บทที่ 3

วิธีดำเนินงานวิจัย

ในบทนี้จะอธิบายถึงชุดข้อมูลข้อความแสดงความคิดเห็นที่เกี่ยวกับโรงแรม ซึ่งรวบรวมมาจากเว็บไซต์ TripAdvisor ที่ใช้ในโครงงานนี้ และวิธีการดำเนินงาน ดังนี้

3.1 ชุดข้อมูล (Dataset)

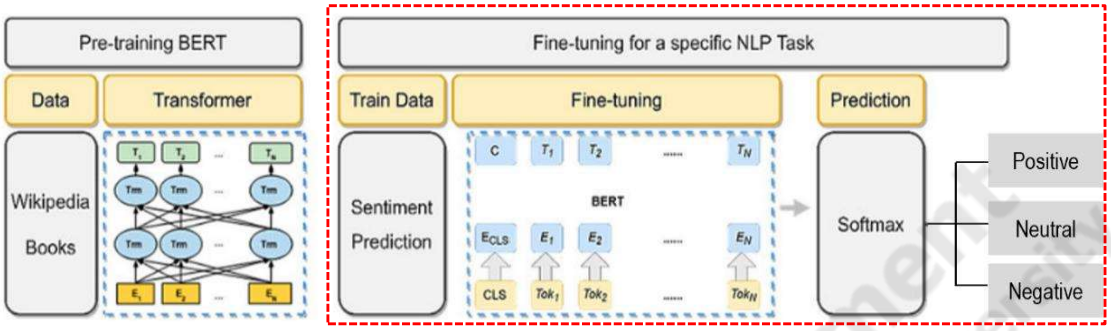
ในงานวิจัยฉบับนี้ใช้ชุดข้อมูลบทวิจารณ์โรงแรมจาก TripAdvisor ที่สามารถดาวน์โหลดได้จาก www.tripadvisor.com โดยมีลักษณะของข้อมูลดังภาพประกอบที่ 3.1

	A	B	C	D
1		Text	RateType	
2	0	The story started after we ate at Baoshuan and headed downstairs to exit the hotel with allsmiles. Then, tho	1	
3	1	This is worst and pathetic experience with The Oberoi New Delhi Zahir Hussain marg. Staffs doesn't know hc	1	
4	2	Very bad Experience with property ever in my life and there is two Staff Member Vikrant and Jatin they miss	1	
5	3	This is very bad property and pathatic services. Staff doesn't have manner to talk with customers. There is t	1	
6	4	On our second night, as we were about to enter the elevator, a guy in black from the front desk came rushir	1	
7	5	I checked into this hotel on Saturday and given below has been experienced Good experience The driver w	1	
8	6	Coming from Europe I chose the Oberoi due to its excellent reputation... Before departure getting an exrtra l	1	
9	7	Seems like I over rated / over expected from this big old brand an otherwise excellent engagement with the	1	
10	8	Wanted to stay at the hotel and after speaking to the Sales Director, a room was booked for me. On the day	1	
11	9	One of the worse dining experience as we booked for lunch buffet. The service was poor, empty dishes and	1	
12	10	The hotel was oldfashioned and outdated in a bad way. Even the superior rooms were small and desperatel	1	
13	11	First off we have stayed at the Oberoi on two different occasions while in Delhi and it truly is a beautiful hot	1	
14	12	After being left at the airport for over an hour only to get lost for two hours in a local taxi, the oberoi started	1	
15	13	I'm in Delhi every month for work but this was the first time I have stayed at the Oberoi which on this occas	1	
16	14	I had money 180 stolen from my room on the last day of our holiday. Luckily I had gone to put the AED in	1	
17	15	Worst experience I've ever had in a hotel in Dubai I will avoid staying at this hotel from the staff, services, a	1	
18	16	Service level was clearly shown before arrival when I tried to contact hotel and Accor customer service to up	1	
19	17	I frequently travel, 2 weeks every month and in different hotels experiencing different levels of services and	1	
20	18	On checking in the lovely male receptionist offered us an upgrade for 1000 dirims. In total After two nights	1	
21	19	We were looking forward to our stay at the Fairmont hotel also due to its pool, beach and garden which can	1	
22	20	Bad experience, the air conditioning is very, very, very bad, and there is a female employee in the reception	1	
23	21	My family and I checked in yesterday to the fairmont on the palm. Quite frankly, check in process was unpro	1	
24	22	We booked Gold rooms and expected to be on levels 8 or 9 as per the website. We were allocated rooms or	1	
25	23	READ BEFORE YOU TAKE YOUR FAMILY!Don't recommend this hotel for the price. I took my family of 5 her	1	
26	24	Probably the worst hotel Ive ever stayed in in Dubai. Arrived at 9am had to wait until 3pm for a room, I boo	1	
27	25	We have been regularly visiting Little Miss India, and the experience have always been nice. This time, it was	1	
28	26	This is the 56th time I have stayed at the Fairmont Palm in the last 8 years. There was been significant dete	1	

ภาพประกอบที่ 3.1 ชุดข้อมูลบทวิจารณ์โรงแรมจาก TripAdvisor

โดยข้อมูลที่ดาวน์โหลดมา มีจำนวน 30,145 บทวิจารณ์ โดยลักษณะของข้อมูลใน 1 ความคิดเห็นจะประกอบด้วย ข้อความแสดงบทวิจารณ์ อันดับคะแนนของบทวิจารณ์

3.2 กรอบการดำเนินงาน



ภาพประกอบที่ 3.2 กรอบการดำเนินงานของโมเดลทรานสฟอร์มเมอร์สำหรับการจำแนกความรู้สึก

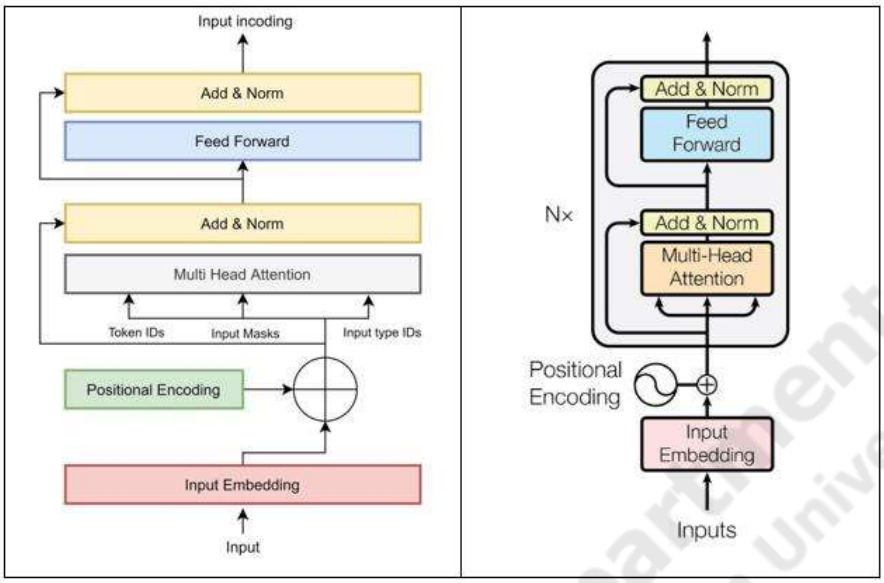
กรอบการดำเนินงานในการประยุกต์โมเดลแบบทรานสฟอร์มเมอร์สำหรับการจำแนกความรู้สึก แบ่งเป็น 2 ส่วนหลักๆ คือ

ส่วนที่ 1 คือ Pre-Training BERT คือการสร้างโมเดลด้วยข้อมูลเอกสารจำนวนมากๆ ซึ่งเป็นเอกสารที่ไม่จำเป็นต้องมีคลาสลาเบล เพื่อให้โมเดลเกิดความเข้าใจ Language Model นั่นคือ ทำให้โมเดลเข้าใจในลักษณะของภาษาหรือการใช้ภาษา

ส่วนที่ 2 คือ Fine-tuning for a specific NLP Task ซึ่งก็คือ การนำข้อมูลที่เกี่ยวข้องกับงานที่ต้องการประมวลผลมาทำการปรับค่าในโมเดลที่ได้จาก Pre-training โดยข้อมูลที่นำมา Fine-tuning ควรเป็นข้อมูลที่มีคลาสลาเบล เพื่อใช้ในการปรับ weight เพื่อให้กลายเป็น Decision Model ที่ต่อยอดมาจาก BERT จากนั้นจะนำ Decision Model ที่ได้จาก BERT ไปใช้ร่วมกับฟังชัน SoftMax เพื่อจำแนกความรู้สึกของเอกสารได้

3.3 Pre-training of BERT

จากรูปเป็นในการสร้างภาพประกอบที่ 3.3 Language Model แบบ BERT โดยมีขั้นตอนดังนี้



ภาพประกอบที่ 3.3 Language แบบ BERT

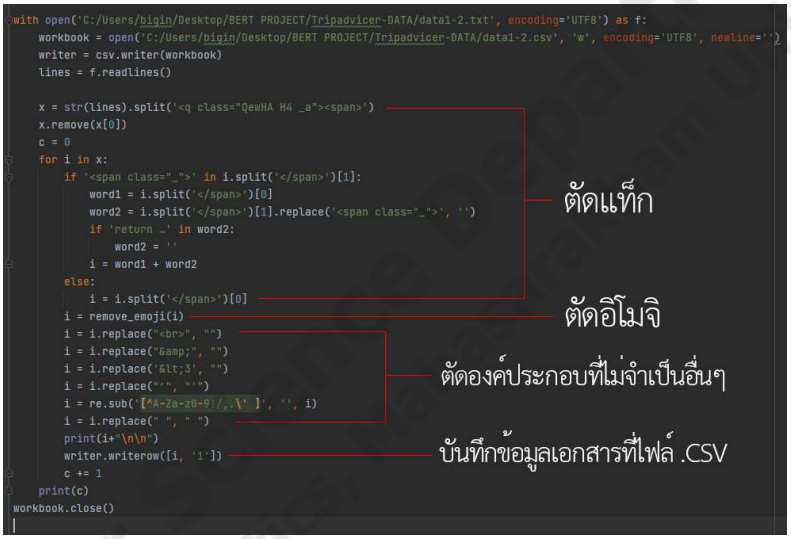
ที่มา : https://www.researchgate.net/figure/Transformer-Encoder-Architecture-BERT-or-Bidirectional-Encoder-Representations-from_fig1_349880253

ขั้นตอนที่ 1 : การอ่านข้อมูลและทำความสะอาดข้อมูล

ก่อนที่จะนำข้อมูลเอกสาร เข้าสู่กระบวนการตัดคำ เนื่องจากข้อมูลรวบรวมมากการเว็บ Tripadvisor ข้อมูลเอกสารเก็บในโครงสร้างแบบ HTML ซึ่งเป็นโครงสร้างข้อมูลที่มีแท็ก (Tag) กำกับเพื่ออธิบายข้อมูลเอกสาร สามารถแสดงตัวอย่างจากเว็บ Tripadvisor ได้จากภาพประกอบที่ 3.4 ตัวอย่างข้อมูลเอกสารจากเว็บ Tripadvisor

ภาพประกอบที่ 3.4 ตัวอย่างข้อมูลเอกสารจากเว็บ Tripadvisor

ซึ่งแท็กเหล่านี้เป็นส่วนที่ไม่มีความจำเป็นต่อการประมวล จึงจำเป็นต้องตัดแท็กเหล่านี้ออกจากข้อมูลเอกสารเสียก่อน ในขณะที่ส่วน “บริบทข้อความ (Context)” ซึ่งเป็นส่วนที่ใช้เพื่อเป็นข้อมูลเอกสาร ยังมีองค์ประกอบบางส่วนที่ไม่มีความจำเป็นต่อการประมวลผล ยกตัวอย่างเช่น อิโมจิ (Emoji) อักษรพิเศษเป็นต้น จึงจำเป็นต้องตัดองค์ประกอบเหล่านี้ออกด้วย ซึ่งกระบวนการตัดแท็กและตัดองค์ประกอบบางส่วนที่ไม่จำเป็นของข้อมูลเอกสาร สามารถทำได้ด้วยการนำข้อมูลโครงสร้าง HTML เข้าสู่โปรแกรมสำหรับตัดคำ เมื่อข้อมูลโครงสร้าง HTML ผ่านกระบวนการตัดคำจนได้ข้อมูลเอกสารที่พร้อมสำหรับการประมวลผลแล้ว จะนำข้อมูลเอกสารไปเก็บไว้ที่ไฟล์ .CSV สามารถแสดงขั้นตอนต่างๆ ได้จากภาพประกอบที่ 3.5



ภาพประกอบที่ 3.5 ตัวอย่างโปรแกรมสำหรับตัดคำ

ขั้นตอนที่ 2 : การตัดคำ (Tokenization)

การตัดคำเพื่อสร้างโมเดลของ BERT จะใช้ WordPiece Tokenizer ในการแยกข้อความออกเป็น “คำ” ให้อยู่ในรูปแบบที่เรียกว่า “รูปแบบของคำเต็ม (Full Form)” หรือเป็น “ชิ้นส่วนของคำ (Word Pieces)” การใช้ WordPiece Tokenizer จะช่วยลดจำนวนคลังในคลังคำศัพท์ และแก้ปัญหา Out-of-Vocabulary (OOV) [5] อีกด้วย

ในกรณี แยกข้อความออกเป็น “คำ” ให้อยู่ในรูปแบบที่เรียกว่า “รูปแบบของคำเต็ม (Full Form)” นั่นคือหนึ่งคำที่ตัดได้จะพิจารณาเป็นหนึ่งโทเค็น (Token) ซึ่งการตัดคำแบบนี้จะอิงตามคลังคำของ WordPiece โดยมีอยู่ทั้งสิ้นจำนวน 30,000 คำ [18] สามารถแสดงตัวอย่างคำของ WordPiece ใน “รูปแบบของคำเต็ม” ได้ดังตารางที่ 3.1

ตารางที่ 3.1 การตัดคำด้วย WordPiece ที่ได้รูปแบบของคำเต็ม

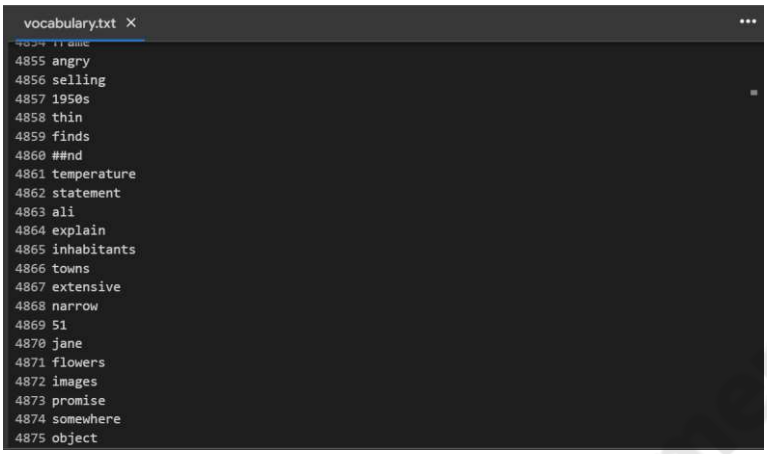
Word	Token(s)
surf	[‘surf’]
snow	[‘snow’]

ในกรณีที่ทำการตัดคำแบบแยกข้อความออกเป็น “ชิ้นส่วนของคำ” นั้น มักพบในการตัดคำที่เป็นคำผสม หรือคำที่มีการเติม Prefix หรือ Suffix ในคำที่เป็นคำดั้งเดิมหรือรากศัพท์ (Root Form) หรือเป็นคำนามเฉพาะ เป็นต้น นั่นคือหนึ่งชิ้นส่วนของคำที่ตัดได้จะพิจารณาเป็นหนึ่งโทเค็น โดย “ชิ้นส่วนของคำ” ที่มาผสมเข้าไป หรือ เป็นส่วนของ Prefix หรือ Suffix ในคำต้นฉบับ หรือชิ้นส่วนของคำนามเฉพาะ จะมีการเติม ## หน้าคำเหล่านั้น เพื่อบ่งบอกว่าเป็นชิ้นส่วนคำที่ถูกตัดออกมาและไม่ใช่คำที่อยู่ในคลังคำของ WordPiece สามารถแสดงตัวอย่างการแยก “คำ” และ “ชิ้นส่วนของคำ” ได้ดังตารางที่ 3.2

ตารางที่ 3.2 การตัดคำด้วย WordPiece ที่ได้รูปแบบของชิ้นส่วนของคำ

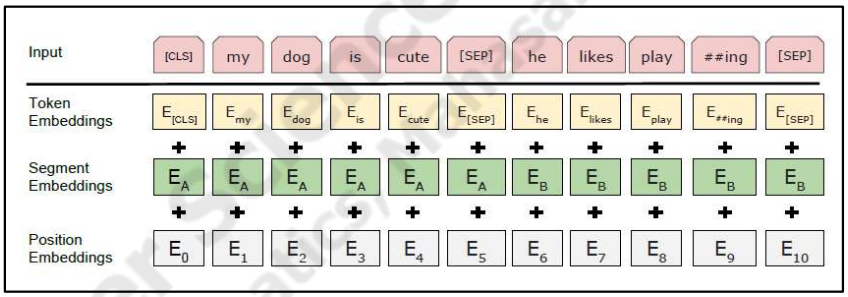
Word	Token(s)
surfing	[‘surf’, ‘##ing’]
surfboarding	['surf', '##board', '##ing']
snowboard	[‘snow’, ‘##board’]
snowboarding	['snow', '##board', '##ing']
Kiatnumchai	[‘Kia’, ‘##t’, ‘##num’, ‘##chai’]

การตัดคำแบบแยกข้อความออกเป็น “ชิ้นส่วนของคำ” นั้นมีกระบวนการตัดคำแบบ greedy algorithm [21] คือจะตัดให้ได้มากที่สุด ที่สามารถเป็นไปได้ในคลังคำศัพท์ เช่น คำว่า ‘surfing’ ในคลังคำศัพท์มีคำว่า ‘surf’ แต่ไม่มีคำว่า ‘surfi’ ในคลังคำศัพท์จึงตัดได้คำว่า ‘surf’ ซึ่งเป็นคำที่สามารถที่สุดที่สามารถตัดได้และมีในคลังคำศัพท์ (สามารถตัวอย่างคำใน “คำ” ในคลังคำศัพท์ได้ดังภาพประกอบที่ 3.6)



ภาพประกอบที่ 3.6 ตัวอย่างคำใน “คำ” ในคลังคำศัพท์

เมื่อผ่านกระบวนการตัดคำแล้ว ในขั้นตอนถัดไปจะเป็นการเตรียมแต่ละโทเค็นในข้อมูลเอกสารให้พร้อมที่จะเข้าสู่โมเดลแบบ BERT ซึ่งโมเดลแบบ BERT จะมีการทำงานทั้งหมด 3 ขั้นตอนดังนี้ (สามารถแสดงขั้นตอนทั้งหมดได้ดังภาพประกอบที่ 3.7)



ภาพประกอบที่ 3.7 ขั้นตอนการเตรียมแต่ละโทเค็นในข้อมูลเอกสารก่อนเข้าสู่โมเดลแบบ BERT

ขั้นตอนที่ 3 : Token Embeddings

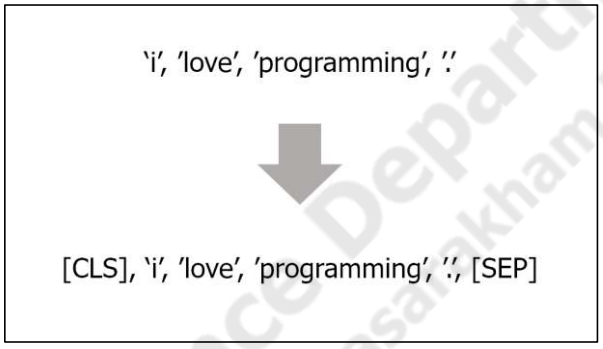
Token embedding เป็นกระบวนกระเปลี่ยนคำให้กลายเป็นตัวเลขรูปแบบเวกเตอร์ โดยใช้หลักการของ WordPiece Embedding จะเปลี่ยนคำแต่ละที่จะได้การตัดคำเป็นตัวเลขจำนวนเต็มที่สามารถบ่งบอกว่าคำเหล่านั้นจะถูกเปลี่ยนเป็นเวกเตอร์ใดใน lookup table ของคลังคำศัพท์ ซึ่งมีจำนวนคำศัพท์อยู่ 30,000 คำศัพท์ ในขณะที่ขนาดของเวกเตอร์ใน lookup table มีขนาด 768 มิติ เพื่อที่จะสามารถแบ่งเวกเตอร์ของแต่ละคำเป็น 12 ส่วนเพื่อให้สอดคล้องกับจำนวน Head Attention ที่ใช้ในขั้นตอนของ Multi-Head Attention และก่อนที่จะทำกระบวนการในกาทำ Token embedding โมเดล BERT จะเพิ่มโทเค็นพิเศษนั้นคือ โทเค็น [CLS] ไว้หน้าสุดของเอกสารข้อความ และเพิ่มโทเค็น [SEP] โดยที่

โทเค็น [CLS] เป็นโทเค็นพิเศษสำหรับการจำแนกประเภท (Classification) ถูกใช้เป็นตัวแทนของข้อมูลเอกสารนั้นๆ สำหรับงานด้านการจำแนกประเภท

โทเค็น [SEP] เป็นโทเค็นพิเศษสำหรับแยกประโยคหรือสิ้นสุดประโยค เพื่อให้โมเดลรับรูปว่าตำใดเป็นของประโยคใด

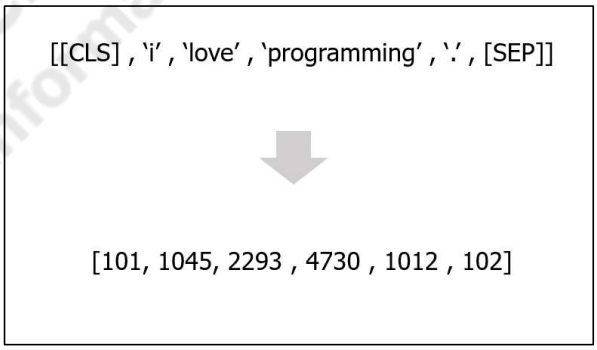
การทำ Token embedding ด้วย WordPiece Embedding มีขั้นตอนดังนี้

(1) เพิ่มโทเค็น [CLS] ไว้ข้างหน้าสุดและเพิ่มโทเค็น [SEP] ไว้ข้างหน้าสุดของข้อมูลเอกสารดังภาพประกอบที่ 3.8



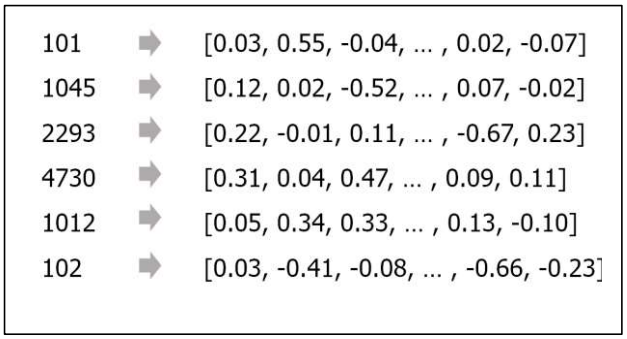
ภาพประกอบที่ 3.8 การเพิ่มโทเค็มพิเศษ [CLS] และ [SEP]

(2) เปลี่ยนแต่ละคำให้เป็นตัวเลขจำนวนเต็มสำหรับบอกที่อยู่ของเวกเตอร์ใน lookup table ดังภาพประกอบที่ 3.9



ภาพประกอบที่ 3.9 การเปลี่ยนคำให้เป็นตัวเลขสำหรับบอกที่อยู่ของเวกเตอร์ใน lookup table

(3) นำตัวเลขของแต่ละคำที่เป็นเหมือนตัวบ่งบอกที่อยู่ของเวกเตอร์ใน lookup table เข้าตรวจสอบกับ lookup table เพื่อดึงเวกเตอร์ขนาด 786 มิติที่ของแต่ละคำในข้อมูลเอกสารดังภาพประกอบที่ 3.10



ภาพประกอบที่ 3.10 การดึงเวกเตอร์ขนาด 786 มิติที่ของแต่ละคำในข้อมูลเอกสาร

เมื่อทำขั้นตอนข้างต้นเสร็จแล้ว เราจะได้กลุ่มของเวกเตอร์ที่บ่งบอกถึงคำต่างๆในข้อมูลเอกสาร และสามารถแสดงออกมาในรูปแบบเมทริกซ์ได้ดังตารางที่ 3.3

ตารางที่ 3.3 เมทริกซ์แสดงเวกเตอร์ของแต่ละคำในข้อมูลเอกสาร

	d_0	d_1	d_2	$d_{...}$	d_{766}	d_{767}
[CLS]	0.03	-0.55	-0.04	...	0.02	-0.07
i	0.12	0.02	-0.52	...	0.07	-0.02
love	0.22	-0.01	0.11	...	-0.67	0.23
programming	0.31	0.04	0.47	...	0.09	0.11
.	0.05	0.34	0.33	...	0.13	-0.10
[SEP]	0.03	-0.41	-0.08	...	-0.66	-0.23

ขั้นตอนที่ 5 : Segment Embeddings

Segment Embeddings คือการฝั่งตำแหน่งของประโยคในข้อมูลเอกสารเพื่อให้โมเดลรับรู้ว่าคำไหนอยู่ในประโยคลำดับที่เท่าใดในข้อมูลเอกสาร ซึ่งโมเดลแบบ BERT จะสามารถรับประโยคเข้าสู่โมเดลได้ครั้งละ 2 ประโยค [23] หากกำหนดให้ “i love programming.” เป็นประโยคแรกในชุดข้อมูลเอกสาร ประโยคนี้จะถูกฝัง Segment Embeddings ให้เป็นเวกเตอร์ขนาด 768 มิติโดยที่ทุกมิติมีค่าเท่ากับ 0 (index ที่ 0) และกำหนดให้ “i love my job.” เป็นประโยคที่ 2 ในชุดข้อมูลเอกสารประโยคนี้จะถูกฝัง Segment Embeddings ให้เป็นเวกเตอร์ขนาด 768 มิติโดยที่ทุกมิติมีค่าเท่ากับ 1 (index ที่ 1)

ตารางที่ 3.4 ตัวอย่างการสร้าง Segment Embeddings

	d_0	d_1	d_2	$d_{...}$	d_{766}	d_{767}
[CLS]	0	0	0	...	0	0
i	0	0	0	...	0	0
love	0	0	0	...	0	0
programming	0	0	0	...	0	0
.	0	0	0	...	0	0
[SEP]	0	0	0	...	0	0
i	1	1	1	...	1	1
love	1	1	1	...	1	1
my	1	1	1	...	1	1
job	1	1	1	...	1	1
.	1	1	1	...	1	1
[SEP]	1	1	1	...	1	1

ขั้นตอนที่ 4 : Positional encoding

Positional encoding เนื่องด้วยว่า Transformer ตัดการทำ recurrent (จาก RNNs) และ convolute (จาก CNNs) ออกไป ปัญหาที่เกิดขึ้นคือ ในขณะที่กำลังประมวลผลคำใดคำหนึ่งอยู่ โมเดลไม่มีสิ่งที่จะช่วยในการระบุตำแหน่งของคำปัจจุบัน ว่าตอนนี้กำลังประมวลผลคำที่เท่าไรในประโยค คำไหนอยู่ด้านหน้า คำไหนอยู่ด้านหลัง จึงได้มีการเพิ่มส่วนของ Positional Embedding เข้าไป เพื่อช่วยให้โมเดลสามารถรับรู้ตำแหน่งของคำที่กำลังพิจารณาอยู่ ซึ่งใช้ค่า sin/cos ในการแทน position ต่างๆ ตามสมการด้านต่อไปนี้

$$\mathrm{PE}_{(\mathrm{pos},2\mathrm{i})} = \sin(\mathrm{pos}/10000^{2\mathrm{i}/\mathrm{d}_{\mathrm{model}}}) \quad 3.1$$

$$\mathrm{PE}_{(\mathrm{pos},2\mathrm{i}+1)} = \cos(\mathrm{pos}/10000^{2\mathrm{i}/\mathrm{d}_{\mathrm{model}}}) \quad 3.2$$

โดย *pos* คือตำแหน่งของคำในเอกสาร

i คือตำแหน่งของแต่ละข้อมูลที่อยู่ในเวกเตอร์ของแต่ละคำ

d_{model} คือขนาดของเวกเตอร์ของคำ

จากสมการข้างต้นสามารถแจกแจงการแทนสมการให้กับแต่ละเวกเตอร์ได้ดังตารางที่ 3.5

ตารางที่ 3.5 ค่า Positional encoding ของแต่ละเวกเตอร์ในข้อมูลเอกสาร

	d_0	d_1	d_2	$d_{...}$	d_{766}	d_{767}
[CLS]	$\sin\left(\frac{0}{10000^{\frac{0}{768}}}\right)$	$\cos\left(\frac{0}{10000^{\frac{0}{768}}}\right)$	$\sin\left(\frac{0}{10000^{\frac{1}{768}}}\right)$	...	$\sin\left(\frac{0}{10000^{\frac{766}{768}}}\right)$	$cos\left(\frac{0}{10000^{\frac{766}{768}}}\right)$
i	$\sin\left(\frac{1}{10000^{\frac{0}{768}}}\right)$	$\cos\left(\frac{1}{10000^{\frac{0}{768}}}\right)$	$\sin\left(\frac{1}{10000^{\frac{1}{768}}}\right)$	...	$\sin\left(\frac{1}{10000^{\frac{766}{768}}}\right)$	$cos\left(\frac{1}{10000^{\frac{766}{768}}}\right)$
love	$\sin\left(\frac{2}{10000^{\frac{0}{768}}}\right)$	$\cos\left(\frac{2}{10000^{\frac{0}{768}}}\right)$	$\sin\left(\frac{2}{10000^{\frac{1}{768}}}\right)$	...	$\sin\left(\frac{2}{10000^{\frac{766}{768}}}\right)$	$cos\left(\frac{2}{10000^{\frac{766}{768}}}\right)$
programming	$\sin\left(\frac{3}{10000^{\frac{0}{768}}}\right)$	$\cos\left(\frac{3}{10000^{\frac{0}{768}}}\right)$	$\sin\left(\frac{3}{10000^{\frac{1}{768}}}\right)$	...	$\sin\left(\frac{3}{10000^{\frac{766}{768}}}\right)$	$cos\left(\frac{3}{10000^{\frac{766}{768}}}\right)$
.	$\sin\left(\frac{4}{10000^{\frac{0}{768}}}\right)$	$\cos\left(\frac{4}{10000^{\frac{0}{768}}}\right)$	$\sin\left(\frac{4}{10000^{\frac{1}{768}}}\right)$	...	$\sin\left(\frac{4}{10000^{\frac{766}{768}}}\right)$	$cos\left(\frac{4}{10000^{\frac{766}{768}}}\right)$
[SEP]	$\sin\left(\frac{5}{10000^{\frac{0}{768}}}\right)$	$\cos\left(\frac{5}{10000^{\frac{0}{768}}}\right)$	$\sin\left(\frac{5}{10000^{\frac{1}{768}}}\right)$	...	$\sin\left(\frac{5}{10000^{\frac{766}{768}}}\right)$	$cos\left(\frac{5}{10000^{\frac{766}{768}}}\right)$

จากตารางที่ 3.5 สามารถคำนวณค่า Positional encoding ตามสมการที่ 3.1 และ สมการที่ 3.2 ได้ดังนี้

คำนวณค่า Positional encoding ของ [CLS]

$PE_{(0,0)} = \sin(0/10000^{0/768})$
$= \sin(0)$
$= 0$

$PE_{(0,1)} = \cos(0/10000^{0/768})$
$= \cos(0)$
$= 1$

$PE_{(0,2)} = \sin(0/10000^{2/768})$
$= \sin(0)$
$= 0$

$PE_{(0,766)} = \sin(0/10000^{766/768})$
$= \sin(0)$
$= 0$

$PE_{(0,767)} = \cos(0/10000^{766/768})$
$= \cos(0)$
$= 1$

คำนวณค่า Positional encoding ของ “i”

$PE_{(1,0)} = \sin(1/10000^{0/768})$
$= \sin(1)$
$= 0.8415$

$PE_{(1,1)} = \cos(1/10000^{0/768})$
$= \cos(1)$
$= 0.5403$

$PE_{(1,2)} = \sin(1/10000^{2/768})$
$= \sin(0.9763)$
$= 0.8284$

$PE_{(1,766)} = \sin(1/10000^{766/768})$

$= \sin(0.0001)$

$= 0.0001$

$PE_{(1,767)} = \cos(1/10000^{766/768})$

$= \cos(0.0001)$

$= 0.9999$

คำนวณค่า Positional encoding ของ “love”

$PE_{(2,0)} = \sin(2/10000^{0/768})$

$= \sin(2)$

$= 0.9093$

$PE_{(2,1)} = \cos(2/10000^{0/768})$

$= \cos(2)$

$= -0.4161$

$PE_{(2,2)} = \sin(2/10000^{2/768})$

$= \sin(1.9526)$

$= 0.9280$

$PE_{(2,766)} = \sin(1/10000^{766/768})$

$= \sin(0.0002)$

$= 0.0002$

$PE_{(2,767)} = \cos(1/10000^{766/768})$

$= \cos(0.0002)$

$= 0.9999$

คำนวณค่า Positional encoding ของ “programming”

$PE_{(3,0)} = \sin(3/10000^{0/768})$

$= \sin(3)$

$= 0.1411$

$PE_{(3,1)} = \cos(3/10000^{0/768})$

$= \cos(2)$

$= -0.9899$

$PE_{(3,2)} = \sin(3/10000^{2/768})$

= sin (2.9289)

= 0.2111

$PE_{(3,766)}$ = sin ($3/10000^{766/768}$)

= sin (0.0003)

= 0.0003

$PE_{(3,767)}$ = cos ($3/10000^{766/768}$)

= cos (0.0003)

= 0.9999

คำนวณค่า Positional encoding ของ “.”

$PE_{(4,0)}$ = sin ($4/10000^{0/768}$)

= sin (4)

= -0.7568

$PE_{(4,1)}$ = cos ($4/10000^{0/768}$)

= cos (4)

= -0.6536

$PE_{(4,2)}$ = sin ($4/10000^{2/768}$)

= sin (3.9052)

= -0.6915

$PE_{(4,766)}$ = sin ($4/10000^{766/768}$)

= sin (0.0004)

= 0.0004

$PE_{(4,767)}$ = cos ($4/10000^{766/768}$)

= cos (0.0004)

= 0.9999

คำนวณค่า Positional encoding ของ [SEP]

$PE_{(4,0)}$ = sin (5/100000/768)

= sin (5)

= -0.9589

$PE_{(4,1)}$ = cos ($5/10000^{0/768}$)

$= \cos(5)$

$= 0.2837$

$PE_{(4,2)} = \sin(5/10000^{2/768})$

$= \sin(4.8815)$

$= -0.9857$

$PE_{(4,766)} = \sin(5/10000^{766/768})$

$= \sin(0.0005)$

$= 0.0005$

$PE_{(4,767)} = \cos(5/10000^{766/768})$

$= \cos(0.0005)$

$= 0.9999$

จากนั้นนำค่าของจากเวกเตอร์ของ Positional Encoding ของแต่ละตำแหน่งในข้อมูลเอกสารไปบวกเข้ากับค่าของจากเวกเตอร์ของ Word Embedding ของแต่ละตำแหน่งของข้อมูลในเอกสาร ซึ่งสามารถแสดงการคำนวนอย่างง่ายได้ดังตารางที่ 3.6 และสามารถแสดงผลรวมของค่า Positional encoding, Token Embeddings และ Segment Embeddings (แสดงได้ดังตารางที่ 3.7)

ตารางที่ 3.6 ผลรวมของค่า Positional encoding, Token Embedding, Segment Embedding

	d_0	d_1	d_2	$d_{...}$	d_{766}	d_{767}
[CLS]	0.03 + 0 + 0	-0.55 + 1 + 0	-0.04 + 0 + 0	...	0.02 + 0 + 0	-0.07 + 1 + 0
i	0.12 + 0.8415 + 0	0.02 + 0.54 + 0	-0.52 + 0.8284 + 0	...	0.07 + 0.0001 + 0	-0.02 + 0.9999 + 0
love	0.22 + 0.9093 + 0	-0.01 + (-0.4161) + 0	0.11 + 0.9280 + 0	...	-0.67 + 0.0002 + 0	0.23 + 0.9999 + 0
programming	0.31 + 0.1411 + 0	0.04 + (-0.9899) + 0	0.47 + 0.2111 + 0	...	0.09 + 0.0003 + 0	0.11 + 0.9999 + 0
.	0.05 + (-0.7568) + 0	0.34 + (-0.6539) + 0	0.33 + (-0.6915) + 0	...	0.13 + 0.0004 + 0	-0.10 + 0.9999 + 0
[SEP]	0.03 + (-0.9589) + 0	-0.41 + 0.2837 + 0	-0.08 + (-0.9857) + 0	...	-0.66 + (-0.0005) + 0	-0.23 + 0.9999 + 0

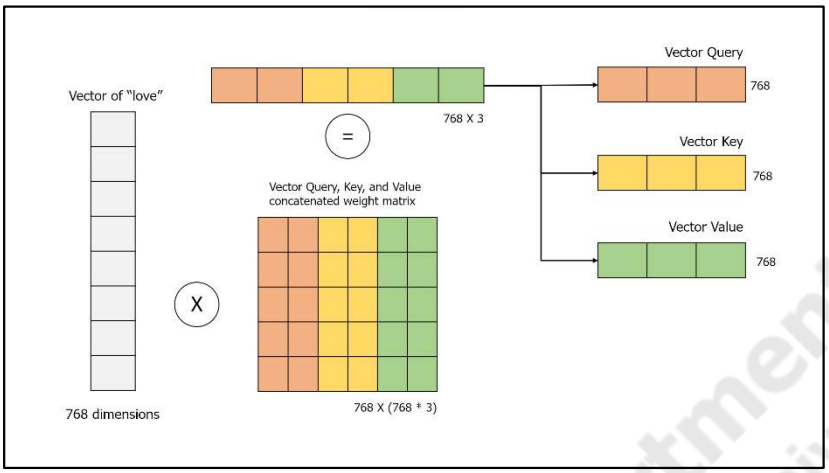
ตารางที่ 3.7 ผลลัพธ์ของผลรวมค่า Positional encoding และ Word embedding

	d_0	d_1	d_2	$d_{...}$	d_{766}	d_{767}
[CLS]	0.03	0.45	-0.04	...	0.02	0.93
i	0.9615	0.56	0.3084	...	0.0701	0.9799
love	1.1293	-0.4261	1.038	...	-0.6698	1.2299
programming	0.4511	-0.9499	0.6811	...	0.0903	1.1099
.	-0.7068	-0.3139	-0.3615	...	0.1304	0.8999
[SEP]	-0.9289	-0.1263	-1.0657	...	-0.6605	0.7699

ขั้นตอนที่ 5: Multi-Head Attention

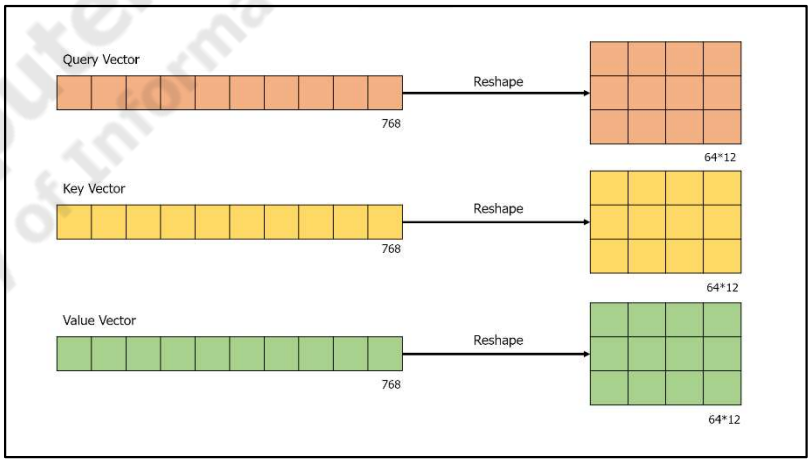
Multi-Head Attention คือกระบวนการที่มี Self-Attention ได้พร้อม ๆ กัน โดยเอาหลาย ๆ Self-Attention มาต่อขนานกันไปเพื่อให้โมเดลสามารถเรียนรู้ความสัมพันธ์ได้หลายมุมมองไปพร้อมๆ กัน ตัวอย่างเช่น “Bobby takes his cats to the park because they love climbing trees.” จะเห็นได้ว่า ‘cats’ มีความสัมพันธ์กับ ‘Bobby + takes’ ในฐานะกรรมของประโยค และคำว่า ‘cats’ ยังมีความสัมพันธ์กับ ‘love + climbing’ ในฐานะที่เป็นประธาน (ในที่นี้คือคำว่า ‘they’ ที่อ้างอิงกลับไปที่คำว่า ‘cats’) และในเชิงความหมาย ‘climbing trees’ มีความสัมพันธ์กับ ‘cats’ เพราะว่าเป็นสิ่งที่พวกมันชอบ แต่ก็ยังมีความหมายโดยนัยว่า ‘the park’ ต้องมี ‘trees’ เพื่อให้พวกมันปีนได้อีกด้วย จากตัวอย่างที่กล่าวข้างต้นทำให้เพียง 1 Self-Attention head ซึ่งสามารถเรียนรู้ได้แค่ความสัมพันธ์รูปแบบเดียว ไม่พอในการเรียนรู้ภาพรวมของทั้งประโยค ขั้นตอนการทำงานของ Multi-Head Attention โดยพิจารณาที่ทำว่า ‘love’ มีดังนี้

(1) สร้างเวกเตอร์ Query (Q), เวกเตอร์ Key (K) และเวกเตอร์ Value (V) เพื่อให้ได้เวกเตอร์ 3 ตัวคือ เวกเตอร์ Query (Q), เวกเตอร์ Key (K) และเวกเตอร์ Value (V) ซึ่งเป็นส่วนสำคัญในการทำ Attention Mechanism ในโอกาสต่อไป การที่จะได้เวกเตอร์ Query (Q), Key (K) และ Value (V) นั้น โดยจะนำเวกเตอร์ที่ได้จากขั้นตอนที่ 1 มาคูณกับ Weight Vector ที่ได้มาจากการสอน (Train) ก็จะได้เวกเตอร์ลัพธ์ขนาด 768 x 3 จากนั้นทำการแยกเวกเตอร์นี้ออกเป็นเวกเตอร์ย่อย 3 เวกเตอร์ได้แก่ เวกเตอร์ Query (Q), เวกเตอร์ Key (K) และเวกเตอร์ Value (V) โดยจะมีขนาดเท่ากับเวกเตอร์ของคำว่า “love” ซึ่งมีขนาดเท่ากับ 786 มิติ (ดังแสดงในภาพประกอบที่ 3.11)



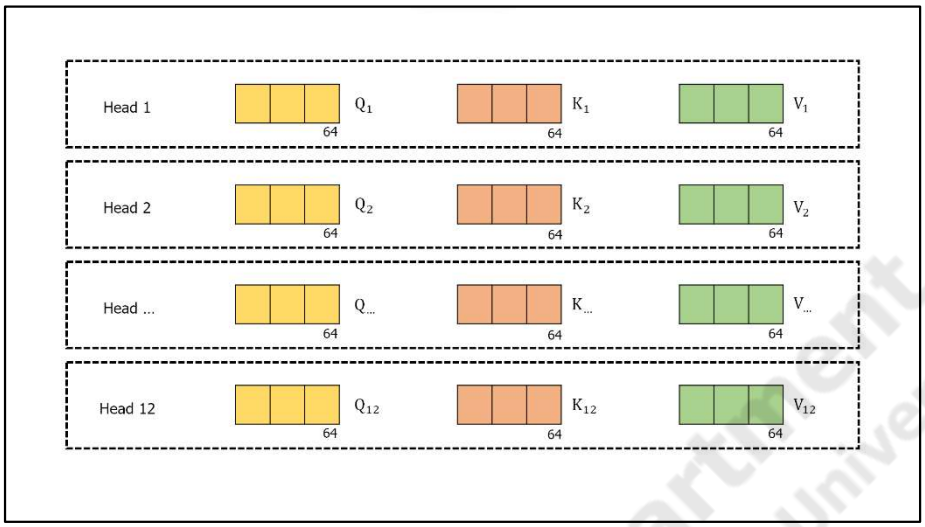
ภาพประกอบที่ 3.11 สร้างเวกเตอร์ Query (Q), Key (K) และ Value (V)

(2) แบ่งเวกเตอร์ให้สอดคล้องกับจำนวน Attention Head สมมติว่าในเลเยอร์นี้กำหนดให้มี Attention Head (h) เท่ากับ 12 ดังนั้นก็ต้องทำการแบ่งเวกเตอร์ Q, เวกเตอร์ K และเวกเตอร์ V ออกเป็น 12 ส่วนเท่าๆ กัน โดยให้เท่ากับจำนวนของ Attention Head ดังนั้นเมื่อเวกเตอร์มีขนาด 768 และจำนวน Attention Head (h) เท่ากับ 12 ดังนั้น ดังนั้น 768/12 ก็จะกับ 64 นั่นคือแต่ละ Attention Head ก็จะได้รับเวกเตอร์ที่มีความยาวขนาด 64 ไปทั้ง 12 Attention Head ซึ่งทำเช่นนี้กับเวกเตอร์ Q, เวกเตอร์ K และเวกเตอร์ V ผลลัพธ์สุดท้ายคือได้เมทริกซ์ 1 ตัว ซึ่งมีขนาด 64 x 12 ซึ่งก็คือขนาดของเวกเตอร์ Q, เวกเตอร์ K และเวกเตอร์ V (d_k, d_v) (ดังภาพประกอบที่ 3.12)



ภาพประกอบที่ 3.12 การแบ่งเวกเตอร์ Q, K และ V ที่มีขนาดเท่ากันให้กับแต่ละ Attention Head

จากนั้นทำการแบ่งของเมทริกซ์ของ Q, K และ V ออกเป็น 12 ส่วนเท่าๆ กัน แล้วจัดสรรให้กับ Attention Head แต่ละตัว ซึ่งในที่นี้มี Attention Head 12 ตัว (ดังภาพประกอบที่ 3.13 Attention Head แต่ละตัวที่ประกอบด้วยส่วนของเวกเตอร์ Q, เวกเตอร์ K และ V)



ภาพประกอบที่ 3.13 Attention Head แต่ละตัวที่ประกอบด้วยส่วนของเวกเตอร์ Q, K และ V

(3) คำนวณหา Attention Score ของแต่ละ Attention Head กระบวนการที่เกิดในแต่ละ Attention Head จะมีลักษณะเหมือนกันในทุก Attention Head ดังนั้นจะแสดงตัวอย่างเฉพาะใน Attention Head ที่ 1 เพราะการทำงานใน Attention Head อื่นๆ จะต่างกันเพียงตัวเวกเตอร์ที่นำเข้าเท่านั้น ซึ่งการคำนวณหา Attention Score มีสมการการคำนวณดังนี้

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_x}})V \quad (3.3)$$

โดย Q คือเวกเตอร์ของคำขอเพื่อเข้าถึงข้อมูล (Query Vector)

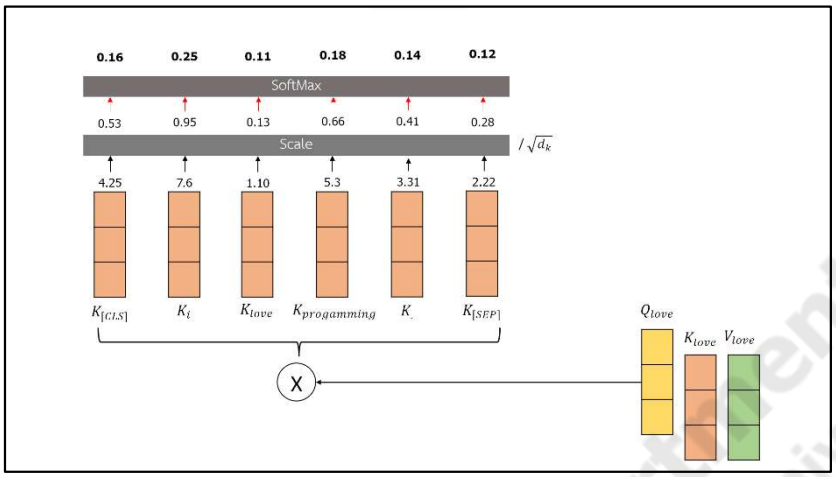
K คือเวกเตอร์ของสิ่งที่บ่งบอกถึงข้อมูล (Key Vector)

V คือเวกเตอร์ของข้อมูล (Value Vector)

T คือตัวเลขที่แสดงตำแหน่งของคำในข้อความเอกสาร

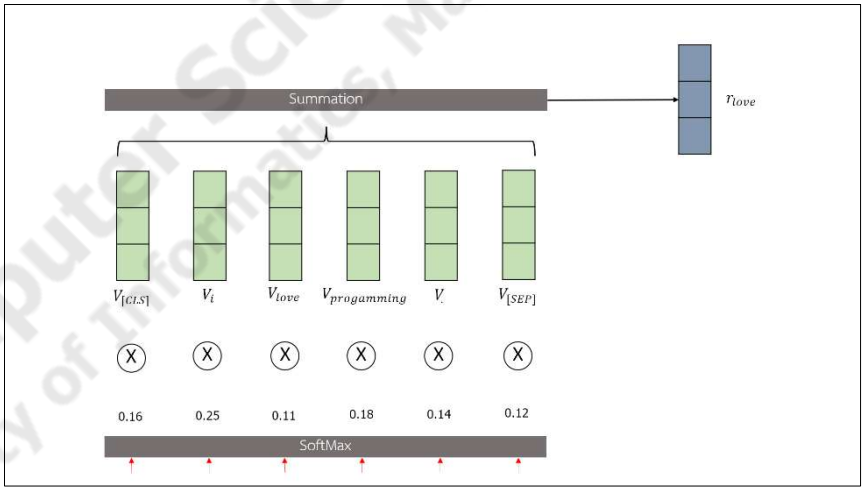
d_k คือขนาดเวกเตอร์ของเวกเตอร์ K

ลำดับแรก คือการนำเวกเตอร์ Q ของคำที่กำลังพิจารณาคือคำว่า “love” ไปคูณกับเวกเตอร์ K ของทุกคำในข้อมูลเอกสาร ซึ่งได้แก่คำว่า [CLS], “I”, “love”, “programming” , “.” และ [SEP] เมื่อได้ผลของการคูณมาแล้วจะนำไป scale ด้วยการหารกับ √(d_k) ซึ่งก็คือ √64 นั้นเอง และนำเข้าฟังก์ชัน SoftMax เพื่อให้เลขที่ได้อยู่ในรูปแบบตัวเลขความน่าจะเป็น (ดังภาพประกอบที่ 3.14)



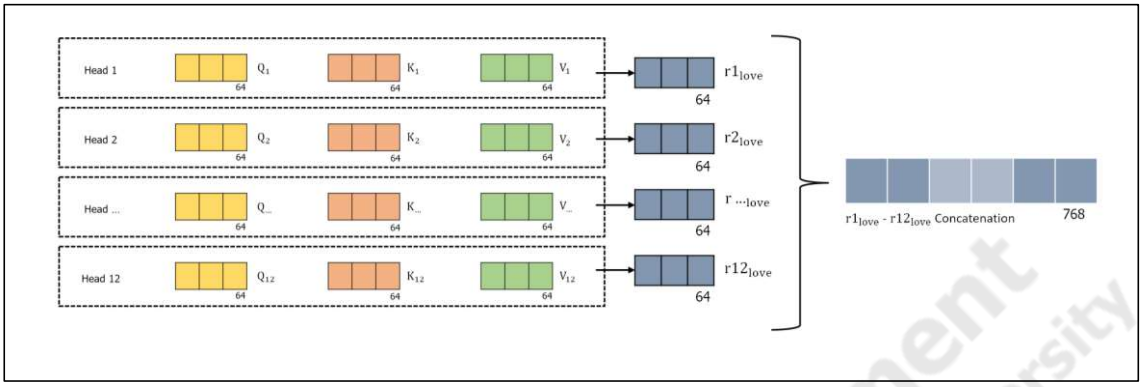
ภาพประกอบที่ 3.14 แสดงการคำนวณ Attention Score ของแต่ละ Attention Head

ลำดับต่อมานำค่าที่ได้ไปคูณกับเวกเตอร์ V ของทุกๆ คำ การคูณตรงนี้เป็นเสมือนการให้น้ำหนัก (Weight) ว่าควรพิจารณาคำไหนมากที่สุด ซึ่งคำว่า “*i*” จะมีค่ามากที่สุด เพราะค่า Score ที่นำไปคูณมีค่ามากที่สุด หลังจากที่นำ Score ไปคูณกับเวกเตอร์ *V* ทุกตัว ก็จะนำค่าที่ได้มาหาผลรวม (Sum) จากนั้นก็จะได้เวกเตอร์ *r* (ภาพประกอบที่ 3.15)



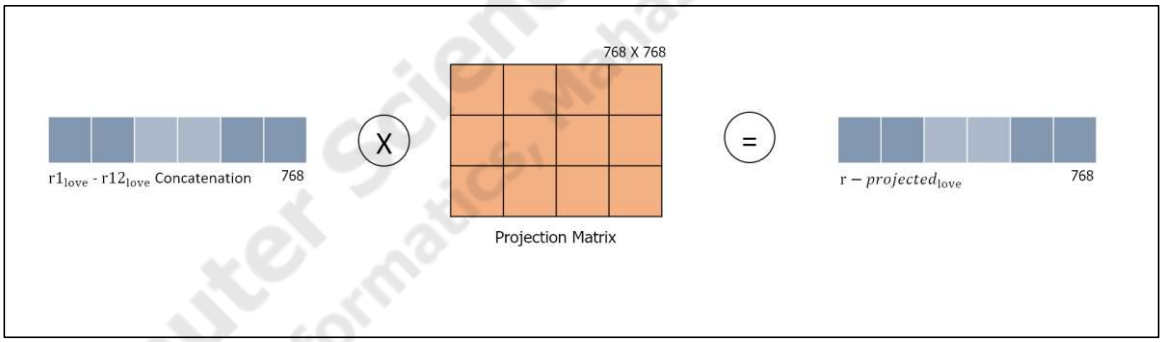
ภาพประกอบที่ 3.15 การคูณเวกเตอร์ V และการสร้างเวกเตอร์ r

(4) การรวบรวมเวกเตอร์ r เอาต์พุตจากแต่ละ Attention Head หลังจากที่ได้เวกเตอร์ r ครบทุก Attention Head ซึ่งขนาดเวกเตอร์ r จะมีขนาดเท่ากับ 64 ตามขนาดของเวกเตอร์ถูกแบ่งไว้ก่อนหน้า ในตัวอย่างนี้จะได้เวกเตอร์ r มา 12 ตัว (ดังภาพประกอบที่ 3.17) จากนั้นเอาเวกเตอร์ r1 ถึงเวกเตอร์ r12 มา concatenation กัน ให้ได้เวกเตอร์เดียวที่จะมีขนาดเท่ากับเวกเตอร์ของคำตั้งต้น ซึ่งมีขนาดเท่ากับ 768



ภาพประกอบที่ 3.16 Concatenation เวกเตอร์ r ทุก Attention Head เข้าด้วยกัน

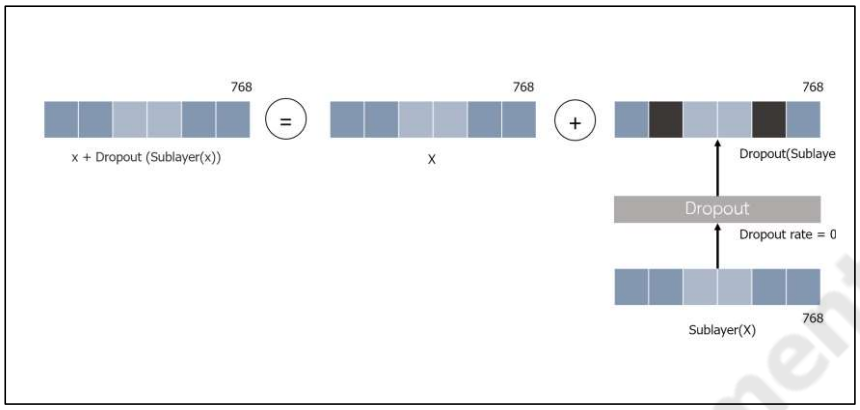
(5) คูณเวกเตอร์ r ด้วยเมทริกซ์ Projection คือนำเวกเตอร์ที่ได้จากนั้นตอนที่ 4 มาทำ Projection ด้วยการคูณกับเมทริกซ์ Projection ซึ่งเป็นเมทริกซ์ที่ได้จากการสอน จากนั้นก็จะได้เวกเตอร์มาตัวหนึ่ง สมมติให้ชื่อว่าเวกเตอร์ r-Projected ซึ่งมีขนาดเท่ากับ 768 เช่นเดิม ซึ่ง ณ จุดๆ นี้ถือว่าเป็นการสิ้นสุดการทำ Attention Mechanism (แสดงดังภาพประกอบที่ 3.17)



ภาพประกอบที่ 3.17 แสดงการสร้างเวกเตอร์ r-Projected

ขั้นตอนที่ 6 : Add & Norm

Add & Norm ในกระบวนการนี้มีการทำงานอยู่ทั้งหมด 2 ส่วนคือ Residual Connections (หรือที่รู้จักในชื่อ ResNet) คือการนำเวกเตอร์ก่อนหน้าที่จะเข้าสู่ Sublayer ซึ่งในที่นี้หมายถึง Multi-head Attention มาบวกกับเวกเตอร์ที่ถูกประมวลผลด้วย Sublayer มาแล้ว แต่ก่อนที่จะนำเวกเตอร์มาบวกกันได้นั้น จะต้องนำเวกเตอร์ที่ถูกประมวลผลด้วย Sublayer มาผ่านการทำ Dropout เสียก่อน โดย Dropout คือการสุ่มให้เวกเตอร์บางส่วนกลายเป็น 0 เพื่อแก้ปัญหา overfitting โมเดล โดยโมเดล Tramsformer นั้นตั้ง dropout rate = 0.1 จากนั้นจึงนำเวกเตอร์ทั้งสองตัวมาบวกกันดังสมการที่ 3.3 (แสดงดังภาพประกอบที่ 3.18)



ภาพประกอบที่ 3.18 กระบวนการทำ Residual Connections

ส่วนที่ 2 คือ Layer Normalization เมื่อได้ผลลัพธ์จากกระบวนการ Residual Connections ก็จะเข้าสู่กระบวนการ Layer Normalization โดยมีหลักการทั้งหมด 4 ขั้นตอนดังนี้

(1) คำนวนค่าเฉลี่ย (mean)

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{3.4}$$

โดย μ คือ ค่าเฉลี่ย

m คือ จำนวนข้อมูลในเวกเตอร์

x_i คือ ข้อมูลในเวกเตอร์ตัวที่ i

หากกำหนดให้เวกเตอร์ของ “love” มีข้อมูลดังนี้

ตารางที่ 3.8 ข้อมูลของเวกเตอร์ “love”

love	0.6	0.23	-0.11	0.45	0.37	-0.27

$$\mu = \frac{(0.6 + 0.23 + (-0.11) + 0.45 + 0.37 + (-0.27))}{6}$$

$$\mu = \frac{1.27}{6}$$

$$\mu = 0.2117$$

(2) หาค่าความแปรปรวน (variance)

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu)^2 \quad (3.5)$$

โดย σ^2 คือ ค่าความแปรปรวน

μ คือ ค่าเฉลี่ย

m คือ จำนวนข้อมูลในเวกเตอร์

x_i คือ ข้อมูลในเวกเตอร์ตัวที่ i

$$\sigma^2 = \frac{1}{6}\left((0.6\text{-}0.2117)^2+(0.23\text{-}0.2117)^2+(-0.11\text{-}0.2117)^2 +(0.45\text{-}0.2117)^2+(0.37\text{-}0.2117)^2+(-0.27\text{-}0.2117)^2\right)$$

$$\sigma^2 = \frac{1}{6}(0.5685)$$

$$\sigma^2 = 0.095$$

(3) ปรับให้เป็นค่ามาตรฐาน (Normalize)

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 - \varepsilon}} \quad (3.6)$$

โดย $\hat{x}_i$ คือ ข้อมูลในเวกเตอร์ตัวที่ i

μ คือ ค่าเฉลี่ย

σ^2 คือ ค่าความแปรปรวน

ε คือ ค่า epsilon (จะถูกโมเดลปรับค่าในทุกรอบ ในที่นี้กำหนดให้ = 0)

$$\hat{x}_0 = \frac{0.6 - 0.2117}{\sqrt{0.095 - 0}}$$

$$\hat{x}_0 = \frac{0.3883}{0.3082}$$

$$\hat{x}_0 = 1.2599$$

$$\hat{x}_1 = \frac{0.23 - 0.2117}{\sqrt{0.095 - 0}}$$

$$\hat{x}_1 = \frac{0.0183}{0.3082}$$

$$\hat{x}_1 = 0.0594$$

$$\hat{x}_2 = \frac{-0.11 - 0.2117}{\sqrt{0.095 - 0}}$$

$$\hat{x}_2 = \frac{-0.3217}{0.3082}$$

$$\hat{x}_2 = -1.0438$$

$$\hat{x}_3 = \frac{0.45 - 0.2117}{\sqrt{0.095 - 0}}$$

$$\hat{x}_3 = \frac{0.2383}{0.3082}$$

$$\hat{x}_3 = 0.7732$$

$$\hat{x}_4 = \frac{0.37 - 0.2117}{\sqrt{0.095 - 0}}$$

$$\hat{x}_4 = \frac{0.1583}{0.3082}$$

$$\hat{x}_4 = 0.5136$$

$$\hat{x}_5 = \frac{-0.27 - 0.2117}{\sqrt{0.095 - 0}}$$

$$\hat{x}_5 = \frac{-0.387}{0.3082}$$

$$\hat{x}_5 = -1.2557$$

เมื่อทำการปรับให้ด้วยค่ามาตรฐานแล้วเวกเตอร์ของ “love” จะแสดงได้ดังนี้

ตารางที่ 3.9 ข้อมูลของเวกเตอร์ “love” หลังจากปรับด้วยค่ามาตรฐาน

love	1.299	0.0594	-1.0438	0.7732	0.5136	-1.2557

(4) ปรับด้วยค่าสเกลและชิพ (Scale and shift)

$$\text{Norm}(x_i) = \gamma \hat{x}_i + \beta \qquad (3.7)$$

โดย $\hat{x}_i$ คือ ค่ามาตรฐานของข้อมูลในเวกเตอร์ตัวที่ i

γ คือ ค่า gamma (จะถูกโมเดลปรับค่าในทุกรอบ ในที่นี้กำหนดให้ = 0.5)

β คือ ค่า beta (จะถูกโมเดลปรับค่าในทุกรอบในที่นี้กำหนดให้ = 0.3)

$$\text{Norm}(x_0) = 0.5(1.299) + 0.3$$
$$\text{Norm}(x_0) = 0.9495$$

$$\text{Norm}(x_1) = 0.5(0.0594) + 0.3$$
$$\text{Norm}(x_1) = 0.3297$$

$$\text{Norm}(x_2) = 0.5(-1.0438) + 0.3$$
$$\text{Norm}(x_2) = -0.2219$$

$$\text{Norm}(x_3) = 0.5(1.299) + 0.3$$
$$\text{Norm}(x_3) = 0.6866$$

$$\text{Norm}(x_4) = 0.5(0.5136) + 0.3$$
$$\text{Norm}(x_4) = 0.5568$$

$$\text{Norm}(x_5) = 0.5(1.299) + 0.3$$
$$\text{Norm}(x_5) = -0.3279$$

เมื่อเสร็จสิ้นกระบวนนี้เวกเตอร์ของ “love” จะแสดงได้ดังนี้

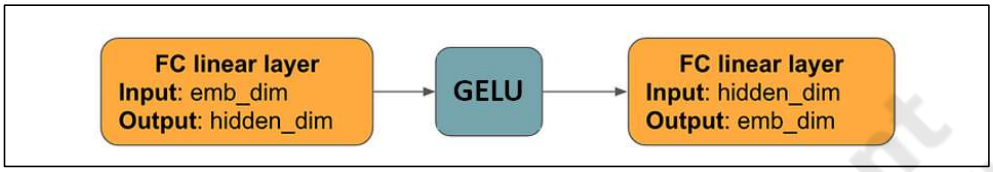
ตารางที่ 3.10 ข้อมูลของเวกเตอร์ “love” หลังจากปรับด้วยค่าสเกลและชิพ

love	0.9495	0.3297	-0.2219	0.6866	0.5568	-0.3279

ขั้นตอนที่ 7 : Feed Forword

Feed Forword เป็นกระบวนการที่นำ Feed-forward neural networks ที่ถือเป็นโมเดลที่มีโครงสร้างที่เรียบง่ายที่สุด เพราะว่า การดำเนินการของข้อมูลจะเป็นไปในทิศทางเดียว ก็คือ รับข้อมูลจาก input layer แล้วส่งไปต่อไปยัง hidden layer เรื่อยๆ จนกระทั้งถึง output layer ก็จะหยุด โดย

การทำ Feed-forward neural networks จำมี Activation function อยู่หนึ่งตัวนั้นคือ ฟังก์ชัน gelu โดยสามารถแสดงการทำงานของขั้นตอนนี้ดังภาพประกอบที่ 3.19



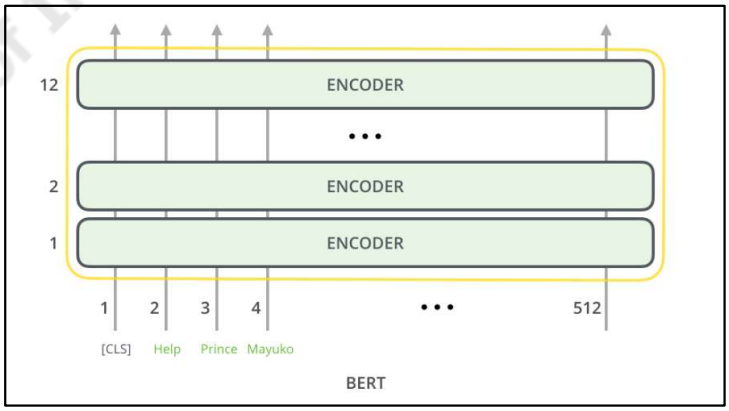
ภาพประกอบที่ 3.19 การทำงานของ Feed-forward neural networks

ที่มา : https://medium.com/dissecting-bert/dissecting-bert-part-1-d3c3d495cdb3

ขั้นตอนที่ 8 : Add & Norm

Add & Norm ขั้นตอนนี้ เป็นขั้นตอนเดียวกับขั้นตอนที่ 6 เพียงเปลี่ยนเวกเตอร์ที่ทำการบวกกันเป็น เวกเตอร์ก่อนจะเข้า Feed-forward neural networks กับ เวกเตอร์หลังเข้า Feed-forward neural networks เพียงเท่านั้น เมื่อจบขั้นตอนนี้ก็ถือเป็นการจบการสร้าง pre-training BERT ใน 1 Encoder แล้ว

โดยในการทำ pre-training BERT นั้นจะมี Encoder Block ซึ่งคือชั้นของ Encoder ที่ต้องเรียงกันทั้งหมด 12 ชั้น เพื่อให้โมเดลปรับค่าน้ำหนักของข้อมูลเอกสารเพื่อให้โมเดลเข้าใจโครงสร้างทางภาษา ลำดับของภาษา และการใช้ภาษา นั้นก็คือ Language Model นั้นเอง (แสดงดังภาพประกอบที่ 3.20)



ภาพประกอบที่ 3.20 ชั้นของ Encoder ภายในโมเดลแบบ BERT

ที่มา : http://jalammar.github.io/illustrated-bert/

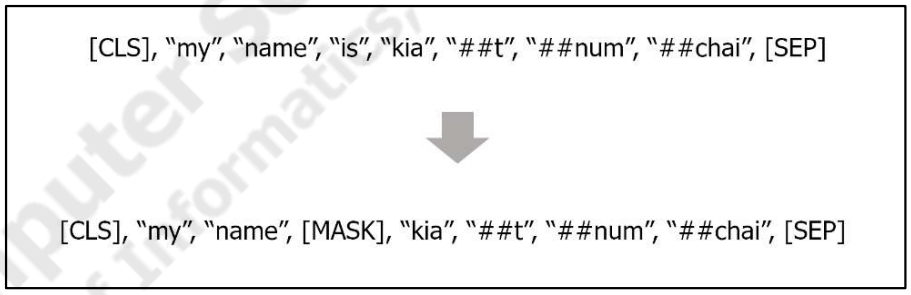
ขั้นตอนที่ 9 : Task of Pre-training BERT

เมื่อผ่านการทำ Pre-training BERT การด้วย Encoder ทั้ง 12 ชั้นแล้วโมเดลแบบ BERT จะท้าทายตัวเองด้วยงานทั้งหมด 2 งานด้วยกัน เพื่อปรับค่าน้ำหนักของโมเดล เพื่อให้โมเดลเข้าใจโครงสร้างทางภาษามากยิ่งขึ้น ซึ่งงานที่โมเดลแบบ BERT ให้ท้าทายตัวเองมีดังนี้

(1) Mask Language Model (MLM)

ก่อนที่จะนำ word sequence เข้าสู่ BERT จะมีการแทน “คำ” ในแต่ละ sequence ด้วยโทเค็น [MASK] เรียกว่า “Masked Word” จำนวน 15% หลังจากนั้นโมเดลจะทำการทำนายค่าดั้งเดิม (Original Value) ของ Masked Word ด้วยคำที่ไม่มี MASK ที่เรียกว่า “Non-masked Word” ซึ่งเป็นบริบทที่อยู่รอบ “Masked Word” ใน word sequence นั้นๆ สำหรับการทำนายคำที่เป็นเอาต์พุต จะมีการดำเนินการดังนี้

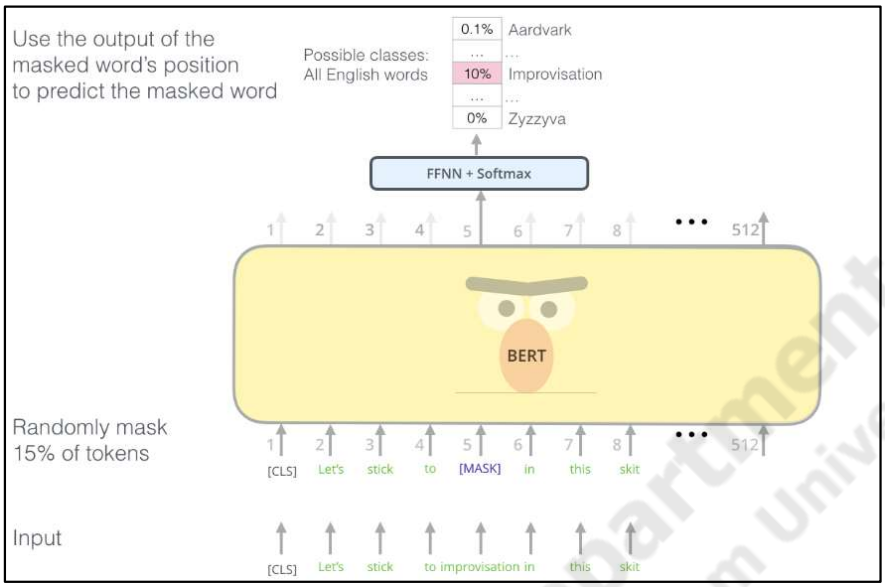
ขั้นตอนที่ 1: สุ่มเปลี่ยนคำในข้อมูลเอกสารที่จะนำเข้าสู่โมเดลแบบ BERT เป็นโทเค็น [MASK] จำนวน 15% จากนั้นนำเข้าสู่กระบวนการ Input Embedding และ Positional encoding เพื่อสร้างเวกเตอร์ของแต่ละคำในข้อมูลเอกสาร และนำเข้าสู่โมเดลแบบ BERT



ภาพประกอบที่ 3.21 การสุ่มเปลี่ยนคำในข้อมูลเอกสารจำนวน 15% เป็นโทเค็น [MASK]

ขั้นตอนที่ 2 : เมื่อเวกเตอร์ของแต่ละคำในชุดข้อมูลเอกสารผ่านการเทรนด้วยโมเดลแบบ BERT แล้ว จะนำเวกเตอร์ของ “Masked Word” เข้าสู่โมเดลเพื่อทำนายค่าดั้งเดิมเดิม โดยโมเดลสำหรับทำนายค่าดั้งเดิมจะประกอบด้วย 2 ส่วนคือ Feed-forward Neural Network และ ฟังก์ชัน SoftMax ที่จะต่อเข้ากับโมเดลแบบ BERT ที่ encoder layer ชั้นบนสุดของโมเดลแบบ BERT นั้นเอง

ขั้นตอนที่ 3 : เมื่อนำเข้าสู่โมเดลสำหรับทำนายค่าดั่งเดิมแล้ว จะได้ตัวเลขความน่าจะเป็นของคำแต่คำที่ควรมาแทน “Masked Word” (แสดงการทำงานทั้งหมดได้ดังภาพประกอบที่ 3.22)



ภาพประกอบที่ 3.22 ขั้นตอนการทำงานในส่วนของ Mask Language Model

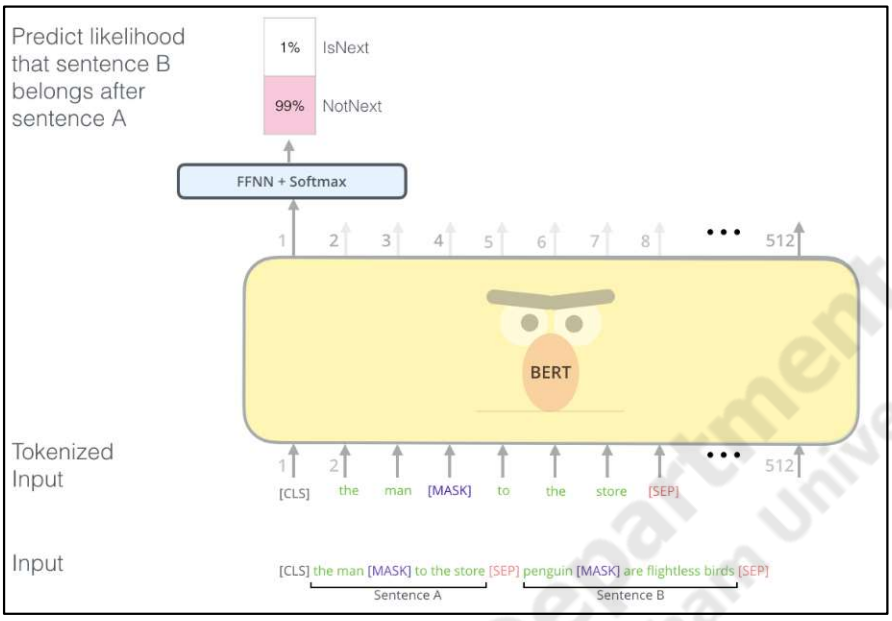
(2) Next Sentient Prediction (NSP)

ในขั้นตอน Next Sentient Prediction ของโมเดลแบบ BERT จะนำโทเค็น [CLS] ของ encoder ชั้นบนสุดไปต่อเข้ากับโมเดลสำหรับทำนายคู่ประโยค ในการทำ Next Sentient Prediction จะรับประโยคเข้ามาเป็นคู่ และจะเรียนรู้เพื่อทำนาย ถ้าประโยคที่สองในคู่ประโยคที่รับเข้ามาเป็น subsequence ในข้อมูลเอกสารต้นฉบับ ระหว่างการเรียนรู้นั้น 50% ของข้อมูลอินพุตจะเป็นคู่ประโยค (Pairs of Sentences) โดยประโยคที่สองที่ตามหลังประโยคแรกต้องเป็น subsequent sentence ในข้อมูลเอกสารต้นฉบับ ในขณะที่ อีก 50% ของข้อมูลอินพุตที่เหลือ จะเป็น random sentence จากคลังเอกสารที่ถูกเลือกมาเป็นประโยคที่สอง ดังนั้นเพื่อช่วยให้โมเดลแยกแยะระหว่างสองประโยคในการเรียนรู้ได้ ในการทำนายว่าประโยคที่สองเชื่อมต่อกับประโยคแรกจริงหรือไม่ ให้ดำเนินการตามขั้นตอนต่อไปนี้:

ขั้นตอนที่ 1 : นำอินพุตทั้งหมดเข้าสู่โมเดลแบบ BERT

ขั้นตอนที่ 2 : [CLS] จะถูกนำเข้าสู่โมเดลที่ต่ออยู่กับ encoder ชั้นบนสุดของ BERT ในโมเดลจะประกอบด้วยชั้นของ Fully-Connected layer และ ฟังก์ชัน SoftMax ซึ่งจะได้เวกเตอร์ผลลัพธ์ขนาด 2×1 ที่บ่งบอกถึงความน่าจะเป็นที่ 2 ประโยคนั้นเป็นประโยคที่ต่อกัน ด้วยฟังก์ชัน SoftMax

สามารถแสดงขั้นตอนการทำงานในส่วนของ Next Sentient Prediction ได้ดัง



ภาพประกอบที่ 3.23 ขั้นตอนการทำงานในส่วนของ Next Sentient Prediction

ในขั้นตอนของการเรียนรู้โมเดล BERT นั้น Masked LM และ Next Sentence Prediction จะได้รับการเรียนร่วมกัน โดยมีเป้าหมายในการลด Loss Function ที่รวมกันจากทั้งสองกลยุทธ์

3.4 Fine-tuning BERT

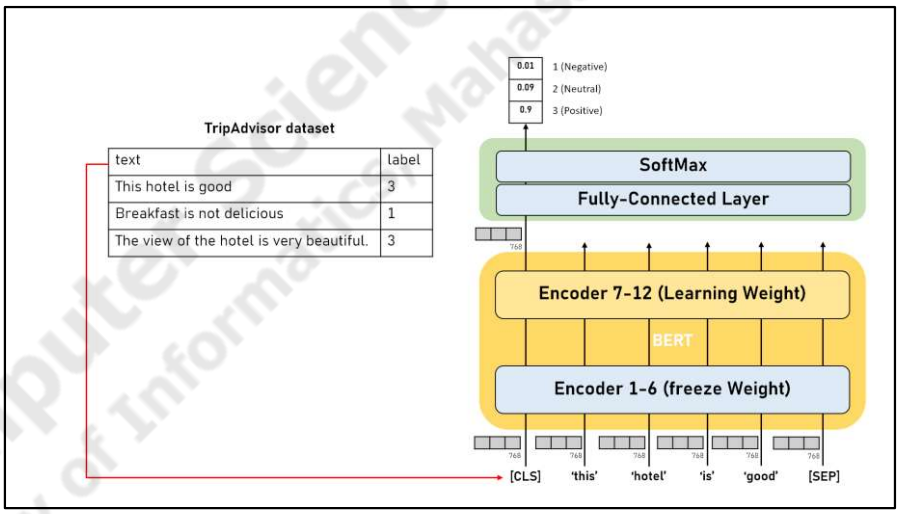
Fine-tuning BERT คือการปรับโมเดลแบบ BERT ให้ทำงานเฉพาะ (specific task) ต่างๆได้ เพื่อที่จะให้โมเดลแบบ BERT สามารถทำงานด้านการจำแนกความรู้สึก (Sentiment Classification) ได้นั้น จะต้องนำข้อมูล ที่รวบรวมจาก Tripadvisor ที่เป็นชุดข้อมูลเอกสารแบบมีคลาสลาเบล เข้าสู่กระบวนการนี้ ในขณะเดียวกันโมเดลแบบ BERT จะถูกเชื่อมต่อเข้ากับโมเดลสำหรับการจำแนกความรู้สึก ซึ่งจะประกอบด้วย Fully-connected Layer และฟังก์ชัน SoftMax โดยโมเดลสำหรับการจำแนกความรู้สึกนี้ จะต่อกับเข้ากับโมเดลแบบ BERT ที่ตำแหน่งของโทเค็น [CLS] ซึ่งเป็นโทเค็นสำหรับงานจำแนกเอกสาร (Text Classification) ที่ encoder ชั้นบนสุด นอกจากนั้นโมเดลแบบ BERT ยังมีการแช่การเรียนรู้ของ encoder ชั้นที่ 1 – 6 ไม่ให้ปรับค่าน้ำหนักและพารามิเตอร์ไปในระหว่างการทำกระบวนการ Fine-tuning และเปิดให้ encoder ชั้นที่ 7 – 12 ปรับค่าน้ำหนักและพารามิเตอร์ไปพร้อมๆกับโมเดลสำหรับการจำแนกความรู้สึก เมื่อสิ้นสุดกระบวนการแล้วก็จะได้โมเดลสำหรับการจำแนกความรู้สึก (Sentiment Classification Model) เพื่อใช้ในงานเฉพาะที่ต้องการ ซึ่งขั้นตอนการดำเนินงานดังนี้

(1) นำโมเดลสำหรับจำแนกความรู้สึกเชื่อมต่อกับโมเดลแบบ BERT ที่ตำแหน่งของโทเค็น [CLS] ชั้น encoder ชั้นบนสุด ซึ่งโมเดลสำหรับจำแนกความรู้สึกประกอบด้วย Fully-Connected layer และฟังก์ชัน SoftMax กำหนดให้เวกเตอร์ผลลัพธ์ที่ได้จากโมเดลสำหรับจำแนกความรู้สึกมีขนาด 3X1 ซึ่งบ่งบอกถึงคลาสลาเบลทั้ง 3 คลาสได้แก่ ความรู้สึกบวก (Positive) ความรู้สึกเป็นกลาง (Neutral) และความรู้สึกเชิงลบ (Negative)

(2) ทำการแช่การเรียนรู้ของ encoder ชั้นที่ 1 – 6 ไม่ให้ปรับค่าน้ำหนักและพารามิเตอร์ไปในระหว่างการทำกระบวนการ Fine-tuning เพื่อไม่ให้โมเดลยึดอยู่กับค่าน้ำหนักของโมเดลแบบ BERT มากจนเกินไปจนนำไปสู่ปัญหา overfitting [22] ได้และเปิดให้ encoder ชั้นที่ 7 – 12 ปรับค่าน้ำหนักและพารามิเตอร์ไปพร้อมๆกับโมเดลสำหรับการจำแนกความรู้สึก

(3) นำข้อมูลที่รวบรวมจาก Tripadvisor เข้าสู่โมเดลเพื่อให้โมเดลเทรนปรับค่าน้ำหนักให้เหมาะสมกับงานจำแนกความรู้สึก

สามารถแสดงขั้นตอนการ Fine-tuning BERT สำหรับงานจำแนกความรู้สึกได้ดังภาพประกอบที่ 3.24 ขั้นตอนการ Fine-tuning BERT สำหรับงานจำแนกความรู้สึก



ภาพประกอบที่ 3.24 ขั้นตอนการ Fine-tuning BERT สำหรับงานจำแนกความรู้สึก

เมื่อสิ้นสุดกระบวนการข้างต้นทั้งหมดแล้วก็จะได้โมเดลสำหรับการจำแนกความรู้สึก (Sentiment Classification Model) ที่สามารถนำไปใช้ในการจำแนกความรู้สึกบทวิจารณ์โรงแรมแล้ว

3.5 ตัวอย่างการคำนวณการประเมินในแบบ Multi-class

ในโครงงานนี้ใช้การประเมินประสิทธิภาพด้วยสมการในการคำนวณเพื่อประเมินประสิทธิภาพในรูปแบบต่าง ได้แก่ ค่าความถูกต้อง (Accuracy) ค่าความระลึก (Recall) ค่าความแม่นยำ (Precision)

และค่าเอฟ (F1, F-score) โดยการที่จะใช้สมการในเหล่านี้ในการคำนวณจะนำค่าจาก คอนฟิวชั่นเมทริกซ์ มาใช้ในการคำนวณ

กำหนดให้ คอนฟิวชั่นเมทริกซ์ มีทั้งหมด 3 class ได้แก่ ความรู้สึกบวก (Positive) ความรู้สึกเป็นกลาง (Neural) และความรู้สึกลบ (Negative) และมีข้อมูลดังตารางที่ 3.11

ตารางที่ 3.11 ตัวอย่าง คอนฟิวชั่นเมทริกซ์แบบ 3 class

		Actual Results		
		Positive	Neural	Negative
Prediction Results	Positive	1918	66	23
	Neural	63	1716	243
	Negative	11	164	1825

จากตารางที่ 3.11 สามารถคำนวณค่าการประเมินประสิทธิภาพต่างๆได้ดังนี้

ค่าความถูกต้อง (Accuracy: Acc)

จะใช้ประเมินการทำนายกลุ่มหรือคลาสที่ถูกต้องจากจำนวนข้อมูลชุดทดสอบทั้งหมด มีสมการในการคำนวณดังนี้

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.8}$$

โดย *TP* คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าจริง และมันจริง

TN คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าไม่จริง และมันไม่จริง

FP คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าจริง แต่ว่ามันไม่จริง

FN คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าไม่จริง แต่มันเป็นจริง

$$ACC = \frac{7 + 2 + 1}{7 + 8 + 9 + 1 + 2 + 3 + 3 + 2 + 1}$$

$$ACC = \frac{10}{0.2778}$$

$$ACC = 0.2778$$

ค่าความระลึก (Recall)

ค่าที่พิจารณาว่าข้อมูลที่ในความเป็นจริงบอกว่า “จริง” และเมื่อถูกพิจารณาด้วยโมเดลแล้วโมเดลจะทำนายผล “จริง” ได้ถูกต้องและตรงกับคำตอบ มีสมการในการคำนวณดังนี้

$$Recall = \frac{TP}{TP + FN} \tag{3.9}$$

โดย *TP* คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าจริง และมันจริง

FN คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าไม่จริง แต่มันเป็นจริง

$$Recall_{Positive} = \frac{7}{7 + 1 + 3}$$

$$Recall_{Positive} = \frac{7}{11}$$

$$Recall_{Positive} = 0.6364$$

$$Recall_{Neural} = \frac{2}{8 + 2 + 2}$$

$$Recall_{Neural} = \frac{2}{12}$$

$$Recall_{Neural} = 0.1667$$

$$Recall_{Negative} = \frac{1}{9 + 3 + 1}$$

$$Recall_{Negative} = \frac{1}{13}$$

$$Recall_{Negative} = 0.0769$$

และสามารถหาค่าเฉลี่ย (Average) ของค่า Recall ได้โดยมีสมการในการคำนวณดังนี้

$$Average_{Recall} = \frac{\sum Recall}{n} \tag{3.10}$$

โดย $\sum Recall$ คือ ค่าผลรวมของค่า Recall ของแต่ละคลาส

n คือ จำนวนคลาส

$$Average_{Recall} = \frac{0.6364 + 0.1667 + 0.0769}{3}$$

$$Average_{Recall} = \frac{0.88}{3}$$

$$Average_{Recall} = 0.2933$$

ค่าความแม่นยำ (Precision)

ค่าของการทำนายกลุ่มข้อมูลที่โมเดลพิจารณาจาก จำนวนข้อมูลที่มีทำนายกลุ่มว่าเป็น “จริง” ทั้งหมดนั้น แท้ที่จริงแล้วถูกต้องมากน้อยเพียงใด เมื่อเปรียบเทียบกับจำนวนข้อมูลทั้งหมดในกลุ่มที่เป็น “จริง” มีสมการในการคำนวณดังนี้

$$Precision = \frac{TP}{TP + FP} \quad (3.11)$$

โดย *TP* คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าจริง และมันจริง

FP คือ ค่าที่ตัวจำแนกเอกสารข้อมูลทำนายว่าจริง แต่ว่ามันไม่จริง

$$Precision_{Positive} = \frac{7}{7 + 8 + 9}$$

$$Precision_{Positive} = \frac{7}{24}$$

$$Precision_{Positive} = 0.2917$$

$$Precision_{Neural} = \frac{2}{1 + 2 + 3}$$

$$Precision_{Neural} = \frac{2}{6}$$

$$Precision_{Neural} = 0.3333$$

$$Precision_{Negative} = \frac{1}{3 + 2 + 1}$$

$$Precision_{Negative} = \frac{1}{6}$$

$$Precision_{Negative} = 0.1667$$

และสามารถหาค่าเฉลี่ย (Average) ของค่า Precision ได้โดยมีสมการในการคำนวณดังนี้

$$Average_{Precision} = \frac{\sum Precision}{n} \quad (3.12)$$

โดย $\sum Precision$ คือ ค่าผลรวมของค่า Precision ของแต่ละคลาส

n คือ จำนวนคลาส

$$Average_{Precision} = \frac{0.2917 + 0.3333 + 0.1667}{3}$$

$$Average_{Precision} = \frac{0.7917}{3}$$

$$Average_{Precision} = 0.2639$$

ค่าเอฟ (F1, F-score)

เป็นตัววัดที่เป็นการสมดุลค่า (Balanced Score) ระหว่างค่าความระลึกและค่าความแม่นยำด้วยค่าเฉลี่ยแบบฮาร์โมนิค (Harmonic Mean) มีสมการในการคำนวณดังนี้

$$F1 = 2 * \frac{Recall * Precision}{Recall + Precision} \quad (3.13)$$

โดย *Recall* คือ ค่าความละลึก

Precision คือ ค่าความแม่นยำ

$$F1_{Positive} = 2 * \frac{0.6364 * 0.2917}{0.6364 + 0.2917}$$

$$F1_{Positive} = 2 * \frac{0.1856}{0.9281}$$

$$F1_{Positive} = 0.4000$$

$$F1_{Neural} = 2 * \frac{0.1667 * 0.3333}{0.1667 + 0.3333}$$

$$F1_{Neural} = 2 * \frac{0.0556}{0.9281}$$

$$F1_{Neural} = 0.1198$$

$$F1_{Negative} = 2 * \frac{0.0769 * 0.1667}{0.0769 + 0.1667}$$

$$F1_{Negative} = 2 * \frac{0.0128}{0.2436}$$

$$F1_{Negative} = 0.1051$$

และสามารถหาค่าเฉลี่ย (Average) ของค่า F1 ได้โดยมีสมการในการคำนวณดังนี้

$$Average_{F1} = \frac{\sum F1}{n} \quad (3.14)$$

โดย $\sum F1$ คือ ค่าผลรวมของค่า F1 ของแต่ละคลาส

n คือ จำนวนคลาส

$$Average_{F1} = \frac{0.4000 + 0.1198 + 0.1051}{3}$$

$$Average_{F1} = \frac{0.6249}{3}$$

$$Average_{F1} = 0.2083$$