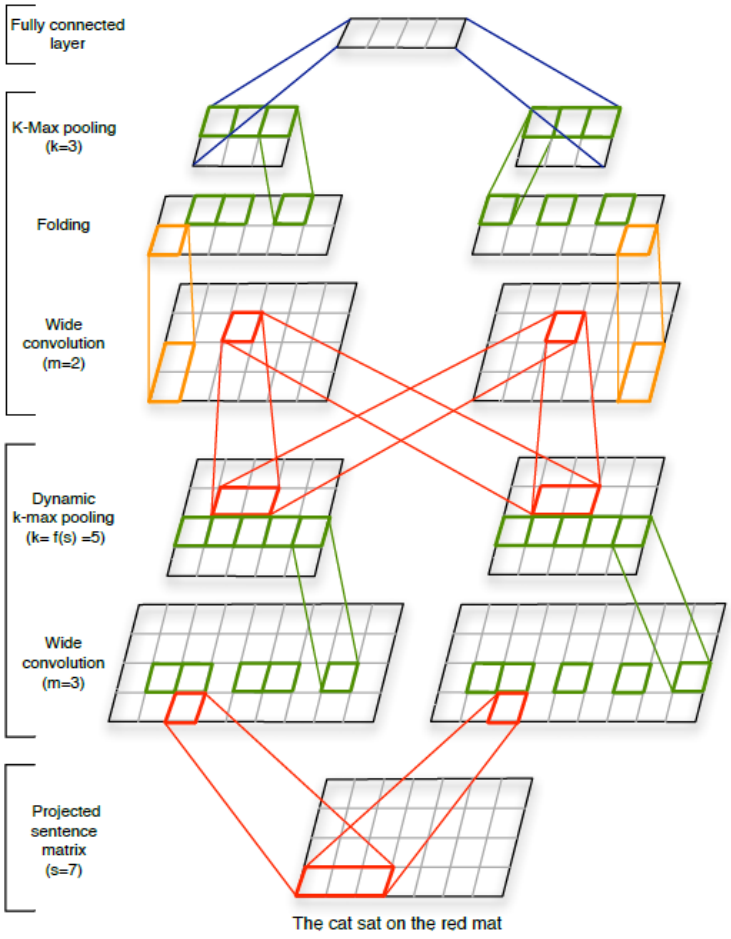
รูปที่ 2.11 โครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำที่ใช้ในการจำแนกประเภทข้อความ (อ้างอิงจาก Fig. 1 ใน [13])

โดยในชั้นของข้อมูลรับเข้าจะเป็นเมทริกซ์ที่มี 2 ช่องสัญญาณ คือช่องสัญญาณแบบคงที่ และช่องสัญญาณที่มีการเรียนรู้ ทั้งนี้ เมทริกซ์ส่วนนี้ถูกสร้างขึ้นมาด้วยการประกอบเวกเตอร์ของคำเข้าด้วยกัน โดยการนำมาต่อกันตามลำดับของคำที่ปรากฏในข้อความดั้งเดิม จากนั้นจึงเป็นชั้นคอนโวลูชัน โดยจะมีตัวกรองที่มีขนาดแตกต่างกันออกไป โดยตัวกรองเหล่านี้ จะมีขนาดในแนวแกนหนึ่งเท่ากับความยาวของเวกเตอร์คำ ส่วนอีกในแนวแกนหนึ่งจะมีขนาดตามที่กำหนดไว้ การเลื่อนตัวกรองเพื่อทำการคอนโวลูชันจะมีการเลื่อนไปเพียงแค่แนวแกนเดียว ฟีเจอร์เวกเตอร์ที่ได้จากแต่ละตัวกรอง

จะถูกนำไปเลือกฟีเจอร์ที่มีค่ามากที่สุดมาเพียงหนึ่งค่าต่อหนึ่งตัวกรอง หรือที่มีชื่อเรียกว่า max over time pooling จากนั้นจึงนำไปเข้าสู่ชั้นการเชื่อมโยงเต็มรูปแบบเพื่อทำการจำแนกประเภทข้อความต่อไป ทั้งนี้ การทำ max over time pooling เป็นวิธีการรวมรูปแบบหนึ่งที่จะทำให้สามารถสร้างฟีเจอร์เวกเตอร์ที่มีขนาดคงที่ จากข้อมูลรับเข้าที่มีขนาดไม่คงที่ได้ จึงทำให้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำนี้สามารถรับข้อมูลที่มีความยาวใด ๆ ได้ เน็ตเวิร์กชนิดนี้สามารถเอาชนะแบบจำลองอื่น ๆ ได้แก่ ซัพพอร์ตเวกเตอร์แมชชีน รวมถึงนิวรอลเน็ตเวิร์กแบบต่าง ๆ ในการจำแนกข้อความไปถึง 4 ใน 7 ชุดข้อมูล ทั้งนี้ ชุดข้อมูลเหล่านี้เป็นการจำแนกรีวิวภาพยนต์ที่ทางด้านบวกและลบ การจำแนกรีวิวสินค้า และการจำแนกประเภทของคำถาม

2.5.1.2 นิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำแบบพลวัต โดย N. Kalchbrenner และคณะ (Dynamic Convolutional Neural Network หรือ DCNN)

ในหัวข้อนี้จะกล่าวถึงนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำอีกรูปแบบหนึ่งซึ่งสามารถรับข้อมูลที่ความยาวใด ๆ คือ นิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำแบบพลวัต (Dynamic Convolutional Neural Network หรือ DCNN) [12] ซึ่งจะถูกนำไปใช้เป็นหนึ่งในแบบจำลองเพื่อจะเปรียบเทียบกับวิธีระดับตัวอักษรที่เสนอในงานวิจัยฉบับนี้ โดยในปี 2014 N. Kalchbrenner, E. Grefenstette และ P. Blunsom ได้เสนอนิวรอลเน็ตเวิร์กชนิดนี้เพื่อใช้สำหรับการจำแนกประเภทข้อความ [12] โครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันแบบพลวัตแสดงได้ดังรูปที่ 2.12



รูปที่ 2.12 โครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำแบบพลวัต (อ้างอิงจาก Fig. 3 ใน [12])

โดยในชั้นข้อมูลรับเข้า จะสร้างขึ้นมาจากเวกเตอร์ของคำที่ต่อกันตามลำดับของคำในข้อความ สำหรับในชั้นคอนโวลูชัน จะเป็นการทำคอนโวลูชันแบบกว้าง โดยที่ตัวกรองจะเป็นเวกเตอร์ที่มีความกว้างหนึ่งหน่วย และมีความยาวตามที่กำหนดไว้ ผลลัพธ์ของตัวกรองแต่ละตัวจะถูกนำไปเชื่อมต่อกันเป็นเมทริกซ์ สำหรับในชั้นการรวม เน็ตเวิร์กชนิดนี้มีความพิเศษกล่าวคือจะใช้การรวมแบบเคค่ามากสุดพลวัต นั่นคือจะมีการคำนวณค่า k ในทุก ๆ รอบของการป้อนไปข้างหน้า ซึ่งจะทำให้จำนวนในการเลือกข้อมูลมีความเหมาะสมกับข้อมูลที่ความยาวต่าง ๆ กัน กำหนดให้ k_l คือค่า k ที่จะถูกนำไปใช้ในชั้นการรวมที่ l โดย k_{top} คือค่า k แบบคงที่ที่ถูกกำหนดไว้ใช้ในชั้นการรวมสุดท้าย L คือจำนวนชั้นการรวมทั้งหมด และ s คือความยาวของข้อมูลรับเข้าในชั้นการรวม จะสามารถคำนวณค่า k ในแต่ละชั้นได้จากสมการ (48)

$$k_l = max(k_{top}, \left\lceil \frac{L-l}{L} s \right\rceil) \quad (48)$$

จากการที่มีการคำนวณค่า k ใหม่ทุก ๆ รอบ ทำให้ขนาดของเมทริกซ์ที่ใช้ในแต่ละชั้นจะแตกต่างกันไปตามความเหมาะสม แบบจำลองนี้ได้มีการนำมาใช้จำแนกข้อความด้านอารมณ์จากรีวิว

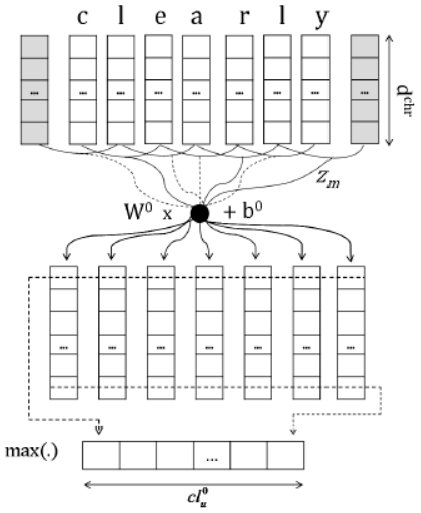
ภาพยนต์และจากทวิตเตอร์ (Twitter) และการจำแนกประเภทของคำถามในภาษาอังกฤษ โดยได้ผลลัพธ์ที่ดีกว่าวิธีอื่น ๆ ได้แก่ ซัพพอร์ตเวกเตอร์แมชชีน นาอีฟเบย์ แมกซิมัมเอนโทรปี และนิวรอลเน็ตเวิร์กแบบหน่วงเวลา ทั้งนี้นิวรอลเน็ตเวิร์กคอนโวลูชันแบบพลวัตของ N. Kalchbrenner และคณะ ให้ความแม่นยำที่ใกล้เคียงกับนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำที่ถูกเสนอโดย Y. Kim และสามารถชนะในการจำแนกอารมณ์ของข้อความ สำหรับงานวิจัยฉบับนี้จะใช้นิวรอลเน็ตเวิร์กคอนโวลูชันแบบพลวัตในการเปรียบเทียบผลการทดลองในบทที่ 4.4

2.5.2 นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร

สำหรับการนำนิวรอลเน็ตเวิร์กคอนโวลูชันมาใช้ในการจำแนกข้อความ โดยที่มีการรับข้อมูลเข้ามาในระดับตัวอักษร สามารถแบ่งออกได้เป็นสองรูปแบบ คือ 1) การใช้ข้อมูลระดับตัวอักษรมาช่วยในการสร้างฟีเจอร์ระดับคำเท่านั้น แต่ยังต้องอาศัยขั้นตอนการตัดคำ และ 2) การนำข้อมูลระดับตัวอักษรไปใช้ในการจำแนกข้อความได้โดยตรง มีรายละเอียดดังงานวิจัยต่อไปนี้

2.5.2.1 นิวรอลเน็ตเวิร์กคอนโวลูชันที่แปลงจากตัวอักษรเป็นประโยค โดย C. N. dos Santos และ M. Gatti (Character to Sentence Convolutional Neural Network หรือ CharSCNN)

ในการใช้งานระดับตัวอักษร C. N. dos Santos และ M. Gatti [14] ได้เสนอการใช้นิวรอลเน็ตเวิร์กคอนโวลูชันจากตัวอักษรเป็นประโยค (Character to Sentence Convolutional Neural Network หรือ CharSCNN) โดยเน็ตเวิร์กจะมีการใช้ข้อมูลระดับตัวอักษรเพื่อสร้างเวกเตอร์ของตัวอักษร และทำคอนโวลูชันบนเมทริกซ์ที่เกิดจากการต่อกันของเวกเตอร์ตัวอักษรในแต่ละคำ เพื่อให้ได้ผลลัพธ์เป็นเวกเตอร์ของคำอีกทีหนึ่ง ดังแสดงในรูปที่ 2.13

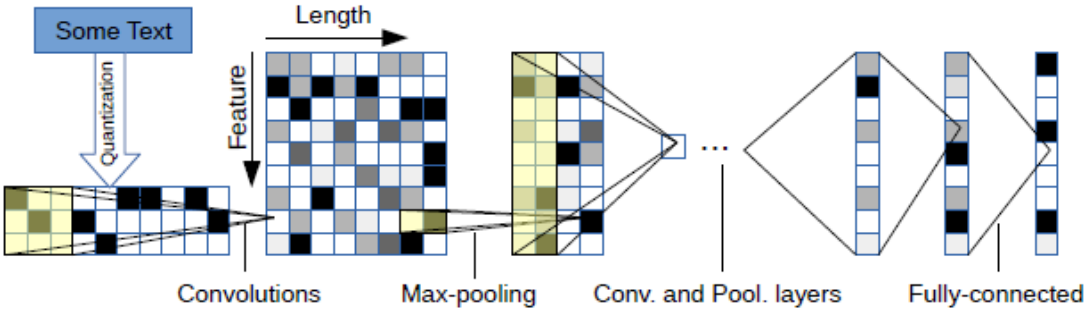


รูปที่ 2.13 การสร้างเวกเตอร์ระดับคำจากเวกเตอร์ระดับตัวอักษร (อ้างอิงจาก Fig. 1 ใน [14])

จากนั้น จึงนำเวกเตอร์คำที่เกิดจากเวกเตอร์ตัวอักษร ไปรวมกับเวกเตอร์ที่สร้างขึ้นมาจากแต่ละคำโดยตรง เพื่อเป็นข้อมูลรับเข้าสำหรับการทำคอนโวลูชันต่อไป จุดประสงค์ของการใช้เวกเตอร์ตัวอักษรเพื่อสร้างเป็นเวกเตอร์ของคำนั้นเพื่อให้เวกเตอร์ของคำเกิดจากส่วนประกอบย่อยของคำ ๆ นั้น และสามารถเรียนรู้คำที่เกิดจากการแปลงรูปได้อย่างมีประสิทธิภาพ จะเห็นได้ว่าวิธีการนี้ เป็นการใช้ข้อมูลตัวอักษรเพื่อเพิ่มประสิทธิภาพให้ข้อมูลระดับคำ แต่ยังจำเป็นต้องมีการตัดคำอยู่ ผลการทดลองจากนิวรอลเน็ตเวิร์กคอนโวลูชันรูปแบบนี้พบว่า การนำเวกเตอร์ตัวอักษรมาใช้งานร่วมด้วยจะให้ความแม่นยำที่ดีกว่าการใช้งานเวกเตอร์คำเพียงอย่างเดียวในการการจำแนกข้อความรีวิวภาพยนต์และการจำแนกอารมณ์จากทวิตเตอร์ในภาษาอังกฤษ

2.5.2.2 นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร โดย X. Zhang และคณะ (Character-level Convolutional Neural Network หรือ Char-CNN)

ในปี 2015 X. Zhang, J. Zhao และ Y. LeCun [15] ได้เสนอนิวรอลเน็ตเวิร์กคอนโวลูชันซึ่งรับข้อมูลเป็นลำดับของตัวอักษร ดังแสดงในรูปที่ 2.14



รูปที่ 2.14 แบบจำลองของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร (อ้างอิงจาก Fig. 1 ใน [15])

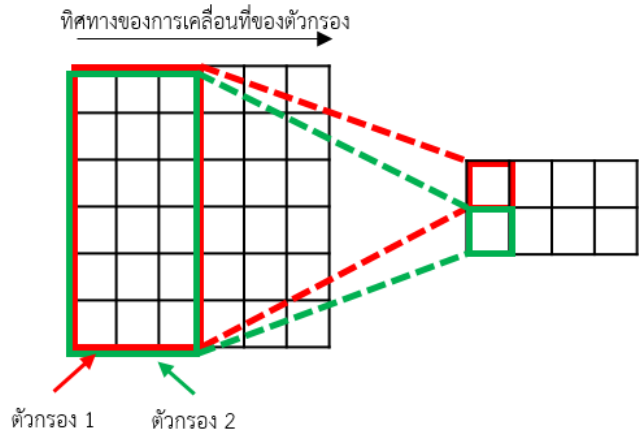
ในชั้นข้อมูลรับเข้า จะเป็นการนำเวกเตอร์วันฮอทมาเชื่อมต่อกันตามลำดับของตัวอักษรในข้อความต้นฉบับ แสดงได้โดย (49)

$$v_a = \begin{bmatrix}1\\0\\0\\\vdots\\0\end{bmatrix}, v_b = \begin{bmatrix}0\\1\\0\\\vdots\\0\end{bmatrix}, \cdots \tag{49}$$

จากสมการ แสดงถึงเวกเตอร์วันฮอทของตัวอักษรในภาษาอังกฤษ v_a แทนเวกเตอร์ของ “a” v_b คือเวกเตอร์วันฮอทของ “b” จากนั้นจึงนำเวกเตอร์วันฮอทเหล่านี้มาประกอบเข้าด้วยกันให้กลายเป็นเมทริกซ์ เช่น การแปลงส่วนของประโยค “a bad cab” ให้กลายเป็นเมทริกซ์ แสดงได้โดย (50)

$$"a\ bad\ cab" \rightarrow \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (50)$$

เมื่อได้เมทริกซ์ที่แสดงข้อความแล้ว ชั้นต่อไปคือชั้นคอนโวลูชัน โดยที่การทำคอนโวลูชันกับข้อมูลที่เป็นข้อความจะเป็นการทำคอนโวลูชันที่มีการเลื่อนตัวกรองเพียงแค่ 1 มิติ เรียกว่าเทมโพรอลคอนโวลูชัน (Temporal Convolution) ดังแสดงในรูปที่ 2.15 จะเห็นได้ว่า ตัวกรองมีขนาดในแนวแกน y เท่ากับขนาดของเมทริกซ์รับเข้า ในขณะที่ในแนวแกน x ตัวกรองจะมีขนาดตามที่กำหนด

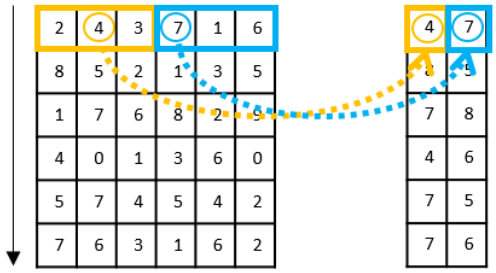


รูปที่ 2.15 แสดงตัวอย่างการทำเทมโพรอลคอนโวลูชัน บนเมทริกซ์ขนาด 6 x 6 และมีตัวกรองความกว้าง 3 จำนวน 2 ตัวกรอง

กำหนดให้เมทริกซ์รับเข้ามีขนาด $f \times l$ โดย f คือจำนวนฟีเจอร์ในระดับตัวอักษรที่กำหนด l คือความยาวของข้อความที่แสดงถึง และมีตัวกรองขนาด $f \times w$ จำนวนทั้งสิ้น n ตัวกรอง จะได้ว่าเมทริกซ์ที่เป็นผลลัพธ์ของการคำเทมโพรอลคอนโวลูชันจะมีขนาด $n \times (l - w + 1)$ ซึ่งก็คือการนำเวกเตอร์ผลลัพธ์ของแต่ละตัวกรองมาต่อกันให้กลายเป็นเมทริกซ์

สำหรับชั้นการรวม ในเน็ตเวิร์กที่เสนอขึ้นมานั้น จะใช้ชั้นการรวมที่มีการเลื่อนขอบเขตที่สนใจใน 1 มิติเช่นเดียวกับการทำคอนโวลูชัน หรือที่มีชื่อเรียกว่าการรวมโดยใช้ค่ามากสุดแบบเทมโพรอล (Temporal Max-Pooling)

รูปที่ 2.16 แสดงการทำการรวมโดยใช้ค่ามากสุดแบบเทมโพรอล กล่าวคือขอบเขตพื้นที่ที่สนใจจะมีขนาด $1 \times w$ โดย w คือขนาดของการทำการรวม และกำหนดให้เมทริกซ์ตั้งต้นมีขนาด $f \times l$ จะได้ว่าผลลัพธ์ของการทำการรวมโดยใช้ค่ามากสุดแบบเทมโพรอลจะได้เมทริกซ์ที่มีขนาด $f \times \frac{l}{w}$



รูปที่ 2.16 แสดงตัวอย่างการทำการรวมโดยใช้ค่ามากสุดแบบเทมโพรอล บนเมทริกซ์ขนาด 6 x 6 และมีขนาดของการทำการรวมเป็น 3

สำหรับโครงสร้างทั้งหมดของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรสรุปไว้ดังตารางที่ 2.2 สังเกตได้ว่า ขนาดของผลลัพธ์ในแต่ละชั้นจะมีขนาดคงที่แน่นอน และในชั้นข้อมูลรับเข้า จะโดนจำกัดความยาวของข้อความไว้ที่ 1014 ตัวอักษร ทั้งนี้ หากข้อความที่จะนำมาจำแนกมีจำนวนตัวอักษรเกินจำนวนที่กำหนด จะโดนตัดออกให้เหลือแค่ 1014 ตัวอักษรเท่านั้น นอกจากโครงสร้างที่ปรากฏในตารางแล้ว ยังมีการใช้ฟังก์ชันขีดแบ่งที่มีการกำหนดค่าขีดแบ่งไว้ที่ 0.000001 ซึ่งทำให้มีความคล้ายคลึงกับการทำฟังก์ชันเรคติไฟต์เชิงเส้น โดยจะเป็นการทำให้ตัวจำแนกสามารถจำแนกข้อมูลที่ไม่อยู่ในรูปแบบเชิงเส้นได้ โดยการใช้ฟังก์ชันขีดแบ่งจะเกิดขึ้นต่อจากทุก ๆ ชั้นคอนโวลูชันและชั้นการเชื่อมโยงเต็มรูปแบบ ยกเว้นเพียงแต่ชั้นการเชื่อมโยงเต็มรูปแบบในชั้นสุดท้าย ซึ่งจะใช้ฟังก์ชันค่าสูงสุดอย่างอ่อนแทน สำหรับการเรียนรู้ของเน็ตเวิร์กจะใช้วิธีการแพร่กระจายย้อนกลับซึ่งใช้ฟังก์ชันต้นทุนเป็นฟังก์ชันลบลอการิทึมของความเป็นไปได้

ตารางที่ 2.2 โครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่เสนอโดย X. Zhang และคณะ [15]

ประเภทของชั้น	พารามิเตอร์	ขนาดของผลลัพธ์
ชั้นข้อมูลรับเข้า	ความยาวตัวอักษร = 1014 จำนวนตัวอักษรทั้งหมดที่สนใจ = 70	70×1014
ชั้นคอนโวลูชันและฟังก์ชันขีดแบ่ง	ขนาดตัวกรอง = 7 จำนวนตัวกรอง = 256	256×1008
ชั้นการรวม	ขนาดของการรวม = 3	256×336
ชั้นคอนโวลูชันและฟังก์ชันขีดแบ่ง	ขนาดตัวกรอง = 7 จำนวนตัวกรอง = 256	256×330
ชั้นการรวม	ขนาดของการรวม = 3	256×110

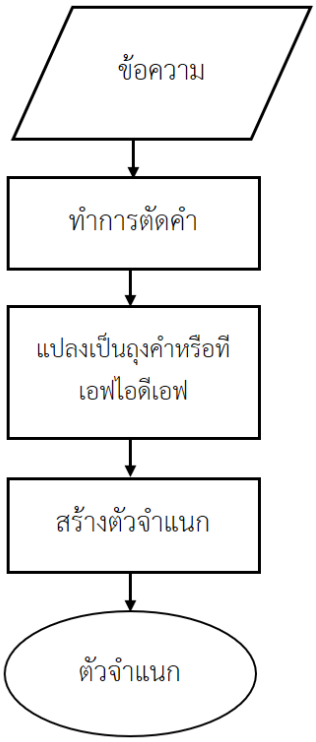
ชั้นคอนโวลูชันและฟังก์ชันขีดแบ่ง	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	256×108
ชั้นคอนโวลูชันและฟังก์ชันขีดแบ่ง	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	256×106
ชั้นคอนโวลูชันและฟังก์ชันขีดแบ่ง	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	256×104
ชั้นคอนโวลูชันและฟังก์ชันขีดแบ่ง	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	256×102
ชั้นการรวม	ขนาดของการรวม = 3	256×34
ชั้นการเชื่อมโยงเต็มรูปแบบและฟังก์ชันขีดแบ่ง	จำนวนนิวรอล = 1024	1024
ชั้นดรอปเอาท์	ความน่าจะเป็น = 0.5	1024
ชั้นการเชื่อมโยงเต็มรูปแบบและฟังก์ชันขีดแบ่ง	จำนวนนิวรอล = 1024	1024
ชั้นดรอปเอาท์	ความน่าจะเป็น = 0.5	1024
ชั้นการเชื่อมโยงเต็มรูปแบบ	จำนวนนิวรอล = 5 (จำนวนประเภทที่ต้องการจำแนก)	5

โดยเน็ตเวิร์กนี้ถูกนำไปทดสอบกับการจำแนกประเภทของข้อความและการจำแนกอารมณ์ของข้อความ โดยการเปรียบเทียบกับวิธีการจำแนกอื่น ๆ รวมถึงนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำและหน่วยความจำระยะสั้นแบบยาว ผลคือ แบบจำลองนี้ให้ความแม่นยำที่ดีกว่าใน 4 ชุดข้อมูลจากทั้งหมด 8 ชุดข้อมูล

บทที่ 3
การใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่รองรับข้อมูลความยาวใด ๆ ในการจำแนกประเภทข้อความภาษาไทย

3.1 การลดขั้นตอนการจำแนกข้อความภาษาไทย ด้วยการใช้ข้อมูลระดับตัวอักษร

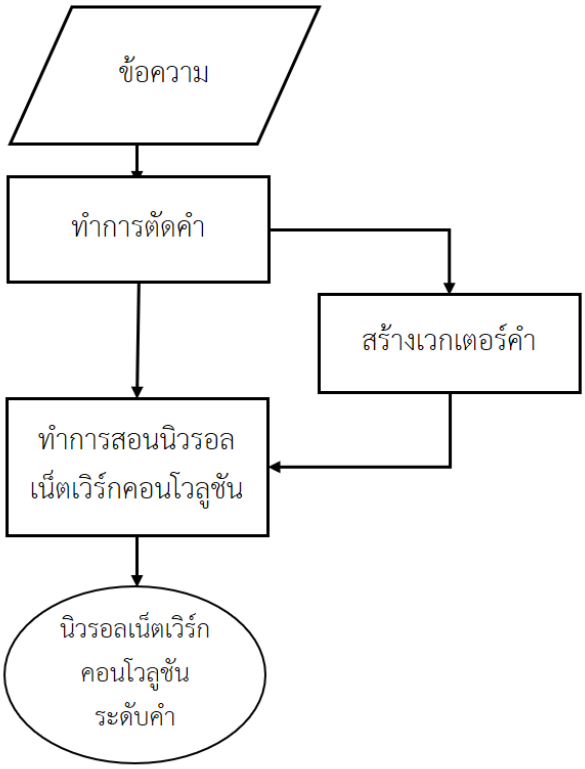
ในการจำแนกประเภทของข้อความภาษาไทยนั้น โดยปกติ ขั้นตอนแรกหลังจากที่มีข้อมูลที่ต้องการใช้งานแล้วก็คือการตัดคำ ซึ่งเป็นการกระทำเพื่อที่จะให้สามารถใช้งานตัวจำแนกได้ เพราะโดยทั่วไป หน่วยย่อยที่สุดที่มีความหมายก็คือข้อมูลระดับคำ ต่อจากนั้น จึงทำการสร้างฟีเจอร์ที่จะนำไปใช้งานด้วยวิธีการต่าง ๆ เช่น สร้างเป็นถุงคำ หรือ ทีเอฟไอดีเอฟ ขั้นตอนต่อไปเมื่อได้ฟีเจอร์เหล่านั้นแล้ว จะนำไปสร้างตัวจำแนกแบบต่าง ๆ ซึ่งขั้นตอนในส่วนนี้จะมีวิธีการเรียนรู้ตามวิธีที่เลือกใช้ ขั้นตอนวิธีทั้งหมดสามารถแสดงได้ดังรูปที่ 3.1



รูปที่ 3.1 ขั้นตอนในการจำแนกข้อความภาษาไทยที่ใช้การแทนข้อความแบบถุงคำหรือทีเอฟไอดีเอฟ

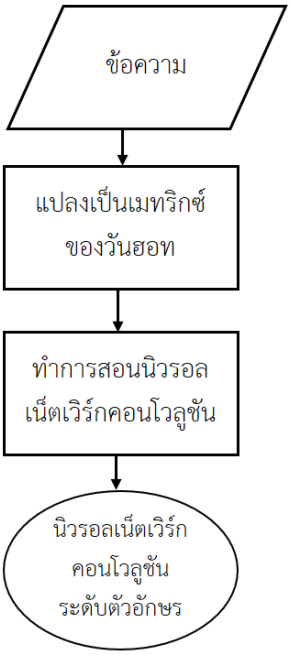
สำหรับการใช้นิวรอลเน็ตเวิร์กระดับคำที่รับข้อมูลเป็นเวกเตอร์ของคำ จะมีขั้นตอนในการจำแนกข้อความคล้ายกับขั้นตอนของตัวจำแนกระดับคำ พิจารณาได้ดังรูปที่ 3.2 นั่นคือ เมื่อทำการตัดคำของข้อความต้นฉบับแล้ว จะนำไปสร้างเป็นเวกเตอร์ของคำ จากนั้น จึงเข้าสู่ขั้นตอนของการเรียนรู้ ซึ่งการเรียนรู้ของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำจะใช้ข้อมูลรับเข้าเป็นเวกเตอร์ ซึ่งเวกเตอร์เหล่านี้จะมีการเรียนรู้จากกระบวนการแพร่กระจายย้อนกลับ แต่ทั้งนี้ หากมีการใช้ค่าเริ่มต้นที่

เหมาะสม จะทำให้การเรียนรู้สามารถทำได้ดีขึ้น จึงมีการนำเวกเตอร์ของคำที่สร้างได้มาเป็นข้อมูลตั้งต้นของเวกเตอร์ที่แทนที่ของแต่ละคำ จากนั้น ขั้นตอนต่อมาจึงเป็นการสร้างแบบจำลองโดยการเรียนรู้จากตัวอย่างข้อความที่มีอยู่



รูปที่ 3.2 ขั้นตอนในการจำแนกข้อความภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำ

จากวิธีในการจำแนกข้อความภาษาไทยทั้งสองวิธีข้างต้นนั้น จะเห็นว่าจำเป็นจะต้องมีการตัดคำก่อน เพื่อที่จะสามารถนำไปใช้งานเป็นฟีเจอร์ของตัวจำแนกต่าง ๆ ได้ ซึ่งขั้นตอนในการตัดคำเป็นขั้นตอนที่สามารถส่งผลต่อประสิทธิภาพโดยรวมของการจำแนก จึงจำเป็นจะต้องเลือกใช้วิธีการตัดคำที่เหมาะสม ในงานวิจัยชิ้นนี้ ต้องการที่จะลดขั้นตอนในการจำแนกข้อความภาษาไทย จึงได้มีการนำวิธีการนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรมาใช้ในการจำแนกข้อความภาษาไทย ซึ่งจะมีผลให้สามารถลดขั้นตอนในการจำแนกทั้งหมดลงได้ ดังแสดงในรูปที่ 3.3 จะเห็นได้ว่าเมื่อเน็ตเวิร์กสามารถรับข้อมูลที่อยู่ในระดับของตัวอักษรได้ จะไม่จำเป็นต้องมีขั้นตอนของการตัดคำอีกต่อไป นอกจากนี้การทำเช่นนี้จะทำให้ประสิทธิภาพของการจำแนกไม่ขึ้นอยู่กับประสิทธิภาพของการตัดคำ

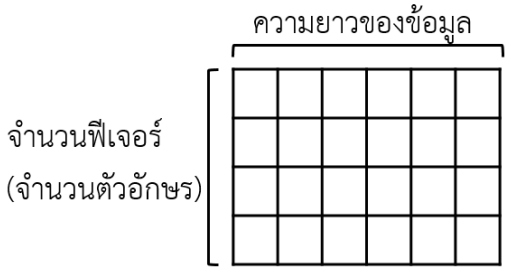


รูปที่ 3.3 ขั้นตอนในการจำแนกข้อความภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร

3.2 การใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรในการรับข้อมูลความยาวใด ๆ

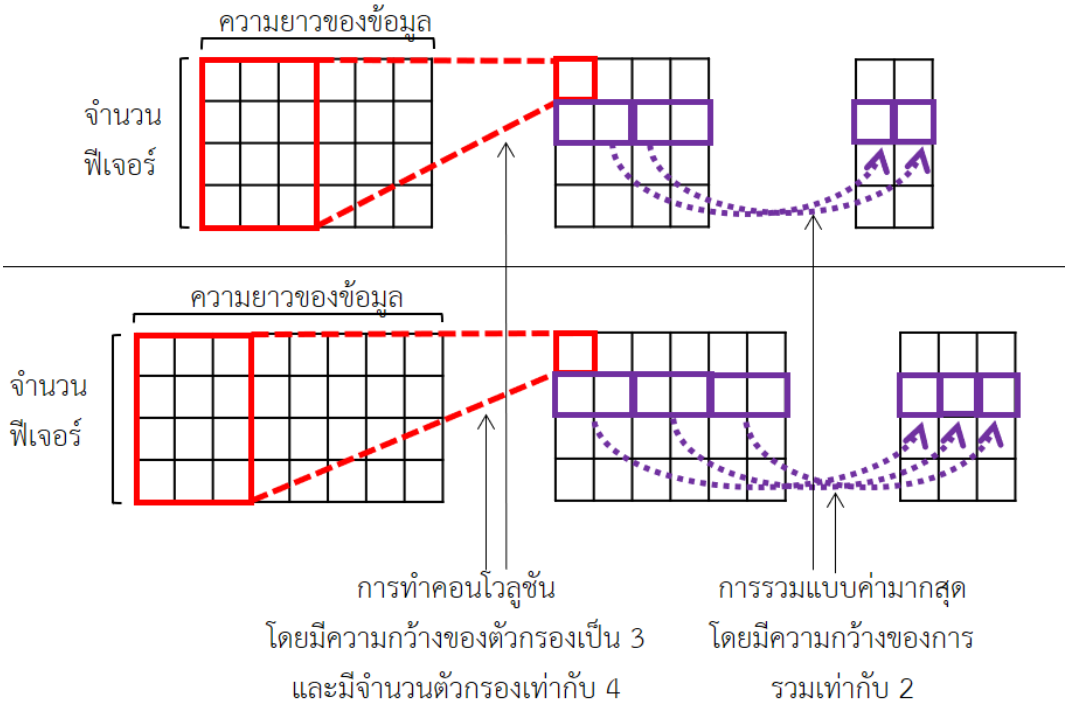
นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่เสนอโดย Zhang และคณะดังรายละเอียดในบทที่ 2.5.2.2 นั้น มีการจำกัดความยาวอยู่ที่ 1,014 ตัวอักษร โดยที่ตัวอักษรที่อยู่นอกเหนือจากความยาวที่กำหนดจะถูกตัดออกและไม่ถูกนำไปใช้ ซึ่งผู้วิจัยมีความเห็นว่าหากมีการปรับปรุงให้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรสามารถรับข้อมูลที่ความยาวใด ๆ ก็ได้ จะส่งผลให้ประสิทธิภาพของตัวจำแนกดีขึ้น เพราะมีตัวอย่างที่ใช้ในการเรียนรู้มากขึ้น

ทั้งนี้ เมื่อพิจารณาถึงวิธีการรับข้อมูลของนิวรอลเน็ตเวิร์กระดับตัวอักษรดังกล่าว จะพบว่าเมทริกซ์ที่เปรียบเสมือนข้อมูลที่ส่งผ่านในเน็ตเวิร์กในแต่ละชั้นนั้น จะมีขนาดที่เกิดจากความยาวของข้อมูล และจำนวนฟีเจอร์ในระดับชั้นนั้น ๆ คูณกัน ดังแสดงในรูปที่ 3.4 สำหรับความยาวของข้อมูลนั้น ในชั้นเริ่มแรกจะมีขนาดตามที่กำหนดไว้ คือ 1,014 และมีการลดหลั่นกันไปตามการทำคอนโวลูชันและการรวม สำหรับอีกมิติหนึ่ง จะมีจำนวนเท่ากับจำนวนฟีเจอร์ในชั้นนั้น ๆ เช่น ในชั้นข้อมูลรับเข้า จะมีจำนวนฟีเจอร์เท่ากับ 70 ซึ่งก็คือจำนวนตัวอักษรที่ถูกนำมาสร้างเป็นเวกเตอร์วันฮอทเพื่อนำมาประกอบกันเป็นเมทริกซ์รับเข้า สำหรับชั้นคอนโวลูชัน เมทริกซ์ผลลัพธ์ที่เกิดจากการทำคอนโวลูชันจะมีขนาดในมิตินี้เท่ากับจำนวนตัวกรองที่กำหนด สามารถดูขนาดของข้อมูลสำหรับเน็ตเวิร์กที่เสนอโดย Zhang และคณะได้ในตารางที่ 2.2



รูปที่ 3.4 ลักษณะของขนาดข้อมูลที่ส่งผ่านในแต่ละชั้นของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร

ในการทำให้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรมีการรับข้อมูลรับเข้าที่ขนาดความยาวใด ๆ สามารถทำได้โดยไม่ต้องมีการเปลี่ยนแปลงโครงสร้างหรือวิธีของเน็ตเวิร์กในชั้นคอนโวลูชันและชั้นการรวมแบบค่ามากสุด เพียงแต่ยกเลิกการจำกัดความยาวในชั้นข้อมูลรับเข้าเท่านั้น พิจารณารูปที่ 3.5 แสดงถึงลักษณะการใช้ข้อมูลรับเข้าที่ความยาวต่างกัน รูปบนแสดงถึงข้อมูลรับเข้าที่มีความยาวน้อยกว่ารูปล่าง สังเกตได้ว่า ความยาวของข้อมูลทั้งจากการผ่านชั้นคอนโวลูชัน และชั้นการรวมแล้วนั้น จะมีอยู่มิติหนึ่งที่เท่ากันนั่นคือจำนวนฟีเจอร์ ส่วนอีกมิติหนึ่งคือความยาวของข้อมูลนั้นจะมีขนาดไม่เท่ากัน โดยนิวรอลเน็ตเวิร์กที่เสนอในงานวิจัยฉบับนี้จะยกเลิกการจำกัดความยาวของข้อมูลรับเข้าและมีการทำงานกับข้อมูลที่ความยาวต่าง ๆ กันดังการอธิบายข้างต้น สำหรับการจัดการกับข้อมูลที่มีความยาวต่างกันในชั้นการเชื่อมโยงเต็มรูปแบบนั้น จะกล่าวถึงในบทที่ 3.3

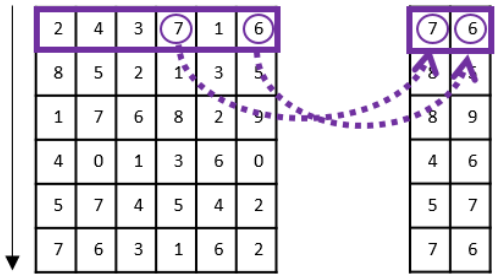


รูปที่ 3.5 ขนาดของข้อมูลในชั้นคอนโวลูชันและชั้นการรวมแบบค่ามากสุด เมื่อใช้ข้อมูลรับเข้าที่ความยาวต่างกัน

3.3 การปรับปรุงนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีข้อมูลรับเข้าความยาวใด ๆ ให้มีขนาดข้อมูลที่เหมาะสมกับชั้นการเชื่อมโยงเต็มรูปแบบ

จากบทที่ 3.2 ซึ่งทำการยกเลิกข้อจำกัดด้านความยาวของข้อมูลรับเข้าของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรไปนั้น เมื่อผ่านชั้นคอนโวลูชันและชั้นการรวมแบบค่ามากสุดไปแล้ว จะทำให้เกิดการที่มีข้อมูลที่มีความยาวไม่เท่ากันเกิดขึ้น ซึ่งในชั้นคอนโวลูชันและชั้นการรวมแบบค่ามากสุดนั้น ไม่มีปัญหาใด ๆ เกิดขึ้น แต่สำหรับชั้นการเชื่อมโยงเต็มรูปแบบ จะไม่สามารถรองรับข้อมูลขนาดต่าง ๆ กันได้ เนื่องจากในชั้นนี้ จะมีน้ำหนักของตัวกรองที่มีขนาดที่กำหนดไว้ค่าหนึ่ง ซึ่งเท่ากับขนาดของข้อมูลในชั้นก่อนหน้า หากข้อมูลในชั้นก่อนหน้ามีขนาดไม่คงที่ จะทำให้ไม่สามารถสร้างเส้นเชื่อมของชั้นการเชื่อมโยงเต็มรูปแบบได้

ผู้วิจัยจึงได้เสนอการนำวิธีการรวมอีกรูปแบบหนึ่งมาใช้งาน คือการใช้การรวมแบบเคค่ามากสุด ซึ่งได้กล่าวถึงไปในบทที่ 2.3.2.2 โดยการรวมแบบเคค่ามากสุดจะต่างกับการรวมแบบค่ามากสุด คือ การรวมแบบเคค่ามากสุดจะไม่ได้ใช้ขอบเขตที่มีความยาวจำกัด แต่จะใช้ขอบเขตทั้งหมดของทั้งฟีเจอร์เวกเตอร์ และทำการเลือกข้อมูลที่มีค่ามากที่สุดมาจำนวน k ตัว ซึ่งเป็นจำนวนที่กำหนดไว้ก่อนแล้ว รูปที่ 3.6 แสดงถึงการรวมแบบเคค่ามากสุดที่คำหนด k = 2 จะสังเกตได้ว่าขอบเขตที่สนใจนั้นจะครอบคลุมเท่ากับความยาวของข้อมูล และจะมีการเลื่อนขอบเขตไปตามฟีเจอร์เวกเตอร์ต่าง ๆ รูปที่ 3.7 แสดงรหัสเทียมของการทำการรวมแบบเคค่ามากสุด



รูปที่ 3.6 การรวมแบบเคค่ามากสุด โดยกำหนด k = 2

ข้อมูลรับเข้า:

M: เมทริกซ์ของข้อมูลรับเข้า ขนาด $f \times l$ โดยที่ f คือจำนวนฟีเจอร์ของข้อมูลรับเข้า และ l คือความยาวของข้อมูลรับเขา

k: ค่าเค หรือจำนวนที่ต้องการใช้ในการเลือกข้อมูล

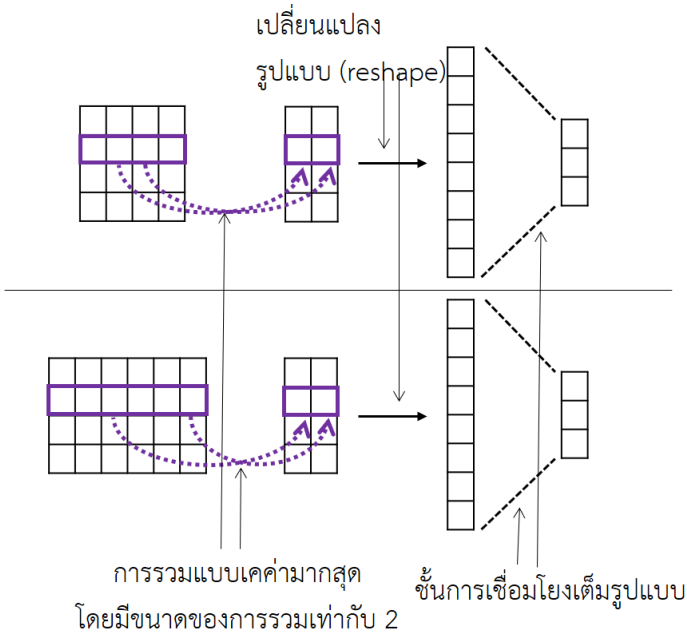
ข้อมูลส่งออก:

O: เมทริกซ์ผลลัพธ์ขนาด $f \times k$

1: O = new matrix with dimension $f \times k$
2: for i = 1 to f
3: select most maximum value $v_1, v_2, \ldots, v_k$ from $M_{i,}$ with a retaining order
4: $O_{i,} = v_1, v_2, \ldots, v_k$
5: return O

รูปที่ 3.7 รหัสเทียมของการทำการรวมแบบเคค่ามากสุด

รูปที่ 3.8 แสดงถึงการนำชั้นการรวมแบบเคค่ามากสุดเข้าไปใช้ นั่นคือ จะมีการใช้ชั้นการรวมแบบเคค่ามากสุดในชั้นก่อนหน้าชั้นการเชื่อมโยงเต็มรูปแบบ ผลของการใช้ชั้นการรวมแบบเคค่ามากสุดจะทำให้ผลลัพธ์ของการรวมมีขนาดที่เท่ากันเสมอ ทำให้สามารถสร้างเส้นเชื่อมของนิวรอลเน็ตเวิร์กในชั้นการรวมอย่างเต็มรูปแบบซึ่งมีขนาดคงที่ได้ จากรูปด้านบนและด้านล่างจะเป็นการเปรียบเทียบขนาดของข้อมูลที่มีความต่างกัน แต่สามารถใช้เน็ตเวิร์กเดียวกันในการจำแนกได้



รูปที่ 3.8 การใช้ชั้นการรวมแบบเคค่ามากสุดและขนาดของผลลัพธ์ เปรียบเทียบระหว่างข้อมูลสองชุดที่ความยาวไม่เท่ากัน

ตารางที่ 3.1 เป็นการสรุปโครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีการปรับปรุงโดยการไม่กำหนดความยาวของข้อมูลตั้งต้นและการใช้การรวมแบบเคค่ามากสุด ทั้งนี้ ในการทดลองยังมีการเปลี่ยนจากการใช้ฟังก์ชันขีดแบ่งเป็นฟังก์ชันเรคติไฟต์เชิงเส้นอีกด้วย โดยจำนวนตัวอักษรที่นำมาเป็นฟีเจอร์ จะรวมตัวอักษรภาษาไทยเข้าไปด้วย ทำให้จำนวนทั้งหมดกลายเป็น 151 ตัวอักษร จะสังเกตได้ว่า ขนาดตัวกรองและจำนวนตัวกรองในชั้นคอนโวลูชัน ขนาดของการรวม และจำนวนนิวรอลในชั้นการเชื่อมโยงเต็มรูปแบบยังมีค่าเท่ากับเน็ตเวิร์กดั้งเดิมที่มีการจำกัดความยาวของข้อมูลรับเข้า ทั้งนี้ ในการเรียนรู้ของเน็ตเวิร์กยังคงใช้วิธีการแพร่กระจายย้อนกลับโดยใช้ฟังก์ชันลบลอการิทึมของความเป็นไปได้เช่นเดิม

ตารางที่ 3.1 โครงสร้างของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่ปรับปรุงแล้ว

ประเภทของชั้น	**พารามิเตอร์**	**ขนาดของผลลัพธ์**
ชั้นข้อมูลรับเข้า	ความยาวตัวอักษร = l_0 จำนวนตัวอักษรทั้งหมดที่สนใจ = 151	$151 \times l_0$
ชั้นคอนโวลูชันและฟังก์ชันเรคติไฟต์เชิงเส้น	ขนาดตัวกรอง = 7 จำนวนตัวกรอง = 256	$256 \times l_1$, โดย $l_1 = l_0 - 6$
ชั้นการรวม	ขนาดของการรวม = 3	$256 \times l_2$, โดย $l_2 = \left\lfloor \frac{l_1}{3} \right\rfloor$
ชั้นคอนโวลูชันและฟังก์ชันเรคติไฟต์เชิงเส้น	ขนาดตัวกรอง = 7 จำนวนตัวกรอง = 256	$256 \times l_3$, โดย $l_3 = l_2 - 6$
ชั้นการรวม	ขนาดของการรวม = 3	$256 \times l_4$, โดย $l_4 = \left\lfloor \frac{l_3}{3} \right\rfloor$
ชั้นคอนโวลูชันและฟังก์ชันเรคติไฟต์เชิงเส้น	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	$256 \times l_5$, โดย $l_5 = l_4 - 2$
ชั้นคอนโวลูชันและฟังก์ชันเรคติไฟต์เชิงเส้น	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	$256 \times l_6$, โดย $l_6 = l_5 - 2$
ชั้นคอนโวลูชันและฟังก์ชันเรคติไฟต์เชิงเส้น	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	$256 \times l_7$, โดย $l_7 = l_6 - 2$
ชั้นคอนโวลูชันและฟังก์ชันเรคติไฟต์เชิงเส้น	ขนาดตัวกรอง = 3 จำนวนตัวกรอง = 256	$256 \times l_8$, โดย $l_8 = l_7 - 2$
ชั้นการรวม	ขนาดของการรวม = 3	256×34

ชั้นการเชื่อมโยงเต็มรูปแบบและฟังก์ชันเรคติไฟต์เชิงเส้น	จำนวนนิวรอล = 1024	1024
ชั้นดรอปเอาท์	ความน่าจะเป็น = 0.5	1024
ชั้นการเชื่อมโยงเต็มรูปแบบและฟังก์ชันเรคติไฟต์เชิงเส้น	จำนวนนิวรอล = 1024	1024
ชั้นดรอปเอาท์	ความน่าจะเป็น = 0.5	1024
ชั้นการเชื่อมโยงเต็มรูปแบบ	จำนวนนิวรอล = 5 (จำนวนประเภทที่ต้องการจำแนก)	5

บทที่ 4
การทดลองและผลการทดลอง

วิทยานิพนธ์ฉบับนี้ ได้แบ่งการทดลองออกเป็น 3 ส่วน เพื่อแสดงให้เห็นถึงผลลัพธ์ของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่นำเสนอขึ้นมาเมื่อเปรียบเทียบกับวิธีที่มีอยู่ก่อนหน้าแล้ว รวมถึงผลลัพธ์ของการสร้างเวกเตอร์ของคำในภาษาไทย โดยแบ่งออกได้ดังนี้ 1) การทดลองเปรียบเทียบผลของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่เสนอขึ้นมา กับวิธีการนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรดั้งเดิม 2) การเปรียบเทียบผลของวิธีการที่เสนอขึ้นมา กับวิธีการที่ได้รับความนิยมซึ่งใช้ข้อมูลนำเข้าเป็นข้อมูลระดับคำ และ 3) ผลของการสร้างคลังเวกเตอร์ของคำ ทั้งนี้ ในการเปรียบเทียบผลลัพธ์จะเน้นในแง่มุมของประสิทธิภาพของวิธีการต่าง ๆ

4.1 ระบบที่ใช้ในการทดลอง

ระบบที่ใช้ในการทดลอง สามารถสรุปได้ดังนี้

4.1.1 คอมพิวเตอร์ที่ใช้ทำการทดลอง

การทดลองต่อไปนี้ ทำบนเครื่องคอมพิวเตอร์ที่มีหน่วยประมวลผลกลาง Intel Core i3-6100 ความเร็ว 3.7 Ghz มีหน่วยความจำขนาด 16 GB มีหน่วยประมวลผลกราฟฟิกคือ Nvidia GTX960 และมีหน่วยความจำกราฟฟิกขนาด 4 GB โดยใช้ระบบปฏิบัติการณ์ Ubuntu 14.04 64 bits

ทั้งนี้ การทดลองด้วยแบบจำลองนิวรอลเน็ตเวิร์กคอนโวลูชันทั้งระดับตัวอักษรและระดับคำจะมีการใช้หน่วยประมวลผลกราฟฟิกมาช่วย สำหรับแบบจำลองอื่น ๆ จะประมวลผลโดยใช้หน่วยประมวลผลกลางเพียงอย่างเดียว

4.1.2 การเขียนโปรแกรม

สำหรับการเขียนโปรแกรมนิวรอลเน็ตเวิร์กคอนโวลูกชันระดับตัวอักษรทำโดยใช้ภาษา Lua และเฟรมเวิร์ก Torch7 โดยมีการปรับปรุงเพิ่มเติมจากต้นฉบับซึ่งเปิดเป็นโอเพนซอร์ส (open source)

สำหรับโปรแกรมนิวรอลเน็ตเวิร์กคอนโวลูชันพลวัตระดับคำนั้น ได้ใช้โอเพนซอร์สในภาษา Python และเฟรมเวิร์ก Theano

และสำหรับตัวจำแนกแบบอื่น ๆ ได้เขียนขึ้นมาโดยใช้ภาษา Python โดยใช้ไลบรารี scikit learn ทั้งนี้ ผู้สนใจสามารถติดต่อขอซอร์สโค้ดจากผู้วิจัยได้

4.2 ข้อมูลที่ใช้ในการทดลอง

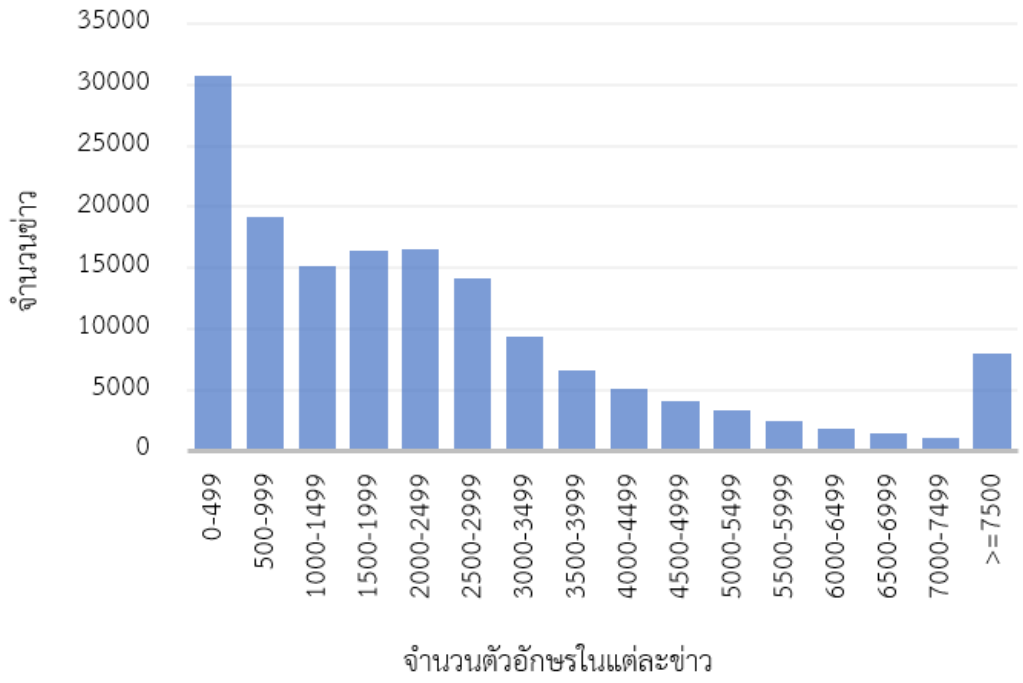
ข้อมูลที่ใช้ในการทดลองเป็นข้อมูลข่าวจากหนังสือพิมพ์ของสำนักข่าวต่าง ๆ ในประเทศไทย ทั้งนี้ ได้ทำการรวบรวมข้อมูล และคัดเลือกเฉพาะข่าวที่เป็นภาษาไทยมาทั้งสิ้นจำนวน 155,000 ข่าว โดยแบ่งออกเป็น 5 หมวดหมู่ที่เท่ากันหมวดหมู่ละ 31,000 ข่าว ตัวอย่างของข่าวในแต่ละหมวดหมู่แสดงได้ดังตารางที่ 4.1

ตารางที่ 4.1 ตัวอย่างของข่าวที่ใช้ในการทดลอง แยกตามหมวดหมู่

หมวดหมู่	ตัวอย่างของเนื้อข่าว
1. อสังหาริมทรัพย์และธุรกิจที่ดิน	เผยคลังเตรียมชงครม.ค่าธรรมเนียมโอน-จดจำนองบ้านเหลือ 0.01% กระตุ้นภาคอสังหาริมทรัพย์ ด้านปลัดคลังคนใหม่พร้อมคลอดมาตรการกระตุ้นภาคธุรกิจต่างๆ ระหว่างที่การส่งออกยังไม่ฟื้นตัว
2. คอมพิวเตอร์และอุปกรณ์	แอปเปิ้ล บริษัทยักษ์แห่งโลกเทคโนโลยีการสื่อสาร เดินหน้าด้านพลังงานทดแทน ด้วยการลงทุน 1.9 พันล้านดอลลาร์สหรัฐ ในการเนรมิตศูนย์ข้อมูลใน "Athenry" ประเทศไอร์แลนด์ และใน "Viborg" ประเทศเดนมาร์ก ให้ขับเคลื่อนด้วยพลังงานทดแทนอย่างเต็มรูปแบบ
3. การเงินและธนาคาร	นางจันทวรรณ สุจริตกุล ผู้ช่วยผู้ว่าการ สายตลาดการเงิน ธนาคารแห่งประเทศไทย (ธปท.) ออกประกาศว่าในเดือน มิ.ย. 2559 ได้ออกประมูลขายพันธบัตรวงเงิน 5 หมื่นล้านบาท อายุ 14 วัน วันที่ 24 มิ.ย. และกำหนดวันชำระเงิน วันที่ 28 มิ.ย. ครบอายุพันธบัตรในวันที่ 12 ก.ค.นี้
4. รัฐธรรมนูญและกฎหมาย	ถึงแม้ว่าคณะกรรมการร่างรัฐธรรมนูญ หรือกรธ.จะยังไม่ตอบรับหรือปฏิเสธ "ข้อเสนอ" ของ คสช.ให้มีบทเฉพาะกาล สรรหาสมาชิกวุฒิสภาจำนวน 250 คนเอาไว้ในร่างรัฐธรรมนูญที่กำลังแก้ไขครั้งสุดท้ายหรือไม่ แต่คำถามก็คือ คสช.ต้องการอะไร อยากให้บ้านเมืองเป็นอย่างไร ถึงได้ขอเวลาให้สมาชิกวุฒิสภาที่มาจากการสรรหานี้อยู่นานถึง 5 ปี
5. กีฬา	กุนซือใหญ่ "จูบิโล อิวาตะ" แย้มให้ความสนใจ อยากคว้าตัว "เมสซี่ เจ" ชนาธิป สรงกระสินธ์ กัปตันทีมชาติไทยชุดปรีโอลิมปิกของบีอีซี เทโรศาสน มาลุยแข้งในศึกเจลีก หลังประทับใจฟอร์มการเล่นที่เข้าตาในศึก ยู-23 ชิงแชมป์เอเชียที่กาตาร์

4.2.1 สถิติในระดับตัวอักษร

จากข่าวที่รวบรวมมา มีความยาวโดยเฉลี่ยของแต่ละข่าวที่ 2,703 ตัวอักษร โดยมีค่าเบี่ยงเบนมาตรฐานที่ 3,807 ตัวอักษร สามารถแสดงฮิสโตแกรมของจำนวนตัวอักษรในข่าวได้ดังรูปที่ 4.1



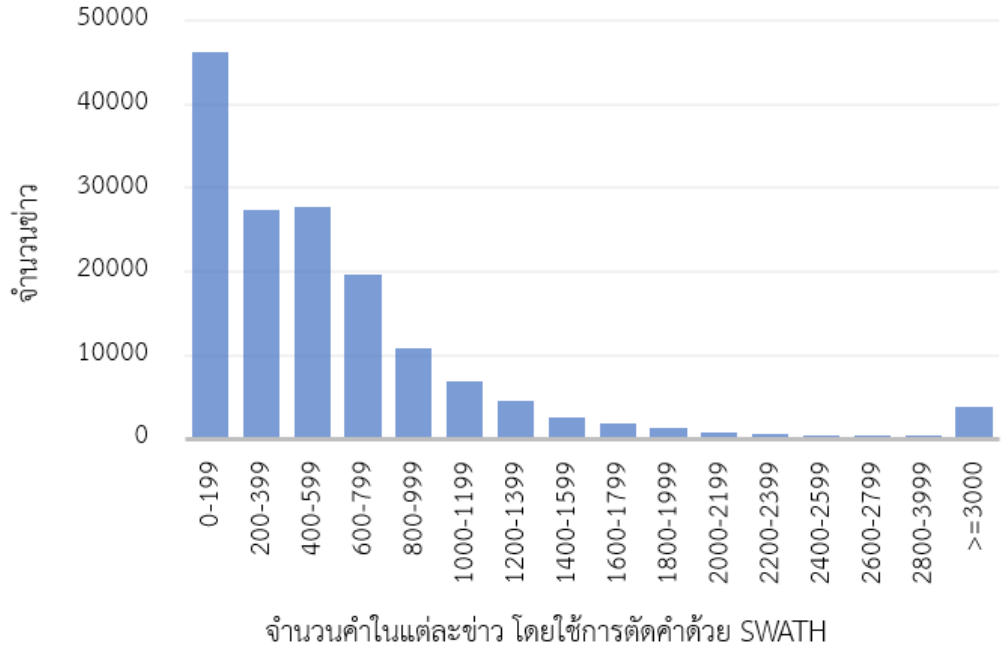
รูปที่ 4.1 ฮิสโตแกรมของจำนวนตัวอักษรในข่าว

4.2.2 สถิติในระดับคำและวิธีการตัดคำ

ในการทดลองที่มีการใช้ข้อมูลรับเข้าในระดับคำ จะทำการทดลองด้วยโปรแกรมที่ใช้ตัดคำ 2 โปรแกรม ได้แก่ SWATH และ Lexto

4.2.2.1 SWATH

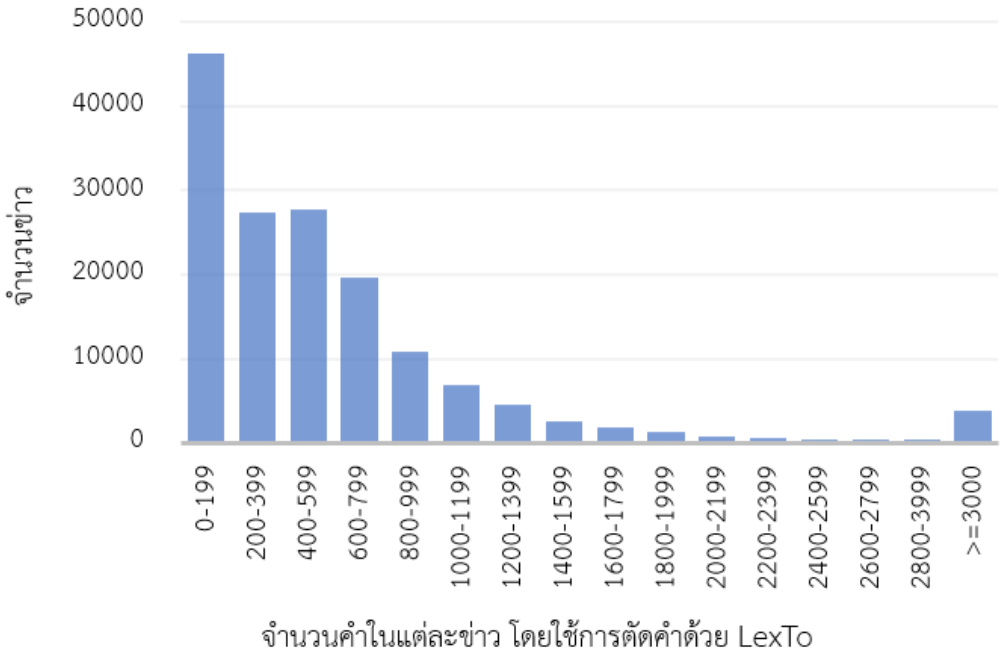
เป็นโปรแกรมตัดคำที่ใช้การเรียนรู้ของเครื่องด้วยวิธีริปเปอร์และวินโนว์มาใช้ในการหาคุณลักษณะของคำ [19] เพื่อเพิ่มประสิทธิภาพของผลลัพธ์ โดยหลังจากการใช้โปรแกรม SWATH กับข้อมูลข่าวแล้ว จะได้ค่าเฉลี่ยของจำนวนคำต่อข่าวเป็น 620 คำ และมีค่าเบี่ยงเบนมาตรฐานที่ 878 คำ และสามารถแสดงฮิสโตแกรมของจำนวนคำได้ดังรูปที่ 4.2



รูปที่ 4.2 ฮิสโตแกรมของจำนวนคำในข่าว ซึ่งทำการตัดคำด้วย SWATH

4.2.2.2 LexTo

เป็นโปรแกรมตัดคำที่ใช้วิธีการเลือกคำที่ยาวที่สุดจากพจนานุกรม สถิติหลังจากใช้โปรแกรม LexTo มาตัดคำพบว่าจะได้ค่าเฉลี่ยของจำนวนคำต่อหนึ่งข่าวที่ 566 คำ และมีค่าเบี่ยงเบนมาตรฐานที่ 878 คำ ทั้งนี้ สามารถแสดงฮิสโตรแกรมของจำนวนคำได้ดังรูปที่ 4.3



รูปที่ 4.3 ฮิสโตแกรมของจำนวนคำในข่าว ซึ่งทำการตัดคำด้วย LexTo

4.2.2.3 ตัวอย่างผลลัพธ์ของการตัดคำ

ตัวอย่างของผลลัพธ์จากการตัดคำด้วยโปรแกรมทั้งสอง แสดงได้ดังตารางที่ 4.2 โดยฝั่งซ้ายคือข้อความที่ใช้โปแกรม SWATH ในการตัดคำ และฝั่งขวาคือข้อความที่ใช้โปรแกรม LexTo ในการตัดคำ

ตารางที่ 4.2 ตัวอย่างผลลัพธ์ของการตัดคำ เปรียบเทียบระหว่างโปรแกรมตัดคำ SWATH และ LexTo

SWATH	LexTo
เผย|คลัง|เตรียม|ชง|ค|รม.ค่า|ธรรมเนียม|โอน-จดจำ|นอง|บ้าน|เหลือ 0.01% กระตุ้น|ภาค|อสังหาริมทรัพย์ ด้าน|ปลัด|คลัง|คน|ใหม่|พร้อม|คลอด|มาตรการ|กระตุ้น|ภาค|ธุรกิจ|ต่างๆ ระหว่าง|ที่|การ|ส่ง|ออก|ยัง|ไม่|ฟื้นตัว	เผย|คลัง|เตรียม|ชง|ครม.|ค่าธรรมเนียม|โอน|-|จดจำ|นอง|บ้าน|เหลือ 0.01|% กระตุ้น|ภาค|อสังหาริมทรัพย์ ด้าน|ปลัด|คลัง|คน|ใหม่|พร้อม|คลอด|มาตรการ|กระตุ้น|ภาค|ธุรกิจ|ต่างๆ ระหว่าง|ที่|การส่งออก|ยัง|ไม่|ฟื้นตัว
แอปเปิ้ล บริษัท|ยักษ์|แห่ง|โลก|เทคโนโลยี|การ|สื่อสาร เดิน|หน้า|ด้าน|พลังงาน|ทดแทน ด้วย|การ|ลงทุน 1.9 พัน|ล้าน|ดอลลาร์|สหรัฐ ใน|การ|เนรมิต|ศูนย์|ข้อมูล|ใน ""Athenry"" ประเทศ|ไอร์แลนด์ และ|ใน ""Viborg"" ประเทศ|เดนมาร์ก ให้|ขับ|เคลื่อน|ด้วย|พลังงาน|ทดแทน|อย่าง|เต็ม|รูปแบบ	แอปเปิ้ล บริษัท|ยักษ์|แห่ง|โลก|เทคโนโลยี|การสื่อสาร เดินหน้า|ด้าน|พลังงาน|ทดแทน ด้วย|การลงทุน 1.9 พัน|ล้าน|ดอลลาร์|สหรัฐ ใน|การ|เนรมิต|ศูนย์|ข้อมูล|ใน ""|Athenry|"" ประเทศ|ไอร์แลนด์ และ|ใน ""|Viborg|"" ประเทศ|เดนมาร์ก ให้|ขับเคลื่อนด้วย|พลังงาน|ทดแทน|อย่างเต็มรูปแบบ
นาง|จัน|ทวรรณ สุจริต|กุล ผู้|ช่วย|ผู้|ว่า|การ สาย|ตลาด|การเงิน ธนาคาร|แห่ง|ประเทศ|ไทย (ธปท.) ออก|ประกาศ|ว่าใน|เดือน มิ.ย. 2559 ได้|ออก|ประมูล|ขาย|พันธบัตร|วงเงิน 5 หมื่น|ล้าน|บาท อายุ 14 วัน วัน|ที่ 24 มิ.ย. และ|กำหนด|วัน|ชำระ|เงิน วัน|ที่ 28 มิ.ย. ครบ|อายุ|พันธบัตร|ใน|วัน|ที่ 12 ก.ค.นี้	นาง|จันท|วรรณ สุจริต|กุล ผู้ช่วย|ผู้ว่าการ สาย|ตลาด|การเงิน ธนาคารแห่งประเทศไทย (|ธปท.|) ออก|ประกาศ|ว่า|ใน|เดือน มิ.ย. 2559 ได้|ออก|ประมูล|ขาย|พันธบัตร|วงเงิน 5 หมื่น|ล้าน|บาท อายุ 14 วัน วันที่ 24 มิ.ย. และ|กำหนด|วัน|ชำระเงิน วันที่ 28 มิ.ย. ครบ|อายุ|พันธบัตร|ใน|วันที่ 12 ก.ค.|นี้

จากการที่ LexTo ใช้วิธีการเลือกคำที่ยาวที่สุดที่มีในพจนานุกรม ส่งผลให้ LexTo มีคำที่ยาวกว่าอย่างเช่น “อย่างเต็มรูปแบบ” แต่ผลลัพธ์จาก SWATH จะโดนตัดคำออกเป็น “อย่าง”, “เต็ม”

และ “รูปแบบ” สอดคล้องกับค่าเฉลี่ยของจำนวนคำต่อข่าวซึ่งผลลัพธ์จาก LexTo จะมีจำนวนคำต่อข่าวน้อยกว่าผลลัพธ์จาก SWATH

4.2.3 การแบ่งข้อมูล

ในการทดลอง ข้อมูลทั้งหมด 31,000 ข่าวในแต่ละหมวดหมู่ จะถูกแบ่งออกเป็น 23,000 ข่าวสำหรับการเรียนรู้ของตัวจำแนก (training set) 3,000 ข่าวเป็นข้อมูลทวนสอบ (validation set) และ 5,000 ข่าวสำหรับเป็นข้อมูลทดสอบ (testing set)

4.3 ผลการทดลองเปรียบเทียบกับนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีความยาวของข้อมูลนำเข้าคงที่

การทดลองนี้ จะทำการเปรียบเทียบแบบจำลองที่ได้ทำการเสนอขึ้นมา คือ นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีความยาวของข้อมูลไม่จำกัด กับแบบจำลองที่มีอยู่ก่อนหน้า คือ นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีความยาวของข้อมูลคงที่

และเนื่องจากข้อจำกัดของหน่วยความจำในหน่วยประมวลผลกราฟฟิก การทดลองสำหรับนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีความยาวของข้อมูลไม่จำกัดนั้น จะกำหนดความยาวมากที่สุดที่ 1014, 2000, 4000 และ 6000 ตัวอักษร ส่วนนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีความยาวของข้อมูลคงที่นั้น จะอ้างอิงตามแบบจำลองดั้งเดิมที่ 1014 ตัวอักษร

ในการเรียนรู้ของนิวรอลเน็ตเวิร์กนั้น ได้ใช้วิธีการ stochastic gradient descent ที่มีอัตราการเรียนรู้ที่ 0.005 และมี โมเมนตัม 0.9 ทั้งนี้ อัตราการเรียนรู้จะลดลงครึ่งหนึ่งทุก ๆ 3 epochs ซึ่งแต่ละ epoch จะประกอบด้วย 10,000 มินิแบท และในแต่ละมินิแบท จะมีข้อมูลจำนวน 32 ข้อมูล

สำหรับตัวอักษรซึ่งจะถูกนำไปแปลงเป็นเวกเตอร์วันฮอทนั้น จะประกอบด้วย 70 ตัวอักษรจากแบบจำลองต้นแบบ คือ

“abcdefghijklmnopqrstuvwxyz0123456789-,;.!?:'\"/\\|_@#$%^&~`+=<>()[]{}”*

รวมกับอีก 81 ตัวอักษรในภาษาไทยที่เพิ่มเข้าไป โดยตารางที่ 4.3 แสดงถึงตัวอักษรภาษาไทยตามยูนิโคด โดยตัวอักษรที่ไม่มีขีดฆ่าทั้งหมดจะถูกนำไปใช้เป็นฟีเจอร์ระดับตัวอักษรเพื่อสร้างวันฮอทเวกเตอร์และข้อมูลรับเข้าต่อไป

ตารางที่ 4.3 ตารางแสดงตัวอักษรในภาษาไทยตามยูนิโคด ตัวอักษรที่ไม่มีขีดฆ่าทั้งสิ้น 81 ตัวจะถูกนำมาใช้ในการสร้างวันฮอทเวกเตอร์ร่วมกับตัวอักษรดั้งเดิมในภาษาอังกฤษ

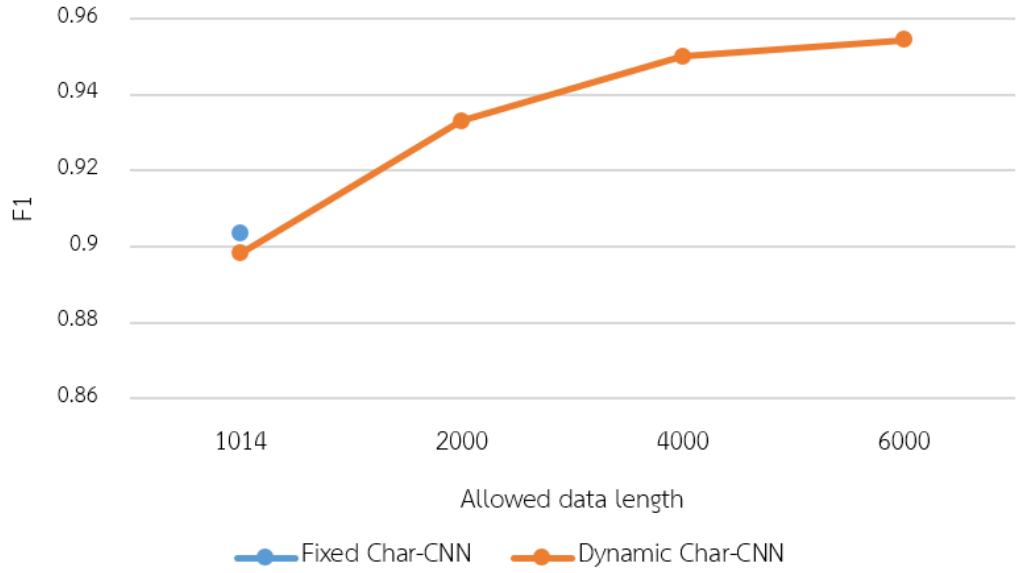
ยูนิโคด	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0E0x		ก	ข	ฃ	ค	ฅ	ฆ	ง	จ	ฉ	ช	ซ	ฌ	ญ	ฎ	ฏ
U+0E1x	ฐ	ฑ	ฒ	ณ	ด	ต	ถ	ท	ธ	น	บ	ป	ผ	ฝ	พ	ฟ
U+0E2x	ภ	ม	ย	ร	ฤ	ล	ฦ	ว	ศ	ษ	ส	ห	ฬ	อ	ฮ	ฯ
U+0E3x	ะ	ั	า	ำ	ิ	ี	ึ	ื	ุ	ู	ฺ					฿
U+0E4x	เ	แ	โ	ใ	ไ	ๅ	ๆ	็	่	้	๊	๋	์	๎	๏	๐
U+0E5x	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙	๚	๛				

จากผลการทดลองในตารางที่ 4.4 เป็นการเปรียบเทียบนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรซึ่งได้เสนอในงานวิจัยฉบับนี้ เปรียบเทียบกับนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรแบบดั้งเดิม ผลปรากฏว่าเน็ตเวิร์กที่ได้เสนอนั้น ให้ความแม่นยำที่ดีกว่าเน็ตเวิร์กแบบดั้งเดิม ทั้งนี้ เน็ตเวิร์กที่สามารถรองรับข้อมูลรับเข้าที่ความยาวมากกว่า จะให้ผลดีกว่าเน็ตเวิร์กที่รองรับความยาวของข้อมูลที่สั้นกว่า

ตารางที่ 4.4 ผลการทดลองเปรียบเทียบระหว่างนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีความยาวข้อมูลรับเข้าคงที่ กับนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่มีข้อมูลรับเข้าความยาวใดก็ได้

วิธีการ	Accuracy	F_1
Fixed Char-CNN (กำหนดความยาวข้อมูลที่ 1014)	90.35%	0.9035
Dynamic Char-CNN (กำหนดความยาวข้อมูลมากที่สุดที่ 1014)	89.83%	0.8982
Dynamic Char-CNN (กำหนดความยาวข้อมูลมากที่สุดที่ 2000)	93.31%	0.9331
Dynamic Char-CNN (กำหนดความยาวข้อมูลมากที่สุดที่ 4000)	95.01%	0.9500
Dynamic Char-CNN (กำหนดความยาวข้อมูลมากที่สุดที่ 6000)	**95.44%**	**0.9544**

รูปที่ 4.4 เป็นการนำเสนอผลการทดลองในรูปแบบกราฟ โดยแกนนอนแสดงถึงความยาวสูงสุดที่เน็ตเวิร์กรองรับ จากกราฟ จะเห็นได้ว่า เน็ตเวิร์กที่รองรับข้อความที่มีความยาวมาก จะยิ่งมีประสิทธิภาพในการจำแนกประเภทของข้อความสูงตามไปด้วย



รูปที่ 4.4 ผลการทดลองในรูปแบบของกราฟเส้นเพื่อแสดงถึงผลของการเพิ่มความยาวที่เน็ตเวิร์กรองรับ

4.4 ผลการทดลองเปรียบเทียบกับตัวจำแนกระดับคำ

ในการทดลองนี้ จะใช้ตัวจำแนกระดับคำได้แก่ ซัพพอร์ตเวกเตอร์แมชชีน นาอีฟเบย์ แมกซิมัมเอนโทรปี และนิวรอลเน็ตเวิร์กคอนโวลูชันระดับคำ ซึ่งในที่นี้ จะเลือกใช้เป็นนิวรอลเน็ตเวิร์กพลวัตในการทดลอง [12] โดยข้อมูลรับเข้าที่ใช้จะมีทั้งรูปแบบถุงคำ และทีเอฟไอดีเอฟ

สำหรับนิวรอลเน็ตเวิร์กพลวัตจะมีการรับข้อมูลที่อยู่ในรูปแบบลำดับของเวกเตอร์ของคำ โดยจะมีการสร้างเวกเตอร์เริ่มต้นโดยใช้โปรแกรม word2vec ด้วย skip-gram โดยกำหนดความยาวของเวกเตอร์ที่ 48 ซึ่งเป็นความยาวที่มีประสิทธิภาพมากที่สุดที่ได้ทดสอบกับข้อมูลภาษาไทยชุดอื่นดังผลงานตีพิมพ์ในภาคผนวก ก สำหรับในชั้นคอนโวลูชันแรกจะใช้ขนาดตัวกรองเป็น 10 โดยมีจำนวนทั้งหมด 6 ตัวกรอง ส่วนในชั้นคอนโวลูชันที่สอง จะใช้ขนาดตัวกรองเป็น 7 และมีจำนวนทั้งหมด 12 ตัวกรอง สำหรับค่า k ในชั้นการรวมชั้นสุดท้าย จะใช้เป็น 5 โดยพารามิเตอร์เหล่านี้ถูกกำหนดให้เหมือนกับพารามิเตอร์จากการทดลองชุดหนึ่งใน [12]

ตารางที่ 4.5 แสดงถึงผลการทดลองการเปรียบเทียบกับตัวจำแนกที่ต้องใช้การตัดคำ โดยทำการตัดคำด้วยโปรแกรม SWATH สำหรับตารางที่ 4.6 จะแสดงถึงผลการทดลองโดยใช้โปรแกรม LexTo ในการตัดคำ

ตารางที่ 4.5 ผลการทดลองของตัวจำแนกที่ต้องใช้การตัดคำโดยใช้โปรแกรม SWATH โดยมีการเปรียบเทียบระหว่างตัวจำแนกในระดับคำ และนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร

วิธีการ	Accuracy	F_1
Naïve Bayes + BoW	87.23%	0.8715
Naïve Bayes + TFIDF	89.00%	0.8896
Maximum Entropy + BoW	94.87%	0.9487
Maximum Entropy + TFIDF	94.74%	0.9474
SVM +BoW	93.78%	0.9379
SVM + TFIDF	95.25%	0.9524
DCNN	**95.96%**	**0.9595**
Dynamic Char-CNN	95.44%	0.9544

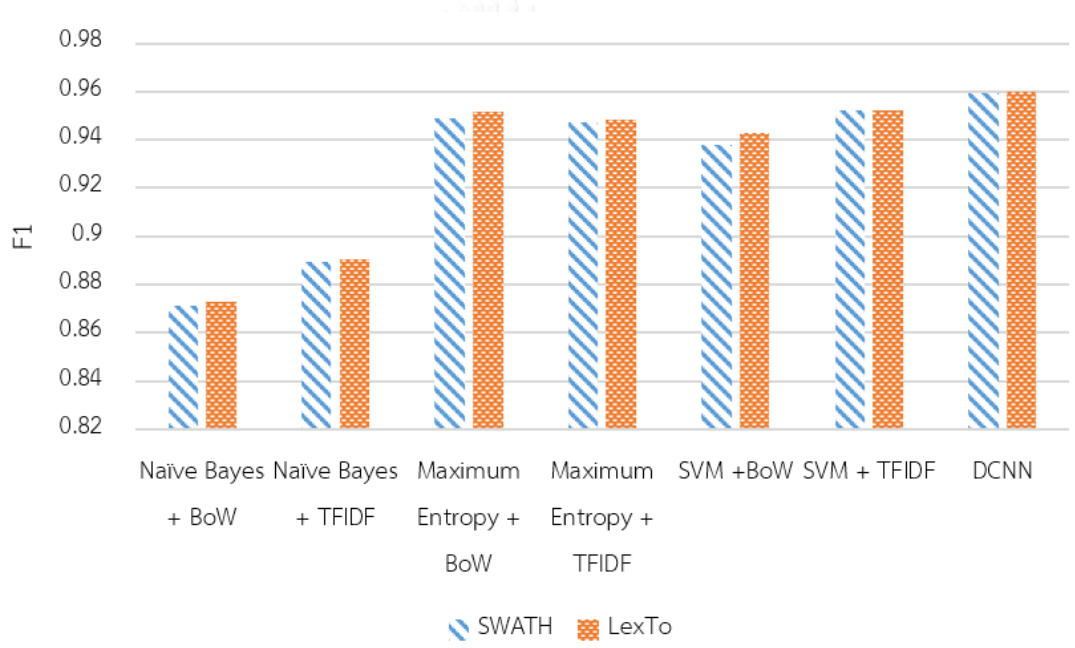
ตารางที่ 4.6 ผลการทดลองของตัวจำแนกที่ต้องใช้การตัดคำโดยใช้โปรแกรม LexTo โดยมีการเปรียบเทียบระหว่างตัวจำแนกในระดับคำ และนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร

วิธีการ	Accuracy	F_1
Naïve Bayes + BoW	87.36%	0.8727
Naïve Bayes + TFIDF	89.12%	0.8907
Maximum Entropy + BoW	95.18%	0.9517
Maximum Entropy + TFIDF	94.84%	0.9484
SVM +BoW	94.26%	0.9426
SVM + TFIDF	95.21%	0.9520
DCNN	**96.02%**	**0.9601**
Dynamic Char-CNN	95.44%	0.9544

จากตารางข้างต้น ทั้งการทดลองโดยใช้โปรแกรม SWATH และ LexTo จะเห็นได้ว่า นิวรอลเน็ตเวิร์คอนโวลูชันระดับคำแบบพลวัตเป็นวิธีการที่ดีที่สุดในการจำแนกประเภทข้อความ ทั้งนี้ วิธีนาอีฟเบย์ เป็นวิธีที่ให้ค่าความแม่นยำน้อยที่สุด และน้อยกว่าวิธีการอื่นค่อนข้างมาก

เมื่อเปรียบเทียบกับวิธีการระดับคำแบบต่าง ๆ ผลปรากฏว่าวิธีการที่เสนอให้ความแม่นยำที่ดีกว่าการจำแนกประเภทข้อความด้วยนาอีฟเบย์ แมกซิมัมเอนโทรปี และซัพพอร์ตเวกเตอร์แมชชีน แต่ทั้งนี้ วิธีที่เสนอจะให้ความแม่นยำน้อยกว่าวิธีการนิวรอลเน็ตเวิร์กคอนโวลูชันที่ใช้ฟีเจอร์ระดับคำ

รูปที่ 4.5 แสดงถึงผลการทดลองในรูปแบบกราฟแท่ง โดยมีการเปรียบเทียบระหว่างวิธีการตัดคำด้วย SWATH และ LexTo จะสังเกตได้ว่าการตัดคำด้วยโปรแกรม LexTo จะสามารถนำไปใช้ในการจำแนกประเภทข้อความได้อย่างมีประสิทธิภาพมากกว่าการตัดคำด้วย SWATH นอกจากนี้ เมื่อเปรียบเทียบประสิทธิภาพของการแทนข้อความด้วยถุงคำและทีเอฟไอดีเอฟแล้ว จะพบว่าการแทนข้อความด้วยทีเอฟไอดีเอฟจะให้ความแม่นยำที่สูงกว่า โดยเฉพาะเมื่อใช้ตัวจำแนกเป็นนาอีฟเบย์หรือซัพพอร์ตเวกเตอร์แมชชีน



รูปที่ 4.5 ผลการทดลองในรูปแบบของกราฟแท่งเพื่อเปรียบเทียบประสิทธิภาพของตัวจำแนกแบ่งตามโปรแกรมตัดคำ

4.5 การสร้างคลังเวกเตอร์ของคำในภาษาไทย

จากการเก็บข้อมูลข่าวภาษาไทยทั้งสิ้น 155,000 ข่าวนั้น เป็นข้อมูลภาษาไทยที่สามารถนำไปใช้ต่อยอดได้ จึงได้นำข้อมูลเหล่านั้นมาสร้างเป็นเวกเตอร์ของคำเพื่อการนำไปใช้งานต่อไปในภายภาคหน้าได้ ในการสร้างเวกเตอร์ของคำเหล่านี้จะกำหนดให้เวกเตอร์มีขนาด 48 ตารางที่ 4.7 แสดงถึงผลลัพธ์บางส่วนของคลังของเวกเตอร์คำที่ได้ แบ่งตามวิธีการตัดคำ

ตารางที่ 4.7 ผลการสร้างคลังเวกเตอร์ของคำในภาษาไทย โดยมีการทดสอบคุณสมบัติต่าง ๆ เปรียบเทียบระหว่างการตัดคำด้วยโปรแกรม SWATH และ LexTo

คุณสมบัติ \ วิธีการตัดคำ	**SWATH**	**LexTo**
จำนวนคำทั้งหมด	625,265	179,658
ขนาดของเวกเตอร์	48	48
เวกเตอร์ของคำว่า “ฉัน”	[0.0663, 0.0747, 0.0374, -0.3962, -0.0787, 0.0930, 0.0385, -0.1656, 0.1196, -0.1401, -0.0177, -0.0924, -0.1071, -0.0385, 0.0053, -0.0063, 0.0016, -0.0814, -0.1295, 0.0324, -0.0593, 0.2455, -0.2230, -0.0420, 0.0818, 0.0551, 0.0145, 0.1308, 0.1059, 0.1338, 0.2496, 0.2656, -0.3505, -0.0442, 0.2550, 0.1422, -0.0492, 0.1345, -0.0001, 0.0134, 0.2185, -0.0970, -0.0995, -0.1703, -0.0403, 0.2606, 0.0148, -0.0826]	[0.1903, 0.1761, 0.0349, -0.1493, -0.0736, 0.1698, 0.0356, -0.1809, -0.0138, 0.0785, 0.0491, 0.1145, -0.0688, 0.1425, -0.0610, -0.0136, 0.1951, -0.0784, 0.2260, -0.0601, -0.0722, 0.2415, -0.2419, -0.1376, -0.0562, 0.1127, 0.1039, -0.1187, 0.1042, -0.1070, -0.0611, -0.0359, -0.2649, 0.1041, 0.1588, 0.0321, 0.3339, 0.0465, -0.0045, 0.0158, 0.3194, 0.1176, -0.2431, -0.1374, 0.0444, 0.0691, 0.0684, -0.2363]
คำที่มีค่า cosine similarity กับคำว่า “ฉัน” มากที่สุด 10 อันดับ พร้อมค่า cosine similarity	เธอ, 0.8155 ผม, 0.7723 ใครๆ, 0.7712 เถิด, 0.7490 ค่ะ, 0.7390 เจ้านาย, 0.7354 คุณ, 0.7309 พ่อ, 0.7284	คุณ, 0.7758 พวกคุณ, 0.7547 คุณพ่อ, 0.7539 อย่างไรเล่า, 0.7522 อั๊ว, 0.7408 เธอ, 0.7402 คุณตา, 0.7323 พ่อ, 0.7272

	แหละ, 0.7256 กระผม, 0.7253	คุณแม่, 0.7150 พ่อคุณ, 0.7134
ผลลัพธ์คำที่ใกล้เคียงที่สุดของเวกเตอร์ “นาย” – (“ชาย” – “หญิง”) และค่า cosine similarity	"นาย, 0.2695 นาง, 0.2399 นูญถาวร, 0.2320 นอก-นาย, 0.2248 ธนภา, 0.2233 ตู่-ทั้ง, 0.2215 www.facebook.com/ betagrofoodsafetysociety, 0.2209 รัฐมนตรี-นาย, 0.2194 €ดร.แก้ว€, 0.2187 โอ-ปัฏ, 0.2175	พ.ต.ท., 0.2451 นาง, 0.2308 พล.ต.ต., 0.2276 พล.ต.ท., 0.2251 รศ.ดร., 0.2206 น.ส., 0.2181 น.พ., 0.2180 พล.ท., 0.2143 พ.ต.อ., 0.2126 นพ., 0.2064

จากผลลัพธ์ที่ได้ของเวกเตอร์ของคำ จะเห็นได้ว่าสามารถสร้างคลังของเวกเตอร์ที่สามารถนำไปใช้ประโยชน์ต่อไปในอนาคตได้ โดยเวกเตอร์ที่มีความหมายใกล้เคียงกัน จะมีเวกเตอร์ที่มีค่าใกล้เคียงกันด้วย ทั้งนี้ วิธีการตัดคำในภาษาไทยที่ต่างกัน จะทำให้ได้ผลลัพธ์ที่เวกเตอร์คำที่ต่างกันด้วย

สำหรับผลลัพธ์ของเวกเตอร์คำที่ใกล้เคียงเวกเตอร์ “นาย” – (“ชาย” – “หญิง”) ที่สุดนั้น พบว่า วิธีการตัดคำโดยใช้ LexTo จะให้ผลที่แม่นยำกว่า นั่นคือ เจอคำว่า “นาง” ในอันดับที่ 2 และเจอคำว่า “น.ส.” ในอันดับที่ 6 ทั้งนี้ สาเหตุที่การตัดคำโดยใช้ SWATH ได้ผลลัพธ์ที่ไม่ดีในเรื่องนี้สามารถมีสาเหตุได้จากการตัดคำที่ไม่เหมาะสม เช่น การตัดคำว่า “"นาย” โดยมีเครื่องหมายฟันหนูติดไปด้วย

บทที่ 5
สรุปการวิจัยและแนวทางการวิจัยในขั้นถัดไป

5.1 สรุปการวิจัย

วิทยานิพนธ์ชิ้นนี้ ได้เสนอวิธีการจำแนกประเภทข้อความในภาษาไทยโดยไม่ต้องมีการตัดคำ ด้วยการใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร ซึ่งได้ทำการปรับปรุงแบบจำลองเพื่อให้มีประสิทธิภาพในการจำแนกที่ดีขึ้น โดยการทำให้เน็ตเวิร์กสามารถรับข้อมูลความยาวเท่าใดก็ได้ และยังคงใช้จำนวนพารามิเตอร์ขนาดเท่าเดิม นิวรอลเน็ตเวิร์กที่เสนอนั้นจะกระทำกับข้อมูลที่มีความยาวไม่เท่ากันในชั้นคอนโวลูชันและชั้นการรวม ในขณะที่ก่อนถึงชั้นการเชื่อมโยงเต็มรูปแบบ จะใช้การรวมแบบเคค่ามากสุดทำให้ความยาวของข้อมูลเท่ากัน และสามารถเข้าสู่เน็ตเวิร์กส่วนจำแนกประเภทได้

จากการทดสอบวิธีการที่เสนอขึ้นมากับข้อมูลข่าวในภาษาไทยพบว่า เน็ตเวิร์กที่เสนอนั้นมีความแม่นยำในการจำแนกข้อความมากกว่านิวรอลเน็ตเวิร์กระดับตัวอักษรแบบดั้งเดิม นอกจากนี้วิธีการที่เสนอยังให้ผลลัพธ์ที่ดีกว่าวิธีการแบบดั้งเดิมที่ต้องใช้การตัดคำ ได้แก่ ซัพพอร์ตเวกเตอร์แมชชีน นาอีฟเบย์ และแมกซิมัมเอนโทรปี ยกเว้นเพียงแต่วิธีนิวรอลเน็ตเวิร์กคอนโวลูชันพลวัตระดับคำ แต่ทั้งนี้ ยังให้ผลอยู่ในระดับที่ใกล้เคียงกัน และเมื่อเปรียบเทียบในด้านอื่น ๆ แล้ว วิธีการที่เสนอจะมีข้อดีคือสามารถทำการจำแนกประเภทของข้อความได้โดยไม่ต้องใช้การตัดคำ แต่จะต้องใช้หน่วยความจำที่มากกว่านิวรอลเน็ตเวิร์กคอนโวลูชันพลวัตระดับคำเพราะมีการใช้เวกเตอร์ระดับตัวอักษรแทนการใช้เวกเตอร์ระดับคำ

สำหรับประโยชน์ที่ได้จากการใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรมาจำแนกประเภทข้อความภาษาไทยนั้น จะทำให้สามารถลดขั้นตอนลงได้ โดยการไม่จำเป็นต้องทำการตัดคำ นอกจากนี้ วิธีการนิวรอลเน็ตเวิร์กคอนโวลูชันยังให้ความแม่นยำที่ดี ซึ่งเป็นผลมาจากการที่มีการใช้งานข้อมูลรับเข้าที่ยึดตามลำดับของคำดั้งเดิม ซึ่งต่างจากวิธีการที่ใช้ข้อมูลรับเข้าแบบถุงคำหรือทีเอฟไอดีเอฟ ซึ่งไม่มีการใช้งานลำดับของคำ นอกจากนี้ การใช้ข้อมูลในระดับตัวอักษรจะทำให้ประสิทธิภาพของการจำแนกข้อความไม่ขึ้นอยู่กับวิธีการตัดคำอีกด้วย

ในส่วนของข้อมูลข่าวภาษาไทยที่ได้เก็บข้อมูลมานั้น ได้นำมาสร้างคลังเวกเตอร์ของคำในภาษาไทย เพื่อนำไปใช้ประโยชน์ในงานอื่น ๆ ต่อไป

5.2 แนวทางการวิจัยในขั้นถัดไป

ในงานวิจัยชิ้นนี้ ได้ใช้โครงสร้างและจำนวนพารามิเตอร์ต่าง ๆ ของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรที่เลียนแบบมาจากโครงสร้างแบบดั้งเดิม การปรับปรุงจำนวนพารามิเตอร์ รวมถึงลำดับชองชั้นต่าง ๆ จะสามารถช่วยให้วิธีการมีความแม่นยำมากขึ้นได้

ข้อมูลที่นำมาใช้ในการทดลองนี้ เป็นข้อมูลที่ไม่ได้มีขนาดใหญ่มาก ซึ่งจากผลงานวิจัยอื่น ๆ ที่เคยมีมานั้น การเรียนรู้แบบเชิงลึกจะได้ผลดียิ่งขึ้นเมื่อมีขนาดของข้อมูลที่ใหญ่ขึ้น และนอกจากนี้ ในงานวิจัยของนิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรในภาษาอังกฤษแต่เดิมนั้น จะให้ประสิทธิภาพที่ดีกว่านิวรอลเน็ตเวิร์คระดับคำเมื่อข้อมูลมีหลักล้านขึ้นไป การทดสอบกับข้อมูลขนาดใหญ่ขึ้นจะช่วยยืนยันผลลัพธ์ให้มีความชัดเจนมากยิ่งขึ้นได้

และจากการที่แบบจำลองระดับตัวอักษรนั้นมีการเรียนรู้มาจากส่วนย่อยที่สุดที่ประกอบขึ้นมาเป็นคำได้โดยตรง ทำให้ยังมีข้อดีอีกอย่างหนึ่ง คือการที่แบบจำลองนั้นสามารถจะมองว่าคำที่ไม่เหมือนกัน แต่มาจากรากศัพท์เดียวกัน เช่นคำว่า “ระลึก” กับ “รำลึก” มีความหมายที่ใกล้เคียงกันได้ แต่ทั้งนี้คำเหล่านั้นจะต้องมีความคล้ายคลึงกันในแง่ของตัวอักษร และไม่มีความแตกต่างกันมากจนเกินไป และสำหรับงานเขียนที่มีความผิดพลาดเยอะ เช่นการสะกดผิด หรือการพิมพ์ผิด หากใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษรแล้ว จะไม่มีผลเสียจากตัวอักษรที่สะกดผิดมากเท่าการใช้ฟีเจอร์ระดับคำ ผู้วิจัยมีความเห็นว่าข้อดีตรงส่วนนี้ของการจำแนกระดับตัวอักษรสามารถนำไปเป็นแนวทางวิจัยต่อไปได้

รายการอ้างอิง

[1] N. Chirawichitchai, "Emotion classification of Thai text based using term weighting and machine learning techniques," 2014, pp. 91-96.

[2] P. Sarakit, T. Theeramunkong, C. Haruechaiyasak, and M. Okumura, "Classifying emotion in Thai youtube comments," *2015 6th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES),* pp. 1-5, 2015 2015.

[3] K. Sukhum, S. Nitsuwat, and C. Haruechaiyasak, "Opinion Detection in Thai Political News Columns Based on Subjectivity Analysis," 2011.

[4] N. Chirawichitchai, P. Sa-nguansat, and P. Meesad, "Developing an effective Thai Document Categorization Framework base on term relevance frequency weighting," 2010, pp. 19-23.

[5] S. Janpla, "THE EFFECTIVENESS OF AUTOMATED THAI DOCUMENTS CATEGORIZATION BASED ON MACHINE LEARNING," *Journal of Theoretical & Applied Information Technology,* vol. 66, 2014 2014.

[6] C. Haruechaiyasak, A. Kongthon, P. Palingoon, and C. Sangkeettrakarn, "Constructing thai opinion mining resource: A case study on hotel reviews," 2010, pp. 64-71.

[7] T. Nomponkrang and K. Woraratpanya, "Thai-sentence classification using conceptual graph," 2010, pp. V2-479.

[8] C. Haruechaiyasak and A. Kongthon, "Mining associative and comparative patterns for Thai sentiment analysis," 2015, pp. 1-6.

[9] Y. LeCun, L. o. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, pp. 2278-2323, 1998 1998.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," 2012, pp. 1097-1105.

[11] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," 2008, pp. 160-167.

[12] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188,* 2014 2014.

[13] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882,* 2014 2014.

[14] C. N. d. Santos and M. Gatti, "Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts," in *the 25th International Conference on Computational Linguistics*, 2014, pp. 69-78.

[15] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," 2015, pp. 649-657.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013, pp. 3111-3119.

[17] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," 2013, pp. 1532-1543.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research,* vol. 15, pp. 1929-1958, 2014 2014.

[19] P. Charoenpornsawat, "Feature-based Thai Word Segmentation," Master, Computer Engineering, Chulalongkorn University, Bangkok, Thailand, 1999.

ภาคผนวก ก

A Study of Sentiment Analysis Using Deep Learning Techniques on Thai Twitter Data

Peerapon Vateekul, Thanabhat Koomsubha
Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
Peerapon.V@Chula.ac.th, Thanabhat.K@Student.Chula.ac.th

***Abstract*— Sentiment analysis is very important for social listening, especially, when there are millions of Twitter users in Thailand nowadays. Almost all prior works are based on classical classification techniques, e.g., SVM, Naïve Bayes, etc. Recently, the deep learning techniques have shown promising accuracy in this domain on English tweet corpus. In this paper, we propose the first study that applies deep learning techniques to classify sentiment of Thai Twitter data. There are two deep learning techniques included in our study: Long Short Term Memory (LSTM) and Dynamic Convolutional Neural Network (DCNN). A proper data preprocessing has been conducted. Moreover, we also investigate an effect of word orders in Thai tweets. The results show that the deep learning techniques significantly outperform many classical techniques: Naïve Bayes and SVM, except Maximum Entropy.**

Keywords*— *Sentiment Analysis; Thai Twitter Data; Deep Learning; Long Short Term Memory; Dynamic Convolutional Neural Network

I. INTRODUCTION

Social media has been widely used and become an important communication tool since the age of Internet. It is an effective way to spread out information and express opinions. Since many people use social media every day, a large amount of reviews, feedbacks, article have been created. Many organizations use social media to reach out their customers. It is important for organizations to automatically identify each customer review whether it is positive or negative; this is called "sentiment analysis."

Twitter have been created since 2006 and gained popularity nowadays. A sentiment analysis on Twitter was introduced in 2009 [1]. The limitation of 140 characters per tweet makes Twitter easier to classify the sentiment [2]. However, commonly used classical techniques, e.g., Multinomial Naïve Bayes (NB), Support Vector Machine (SVM) and Maximum Entropy (MaxEnt), are based on bag-of-words model which the sequence of words is ignored. This results in inefficient sentiment analysis because the sequence of words can affect the emotion. For example, "bad" and "not" are both negative, but the phrase "not bad" which is composed of these two words has positive meaning.

"Deep Learning" is a deep machine learning architecture consisting of many layers of perceptron inspired by our brain. There have been many success researches reported by employing deep learning in sentiment analysis. Long Short Term Memory (LSTM) and Dynamic Convolutional Neural Network (DCNN) are methods that learn from a sequence of words. Both methods outperform classical methods using bag-of-words on Twitter data [3, 4]. The report also shows that DCNN has a higher accuracy than LSTM. However, most researches were done on English Twitter data. To the best of our knowledge, there have never been any study of deep learning on Thai Twitter data which the meaning of a group of words can depend on the sequence of words, e.g., "โกงความตาย" (comeback) implies positive emotion while each single word, i.e. "โกง" (cheat) and "ความตาย" (death), gives negative sentiment.

In this paper, we aim to study the sentiment analysis on Twitter data in Thai by employing well known deep learning techniques. Our study has three main objectives – (*i*) to study the effect of each parameter on deep neural network, (*ii*) to compare LSTM and DCNN to other methods using bag-of-words, and (*iii*) to investigate how the important of sequence of words in Thai Twitter data. We prepare emotional data by searching the known emoticons in each tweet. We also present the preprocessing step for Thai Twitter data. To illustrate the result, intensively experiments were conducted. The result shows that DCNN is better than LSTM in term of accuracy and both deep learning techniques have higher accuracy compared to classical methods except MaxEnt. Finally, we also show that the sequence of words in Thai is important.

This paper is organized as follows. Section 2 reports the related works. Section 3 describes model of word vector, LSTM and DCNN. Section 4 shows how our Twitter data collection process works. Section 5 describes the experiment and result. Section 6 is a conclusion.

II. RELATED WORK

The sentiment analysis on Twitter data was early adopt in 2009, Go et al. used an automated system to prepare training data. In the labeling process, they divided their collected tweets into two sets, i.e. positive and negative, using predefined emoticon. Tweets containing emoticon ":)" or ":D" were labeled as positive whereas tweets containing emoticon ":(" or ":-(" were labeled as negative. And in the classification process, they used bag-of-words feature classifiers: NB, MaxEnt and SVM with n-gram and part-of-speech. These classifiers defeated the

978-1-5090-2033-1/16/$31.00 ©2016 IEEE

baseline method, which used a set of known keywords to classify tweets.

Neural network is a model for machine learning inspired by human brain. It consists of many neurons that form a large network. In 2003, Bengio et al. used neural network for language modeling and outperformed the state-of-the-art n-grams model [5]. Neural network has a flexible architecture. It can have various number of nodes per layer, with various number of hidden layers and weights connected in between. The more layers a neural network has, the more complex model the network can learn. A neural network with multiple hidden layers is called Deep Learning. However, simple feed forward neural network can not gain a profit by only adding layers because its training process is ineffective [6]. In 2007, Bengio et al. proposed an unsupervised pre-training process called auto-encoders, as it represents a process of encoding large features to smaller features. They found that the model with unsupervised pre-training weights surpasses the model without pre-training weights [7].

One architecture of deep learning, Recurrent Neural Network (RNN) was applied in language modeling on speech recognition by Mikolov et al. in 2010 [8]. They show that RNN outperforms n-gram technique. The advantage of RNN in language modeling is a using of previous state to compute its current state, which is similar to the context in most of natural languages. However, simple RNN has a problem in passing the information in a long sequence. A solution to this issue is LSTM, a RNN with additional long term memory, that was proposed in 1977 [9]. In 2015, Wang et al. proposed LSTM with Trainable Lookup-Table (LSTM-TLT). They replaced fixed lookup-table of word vector by trainable lookup-table. Their trainable lookup-table also pre-trained by word2vec (Mikolov et al., 2013 [10]). LSTM-TLT beat state of the art techniques in Twitter sentiment analysis.

Another type of deep learning technique, Convolutional Neural Network (CNN) was introduced in 1998 by LeCun et al. on the document recognition task [11]. CNN consists of many layers that perform different functions. One key layer is the convolutional layer. This layer is used for extracting information from group of neighbor inputs. In 2012, CNN was used in image recognition task and outperformed other methods [12]. In the same year, DCNN - a CNN with dynamic k-max pooling layer - which is suitable for various input lengths was proposed by Kalchbrenner et al. It successfully defeated other models in Twitter sentiment classification. Pre-training word vectors was also used with CNN in sentence classification [13] and Twitter sentiment analysis [14].

There are some of researches for Thai sentiment analysis. In 2013, Wunnasri et al. proposed a method based on k-Nearest Neighbor (kNN) to solve unbalanced sentiment data from Thai Twitter [15]. Later, Chirawichitchai found that SVM outperforms kNN, NB and Decision Tree in emotion classification [16]. And in 2015, Sarakit et al. classified emotion data from Thai comments on Youtube using SVM, NB and Decision Tree [17]. However, most of Thai text researches use bag-of-words model.

III. MODELS

A. Word Vectors

In conventional methods, bag-of-words is popularly used as a document representation. It is a vector that has the same length as the number of words in dictionary. Each value in the vector indicates the frequency of that word in the document. However, with a large amount of words in natural language, a document representation based on bag-of-words is usually large. In addition, sparsity is likely to occur and causes difficulty in the training process.

Word vector is a smaller vector used to represent a word instead of a whole document. The length of word vector is adjustable and independent from the size of dictionary. In this study, we use word2vec to train initial word vectors for LSTM and DCNN models. With these word vectors trained by word2vec, a group of words having similar meaning also have similar word vectors.

B. Long Short Term Memory

LSTM is a RNN with an additional internal memory cell. Fig. 1 shows LSTM architecture, there are 3 internal gates: input gate, forget gate and output gate. Each gate indicates the controller of an information flow. The gates are computed as:

$$G_i^t = \sigma(W_i x^t + U_i h^{t-1} + b_i) \quad (1)$$

$$G_f^t = \sigma(W_f x^t + U_f h^{t-1} + b_f) \quad (2)$$

$$G_o^t = \sigma(W_o x^t + U_o h^{t-1} + b_o) \quad (3)$$

Where G^t is the gate at time t, x^t is the input at time t, h^{t-1} is hidden activation at time $t-1$. U and W represent the weight matrix of each gates. b is bias. Subscript i, f, and o indicate the variables for input gate, forget gate and output gate. σ is sigmoid function. The cell state C at time t can be calculated from:

$$C^t = G_f^t \times C^{t-1} + G_i^t \times tanh(W_C x^t + U_C h^{t-1} + b_C) \quad (4)$$

Subscript C indicates the variable for cell state. $tanh$ is hyperbolic tangent function. From (4), it can be seen that C^t is a result of adding the previous cell state C^{t-1} with the current input x^t by a proportion of gate value. Next, the hidden activation h^t is calculated from:

$$h^t = G_o^t \times tanh(C^t) \quad (5)$$

After the last input of the sequence, h^t will also represent the network output as shown on Fig. 2 - an unfolded version of LSTM with fully connected layer. The figure illustrates an expanded LSTM through time. The sequence of word vectors is

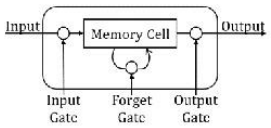


Figure 1. Long Short Term Memory.

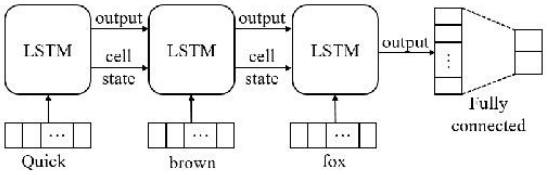


Figure 2. Unfolded Long Short Term Memory with Fully Connected Layer.

used in each time step. The output and cell state of previous LSTM are used in current LSTM. Finally, the output from last LSTM cell is feed into a fully connected layer with softmax classification. This network is trained by backpropagation.

C. Dynamic Convolutional Neural Network

Fig. 3 represents an architecture of DCNN. A network input is the sentence matrix $\mathbf{s} \in \mathrm{R}^{d \times s}$, s is the sentence length and d is the word vector length. The sentence matrix s consists of s word vectors $w_i \in \mathrm{R}^d$, w_i is the word vector of i^{th} word in a sentence, shows in (6),

$$\mathbf{s} = \begin{bmatrix} | & | & | \\ w_1 & \cdots & w_s \\ | & | & | \end{bmatrix} \tag{6}$$

Wide convolutional layer is a convolutional layer that uses zero padding on the border of input. In this layer, there are some filter matrices $\mathbf{m} \in \mathrm{R}^{d \times m}$, m is the filter size. The filters operate one-dimensional convolution on a row of sentence matrix. This operation is similar to creating m-gram features from an original sentence. A result of wide convolutional layer, represented by matrix $\mathbf{c}$ of dimension $d \times (s+m-1)$, is calculated from:

$$\mathbf{c}_{i,j} = \mathbf{m}_{i,:}\mathbf{s}_{i,j-m+1:j} \tag{7}$$

Folding layer is a layer that sums two adjacency rows into one row. From a matrix with d rows, this layer makes a new matrix with $d/2$ rows. It is used to increase the dependence between the rows, which are not involved with one another before the fully connected layer.

Afterward, there is a dynamic k-max pooling layer. This layer performs a selection of the top maximum k values on the column of sentence matrix. The dynamic k value is used instead of fixed k value. The k value for layer l is calculated from:

$$k_l = max(k_{top}, \left\lceil \frac{L-l}{L} s \right\rceil) \tag{8}$$

Given L is the total number of convolutional layers, l is the index of current convolutional layer which the calculated value k will apply to and k_{top} is the fixed k value for the topmost convolutional layer. These pooling strategy makes DCNN suitable for any various lengths of input. After k-max pooling layer, a non-linear function, which is hyperbolic tangent for this study, is applied.

Each network has arbitrary number of convolutional layer, folding layer, and pooling layer connected together. And on the top, there are dropout layer and fully connected layer with softmax classification.

IV. Our Twitter Data Collection

Our data collection process shows on Fig. 4. While Twitter API allows us to search tweets in a specific language, collecting tweets in Thai is still a complex situation. This is because the API does not support searching a keyword in non-space separated languages. To carry out data collection, we use our known Thai Twitter users as a seed and get their followers to collect more users. We filter out users that have no Thai tweet by checking Thai letters to get sample Thai users. And from this set of users, we then collect only tweets that contain Thai characters.

In the labeling process, we use pre-classify emotions to classify tweets into positive or negative. We label any tweet to each class when it contains emoticons corresponding to that class, and does not contain any emoticon corresponding to

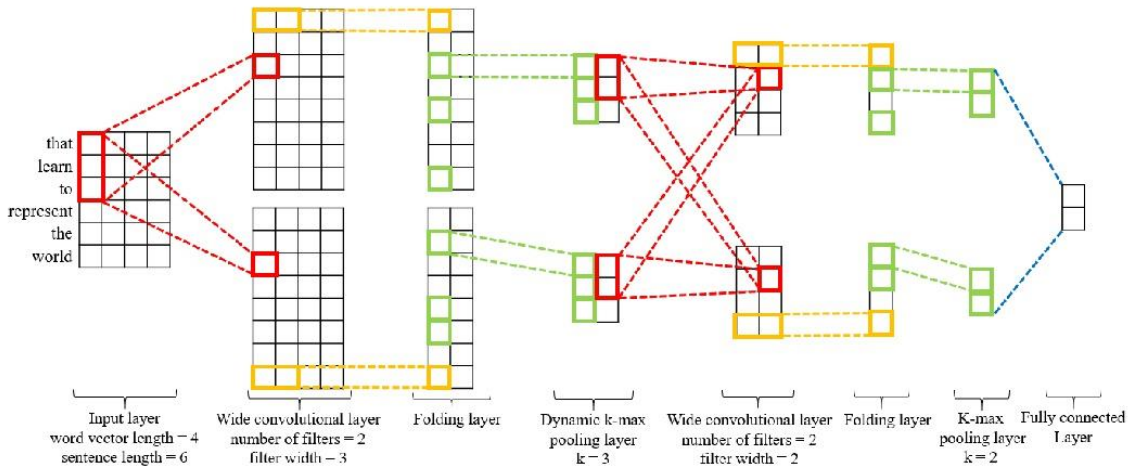


Figure 3. Dynamic Convolutional Neural Network

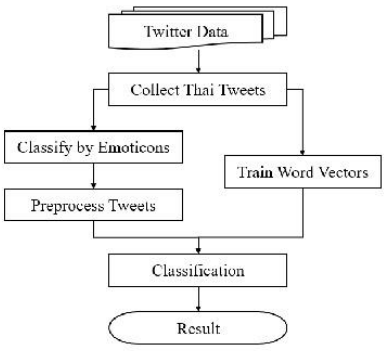


Figure 4. Our Twitter data collection and classification process.

another class. The pre-classify emoticons are shown on Table I. And the number of tweets with emoticons are listed in Table II.

TABLE I. PRE-CLASSIFY EMOTICONS

Positive Emoticons	Negative Emoticons
😂, 😊, 😍, 😁, 😄, ☺, 😉, 😋, 😅, 😌, , 😏, :), ^^, :D	😓, ㅠ, ☹, 😞, 😕, 😢, 😩, 😐, :(, TT, T_T,T-T, T T, T^T

TABLE II. NUMBER OF COLLECTED TWEETS GROUPED BY EMOTIONS

Group	Number of Tweets
All	3,813,173
Positive Tweets	140,414
Negative Tweets	33,349

We also preprocess tweets by these steps accordingly.

1) Remove pre-classify emoticons from a tweet. This step prevents a classifier from overfitting with these supervised emoticons.
2) Remove characters that are not in Thai character Unicode range (U+0E00 to U+0E7F)
3) Replace a sequence of duplicated characters longer than two with one character.
4) Remove ๆ (U+0E46). This character is only used for repeating the previous word but it does not change the polarity of sentence. We remove it to reduce the number of words in dictionary after applying words segmentation.
5) Remove single-character word. It does not have a meaning in Thai.
6) Break a tweet into word segments. This is an important part of preprocessing because Thai is a non-space separated language. We use KUCut [18] library to perform this task.

Table III shows the results of features reduction for each preprocessing step. After applying all steps, the feature size of this Thai Twitter corpus is reduced to 28,890 words.

TABLE III. NUMBER OF FEATURES AFTER EACH PREPROCESSING STEP

Preprocessing	Number of Features	Percent of Original
None	33,803	100.00%
repeated character	32,141	95.08%
ๆ (U+0E46)	30,521	90.29%
word with length 1	33,745	99.83%
All	28,890	85.47%

V. EXPERIMENTS

In this study, we use 3-folds cross validation in the verification process. Each subset contains a sample of 11,000 positive tweets and 11,000 negative tweets from labeled data. We select 10% of training data and use them for validation. From the total of 28,890 words, we select 12,174 words that appear more than once in our corpus to use in the experiments.

We measure the results by accuracy value which is computed as:

$$Accuracy = \frac{number\ of\ correct\ results}{number\ of\ total\ results} \quad (9)$$

In this section, we organize five experiments as follow. Experiment 1, 2 and 3 are conducted to find the best parameters for deep learning. Experiment 4 is a comparison between deep learning and classical methods. Experiment 5 is a study of an important of sequence in Thai Twitter data.

A. Experiment 1: Different Sizes of Word Vectors for Deep Learning Techniques

Word vector is a vector representation of an original word. We use word vectors for both LSTM and DCNN. Our word vectors are initially trained by word2vec with skip-gram model and be trained from backpropagation later.

The LSTM using in this experiment receives a sequence of word vectors and creates an output with the length of 50. Then, in the output layer of LSTM, there is a fully connected layer. We use softmax function to classify data into two classes. This network is trained by stochastic gradient descent with cross entropy loss functions.

The DCNN using in this experiment has two convolutional and two folding layers. First convolutional layer has 6 filters with width of 7. Second convolutional layer has 14 filters with width of 5. The top of network has a dropout layer with 0.5 probability and fully connected layer with softmax classification. Because there are two folding layers, we have to use word vector length that can be divided by 4. The fixed k in the topmost pooling layer is 4. This network is trained by "Adaptive Gradient (Adagrad)" [19] with cross entropy loss function.

The result of this experiment shows in Table IV. The using of word vectors with a length under 12 is less accurate than using the larger word vectors. The accuracy is increased as the word vector length is increased at the beginning, but it is not increased after a certain length. Therefore, we will use the word vector length at 48 in the remaining experiments to optimize a computational time.

TABLE IV. ACCURACY ON VARIOUS WORD VECTOR LENGTHS. BOLDFACE IS THE WINNER.

Classifier	Accuracy (%) by Word Vector Length						
	4	8	12	24	48	96	192
LSTM	73.79	74.59	74.87	75.06	75.03	**75.12**	74.88
DCNN	74.19	74.48	75.06	75.23	**75.35**	75.25	74.95

B. Experiment 2: Paramater Selection for LSTM

In this experiment, we aim to find the best output sizes of LSTM, which can be seen as a size of hidden nodes in the network. We use the best architecture of LSTM network from previous experiment. We also experiment without the hidden node by immediate applying softmax classification on LSTM output. Table V shows the result of the experiment in term of accuracy. We find that the best accuracy comes from using 5 hidden nodes. Moreover, the result shows that an accuracy of the network without hidden node is less than the network with hidden node. This means that hidden node is necessary to be included in our network. Since 5 hidden nodes give the best accuracy, we will use this configuration in remaining experiments.

TABLE V. ACCURACY ON VARIOUS NUMBER OF HIDDEN NODES IN LSTM NETWORK. BOLDFACE IS THE WINNER.

Classifier	Accuracy (%) by Number of Hidden Nodes				
	No hidden node	5	10	20	50
LSTM	73.93	**75.30**	75.20	75.07	75.03

C. Experiment 3: Parameter Selection for DCNN

In this experiment, we use the same DCNN architecture as described in Experiment 1, except that we try to change the number of filters and filter width. Table VI shows the accuracies from the experiment on filter widths. The first number in each column indicates the filter width of first convolutional layer, while the second number is for second convolutional layer. We found that the filter width of 7 and 5 in each layer give the best accuracy. In the same way, Table VII shows the accuracies from the experiment on various number of filters. From the result, there are two groups that give the best accuracy. However, we will use 3 filters on the first convolutional layer and 6 filters on the second convolutional layer because the smaller number of filters requires the lesser computational time.

TABLE VI. ACCURACY ON VARIOUS FILTER WIDTH OF DCNN. FIRST NUMBER ON EACH COLUMN IS THE FILTER WIDTH ON FIRST CONVOLUTIONAL LAYER. SECOND NUMBER IS FOR THE SECOND CONVOLUTIONAL LAYER. BOLDFACE IS THE WINNER.

Classifier	Accuracy (%) by Filter Width			
	4, 2	5, 3	7, 5	10, 7
DCNN	75.10	75.23	**75.35**	75.28

TABLE VII. ACCURACY ON VARIOUS NUMBER OF FILTERS OF DCNN. FIRST NUMBER ON EACH COLUMN IS THE FILTER WIDTH ON FIRST CONVOLUTIONAL LAYER. SECOND NUMBER IS FOR THE SECOND CONVOLUTIONAL LAYER BOLDFACES ARE THE WINNER.

Classifier	Accuracy (%) by Number of Filters				
	2, 2	2, 4	3, 6	4, 8	6, 14
DCNN	75.24	75.23	**75.35**	75.18	**75.35**

D. Experiment 4: A Comparison of Deep Learning and Classical Methods

In this experiment, we compare the previous result from LSTM and DCNN with classical methods using bag-of-word model. The features used in these classifiers are prepared by TF-IDF. TF-IDF is a representation of word in document calculated by multiplying term frequency with inverse document frequency. Given D is the document corpus, N is the number of documents, n_t is the number of documents that contain term t, $tf(t, d)$ is the term frequency of term t in document d. TF-IDF is calculated from:

$$tfidf(t, d, D) = tf(t, d) \times log(\frac{N}{n_t}) \quad (10)$$

Table VIII shows the result of experiment. Multinomial Naïve Bayes (NB), Support Vector Machine (SVM) and Maximum Entropy (MaxEnt) are used as baseline methods. We also experiment on another deep learning technique called stacked auto-encoders (SAE) with TF-IDF representation. The result shows that both LSTM and DCNN have higher accuracies compared to other classifiers.

TABLE VIII. ACCURACY OF LSTM AND DCNN COMPARED TO CONVENTILNAL CLASSIFIERS. BOLDFACE IS THE WINNER.

Classifier	Accuracy (%)
LSTM	75.30
DCNN	**75.35**
SAE	74.91
NB	74.05
SVM	74.71
MaxEnt	75.13

We also verify the result by using accuracies from 3-folds cross validation in a statistical technique called paired t-test, which is used to determine if two methods are different from each other. If P-Value of any pair of methods is less than alpha, which is 0.05 in our testing, we will conclude that the two methods have a different mean. Table IX shows P-value from our verification. The result shows that LSTM and DCNN have different mean from NB and SVM. Therefore, we conclude that these two deep learning techniques significantly outperform the two conventional techniques.

TABLE IX. P-VALUE FROM PAIRED T-TEST BETWEEN LSTM, DCNN AND OTHER CLASSIFIERS. LSTM AND DCNN ARE SIGNIFICANTLY DIFFERENT FROM NB AND SVM.

	NB	SVM	MaxEnt	SAE	DCNN
LSTM	**0.025**	**0.021**	0.121	0.087	0.472
DCNN	**0.038**	**0.045**	0.210	0.130	

E. Experiment 5: An Effect of Sequence in Thai Tweets

In this experiment, we study how the sequence of words influences the sentiment analysis with deep learning. We use the same Thai Twitter data as previous experiment but shuffle words in sentences before using in the testing data. Results are shown in Table X. We show that the accuracies of sentiment analysis on shuffled words in sentences are less than the accuracies on original sentences.

TABLE X. ACCURACY ON CLASIFIFYING ORIGINAL TWEET COMPARED TO TWEET WITH SHUFFLED WORDS. BOLDFACE IS THE WINNER.

Classifier	Accuracy (%)	
	Original Sequence	Shuffled Words Sequence
LSTM	**75.30**	75.01
DCNN	**75.35**	75.04

Moreover, we analyze the result of this experiment by walkthrough tweets that is misclassified after shuffled the words. We find that the word that has a different emotion from the sentence's overall meaning can lead to an incorrect classification. For example, from Table XI, the word "forget" tends to be negative, while the whole sentence is positive. We conclude that a sequence of word in Thai is important for the sentiment analysis.

TABLE XI. EXAMPLE OF ORIGINAL TWEET AND SHUFFLED WORD TWEET THAT MAKE AN INCORRECT CLASSIFICATION

Original tweet classifying in positive emotion	วันนี้ วัน แม่ อย่า ลืม บอก รัก แม่ ด้วย นะ คับ (Today is mother day, don't forget to tell a love to mother too, sir.)
Shuffled word tweet classifying in negative emotion	บอก ลืม นะ วัน อย่า คับ วันนี้ รัก ด้วย แม่ แม่ (tell forget, day don't sir today love too mother mother)
Original tweet classifying in negative emotion	ขอบคุณ ครับ สงสัย ผม ต้อง ถอน ฟัน คุด ฟัน กราม ประมาณ ซีก (Thank you sir, I doubt that I have to extract molar wisdom tooth about a piece.)
Shuffled word tweet classifying in positive emotion	ฟัน คุด กราม ซีก สงสัย ต้อง ถอน ประมาณ ครับ ฟัน ผม ขอบคุณ (wisdom tooth molar piece a piece I doubt have to extract about sir tooth I thank you)

VI. CONCLUSION

In this paper, we apply deep learning techniques on our Thai tweets to analyze their sentiments. First, we conduct an experiment to find the best parameters of LSTM and DCNN. Then we show that the best classifier is DCNN, followed by LSTM. Both techniques give significantly higher accuracies than classical techniques such as NB and SVM, but not MaxEnt. Finally, we experiment on sentences with shuffled word order to demonstrate that the sequence of words influences sentiment analysis on Thai Twitter data.

In addition to our study, there is another benefit from using deep learning techniques. A set of trained word vectors from classification process can be used in many linguistic research areas.

For future work, we think that an accuracy can be enhanced by improving the feature mapping and convolutional process.

ACKNOWLEDGMENT

We would like to acknowledge Prof. Dr. Boonserm Kijsirikul for a suggestion of deep learning and Dr. Choochart Haruechaiyasak for an example of Thai Twitter data.

REFERENCES

[1] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," CS224N Project Report, Stanford, vol. 1, p. 12, 2009.

[2] A. Bermingham and A. F. Smeaton, "Classifying sentiment in microblogs: is brevity an advantage?," in Proceedings of the 19th ACM international conference on Information and knowledge management, 2010, pp. 1833–1836.

[3] X. Wang, Y. Liu, C. Sun, B. Wang, and X. Wang, "Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory," in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, 2015, vol. 1, pp. 1343–1353.

[4] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," arXiv preprint arXiv:1404.2188, 2014.

[5] Y. Bengio, R. Ducharme, P. Vincent, F. Morin, and C. Jauvin, "Neural probabilistic language models," in Journal of Machine Learning Research, 2003, pp. 1137–1155.

[6] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," APSIPA Transactions on Signal and Information Processing, vol. 3, 2014.

[7] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, and others, "Greedy layer-wise training of deep networks," Advances in neural information processing systems, vol. 19, p. 153, 2007.

[8] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent Neural Network based Language Model," Interspeech, no. September, 2010, pp. 1045–1048.

[9] S. Hochreiter, S. Hochreiter, J. Schmidhuber, and J. Schmidhuber, "Long short-term memory.," Neural Comput., vol. 9, no. 8, 1997, pp. 1735–80.

[10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in Advances in neural information processing systems, 2013, pp. 3111–3119.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2323, 1998.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[13] Y. Kim, "Convolutional neural networks for sentence classification," arXiv preprint arXiv:1408.5882, 2014.

[14] I. D. Dario Stojanovski , Gjorgji Strezoski, Gjorgji Madjarov, "Twitter Sentiment Analysis Using Deep Convolutional Neural Network," Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), vol. 9121, pp. 515–529, 2015.

[15] W. Wunnasri, T. Theeramunkong, and C. Haruechaiyasak, "Solving unbalanced data for Thai sentiment analysis," Proceedings of the 2013 10th International Joint Conference on Computer Science and Software Engineering, JCSSE 2013, pp. 200–205, 2013.

[16] N. Chirawichitchai, "Emotion classification of Thai text based using term weighting and machine learning techniques," in Computer Science and Software Engineering (JCSSE), 2014 11th International Joint Conference on, 2014, pp. 91–96.

[17] P. Sarakit, T. Theeramunkong, C. Haruechaiyasak, and M. Okumura, "Classifying emotion in Thai youtube comments," 2015 6th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES), pp. 1–5, 2015.

[18] Sudprasert, S., Kawtrakul, A., "Thai word segmentation based on Global and Local Unsupervised learning." In NCSEC, Chonburi, Thailand, 2003.

[19] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," The Journal of Machine Learning Research, vol. 12, pp. 2121–2159, 2011

ภาคผนวก ข

A Character-level Convolutional Neural Network with Dynamic Input Length for Thai Text Categorization

Thanabhat Koomsubha, Peerapon Vateekul
Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
Thanabhat.K@Student.Chula.ac.th, Peerapon.V@Chula.ac.th

***Abstract*— A Character-level Convolutional Neural Network (Char-CNN) is an efficient text categorization method. It can be used in categorization task without a word segmentation step, which is necessary by traditional method for Thai. Currently, the existing model of Char-CNN uses a fixed input length and requires cutting off exceeding characters, which may lead to a missing of important content. In this paper, we propose a new Char-CNN model with a capability to accept any length of input by employing k-max pooling before a fully connected layer. The result shows that our model outperforms a Char-CNN model with a fixed input length on Thai news categorization. Moreover, our proposed method gives a better accuracy than many word-level methods: Naive Bayes, Logistic Regression, Support Vector Machine except a word-level CNN.**

Keywords*— *Character-level Convolutional Neural Network; Dynamic Input Length; Thai Text Categorization; Deep Learning

I. INTRODUCTION

An automatic text categorization system is an important task. Nowadays, there are many of work that needs to divide natural language data into a group e.g. tagging a blog with predefined classes, classifying an emotion from social media, or filtering out a spam from an inbox. We can reduce a human effort by deploy a suitable method.

Many well-known methods have been used in text categorization e.g. Support Vector Machine, Naive Bayes, Logistic Regression [1, 2]. After that, deep learning techniques have been adopted to natural language classification e.g. Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM) [3, 4]. However, these models use a word-level feature and need to break a sentence into a list of words.

Later, a Character-level Convolutional Neural Network (Char-CNN) has been proposed by Zhang et al. [5]. They use Char-CNN to classify large English text data. The result shows that this model can compete with tradition methods in sentiment analysis and text categorization. An advantage of character-level features is that we can use these model without any knowledge of the language. Also, it has a benefit when applying to a language with word morphology.

From the benefit of a character-level feature, we want to apply this method to Thai text. Because Thai is a non-space separated language, therefore, we need to perform a word segmentation before feed any sentence to a traditional method. When applying Char-CNN to Thai, we can simplify the procedure of Thai text categorization. Moreover, there is a study shows that deep learning techniques perform a good result compare to traditional techniques on Thai text [6].

However, an existing Char-CNN model still has a disadvantage because it receives an input matrix with a fixed dimension. We have to clip a document to fit the defined length. Any exceeding character is removed. In this paper, we improve a Char-CNN by extending the model to capture any length of the input. We test our model on Thai news with five categories. Our objective is to improve a Char-CNN by extending the model to accept any input length. We compare our dynamic model to a fixed Char-CNN model. We also compare our method with a word-level model.

This paper is organized as follow. Section 2 shows an existing work. Section 3 describes our proposed model. Experiments appear in section 4. And a conclusion is written in section 5.

II. RELATED WORK

There is an amount of research in text categorization. Major techniques in machine learning have also been applied to this area e.g. Decision Tree, SVM, Naive Bayes, k-NN, and Neural Network [7]. A simple and famous document presentation using for these classifiers is bag of words. Also, Term Frequency-Inverse Document Frequency (TF-IDF) is a good alternative option to represent a text. TF-IDF is an efficient method that it reduces an important of words which appear too frequent.

In 2008, to utilize a characteristic of natural language that an order of word is important, Collobert and Weston use max pooling over time and Time-Delay Neural Network (TDNN) for languages-related tasks [8]. Afterthat, when deep learning techniques are famous, Recurrent Neural Network (RNN) and its successive, Long-Short Term Memory (LSTM), has been used in language modeling. Those methods are suitable for natural language because of an ability to extract features from a sequential data. In the same way, Convolutional Neural Network (CNN), which is popular in computer vision task, has been adopted for text classification. Dynamic Convolutional Neural Network (DCNN) is one type of CNN which is proposed by Kalchbrenner et al. in 2014 [9]. Their method outperforms other

methods on sentiment classification. They use a new pooling layer called a dynamic k-max pooling, which compute a new suitable k value for each iteration. Therefore, their network can read any length of an input. In addition, most of neural network based approach use word embedding, as known as word vector, to represent an input.

In 2014, a CNN which uses some feature from character-level was proposed for part of speech tagging and sentiment analysis [10, 11]. They use characters in a word to create character-level embedding alongside with word-level embedding. Later in 2015, another research on CNN combining with LSTM using character embedding was proposed [12]. But these models still require a word segmentation.

In 2015, Zhang et al. proposed a Character-level Convolutional Neural Network (Char-CNN), shows in Fig. 1. Their model produces a better result than other models including a word-level CNN on sentiment analysis and text categorization. A one-hot encoding has been used as an input for the network. A one-hot vector is a vector that there is only one position with value 1, and other remaining positions have a value 0. The position of value 1 in the vector indicates a represented character. An example of a character presentation shows in (1)

$$v_a = \begin{bmatrix}1\\0\\0\\\vdots\\0\end{bmatrix}, v_b = \begin{bmatrix}0\\1\\0\\\vdots\\0\end{bmatrix}, \ldots \quad (1)$$

The v_a vector represent a character 'a', v_b vector represent a character 'b', and so on. An input matrix is a sequence of one-hot vectors.

A convolution for this network is called temporal convolution. It is a 1-D convolution. Given an input matrix of this layer M with a dimension $l \times n$, l is length, n is number of input features. Given w is a filter width, therefore, a dimension of filter W is $n \times w$. Also, there is a bias b. An output vector of temporal convolution C, which has a dimension of $(l-w+1) \times 1$, is computed by

$$C_t = b + \sum_{j=1}^{n} \sum_{k=1}^{w} W_{j,k} M_{w \times (t-1)+k,j} \quad (2)$$

An output C in (2) is an output of one filter. Normally, there is more than one filter, which sometime be called a feature. An output C of each filter will put together to form an output matrix.

Also, a max pooling in this network is called temporal max pooling. It performs an operation on 1-D. Given M is an input matrix of pooling layer with a dimension of $l \times n$. Given s is a pooling size. An output matrix P with a dimension of $\frac{l}{s} \times n$ is computed by

$$P_{t,n} = max_{k=1}^{s} M_{s \times (t-1)+k,n} \quad (3)$$

A summary of Char-CNN architecture is shown on Table II. We use a small network from the original paper.

III. Proposed Model

Our proposed Char-CNN is based on Zhang's model. A major change occurs at a last pooling layer. We use k-max pooling instead of max-pooling. Given a same variable from (3), an output matrix of k-max pooling P with a dimension of $k \times n$ is computed by

$$P_{*,n} = k\ max_{j=1}^{l} M_{j,n} \quad (4)$$

Fig. 2 shows a difference between max pooling and k-max pooling. Max pooling is a method for down sampling data by using a sliding window on a row of data and select a cell which contains a maximum value to be passed to next layer. A window is then move to a next non-overlapping area. Given l is a data length, s is window size. The length of a max pooling result is

$$|MaxPooling(l, s)| = \frac{l}{s} \quad (5)$$

On the other hand, k-max pooling doesn't have a window. A selecting operation performs for all data in a row. Top k cells which have maximum value are selected to be used in next layer. Given m is a data length, k is k value. The length of a k-max pooling is

$$|KMaxPooling(l, k)| = k \quad (6)$$

When applying k-max pooling to a network. A next layer always has a length of k. We use this advantage to apply this layer before a fully connected layer. Therefore, we can certainly have a matrix which able to fit into a fully connect layer regardless the length of an input.

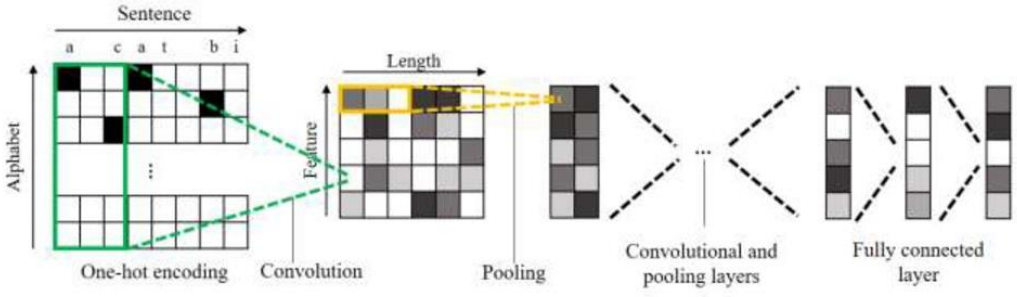


Figure 1. Character-level Convolutional Neural Network (Char-CNN) proposed by Zhang et al. (2015)

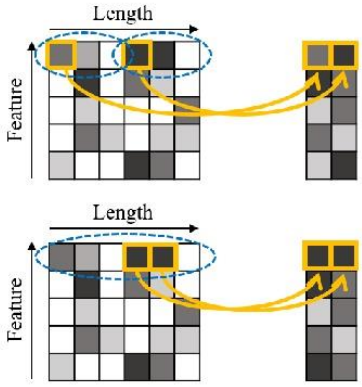


Figure 2. Top: max pooling with a window size = 3.
Bottom: k-max pooling with k = 2

For an input, we don't fix a length of a matrix. We only set a minimum length to 402, which is a minimum number that there are enough data for fully connected layer. Fig. 3 shows our method for various input length. On the convolutional and pooling layers, the length of data in network depends on the length of input. While after the k-max pooling layer, the length of data in every document are equal. We set k value on this layer to 34 to match a last max pooling layer of Zhang's model. Also, other network parameters are still the same with Zhang's model e.g. the filter width and number of features in convolutional layer, and the size of first and second max pooling layer.

There is another difference between Thai and English. Thai alphabet has superscript and subscript characters that place at the same position with another alphabet e.g. "สวัสดี" (Hello). In this case, we use an original order of alphabet from a news source.

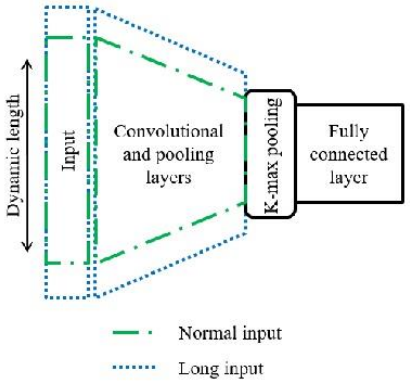


Figure 3. Our proposed Char-CNN with a dynamic input length. The normal length input and long input have a different in length. But both input use a same network.

IV. Dataset

We use Thai news from a newspaper for an experiment. Each news contains only news content and doesn't include any article title because it may explicitly answer a category. A total of 155,000 news is collected equally from five categories: real estate and land, computer and hardware, financial and bank, constitution and law, and sport. Examples of the news are shown in Table I.

TABLE I. Examples of news in each category. We show only few sentence for each example.

Real estate and land	คลังชี้มาตรการกระตุ้นอสังหาฯสูญรายได้ 1.6 หมื่นล้านบาท (Ministry of Finance told that an immovable property support plan made a money loss about 1.6 ten thousand million Baht.)
Computer and hardware	เดลี่เทครายงานว่า แอปเปิ้ล ผู้พัฒนาไอโฟน จากประเทศสหรัฐอเมริกา ส่งจดหมายเชิญสื่อมวลชนร่วมงานเปิดตัวผลิตภัณฑ์ใหม่ประจำปี (Daily Tech report that Apple, an Iphone's inventor from USA, invite mass media to their annual debut ceremony.)
Financial and bank	จีนเดินเกมลดค่า "เงินหยวน" สร้างความได้เปรียบการค้าโลก (China start to reduce Yuan value for their advantage in world trading.)
Constitution and law	รัฐธรรมนูญฉบับมีชัย ใกล้ปรากฏโฉม ตามเส้นตายที่กำหนดไว้ 29 มกราคม (The Meechai's constitution will be available soon by the deadline at 29th January.)
Sport	"ช้างศึก" ทีมชาติไทย พร้อมหักด่าน "ซามูไร" ญี่ปุ่น ทำศึกลูกหนังยู 23 ชิงแชมป์เอเชีย วันนี้ ("War elephant", Thailand national football team, is ready to compete with "Samurai", Japan national football team, in U-23 Asia Championship today.)

An average news length is 2,703 characters with standard deviation (SD) at 3,807 characters. The SD is large because some news has a very long length about 100,000 characters. The histogram of news length is shown in Fig. 4. There are 532,348 unique words from a total of 71,593,778 words. The top 10,000 most frequency words are covered about 96% of all words in the dataset. We split news data into 3 sets: 115,000 news into a training set, 15,000 news into a validating set, and 25,000 news into a testing set. For a character-level model, we only use one preprocessing step that we convert all English uppercase into English lowercase.

V. Experiments

In this paper, we measure results in term of accuracy. An accuracy is computed by

$$Accuracy = \frac{number\ of\ correct\ results}{number\ of\ total\ results} \quad (7)$$

A. Experiment 1: Compare with Fixed Char-CNN Model

In this experiment, we compare our model with a fixed Char-CNN model. We use the small network from Zhang's paper. An input matrix is one-hot encoding with a fixed length at 1014. The first and second convolutional layer's filter have width 7 while

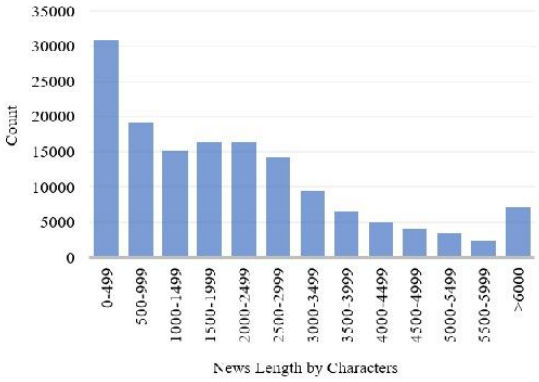


Figure 4. Histogram of news length counted by characters.

the rest convolutional layers have width 3. Every convolutional layer has 256 features. Every max pooling layer has width 3. Lastly, there are three fully connected layers. Two layers with 1024 hidden nodes and one layer with 5 output nodes for each class with softmax function.

For our model, we don't fix the length of the input matrix. However, by the limitation of a memory, we define a max length and observe how the length of the input affects an accuracy. We set the max length of input matrix at 1000, 2000, 4000, and 6000. At the last max pooling layer, we change it into k-max pooling with size 34. Therefore, the input of fully connected layer of both models have the same length at 34 x 256 = 8704. Besides, we change an original threshold layer to rectifier linear unit (ReLU). A summary of our model architecture is shown on Table II.

The input matrix of both models has 151 characters. We use the same 70 English alphanumeric and special characters from the original paper. Also, we add 81 Thai characters. A set of characters that we use for an input features is shown on Table III.

TABLE III. A SET OF CHARCTERS THAT WE USE FOR EXPERIMENTS

Characters from original model	abcdefghijklmnopqrstuvwxyz01234 56789-,;.!?:'\"/|_@#$%^&*~`+=<> ()[]{}
Our additional characters	กขฃคฅฆงจฉชซฌญฎฏฐฑฒณดตถทธนบปผฝพฟ ภมยรฤลฦวศษสหฬอฮะัาำิีึืฺุู฿เแโใไๅๆ็่้๊๋์ ๐๑๒๓๔๕๖๗๘๙

We train the network by backpropagation. A stochastic gradient descent with momentum 0.9 has been used. An initial learning rate is 0.005 and reduces to a half of previous rate every 3 epochs. Each epoch consists of 10000 random minibatch of size 32.

An implementation is done on Torch 7 [13]. We modify the source code from Zhang's Github [14].

TABLE IV. RESULT OF OUR PROPOSED MODEL COMPARE WITH EXISTING MODEL. BOLDFACE IS THE WINNER.

Method	Accuracy (%)
Char-CNN (Zhang et al., 2015, fixed length at 1014)	90.59
Proposed Char-CNN (max length at 1000)	90.67
Proposed Char-CNN (max length at 2000)	93.32
Proposed Char-CNN (max length at 4000)	95.01
Proposed Char-CNN (max length at 6000)	**95.44**

The results on the Table IV show that our dynamic Char-CNN outperforms the fixed Char-CNN in term of accuracy. Moreover, we observe that when we use a longer input matrix, an accuracy is better.

TABLE II. AN AHCHITECTURE OF CHAR-CNN AND A COMPARASION BETWEEN ORIGINAL MODEL (ZHANG ET AL., 2015) AND OUR PROPOSED MODEL. F INDICATES NUMBER OF FEATURES. W IS FILTER WIDTH. S IS POOLING SIZE. H IS HIDDEN NODES. O IS OUTPUT SIZE. ALSO NOTE THAT THERE IS A NON-LINEARITY LAYER AFTER EVERY CONVOLUTIONAL LAYER (THRESHOLDING FOR ZHANG'S MODEL, RELU FOR OUR PROPOSED MODEL)

Zhang et al., 2015			Proposed model		
Layer	**Parameter**	**Length**	**Layer**	**Parameter**	**Length**
Input layer	N/A	1014	Input layer	N/A	l_0
Convolutional layer	F = 256, W = 7	1008	Convolutional layer	F = 256, W = 7	$l_1 = l_0 - 6$
Max pooling layer	S = 3	336	Max pooling layer	S = 3	$l_2 = l_1 / 3$
Convolutional layer	F = 256, W = 7	330	Convolutional layer	F = 256, W = 7	$l_3 = l_2 - 6$
Max pooling layer	S = 3	110	Max pooling layer	S = 3	$l_4 = l_3 / 3$
Convolutional layer	F = 256, W = 3	108	Convolutional layer	F = 256, W = 3	$l_5 = l_4 - 2$
Convolutional layer	F = 256, W = 3	106	Convolutional layer	F = 256, W = 3	$l_6 = l_5 - 2$
Convolutional layer	F = 256, W = 3	104	Convolutional layer	F = 256, W = 3	$l_7 = l_6 - 2$
Convolutional layer	F = 256, W = 3	102	Convolutional layer	F = 256, W = 3	$l_8 = l_7 - 2$
Max pooling layer	S = 3	34	K-max pooling layer	K = 34	34
Fully connected layer	H = 1024	N/A	Fully connected layer	H = 1024	N/A
Dropout	P = 0.5	N/A	Dropout	P = 0.5	N/A
Fully connected layer	H = 1024	N/A	Fully connected layer	H = 1024	N/A
Dropout	P = 0.5	N/A	Dropout	P = 0.5	N/A
Fully connected layer	O = 5	N/A	Fully connected layer	O = 5	N/A

B. Experiment 2: Compare with Word-level Model

In this experiment, we compare our dynamic Char-CNN model with other word-level models. We perform a word segmentation in this experiment using SWATH [15].

Three traditional methods, Naive Bayes (NB), Logistic Regression, and Support Vector Machine (SVM) have been used. We select the top 10,000 frequent words to create a feature matrix because of a memory limitation, an input matrix is large as a number of data multiply with a number of words. An experiment has been done for both bag-of-words (BoW) and TF-IDF.

Another word-level model using in this experiment is Dynamic Convolutional Neural Network (DCNN). We use an initial word vector trained by word2vec with skip-gram model [16]. We set filter width to 10 for first convolutional layer and 7 for second convolutional layer. Number of filters is set to 6 and 12 accordingly. For k value, we set to 5.

TABLE V. RESULT OF OUR PROPOSED MODEL COMPARE WITH WORD-LEVEL MODEL. BOLDFACE IS THE WINNER.

Method	Accuracy (%)
Naïve Bayes, BoW	87.22
Naïve Bayes, TF-IDF	89.00
Logistic Regression, BoW	94.87
Logistic Regression, TF-IDF	94.74
SVM, BoW	93.78
SVM, TF-IDF	95.24
DCNN (Kalchbrenner et al., 2014)	**95.95**
Proposed Char-CNN	95.44

The result is shown in Table V. Our proposed model has a better accuracy than NB, Logistic Regression, and SVM for both bag-of-words and TF-IDF features. However, the best model in this experiment is a word-level DCNN. A factor that might be a reason for this result is data size. As reported in an original Char-CNN paper, we observe that Char-CNN is starting to produce a better accuracy than word-level CNN when data size is about 1 million.

Furthermore, it is worth mentioning that we also test the model similar to DCNN, but use character input. We try to apply both k-max pooling and dynamic k-max pooling to the two earliest pooling layers. But the result is bad. The network doesn't converge. We conclude that when using character-level features, it shouldn't use k-max pooling at the earliest layer because the character features may be messed up.

VI. CONCLUSION

In this paper, we propose a Char-CNN with dynamic length input. We apply the model to Thai news categorization task. We demonstrate that our model which can accept a longer input gives a better accuracy. We also show that our model outperforms traditional method with word-level feature e.g. NB, Logistic Regression, SVM except for word-level CNN. By applying this model, we can simplify the Thai text categorization task by skipping the word segmentation step. Besides, the character-level model still has a potential for another natural language task.

For future work, it could benefit from an experiment on a very large Thai dataset to compare between character-level model and word-level model. Moreover, an investigation of character-level's advantage for a language morphology in Thai is considerable.

ACKNOWLEDGMENT

We would like to acknowledge Prof. Dr. Boonserm Kijsirikul for an invaluable suggestion about a Convolutional Neural Network.

REFERENCES

[1] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant," 1998.

[2] Y. Yang and X. Liu, "A Re-Examination of Text Categorization Methods," ACM Special Interest Group of Information Retrieval (SIGIR), pp. 42–49, 1999.

[3] Y. Kim, "Convolutional neural networks for sentence classification," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1746–1751.

[4] X. Wang, Y. Liu, C. Sun, B. Wang, and X. Wang, "Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory," in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, 2015, vol. 1, pp. 1343–1353.

[5] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in Advances in Neural Information Processing Systems, 2015, pp. 649–657.

[6] P. Vateekul and T. Koomsubha, "A Study of Sentiment Analysis Using Deep Learning Techniques on Thai Twitter Data," presented at the Computer Science and Software Engineering (JCSSE), 13th International Joint Conference on, 2016.

[7] F. Sebastiani, "Machine learning in automated text categorization," ACM Computing Surveys (CSUR), vol. 34, no. 1, pp. 1–47, 2002.

[8] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in Proceedings of the 25th international conference on Machine learning, 2008, pp. 160–167.

[9] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, {ACL} 2014, 2014, pp. 655–665.

[10] C. D. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014, pp. 1818–1826.

[11] C. N. dos Santos and M. Gatti, "Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts," in COLING, 2014, pp. 69–78.

[12] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," arXiv preprint arXiv:1508.06615, 2015.

[13] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in BigLearn, NIPS Workshop, 2011.

[14] X. Zhang, "Crepe", GitHub repository, https://github.com/zhangxiangxiao/Crepe, 2015

[15] P. Charoenpornsawat, "Feature-based Thai Word Segmentation," Master's Thesis, Computer Engineering. Chulalongkorn University, Bangkok, Thailand, 1999.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in Advances in neural information processing systems, 2013, pp. 3111–3119.

ประวัติผู้เขียนวิทยานิพนธ์

นายธนภัทร์ คุ้มสุภา เกิดเมื่อวันที่ 21 พฤศจิกายน พ.ศ. 2533 ที่จังหวัดลพบุรี สำเร็จการศึกษาระดับปริญญาตรีหลักสูตรวิศวกรรมศาสตรบัณฑิต (เกียรตินิยมอันดับ 1) สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2555 และเข้าศึกษาในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2557