Báo cáo đồ án Xử lý ngôn ngữ tự nhiên

Xử lý ngôn ngữ tự nhiên (Trường Đại Học Công Nghệ Thông Tin)

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

ĐỒ ÁN MÔN HỌC

XỬ LÝ NGÔN NGỮ TỰ NHIÊN

PHÂN TÍCH CẢM XÚC VĂN BẢN TIẾNG VIỆT

Giảng viên hướng dẫn	**:**	**TS. Nguyễn Trọng Chỉnh**
Sinh viên thực hiện 1	**:**	**Lê Châu Giang**
Mã sinh viên 1	**:**	**21520213**
Sinh viên thực hiện 2	**:**	**Đoàn Lê Tuấn Thành**
Mã sinh viên 2	**:**	**21521438**
Sinh viên thực hiện 3	**:**	**Nguyễn Ngọc Thức**
Mã sinh viên 3	**:**	**21521506**
Lớp	**:**	**CS221.O22**

Tp HCM, tháng 6 năm 2024

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp.HCM, ngày … tháng 06 năm 2024

GVHD

TS. Nguyễn Trọng Chỉnh

MỤC LỤC

Chương 1:
GIỚI THIỆU BÀI TOÁN

1.1. Đặt vấn đề

Phân tích cảm xúc văn bản, hay còn gọi là sentiment analysis hoặc opinion mining, là một lĩnh vực nghiên cứu trong khoa học máy tính và xử lý ngôn ngữ tự nhiên (NLP), tập trung vào việc xác định, trích xuất và phân tích cảm xúc (sentiment) thể hiện trong văn bản. Mục tiêu chính của phân tích cảm xúc văn bản là nhận diện và phân loại cảm xúc của người viết về một chủ đề nhất định, như tích cực, tiêu cực hay trung lập.
Trong thực tế, phân tích cảm xúc văn bản đóng vai trò rất quan trọng trên nhiều lĩnh vực. Ví dụ, trong lĩnh vực thương mại, phân tích cảm xúc của khách hàng về 1 sản phẩm thành công đồng nghĩa với việc tổng hợp được thông tin về mức độ hài lòng của khách hàng đối với sản phẩm đó, từ đó nhà sản xuất có thể phát huy ưu điểm cũng như cải thiện nhược điểm sao cho phù hợp nhất với nhu cầu sử dụng của khách hàng.

1.2. Mục tiêu nghiên cứu

Trong phạm vi đề tài này, nhóm trình bày về việc sử dụng mô hình BERT (Bidirectional Encoder Representations from Transformers) với cụ thể là **PhoBERT** (Pre-trained language models for Vietnamese), so sánh với phương pháp máy học **SVM** (Support Vector Machine) để thực hiện phân tích cảm xúc trên một số bình luận bằng văn bản tiếng việt. Việc phân tích này có tác dụng trích xuất ý kiến, quan điểm hay đánh giá từ các đoạn văn bản. Mục tiêu là nhận biết và tóm tắt các ý kiến của người dùng về một sản phẩm, dịch vụ, hoặc sự kiện cụ thể.
Tuy nhiên, quá trình thực hiện bài toán phân tích cảm xúc văn bản thường gặp phải nhiều khó khăn. Một trong những vấn đề lớn nhất khi giải quyết bài toán là vấn đề về từ nhiều nghĩa, khi mà một từ có thể hiểu bằng nhiều cách (nghĩa gốc và nghĩa chuyển), việc xác định đúng ngữ cảnh là một thách thức lớn.

Ví dụ: Từ “hay” thường được sử dụng để thể hiện thái độ tích cực, nhưng trong một số trường hợp, người dùng cũng có thể sử dụng từ này với hàm ý mỉa mai như “Bộ phim hay lắm, hay đến mức chỉ muốn bỏ về ngay.”

Chương 2:

BỘ NGỮ LIỆU

2.1. Cách xây dựng bộ ngữ liệu:

Trong đề tài này, nhóm sử dụng bộ dữ liệu các bình luận nhận xét về khách sạn được sử dụng trong bài báo Sentiment Analysis in Code-Mixed Vietnamese-English Sentence-level Hotel Reviews ([2]) được công bố trên tạp chí Pacific Asia Conference on Language, Information and Computation, năm 2022. Bộ dữ liệu được lưu trữ dưới dạng file csv.

	Column1	Column2
2	Tiếng Việt	label
3	khi m nhận phòng đc nv xếp cho phòng ở tầng 1 là phòng twin nhưng p rất xấu , chỉ giống nhà nghỉ .	negative
4	m thực sự thất vọng .	negative
5	nhưng nhân viên ở đây khá nhiệt tình , thân thiện	positive
6	tôi đặt phòng standard double , nhưng thực tế thì ở phòng single (cái này là do tôi nhìn trên bảng giá niêm yết của khách sạn , thấy ghi những phòng nào là phòng single , bao gồm có phòng của gia đình tôi) .	neutral
7	do đó , giường hơi nhỏ , điều hòa ồn , đêm có nhiều muỗi , được cái gần bờ biển , khu vực yên tĩnh , phù hợp vớigia đình có trẻ nhỏ (sáng và chiều ra biển tắm , ăn hải sản tha hồ) .	positive
8	khách sạn giá rẻ , gần biển nhân viên lễ tân thiếu lịch sự với khách , thái độ khó chịu .	negative
9	ý dịch vụ giặt là (cần hỏi giá trước) .	neutral
10	vì không phải tính theo kg đâu mà là đếm cái ăn tiền .	neutral
11	dịch vụ thuê xe cũng khá đắt .	negative
12	khách sạn đã cho tôi có 1 kỳ nghỉ tuyệt vời , tôi rất hài lòng !	positive
13	nhân viên ở khách sạn rất thân thiện , hòa đồng .	positive
14	nhân viên nhiệt tình hỗ trợ khách trong thời gian lưu trú , tôi rất hài lòng .	positive
15	khách san sạch sẽ , nhân viên thân thiện , phục vụ nhanh , gần bái biển , đi vào trung tâm thành phố rất tiện	positive
16	khách sạn gần sát biển , chỉ mất 2p để đi bộ .	positive
17	nhân viên lễ tân khá thân thiện nhưng dịch vụ dọn phòng chưa được tốt (lười thay ga trải giường)	negative
18	tôi và vợ mới cưới rất hài lòng với dịch vụ khách sạn .	positive
19	gần bãi biển và các quán ăn đồ biển phong phú , giá cả vừa phải .	positive
20	từ ks bạn có thể vào trung tâm tp bằng xe máy (thuê tại ks) hoặc taxi khoảng 10 phút qua cầu sông hàn .	positive
21	bạn có thể đi thăm chùa linh ứng , bán đảo sơn trà (rất đẹp , nên đi) có tượng phật bà quan âm chỉ khoảng 20 ' bằng 2 phương tiện trên . đặc biệt đặt phòng qua agoda rất kinh tế mà an toàn .	positive
22	tôi nghĩ chất lượng dich vụ khá tốt với chi phí bỏ ra .	positive
23	gần biển nên rất thuận tiện .	positive
24	tôi khá hài lòng với ks atlantic , nhân viên niềm nở nhiệt tình và còn đổi phòng to hơn cho chúng tôi khi chúng tôi có yêu cầu và tất nhiên không tính phí .	positive
25	mỗi tội phòng hơi nhỏ nhưng vì đi du lịch ko mấy ở ks nên đây ko phải là vấn đề .	negative
26	khách sạn gần biển chỉ đi bộ vài phút là tới biển nên tôi và gd ra biển thường xuyên hơn .	positive
27	chúng tôi chỉ nghỉ một đêm tại đây nhưng thực sự không hài lòng lắm về lựa chọn của mình .	negative
28	view biển bị chắn bởi khách sạn đang xây bên cạnh nhưng được cái là khá gần biển .	negative

Hình 2.1. Ví dụ minh họa về dataset được sử dụng trong đề tài

2.2. Quy tắc chú thích dữ liệu:

2.2.1. Định nghĩa các nhãn cảm xúc

Bộ dữ liệu bao gồm 3 label (positive, negative, neutral) đại diện cho 3 trạng thái cảm xúc:

- Positive: Cảm xúc tích cực, thể hiện thái độ hài lòng của khách hàng đối với khách sạn, dịch vụ tại khách sạn.
- Negative: Cảm xúc tiêu cực, thể hiện thái độ không hài lòng của khách hàng đối với khách sạn, dịch vụ tại khách sạn.
- Neutral: Cảm xúc trung tính, thái độ không rõ ràng hoặc chỉ đưa ra thông tin mà không thể hiện thái độ.

2.2.2. Quy trình chú giải

Các bình luận được gán nhãn dựa trên cảm xúc của người bình luận, định nghĩa của từng label thông qua các tính từ miêu tả trải nghiệm hoặc bày tỏ cảm xúc được sử dụng trong bình luận.

Ví dụ 1 số tính từ được sử dụng để xác định trạng thái cảm xúc:

- Positive: "lý tưởng", "xứng đáng", "nhiệt tình", "tuyệt vời",…
- Negative: "không hài lòng", "thất vọng", "bẩn",…

Khi gán nhãn, trong trường hợp bình luận bao gồm cả tính từ thể hiện thái độ tích cực và tiêu cực, nếu bình luận thiên về cảm xúc nào hơn thì nhãn được gán cho trạng thái cảm xúc đó, nếu bình luận không thiên về cảm xúc nào (tỉ lệ giữa cảm xúc tích cực và tiêu cực ngang bằng nhau) thì nhãn được gán là cảm xúc trung tính.

2.2.3. Ví dụ minh họa

Để dễ dàng hình dung được quy tắc chú thích ngữ liệu sau đây là 1 số ví dụ minh họa chủ giải ngữ liệu:

Nội dung bình luận	Label	Giải thích
1 khách sạn tuyệt vời để nghỉ dưỡng tại nha trang với đội ngũ nhân viên lịch thiệp , hồ bơi rất đẹp nằm ở tầng 2 liên thông với phòng tập thể dục .	Positive	Bình luận này thể hiện thái độ tích cực của khách hàng khi sử dụng các tính từ: "tuyệt vời", "rất đẹp", "lịch thiệp". Các từ này thể hiện mức độ hài lòng cao của khách hàng đối với trải nghiệm tại khách sạn.
lobby chật hẹp , không thoải mái khi khách đến check in , không có cây xanh , thiết kế nhà kính glass cảm giác nóng bức khi vào đến khách sạn	Negative	Bình luận này thể hiện thái độ tiêu cực của khách hàng khi sử dụng các tính từ: "chật hẹp", "không thoải mái", "nóng bức". Các từ này thể hiện thái độ không hài lòng của khách hàng đối với khách sạn và các dịch vụ tại khách sạn.
tôi ở khách sạn yasaka sài gòn nha trang 02 ngày .	Neutral	Bình luận này chỉ đơn thuần đưa ra thông tin mà không thể hiện thái độ của khách hàng.
phòng sạch sẽ (tuy nhiên có hôm dọn phòng hơi chậm , đến chiều chúng tôi đi chơi về nghỉ mới bắt đầu dọn nên không tiện cho khách lắm) , ăn sáng phong phú , nhân viện đều thân thiện nhiệt tình .	Neutral	Bình luận này bao gồm cả thái độ tích cực của khách hàng (Thông qua các tính từ "sạch sẽ", "phong phú", "thân thiện", "nhiệt tình") và thái độ tiêu cực (Thông qua các từ/cụm từ "hơi chậm", "không tiện"). Bình luận không quá thiên về hướng tích cực hay tiêu cực nên có thể xem là thái độ trung tính.

Bảng 2.1. Ví dụ minh họa quy tắc chú thích ngữ liệu

2.3. Thống kê ngữ liệu:

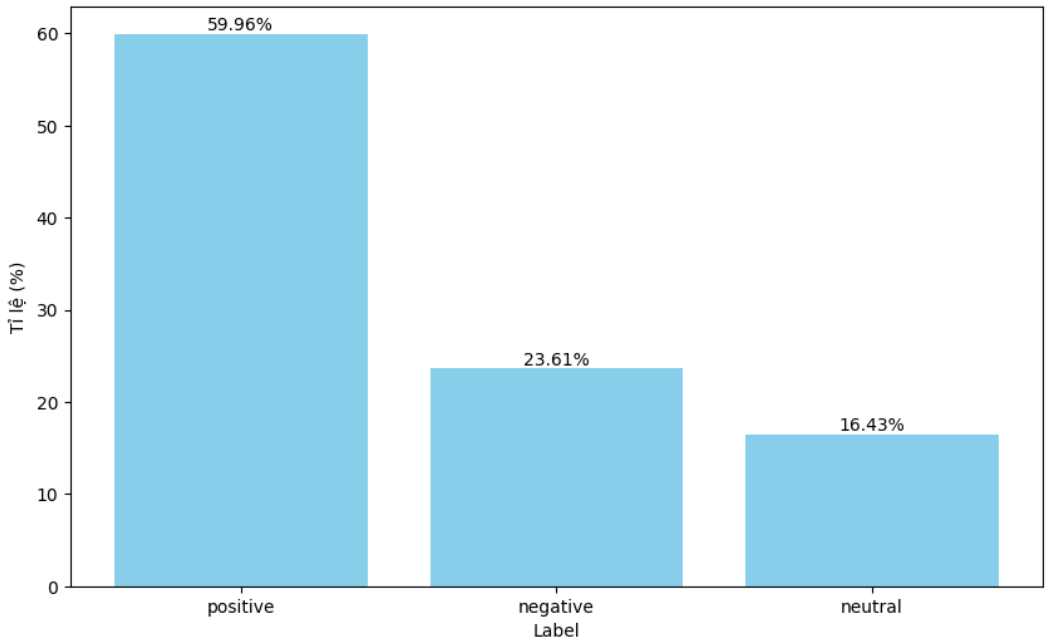
Để hiểu rõ hơn về ngữ liệu được sử dụng trong ngữ liệu, nhóm thực hiện thống kê số lượng và phân bố thông tin ngữ liệu (sau khi đã áp dụng VNCoreNLP để tách từ, phân đoạn văn bản tiếng việt).

	N.o sentences per class			**Length**	**Vocab**
	Positive	Negative	Neutral		
Dataset	1981	780	543	14.52	47990

Bảng 2.2. Thống kê ngữ liệu

Bảng 2 trình bày các thông tin sau đây:

- N.o sentences per class: Thông tin thống kê số lượng mẫu thuộc mỗi class trên toàn bộ bộ dữ liệu
- Length: Số lượng từ trung bình mỗi mẫu trên toàn bộ bộ dữ liệu (không tính các dấu câu)
- Vocab: Tổng số lượng từ được sử dụng trên toàn bộ bộ dữ liệu (Không tính các dấu câu)

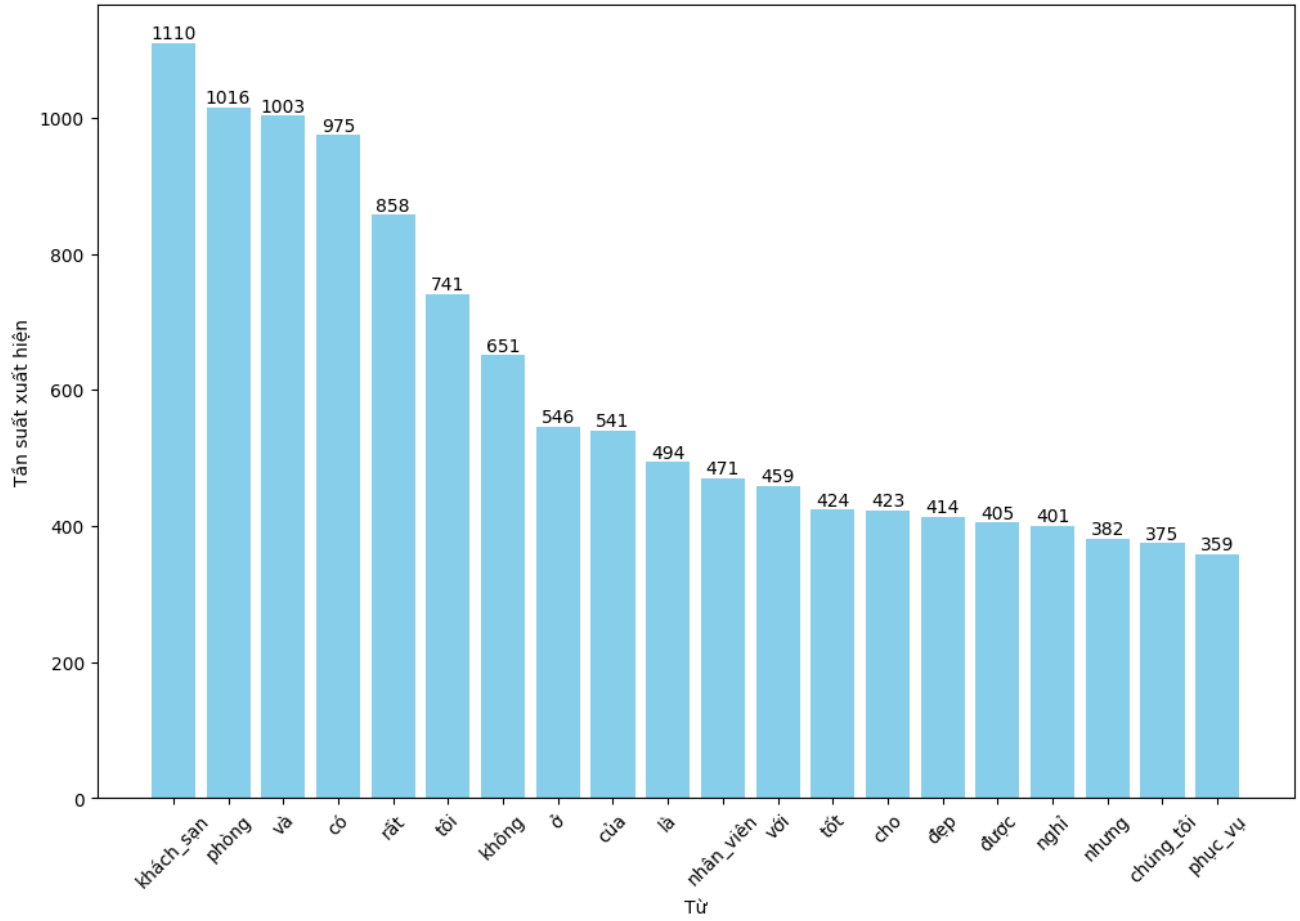


Hình 2.2. Biểu đồ tỉ lệ giữa các label trong dataset

Quan sát bảng 2.2 và hình 2.2 ta có thể thấy rằng, bộ dữ liệu bị mất cân bằng lớn khi số lượng mẫu có label là positive chiếm **59,96%**, trong khi mẫu có label neutral chỉ chiếm **16.43%** và mẫu có label negative chiếm **23.61%**.

➔ Cần cân bằng dữ liệu trước khi đưa vào mô hình dự đoán.

Để có thể hiểu rõ về tần suất xuất hiện của các từ trong ngữ liệu, nhóm thực hiện thống kê các từ xuất hiện trong ngữ liệu (sau khi đã qua word segmentation của VnCoreNLP và một số bước tiền xử lý dữ liệu khác):



Hình 2.3. Biểu đồ thể hiện tuần suất xuất hiện 20 từ phổ biến nhất trong ngữ liệu

2.4. Kiểm tra chú thích:

Để kiểm tra xem các mẫu này có được chú giải theo đúng quy tắc đã trình bày không, nhóm thực hiện kiểm tra và giải thích 75 chú thích được trích xuất từ bộ ngữ liệu.

STT	Label	Nội dung bình luận	Giải thích
1	negative	khi m nhận phòng đc nv xếp cho phòng ở tầng 1 là phòng twin nhưng p rất xấu , chỉ giống nhà nghỉ .	Bình luận này thể hiện sự chê bai về chất lượng bằng tính từ "rất xấu", và so sánh phòng twin "như nhà nghỉ"
2	negative	m thực sự thất vọng .	Bình luận này thể hiện thái độ đối với khách sạn là "thất vọng"
3	positive	nhưng nhân viên ở đây khá nhiệt tình , thân thiện	Bình luận này thể hiện sự hài lòng về nhân viên của khách sạn bằng các từ ngữ "khá nhiệt tình", "thân thiện"
4	neutral	tôi đặt phòng standard double , nhưng thực tế thì ở phòng single (cái này là do tôi nhìn trên bảng giá niêm yết của khách sạn , thấy ghi những phòng nào là phòng single , bao gồm có phòng của gia đình	Bình luận này đã được gán nhãn sai, bởi vì mặc dù không có từ ngữ nào thể hiện sự không hài lòng nhưng về mặc nội dung khách hàng đang đề cập tới sự cố nhầm lẫn khi đặt phòng của khách sạn (negative)

		tôi) .	
5	positive	do đó , giường hơi nhỏ , điều hòa ồn , đêm có nhiều muỗi , được cái gần bờ biển , khu vực yên tĩnh , phù hợp với gia đình có trẻ nhỏ (sáng và chiều ra biển tắm , ăn hải sản tha hồ) .	Bình luận này mặc dù chứa các từ ngữ mang ý nghĩa tiêu cực như "hơi nhỏ", "ồn", "nhiều muỗi", nhưng phía sau đó là các từ "gần biển", "yên tĩnh", "phù hợp" làm cho câu mang ý nghĩa hài lòng nhiều hơn
6	negative	khách sạn giá rẻ , gần biển nhân viên lễ tân thiếu lịch sự với khách , thái độ khó chịu .	Bình luận này ban đầu có các từ mang nghĩa khen như "giá rẻ", "gần biển", nhưng nó mang nghĩa tiêu cực vì sau đó là các từ chỉ ra các nhược điểm như "thiếu lịch sự", "khó chịu"
7	neutral	ý dịch vụ giặt là (cần hỏi giá trước) .	Bình luận này không có từ ngữ thể hiện cảm xúc tích cực hay tiêu cực, về mặc nội dung nó cũng chỉ đề cập tới một lưu ý về dịch vụ giặt là.
8	neutral	vì không phải tính theo kg đâu mà là đếm cái ăn tiền .	Bình luận này không thể hiện cảm xúc cụ thể và cũng không có từ ngữ thể hiện cảm xúc, nó chỉ giải thích về cách tính phí của dịch vụ giặt là
9	negative	dịch vụ thuê xe cũng khá đắt .	Bình luận này chứa từ "khá đắt" thể hiện cảm xúc tiêu cực
10	positive	khách sạn đã cho tôi có 1 kỳ nghỉ tuyệt vời , tôi rất hài lòng !	Bình luận này chứa từ "tuyệt vời", "hài lòng" thể hiện cảm xúc tích cực
11	positive	nhân viên ở khách sạn rất thân thiện , hòa đồng .	Bình luận này chứa các từ thể hiện sự hài lòng thông qua từ "thân thiện", "hòa đồng"
12	positive	nhân viên nhiệt tình hỗ trợ khách trong thời gian lưu trú , tôi rất hài lòng .	Bình luận này thể hiện sự hài lòng thông qua các từ như "nhiệt tình", "hài lòng"
13	positive	khách san sạch sẽ , nhân viên thân thiện , phục vụ nhanh , gần bái biển , đi vào trung tâm thành phố rất tiện	Bình luận này thể hiện sự hài lòng rõ ràng bnagwf hàng loạt các tính từ "sạch sẽ", "thân thiện", "nhanh", "gần", "tiện"
14	positive	khách sạn gần sát biển , chỉ mất 2p để đi bộ .	Bình luận thê hiện sự hài lòng về vị trí của khách sạn thông qua tính từ "gần", "chỉ mất"
15	negative	nhân viên lễ tân khá thân thiện nhưng dịch vụ dọn phòng chưa được tốt (lười thay ga trải giường)	Bình luận này mặc dù ban đầu khen về thái độ phục vụ "thân thiện" của lễ tân, nhưng sau đó thì phàn nàn về dịch vụ dọn phòng "chưa được tốt", "lười"
16	positive	tôi và vợ mới cưới rất hài lòng với dịch vụ khách sạn .	Bình luận này thể hiện sự hài lòng về khách sạn thông qua cụm từ "rất hài lòng"
17	positive	gần bãi biển và các quán ăn đồ biển phong phú , giá cả vừa phải .	Bình luận này thể hiện sự hài lòng thông qua các từ như "gần", "phong phú", "vừa phải"
18	**positive**	từ ks bạn có thể vào trung tâm	Bình luận đã bị gán nhãn sai, nó chỉ đơn

	-> **neutral**	tp bằng xe máy (thuê tại ks) hoặc taxi khoảng 10 phút qua cầu sông hàn .	thuần cung cấp thông tin (neutral)
19	**positive -> neutral**	bạn có thể đi thăm chùa linh ứng , bán đảo sơn trà (rất đẹp , nên đi) có tượng phật bà quan âm chỉ khoảng 20 ' bằng 2 phương tiện trên . đặc biệt đặt phòng qua agoda rất kinh tế mà an toàn .	Bình luận đã bị gán nhãn sai, nó chỉ đơn thuần cung cấp thông tin mặc dù có chứa các từ ngữ mang nghĩa tích cực "rất đẹp", "nên đi", "rất kinh tế", "an toàn" (neutral)
20	positive	tôi nghĩ chất lượng dịch vụ khá tốt với chi phí bỏ ra .	Bình luận này thể hiện cảm xúc tích cực bằng từ "khá tốt"
21	positive	gần biển nên rất thuận tiện .	Bình luận này thể hiện cảm xúc tích cực bằng các từ "gần", "thuận tiện"
22	positive	tôi khá hài lòng với ks atlantic , nhân viên niềm nở nhiệt tình và còn đổi phòng to hơn cho chúng tôi khi chúng tôi có yêu cầu và tất nhiên không tính phí .	Bình luận này thế hiện sự hài lòng về khách sạn với nhiều từ mang nghĩa tích cực như "khá hài lòng", "niềm nở", "nhiệt tình", "không tính phí"
23	**negative -> neutral**	mỗi tội phòng hơi nhỏ nhưng vì đi du lịch ko mấy ở ks nên đây ko phải là vấn đề .	Bình luận đã gán nhãn sai, mặc dù vế đầu chứa từ tiêu cực "nhỏ", nhưng vế sau phủ nhận vấn dề đã nói ở vế trước "không phải vấn đề", tức là với họ không khen cũng không chê về khách sạn (neutral)
24	positive	khách sạn gần biển chỉ đi bộ vài phút là tới biển nên tôi và gd ra biển thường xuyên hơn .	Bình luận thể hiện cảm xúc tích cực bằng các từ "gần", "thường xuyên hơn"
25	negative	chúng tôi chỉ nghỉ một đêm tại đây nhưng thực sự không hài lòng lắm về lựa chọn của mình .	Bình luận thể hiện sự tiêu cực về khách sạn thông qua từ "không hài lòng lắm"
26	**negative -> neutral**	view biển bị chắn bởi khách sạn đang xây bên cạnh nhưng được cái là khá gần biển .	Bình luận này được gán nhãn sai, tuy khách có nói view biển bị chắn nhưng cũng có khen khá gần biển, không thể hiện thái độ quá tiêu cực.
27	positive	đi nghỉ đúng vào dịp mưa va lạnh kéo dài , được cái vị trí và view của khách sạn tương đối ổn nên cũng đỡ đi phần nào ...	Bình luận này thể hiện thái độ khá hài lòng với từ "tương đối ổn"
28	negative	kể ra khách sạn lắp cái điều hòa 2 chiều thì tốt , mua lạnh mà ko có máy sưởi thì cũng hơi thất vọng	Bình luận này thể hiện thái độ hơi không hài lòng với điều hòa thông qua từ "hơi thất vọng"
29	positive	khách sạn sẽ trở nên lý tưởng hơn nếu đến nghỉ vào mùa hè vì địa thế gần biển , có thể dễ dàng ra biển hoặc ngắm cảnh	Bình luận này thể hiện thái độ hài lòng, tích cực thông qua các từ như "lý tưởng", "gần", "dễ dàng"

		biển từ khách sạn .	
30	negative	du lịch dn vào mùa đông không được vui vì trời mưa kéo dài và biển động .	Bình luận này thể hiện thái độ tiêu cực thông qua từ "không được vui"
31	positive	khách sạn xứng đáng 3 * với các trang thiết bị trong phòng , nhân viên ở đây rất nhiệt tình , đặc biệt là nhân viên bảo vệ cũng như khuân hành lý đã giúp gia đình tôi rất nhiều	Bình luận này thể hiện thái độ tích cực, hài lòng thông qua các từ "xứng đáng", "nhiệt tình", "giúp".
32	positive	biển đẹp tuyệt vời , yên tĩnh	Bình luận này thể hiện thái độ tích cực thông qua từ "tuyệt vời"
33	neutral	hài lòng với không gian và dịch vụ của khách sạn , tuy nhiên đồ ăn sáng hơi kém phong phú và gian phòng ăn hơi chật chội nên kh khách đông phải xếp hàng chờ .	Bình luận này thể hiện thái độ trung tính, gồm cả khen với từ "hài lòng" và chê với các từ "kém phong phú", "hơi chật chội".
34	neutral	tôi nghĩ nên linh động cho khách về thời gian trả phòng và nhắc khách của agoda giờ trả phòng	Bình luận này chỉ mang tính chất đưa ra ý kiến, góp ý
35	positive	một kỳ nghỉ tuyệt vời , thích hợp cho các gia đình kể cả có con nhỏ .	Bình luận này thể hiện thái độ tích cực với các từ "tuyệt vời", "thích hợp"
36	**neutral -> Positive**	vị trí hơi xa trung tâm đà lạt nhưng hoàng anh gia lai resort có được khuôn viên rộng rãi thoải mái , tạo cho tôi cảm giác thư giản , nghỉ ngơi , đúng với nhu cầu nghỉ dưỡng của tôi , nơi đây nếu có trang bị hồ bơi nước nóng sẽ rất tốt .	Bình luận này ban đầu có nói về vị trí không thuận tiện, nhưng đoạn sau đều mang ý tích cực với các từ "rộng rãi", "thoải mái", "thư giản", "đúng nhu cầu". Như vậy bình luận thiên về cảm xúc tích cực hơn nên cần được gán nhãn là positive
37	positive	tôi rất thích nghỉ ngơi ở hagl , và luôn chọn hagl làm điểm dừng chân mỗi khi đến dl .	Bình luận này thể hiện thái độ tích cực với các từ "rất thích", "luôn chọn"
38	positive	ở hagl tôi không cần phải đi ra ngoài , vì ở đây có tất cả .	Bình luận này thể hiện thái độ tích cực với từ "có tất cả"
39	negative	tuy nhiên tôi vẫn chưa hài lòng về dịch vu bar va hy vọng lần sau sẽ thấy bar thật sự sống va " hot " hơn .	Bình luận này thể hiện thái độ không hài lòng với dịch vụ bar với từ :chưa hài lòng"
40	positive	thiết kế mang bạn hòa mình vào thiên nhiên , khung cảnh thân thiện tạo cho du khách một cảm giác âm áp gần gũi và thân mật .	Bình luận này thể hiện thái độ hài lòng với các từ "thân thiện", "ấm áp", "gần gũi", "thân mật"
41	neutral	nếu bạn đã trải nghiệm du lịch	Bình luận này không đưa ra nhận xét về

		nghỉ dưỡng ở đà lạt bạn sẽ gặp khó khăn trong vấn đề ẩm thực .	khách sạn mà chỉ đơn thuần cung cấp thông tin cho người đọc.
42	positive	khi bạn đến với hoàng anh gia lai resort bạn hoàn toàn yên tâm và thưởng thức về kỳ nghỉ của bạn hãy tận hưởng cuộc sống và những chuyến nghỉ dưỡng của bạn với những gì resort hoang anh gia lai mang lại cho bạn .	Bình luận này thể hiện thái độ hài lòng với các từ "yên tâm", "thưởng thức", "tận hưởng"
43	neutral	chúc các bạn có được những ngày nghỉ thoải mái và hạnh phúc khi có những lựa chọn đúng cho kỳ nghỉ của mình .	Bình luận này là lời chúc, không thể hiện thái độ gì dù có dùng 1 số từ thể hiện thái độ tích cực
44	positive	nhìn chung hoàng anh là 1 resort cao cấp tiện nghi và sạch sẽ , nhân viên lễ độ , chu đáo , dịch vụ hoàn hảo .	Bình luận này thể hiện thái độ hài lòng với các từ như "cao cấp", "tiện nghi", "sạch sẽ", "lễ độ", "chu đáo", "hoàn hảo".
45	neutral	nếu resort chịu đầu tư thêm các hạng mục giải trí bên trong khu vực để du khách có thể vui chơi thư giãn sau khi tham quan đà lạt thì càng tuyệt vời	Bình luận này đưa ra ý kiến góp ý, xây dựng chứ không thể hiện thái độ
46	negative	phòng nào cũng có muỗi và kiến .	Bình luận này thể hiện thái độ tiêu cực dù không có các từ ngữ thể hiện thái độ trong câu.
47	negative	rất nhiều muỗi ?	Bình luận này thể hiện thái độ tiêu cực dù không có các từ ngữ thể hiện thái độ trong câu.
48	negative	đồ ăn thì dở .	Bình luận này thể hiện thái độ tiêu cực dù không có các từ ngữ thể hiện thái độ trong câu.
49	**Neutral -> negative**	nhân viên tỏ ra thân thiện trong hướng dẫn khách nhưng nhờ hỏi dịch vụ thuê xe thì báo giá cao hơn thực thế nhiều .	Bình luận này đầu câu có khen thái độ nhưng câu sau thể hiện thái độ chê khi nói "báo giá cao hơn thực tế".
50	negative	rất nhiều muỗi và kiến .	Bình luận này thể hiện thái độ chê bai khi nhắc đến "muỗi" và "kiến" - những thứ sẽ mang đến trải nghiệm không tốt cho chuyến đi
51	positive	tôi đánh giá rất cao về thái độ phục vụ của khách sạn hạnh đạt , cả bà chủ khách sạn lẫn nhân viên đều rất nhiệt tình và niềm nở .	Bình luận này thế hiện sự hài lòng và đánh giá cao về thái độ phục vụ của khách sạn, bà chủ và nhân viên thông qua các tính từ "rất cao", "nhiệt tình", "niềm nở".
52	**neutral -> positive**	phòng khách sạn tuy hơi nhỏ nhưng hiện đại , sạch sẽ , giá cả phải chăng .	Bình luận này đã gán nhãn sai, mặc dù có từ "hơi nhỏ" (có thể coi là điểm trừ), nhưng tổng thể vẫn mang tính tích cực

			nhờ các yếu tố hiện đại, sạch sẽ và giá cả hợp lý, tạo cảm giác hài lòng thông qua các từ "hiện đại", "sạch sẽ", "giá cả phải chăng".
53	neutral	những ngày nghỉ cũa tôi ở huế đã trải qua rất tuyệt vời .	Bình luận này chỉ cung cấp thông tin về trải nghiệm chung ở Huế mà không cụ thể về khách sạn, nhân viên, dịch vụ hoặc vị trí.
54	neutral	cám ơn hạnh đạt hotel và agoda .	Bình luận này chỉ đơn thuần là lời cảm ơn mà không cung cấp thông tin cụ thể về cảm xúc, trải nghiệm hoặc đánh giá về khách sạn, nhân viên, dịch vụ hoặc vị trí và không có yếu tố khen ngợi hay chê bai rõ ràng.
55	netrual	một kỳ nghỉ không tồi ở huế .	Bình luận này chỉ cung cấp thông tin rằng kỳ nghỉ ở Huế không phải là trải nghiệm xấu, từ "không tồi" có nghĩa là kỳ nghỉ ổn, nhưng không thể hiện sự hài lòng đặc biệt hoặc không hài lòng.
56	positive	khách sạn rất thuận tiện , còn tiếp tân rất friendly .	Bình luận này thể hiện sự hài lòng về sự thuận tiện của khách sạn và thái độ thân thiện của tiếp tân, thông qua các từ ngữ chỉ sự tích cực: "rất thuận tiện", "rất friendly".
57	positive	tiếp tân giúp ích rất nhiều trong việc hướng dẫn du khách đến huế , thậm chí giúp mua những đặc sản huế khá ngon và hợp lý .	Bình luận này thể hiện sự khen ngợi hài lòng về tiếp tân và đặc sản huế thông qua các từ ngữ tích cực: "giúp ích rất nhiều", "ngon và hợp lý".
58	positive	đây là khách sạn rất thuận tiện cho những ai muốn đi du lịch thắng cảnh ở huế .	Bình luận này nhấn mạnh về sự thuận tiện của khách sạn cho những du khách muốn tham quan cảnh đẹp ở Huế, phản ánh một trải nghiệm tích cực về vị trí và tiện ích của khách sạn.
59	positive	tôi nghĩ rằng mình không có lý do gì để chọn khách sạn khác nếu một lần nữa đi du lịch ở huế .	Bình luận này thể hiện sự hài lòng và sự quyết định của người nói rằng họ sẽ quay lại khách sạn này nếu có dịp đi du lịch ở Huế lần nữa, tạo ra một đánh giá tích cực về khách sạn và trải nghiệm trước đó.
60	neutral	khách sạn cách đại nội khoảng 4,5 km và hoàn toàn có thể đi bộ đến đó trong vòng 45 phút đến 1 tiếng .	Bình luận này cung cấp thông tin về khoảng cách giữa khách sạn và điểm đến (đại nội), không thể xác định cảm xúc tích cực hoặc tiêu cực.
61	positive	tại khách sạn có cho thuê xe máy với chất lượng xe còn rất tốt .	Bình luận này khen ngợi khả năng cho thuê xe máy tại khách sạn và đánh giá cao chất lượng của các xe máy thông qua từ "rất tốt".
62	positive	khách sạn sạch sẽ , nhân viên	Bình luận này thể hiện sự hài lòng về sự

		phục vụ rất nhiệt tình , niềm nở .	sạch sẽ của khách sạn và thái độ tích cực của nhân viên, thông qua các từ ngữ tích cực: "sạch sẽ", "rất nhiệt tình", "niềm nở".
63	positive	trang thiết bị vật chất rất tốt so với giá tiền .	Bình luận này đánh giá cao trang thiết bị vật chất của khách sạn so với giá tiền, thông qua từ "rất tốt".
64	**neutral -> positive**	nếu bạn đi du lịch tại huế và muốn ở một nơi gần trung tâm , giá cả phù hợp nhưng vẫn đầy đủ tiện nghi và được phục vụ tốt thì nên lựa chọn khách sạn hạnh đạt này .	Bình luận này gán nhãn sai, khách sạn Hạnh Đạt đáp ứng các yếu tố quan trọng khi chọn khách sạn, bao gồm vị trí thuận tiện, giá cả hợp lý, tiện nghi đầy đủ và dịch vụ tốt, thông qua các từ: "gần trung tâm", "giá cả phù hợp", "đầy đủ tiện nghi", "được phục vụ tốt".
65	positive	tôi đã có một kì nghỉ tuyệt vời tại vedana .	Bình luận này diễn tả trải nghiệm kỳ nghỉ tích cực tại Vedana thông qua từ "tuyệt vời"
66	neutral	tôi sẽ không bao giờ quên cảnh mặt trời mọc và hoàng hôn trên phá tam giang .	Bình luận này chỉ cung cấp thông tin về trải nghiệm chung ở Huế mà không cụ thể về khách sạn, nhân viên, dịch vụ hoặc vị trí.
67	positive	khách sạn đáng yêu .	Bình luận này thể hiện sự hài lòng và đánh giá tích cực về khách sạn, thông qua từ "đáng yêu".
68	**positive -> neutral**	phong cảnh đẹp yên tĩnh , không khí trong lành rất thích hợp để nghỉ dưỡng .	Bình luận này mô tả phong cảnh và không khí xung quanh địa điểm nghỉ dưỡng, nhưng không đề cập trực tiếp đến khách sạn, nhân viên, dịch vụ hoặc vị trí cụ thể nào. Mặc dù mô tả có cảm giác tích cực về phong cảnh và không khí, nhưng không đủ thông tin để xác định cảm xúc liên quan đến tiêu chí gán nhãn về khách sạn.
69	neutral	tôi đến lúc 10h sáng khi check in phòng nhân viên báo là thời gian check in là 14h chiều .	Bình luận này chỉ cung cấp thông tin về trải nghiệm chung ở Huế mà không cụ thể về khách sạn, nhân viên, dịch vụ hoặc vị trí.
70	negative	nên chờ hơi lâu .	Bình luận này thể hiện sự không hài lòng về thời gian chờ đợi thông qua từ "hơi lâu"
71	**positive -> neutral**	gần resort có quán gái đẳng ăn hải sản rất ngon .	Bình luận này đã gán nhẫn sai, mặc dù có yếu tố tích cực ("hải sản rất ngon"), nhưng nó không liên quan trực tiếp đến trải nghiệm dịch vụ hoặc chất lượng của resort.
72	negative	đường vào resort hơi nhếch nhác và bẩn .	Bình luận này thể hiện sự không hài lòng về tình trạng của đường vào resort, thông qua từ "nhếch nhác" và "bẩn".
73	positive	khi vào khuôn viên resort thi	Bình luận này khen ngợi khuôn viên

		mọi thứ điều ok (xanh sạch đẹp) .	resort thông qua các từ "ok", "xanh sạch đẹp".
74	negative	wc nắp xi er phía dưới rất bẩn vì không được thường xuyên lau chùi .	Bình luận này thể hiện sự không hài lòng về tình trạng vệ sinh của nhà vệ sinh tại khách sạn hoặc resort thông qua các từ "rất bẩn", "không được thường xuyên lau chùi".
75	positive	khách sạn với địa điểm tốt , không khí trong lành yên tĩnh , phục vụ tận tình	Bình luận này thể hiện sự hài lòng và khen ngợi về vị trí, môi trường và dịch vụ của khách sạn, thông qua từ "địa điểm tốt", "không khí trong lành yên tĩnh", "phục vụ tận tình".

Bảng 2.3. Explain Label

Nhóm thực hiện giải thích, chỉnh sửa trên 75 mẫu dữ liệu được trích xuất từ ngữ liệu, như đã trình bày tại bảng 2.3. Theo đó:

- Các mẫu có label sử dụng chữ đen, in thường là các mẫu được label đúng, nhóm thực hiện giải thích, chú giải label đó, đưa ra các từ khóa được sử dụng để phân tích cảm xúc khách hàng trong mẫu đó.
- Các mẫu có label sử dụng chữ đỏ, in đậm là các mẫu chưa được label đúng, nhóm thực hiện giải thích, chú giải nguyên nhân vì sao cần thay đổi label cho mẫu đó, đưa ra các "từ khóa" giúp xác định mức độ hài lòng của khách hàng.

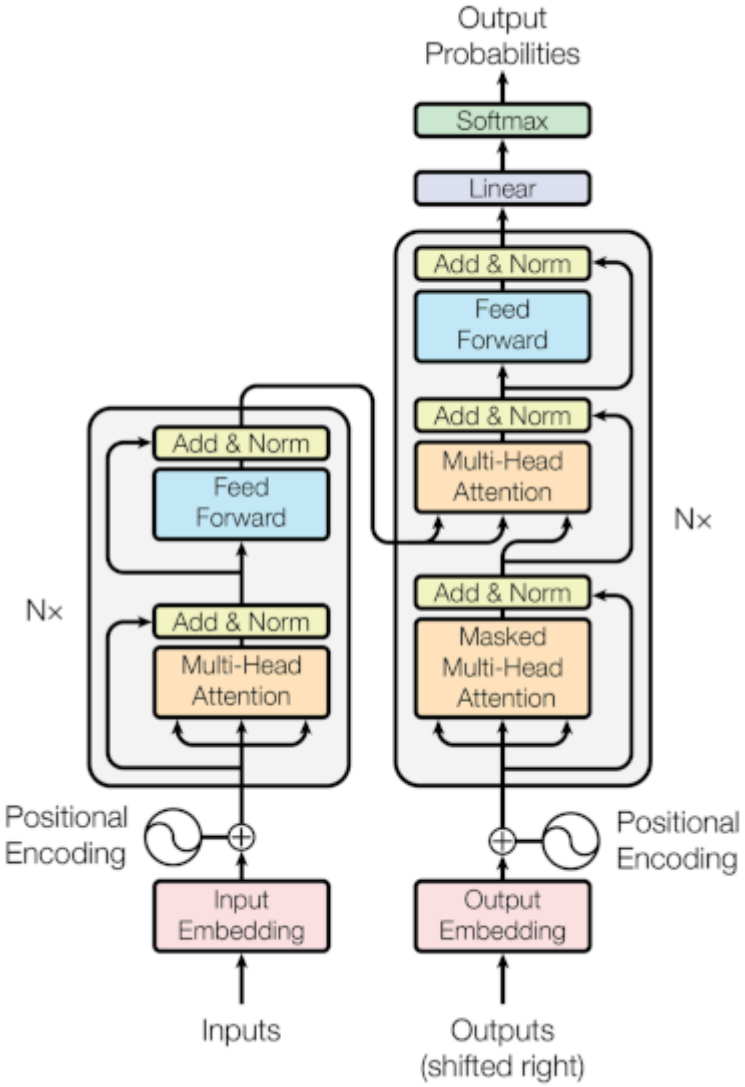
Quan sát **bảng 2.3**, ta có thể thấy rằng, số lượng mẫu được label sai là 10 mẫu trên tổng số 75 mẫu được quan sát, chiếm tỉ lệ 13.33%. Như vậy, ta có thể kết luận rằng, tuy vẫn còn một số mẫu bị gán nhãn sai, nhưng hầu hết các mẫu được gán nhãn đúng với quy tắc chú thích dữ liệu đã được trình bày ở phần 2.2.2.

Chương 3:

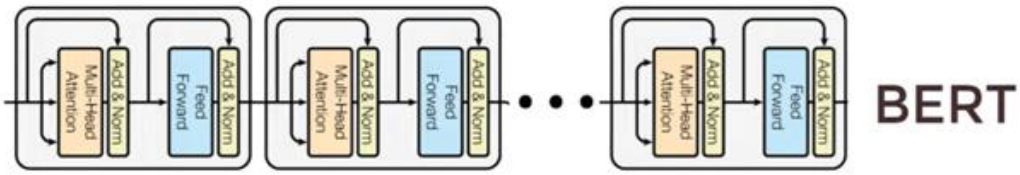
PHƯƠNG PHÁP SỬ DỤNG - PhoBERT

3.1. Kiến trúc Transformer (encoder) và BERT, RoBERTa dến PhoBERT

BERT (Bidirectional Encoder Representations from Transformers) là một mô hình ngôn ngữ tiên tiến được phát triển bởi Google AI vào năm 2018. Dựa trên cấu trúc Transformer, BERT chỉ sử dụng phần encoder của mô hình này. Transformer encoder hoạt động bằng cách áp dụng cơ chế self-attention, cho phép mô hình xem xét toàn bộ câu hoặc đoạn văn bản để hiểu ngữ cảnh của từng từ. Khác với các mô hình trước đây thường chỉ xem xét ngữ cảnh từ trái sang phải hoặc từ phải sang trái, BERT xử lý văn bản theo cách bidirectional (hai chiều). Điều này có nghĩa là BERT đọc toàn bộ câu cùng một lúc, giúp nắm bắt ngữ cảnh tốt hơn và cải thiện độ chính xác trong các tác vụ NLP (xử lý ngôn ngữ tự nhiên) như trả lời câu hỏi, phân loại văn bản, và nhận diện thực thể. BERT đã mở ra một kỷ nguyên mới cho các mô hình ngôn ngữ dựa trên Transformer, với hiệu suất vượt trội trên nhiều benchmarks khác nhau.



Hình 3.1: Kiến trúc mô hình Transformer



Hình 3.2: Kiến trúc mô hình BERT

Khác với mô hình Transformer có cấu trúc gồm hai phần: mã hóa (encoder) và giải mã (decoder), BERT là một ứng dụng cụ thể của Transformer, chỉ sử dụng phần encoder. BERT là mô hình hai chiều, xem xét cả ngữ cảnh trước và sau của một từ. Không giống như Transformer tổng quát, được thiết kế cho các tác vụ như dịch máy, BERT tập trung vào việc tạo ra các biểu diễn ngữ cảnh sâu cho nhiều ứng dụng NLP khác nhau như: Phân loại văn bản, Question Answering, Dịch máy,…

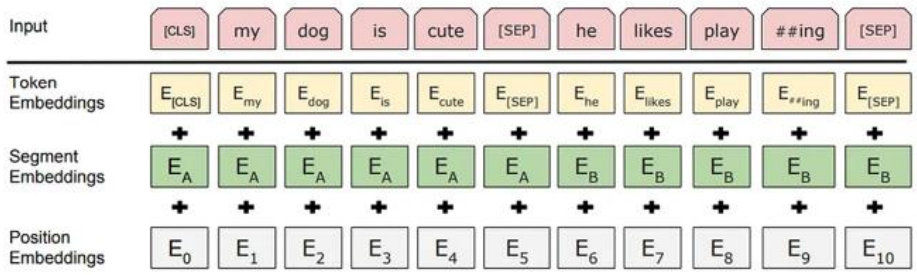
Khác với BERT, RoBERTa được huấn luyện trên một lượng dữ liệu lớn hơn gồm khoảng 160GB văn bản từ tiếng Anh trên mạng, bao gồm cả các tài liệu từ Wikipedia và các trang web khác. Điều này giúp RoBERTa cải thiện độ chính xác và khả năng tổng quát hóa trên nhiều tác vụ NLP.

PhoBERT là một pre-trained được huấn luyện monolingual language, tức là chỉ huấn luyện dành riêng cho tiếng Việt. Việc huấn luyện dựa trên kiến trúc và cách tiếp cận giống RoBERTa. Tương tự như BERT, PhoBERT cũng có 2 phiên bản là PhoBERT base với 12 transformers block và PhoBERT large với 24 transformers block. Chúng ta cùng đi sâu vào phân tích cấu trúc 1 encoder.

3.1.1. Input Embedding

Embedding vector là một thành phần quan trọng trong kiến trúc của BERT và các mô hình xử lý ngôn ngữ tự nhiên hiện đại. Đây là bước đầu tiên trong quá trình xử lý đầu vào của mô hình, chuyển đổi các token (từ hoặc phần của từ) thành các vector số học có kích thước cố định. Embedding vector là biểu diễn vector của mỗi token trong không gian đa chiều, thường có kích thước 768 hoặc 1024 chiều trong BERT. Mục đích chính của embedding là chuyển đổi dữ liệu đầu vào rời rạc (các từ) thành dạng liên tục mà mạng neural có thể xử lý hiệu quả. Embedding vector nắm bắt các đặc trưng ngữ nghĩa và ngữ cảnh của từng token. Trong BERT, embedding vector cuối cùng được tạo thành từ ba loại embedding khác nhau:

- Token Embeddings: Biểu diễn ý nghĩa của từng token. Được học trong quá trình huấn luyện.
- Segment Embeddings: Phân biệt câu trong cặp câu đầu vào. Có hai loại: A (câu đầu) và B (câu sau). Việc này cho phép khối encoder phân biệt giữa các câu trong chuỗi input.
- Position Embeddings: Cho biết vị trí của token trong chuỗi. Đối với BERT thì Position Embeddings sẽ được học, không sử dụng hàm sin/cos như Transformer gốc. Mỗi vị trí (từ 0 đến độ dài tối đa của chuỗi, thường là 512) có một vector embedding riêng.



Hình 3.3: Quá trình embedding input của BERT.

Các embedding này được cộng lại để tạo ra embedding vector cuối cùng cho mỗi token. BERT học các embedding vector trong quá trình huấn luyện trước (pre-training). Ban đầu, các embedding được khởi tạo ngẫu nhiên và sau đó được tinh chỉnh thông qua quá trình học để nắm bắt tốt hơn các đặc trưng ngữ nghĩa và cú pháp của ngôn ngữ.

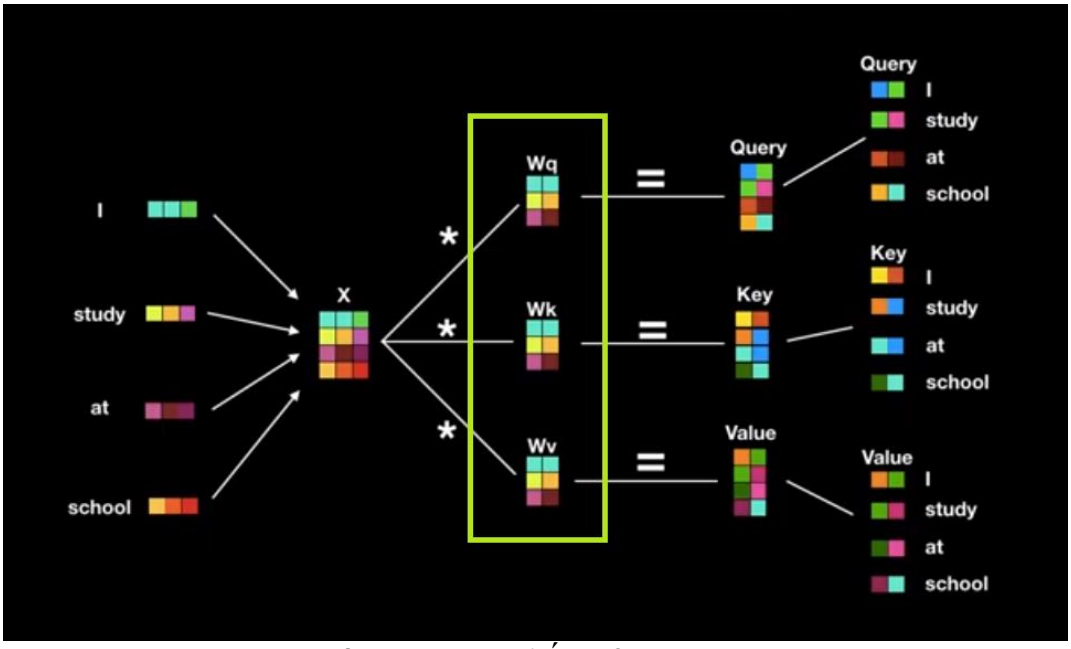
3.1.2. Cơ chế Attention:

Attention là một cơ chế giúp mô hình học cách "chú ý" đến các phần khác nhau của đầu vào khi thực hiện các tác vụ khác nhau. Trong ngữ cảnh của xử lý ngôn ngữ tự nhiên, attention cho phép mô hình tập trung vào các từ liên quan nhất khi mã hóa một câu hoặc đoạn văn bản.

Trong BERT, attention được triển khai dưới dạng multi-head self-attention. Đây là một thành phần quan trọng của kiến trúc Transformer, bao gồm nhiều "đầu" attention độc lập. Mỗi đầu attention học cách chú ý đến các phần khác nhau của đầu vào.
Self-attention là cơ chế cho phép mỗi từ trong một câu có thể tương tác với các từ khác trong cùng câu đó. Điều này giúp mô hình hiểu được ngữ cảnh tổng thể của câu, không chỉ dựa trên các từ xung quanh trực tiếp.

Quá trình self-attention trong BERT diễn ra theo các bước sau:

- Tạo các ma trận Query (Q), Key (K), và Value (V):
  - Mỗi từ trong đầu vào được chuyển đổi thành các vector Query, Key và Value thông qua các ma trận trọng số đã được học. Trong đó: Query (Q): Vector để hỏi (query), được sử dụng để so sánh với tất cả các từ khác trong câu để tính toán mức độ quan trọng của chúng đối với từ hiện tại. Key (K): Vector để chỉ ra (key), cũng được sử dụng để so sánh với tất cả các từ khác trong câu. Value (V): Vector giá trị, được sử dụng để tạo ra đầu ra của mô hình sau khi tính toán self-attention.
  - Nếu X là vector biểu diễn của câu sau khi qua quá trình embedding thì $Q=XW_Q$, $K=XW_K$, $V=XW_V$. Trong đó W_Q, W_K, W_V chính là những hệ số mà model cần huấn luyện. Sau khi nhân các ma trận này với ma trận đầu vào ta thu được ma trận (tương ứng với trong hình là ma trận Query, Key và Value).



Hình 3.4. Cơ chế Self-attention

- Tính toán Attention score:

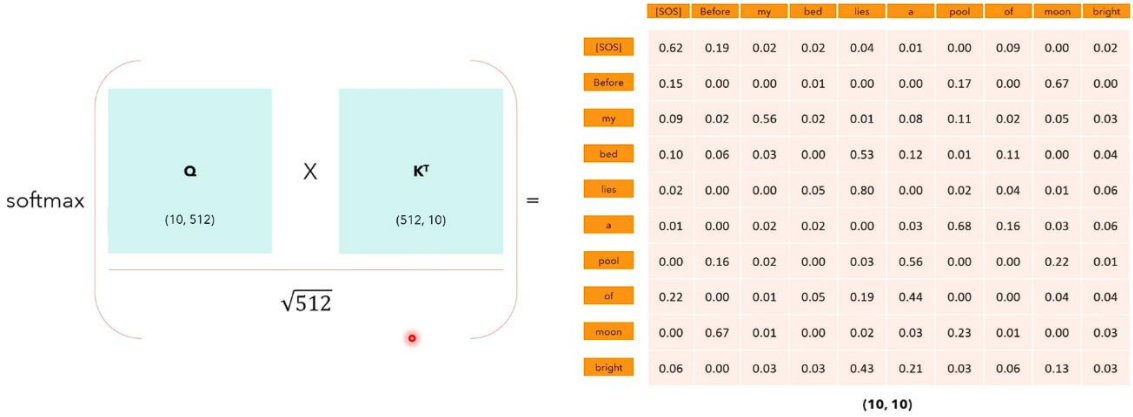
  Attention score được tính bằng cách nhân ma trận Query với ma trận Key rồi chia cho căn bậc hai của kích thước của vector embedding (để ổn định hóa gradient), sau đó áp dụng hàm softmax để có được phân phối xác suất.

  Công thức:

$$\text{Attention scores} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

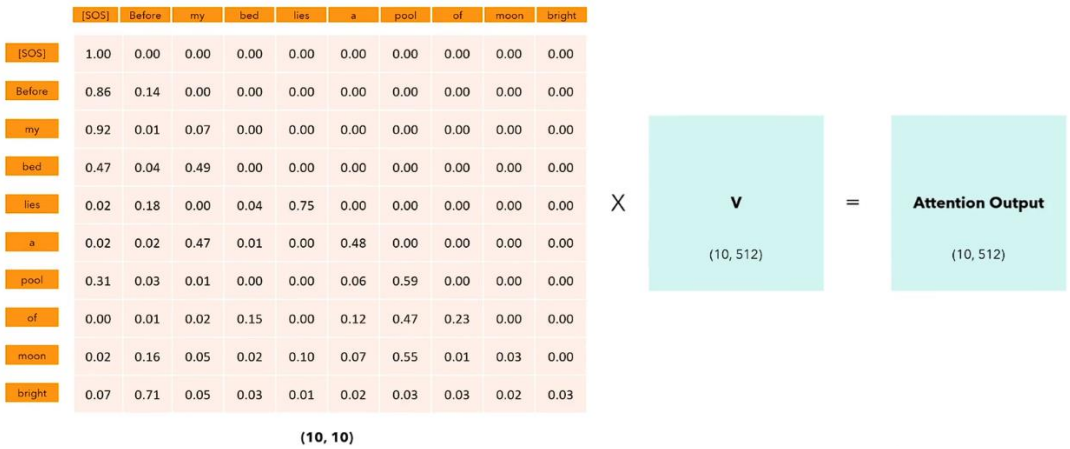
  Trong đó d_k là kích thước của các vector embedding.

  Score của mỗi cặp từ (w_i, w_j) là mỗi cặp từ (w_i, w_j) giá trị trong ma trận phân phối xác xuất. Độ lớn sẽ đại diện cho mức độ attention của từ query tới từ key. Trọng số càng lớn càng chứng tỏ từ w_i trả về một sự chú ý lớn hơn đối với từ w_j.



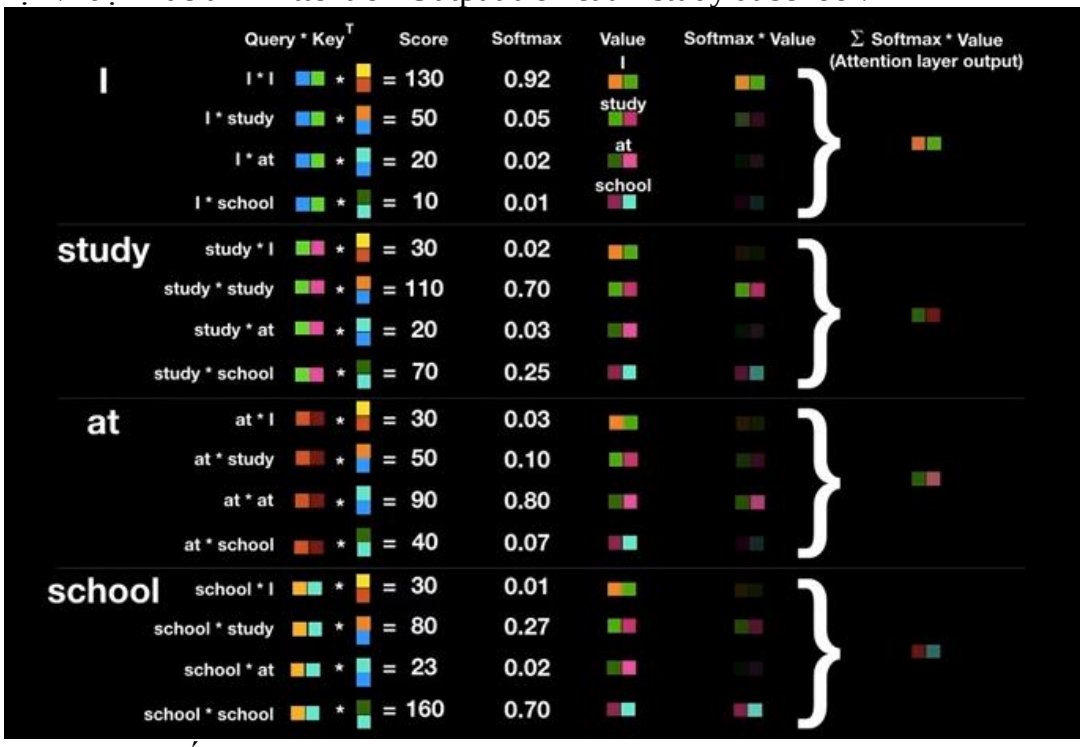
Hình 3.5. Ví dụ tính toán Attention score cho một câu

- Tính toán các Attention Output: Các giá trị Attention được tính bằng cách nhân phân phối xác suất từ bước trên với ma trận Value.



Hình 3.6. Ví dụ tính toán Attention score cho một câu

- Một ví dụ khác tính Attention Output trên câu I study at school:



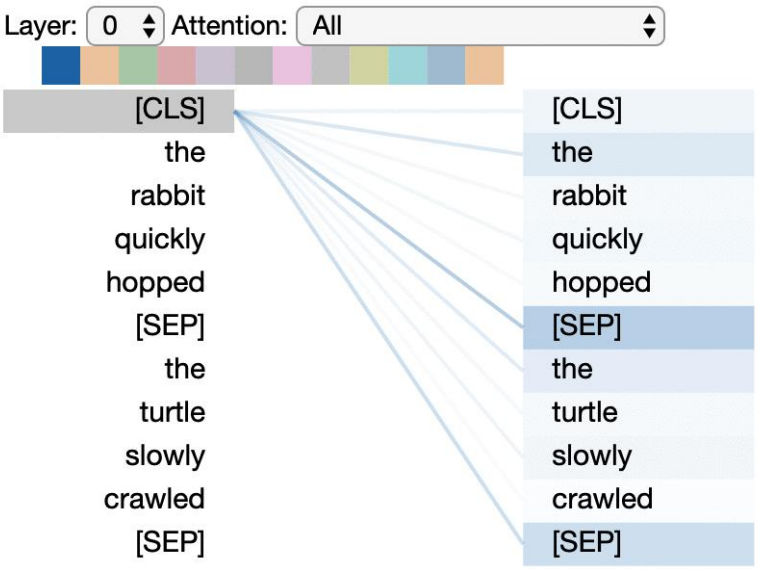
Hình 3.7. Kết quả tính attention vector cho toàn bộ các từ trong câu "I study at school"

- Kết hợp 2 bước trên ta có công thức tính Attention Output như sau:

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax(\frac{\mathbf{Q}\mathbf{K}^\mathbf{T})}{\sqrt{d_k}}\mathbf{V}$$

- Trong đó: Q,K,V là 3 vetor Query, Key, Value tạo ra ở bước 1. d_k là kích thước vector embedding.

- Minh họa tính toán attention của các từ trong 1 head:



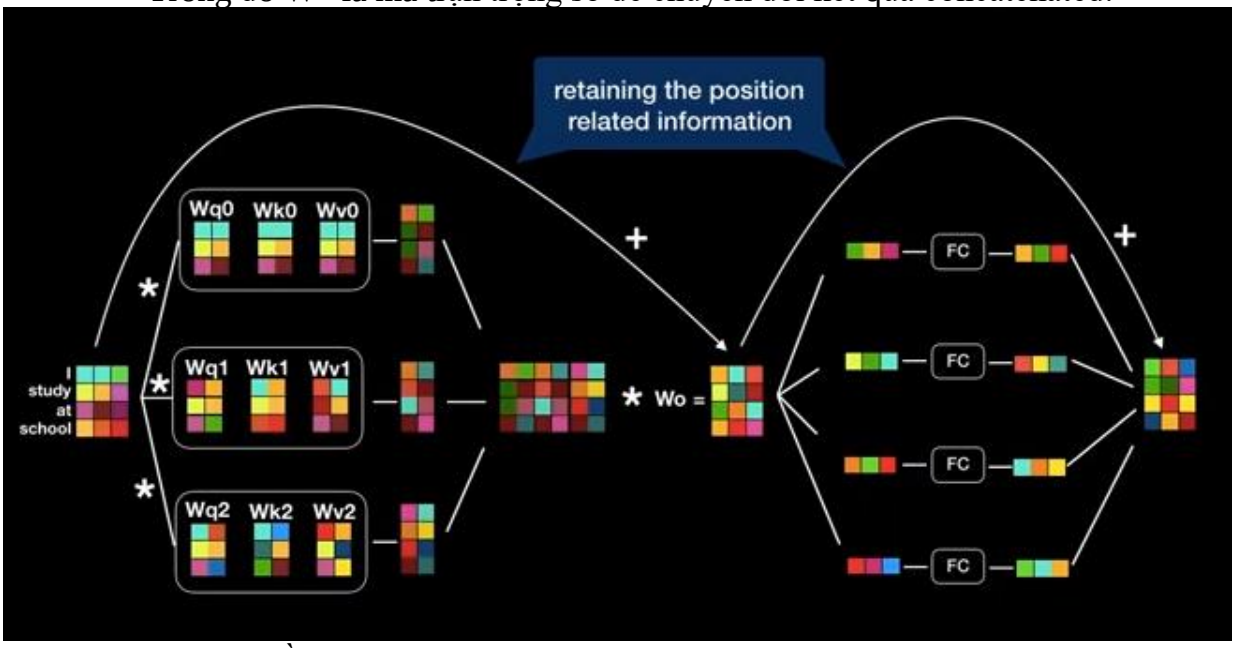
Hình 3.8. Minh họa tính toán attention của các từ trong 1 head

Multi-head attention: Kết quả từ các đầu attention khác nhau được nối lại với nhau và chuyển qua một lớp dense để tạo ra đầu ra cuối cùng của layer attention.

- Nếu có h đầu attention, mỗi đầu tạo ra một kết quả khác nhau:

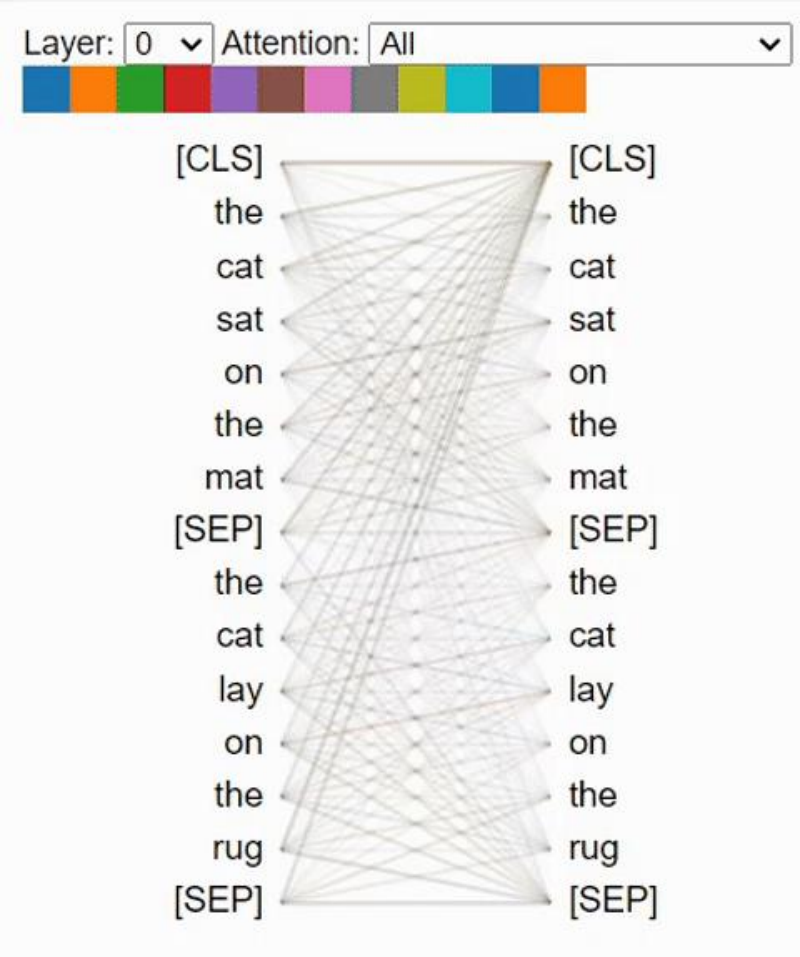
$$\text{Multi-head Attention} = \text{Concat}\left(\text{head}_1, \text{head}_2, \ldots, \text{head}_h\right) W^O$$

- Trong đó W^O là ma trận trọng số để chuyển đổi kết quả concatenated.



Hình 3.9. Sơ đồ của 1 block layer áp dụng multi-head attention layer.

- Ở sub-layer thứ 2 chúng ta sẽ đi qua các kết nối fully connected và trả ra kết quả ở đầu ra có shape trùng với input. Mục đích là để chúng ta có thể lặp lại các block này Nx lần.
- Minh hoạt attention của các từ trong câu ở nhiều head:



Hình 3.10. Minh hoạt attention của các từ trong câu ở nhiều head

3.2. BERT pre-training:

Chúng ta đào tạo BERT bằng cách sử dụng 2 nhiệm vụ dự đoán không giám sát được gọi là Masked Language Model và Next Sentence Prediction.

3.2.1. Mask Language Model task

Mask Language Model (MLM) là một nhiệm vụ trong xử lý ngôn ngữ tự nhiên (NLP) mà mô hình cố gắng dự đoán từ ngữ bị "mask" (che đi). Trong quá trình huấn luyện, một số từ trong câu sẽ bị ngẫu nhiên "che đi" (thay thế bằng token [MASK]), và mô hình phải dự đoán từ đó dựa vào ngữ cảnh còn lại của câu.

Trực quan mà thấy, BERT là một mô hình học sâu được học dựa trên ngữ cảnh 2 chiều là tự nhiên và mạnh mẽ hơn nhiều so với một mô hình chỉ dùng ngữ cảnh từ trái qua phải (hoặc ngược lại).

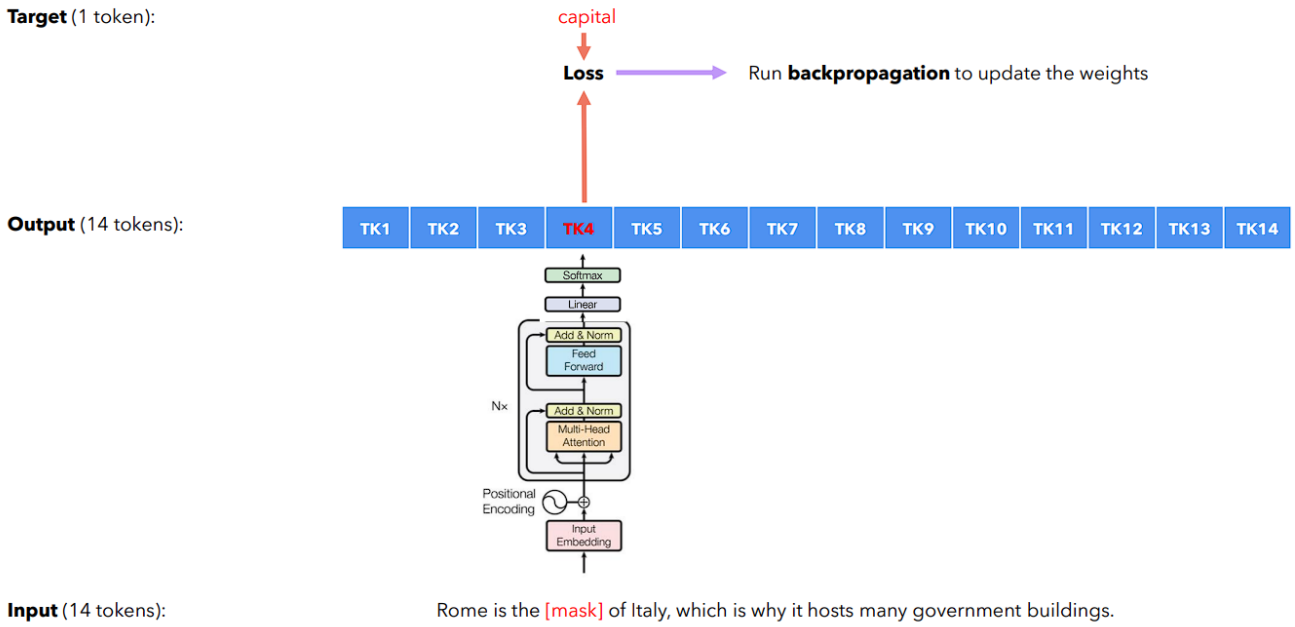
Để đào tạo một mô hình tìm ra đại diện dựa vào ngữ cảnh 2 chiều, chúng ta sử dụng một cách tiếp cận đơn giản để che giấu đi một số token đầu vào một cách ngẫu nhiên và sau đó chúng ta chỉ dự đoán các token được giấu đi. Trong trường hợp này, các hidden vectors ở lớp cuối cùng tương ứng với các tokens được ẩn đi được đưa vào 1 lớp softmax trên toàn bộ tự vựng để dự đoán.

Mặc dù điều này cho phép chúng ta có được một mô hình đào tạo 2 chiều, nhưng có 2 nhược điểm tồn tại. Đầu tiên là chúng ta đang tạo ra một sự không phù hợp giữa pre-train và fine-tuning vì các token được [MASK] không bao giờ được nhìn thấy trong quá trình tinh chỉnh mô hình. Để giảm thiểu điều này, chúng ta sẽ không phải lúc nào

cũng thay thế các từ được giấu đi bằng token [MASK]. Thay vào đó, trình tạo dữ liệu đào tạo chọn 15% tokens một cách ngẫu nhiên và thực hiện các bước như sau:
Ví dụ với câu: "Hà Nội là thủ_đô của Việt Nam" Từ được chọn để mask là từ "thủ đô".

- Thay thế 80% từ được chọn trong dữ liệu huấn luyện thành token [MASK] -> "Hà Nội là [MASK] của Việt Nam"
- 10% các từ được chọn sẽ được thay thế bởi 1 từ ngẫu nhiên. -> " Hà Nội là máy_tính của Việt Nam "
- 10% còn lại được giữ không thay đổi -> "Hà Nội là thủ_đô của Việt Nam"

Minh họa quá trình huấn luyện cho Masked Language Model task:



Hình 3.11. Ví dụ quá trình traning MLM task

3.2.2. Next Sentence Prediction task

Nhiều nhiệm vụ quan trọng trong xử lý ngôn ngữ tự nhiên như Question Answering yêu cầu sự hiểu biết dựa trên mối quan hệ giữa 2 câu văn bản, không trực tiếp sử dụng được các mô hình ngôn ngữ.

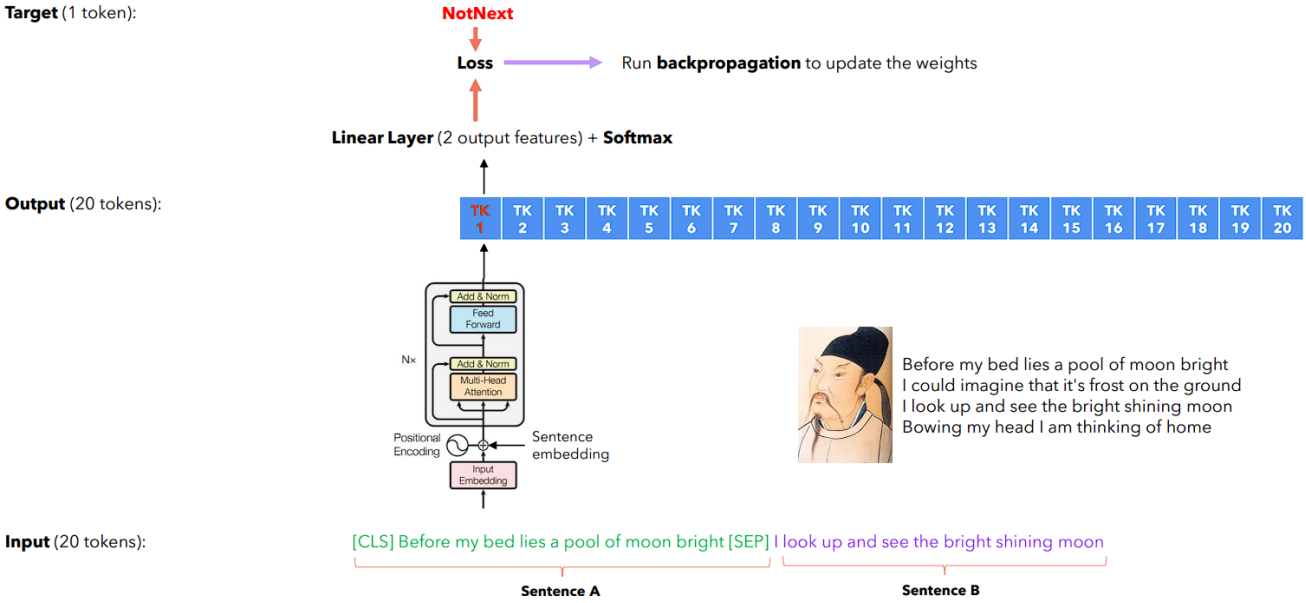
Giải thích 1 số token đặc biệt:

- Token [CLS] là token đặc biệt được thêm vào đầu mỗi chuỗi input. Token này được sử dụng để biểu diễn toàn bộ chuỗi đầu vào và đặc biệt hữu ích cho các nhiệm vụ như phân loại câu hoặc phân loại chuỗi văn bản. Sau khi câu input đi qua các lớp của mô hình BERT, biểu diễn đầu ra tương ứng với [CLS] được sử dụng như là biểu diễn của toàn bộ chuỗi. Trong Next Sentence Prediction task, mô hình sẽ dự đoán một trong hai nhãn: "IsNext" (câu thứ hai là câu tiếp theo) hoặc "NotNext" (câu thứ hai không phải là câu tiếp theo).
- Token [SEP] là token đặc biệt được sử dụng để phân tách các câu trong chuỗi đầu vào. Trong nhiệm vụ NSP, token này xuất hiện ở giữa hai câu và ở cuối chuỗi đầu vào. Chúng ta thêm một segment embedding cho câu A và một segment embedding khác cho câu B như ở phần 3.1.1 Input embedding.

Để đào tạo được mô hình hiểu được mối quan hệ giữa các câu, chúng ta xây dựng một mô hình dự đoán câu tiếp theo dựa vào câu hiện tại, dữ liệu huấn luyện có thể là một corpus bất kỳ nào. Trong quá trình pre-training, BERT được cung cấp các cặp câu.

Khoảng 50% các cặp câu là các câu liên tiếp trong văn bản (IsNext), và 50% là các cặp câu ngẫu nhiên từ văn bản khác nhau (NotNext).

Minh họa quá trình huấn luyện cho Next Sentence Prediction task:



Hình 3.12. Ví dụ quá trình traning NSP task

3.3. BERT fine-tuning

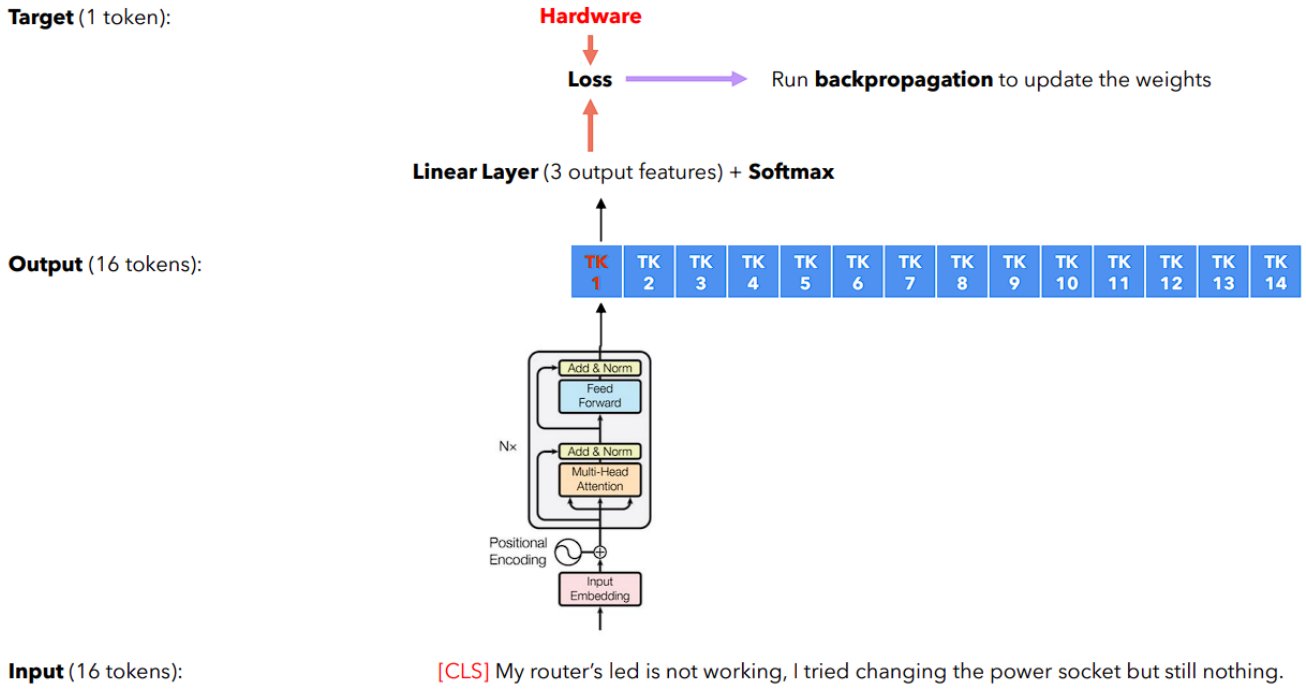
Fine-tuning là quá trình điều chỉnh mô hình BERT đã được huấn luyện trước trên một tập dữ liệu lớn (pre-trained) cho một nhiệm vụ cụ thể bằng cách tiếp tục huấn luyện mô hình đó trên một tập dữ liệu nhỏ hơn của nhiệm vụ đó. Đây là bước quan trọng giúp BERT đạt hiệu suất cao trong các ứng dụng thực tế như phân loại văn bản (text classification) và trả lời câu hỏi (question answering).

3.3.1. Phân loại văn bản (Text classification):

Fine-tuning BERT cho tác vụ phân loại văn bản bao gồm điều chỉnh các tham số của mô hình BERT đã được tiền huấn luyện để phù hợp với dữ liệu cụ thể của tác vụ phân loại. Quá trình này bắt đầu bằng cách thêm một lớp phân loại trên đầu mô hình BERT. Sau đó, mô hình được huấn luyện lại trên tập dữ liệu nhãn cụ thể, sử dụng phương pháp tối ưu hóa để điều chỉnh các tham số sao cho mô hình học cách phân loại các văn bản theo đúng các nhãn tương ứng. Kết quả là một mô hình BERT tinh chỉnh chuyên biệt cho tác vụ phân loại văn bản.

Quá trình fine-tune BERT được thực hiện qua các bước:

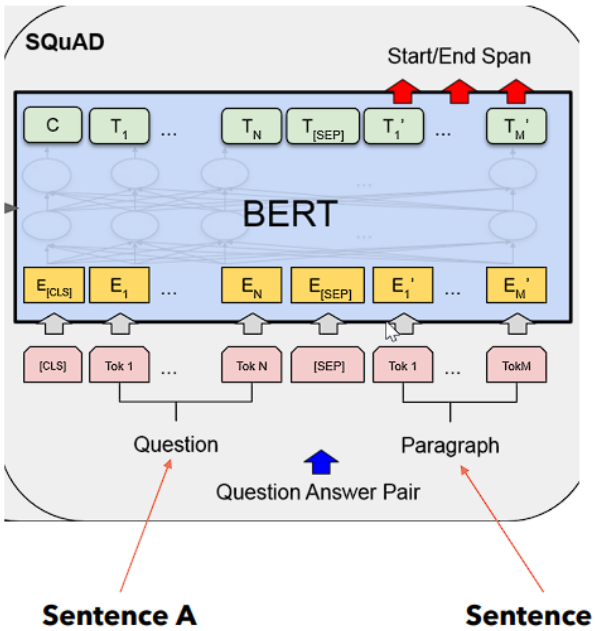
- Thêm token [CLS]: BERT thêm token đặc biệt [CLS] vào đầu mỗi câu đầu vào. Ví dụ: [CLS] Đây là một câu văn bản [SEP].
- Mã hóa câu: BERT mã hóa toàn bộ câu, bao gồm cả token [CLS].
- Sau khi đi qua các lớp của BERT, vector đầu ra của token [CLS] được sử dụng làm đại diện cho toàn bộ câu.
- Vector [CLS] được đưa qua một lớp phân loại (thường là một lớp fully-connected) để dự đoán nhãn.



Hình 3.13. Ví dụ fine-tune bert cho tác vụ phân loại văn bản

3.3.2. Question Answering

Có 2 vấn đề trong tác vụ Question Answering cần được giải quyết đó là:
Chúng ta cần tìm cách để BERT hiểu đâu là phần ngữ cảnh của input, đâu là câu hỏi.
Chúng ta cần phải tìm cách để giúp BERT cho chúng ta biết vị trí của câu trả lời bắt đầu và kết thúc trong input context được cung cấp.
Đối với các tác vụ như question answering thì ta sẽ thêm token khởi tạo là [CLS] ở đầu câu, token [SEP] ở giữa để ngăn cách 2 câu.



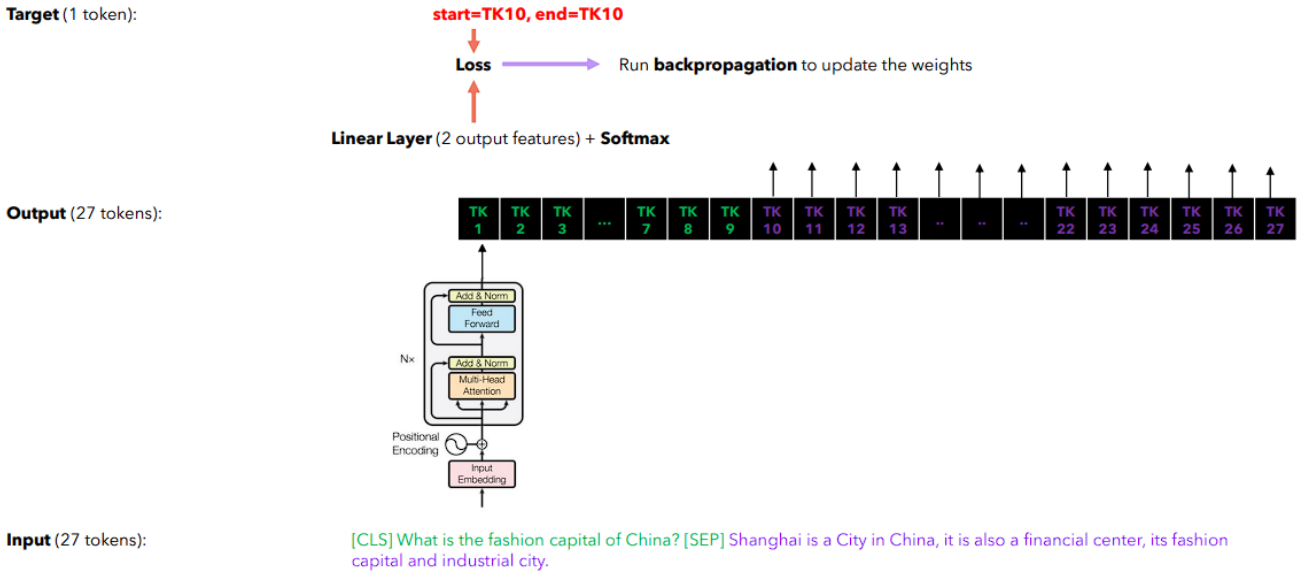
Hình 3.14. Biểu diễn câu hỏi và trả lời trong BERT

Quá trình fine-tuning BERT cho tác vụ này như sau:

- Embedding toàn bộ các token của cặp câu bằng các véc tơ nhúng từ pretrain model. Các token embedding bao gồm cả 2 token là [CLS] và [SEP] để đánh

dấu vị trí bắt đầu của câu hỏi và vị trí ngăn cách giữa 2 câu. 2 token này sẽ được dự báo ở output để xác định các phần Start/End Spand của câu output.

- Truyền các vector nhúng qua kiến trúc multi-head attention của BERT để tạo vector output từ encoder.
- Sử dụng vector output từ encoder để dự đoán vị trí bắt đầu và kết thúc của câu trả lời. Biến đổi qua một hoặc nhiều lớp linear và áp dụng softmax để thu được phân phối xác suất cho các vị trí từ, giúp xác định start và end span cho câu trả lời trong đoạn văn.



Hình 3.15. Ví dụ fine-tune BERT cho tác vụ question answering

3.4. RoBERTa và PhoBERT

RoBERTa (viết tắt của "Robustly Optimized BERT Approach") là một biến thể của mô hình BERT (Bidirectional Encoder Representations from Transformers), được phát triển bởi các nhà nghiên cứu tại Facebook AI. Giống như BERT, RoBERTa là một mô hình ngôn ngữ dựa trên transformer sử dụng self-attention để xử lý các chuỗi đầu vào và tạo ra các biểu diễn ngữ cảnh hóa của từ trong một câu.

Một điểm khác biệt chính giữa RoBERTa và BERT là RoBERTa được huấn luyện trên một tập dữ liệu lớn hơn nhiều và sử dụng một quy trình huấn luyện hiệu quả hơn. Cụ thể, RoBERTa được huấn luyện trên một tập dữ liệu gồm 160GB văn bản, lớn hơn hơn 10 lần so với tập dữ liệu được sử dụng để huấn luyện BERT. Ngoài ra, RoBERTa sử dụng một kỹ thuật dynamic masking trong quá trình huấn luyện, giúp mô hình học được các biểu diễn từ ngữ mạnh mẽ và tổng quát hơn.

RoBERTa có kiến trúc gần như tương tự với BERT, nhưng để cải thiện kết quả trên kiến trúc BERT, các tác giả đã thực hiện một số thay đổi thiết kế đơn giản trong kiến trúc và quy trình huấn luyện của nó. Những thay đổi này bao gồm:

- Loại bỏ Next Sentence Prediction (NSP): Trong dự đoán câu tiếp theo, mô hình được huấn luyện để dự đoán liệu các đoạn văn bản quan sát được đến từ cùng một tài liệu hay từ các tài liệu khác nhau thông qua một mất mát phụ trợ NSP. Các tác giả đã thử nghiệm việc loại bỏ/thêm mất mát NSP vào các phiên bản khác nhau và

kết luận rằng loại bỏ mất mát NSP khớp hoặc cải thiện nhẹ hiệu suất của downstream task.

- Huấn luyện với kích thước batch lớn hơn và chuỗi dài hơn: Ban đầu, BERT được huấn luyện trong 1 triệu bước với kích thước batch là 256 chuỗi. Trong bài báo này, các tác giả đã huấn luyện mô hình với 125 steps của 2 nghìn sequences và 31 steps với 8 nghìn sequences trong một batch. Điều này có hai lợi thế: các batch lớn cải thiện độ khó dự đoán (perplexity) trên mục tiêu mô hình ngôn ngữ bị che và cũng như độ chính xác của nhiệm vụ cuối cùng. Các batch lớn cũng dễ dàng được song song hóa thông qua huấn luyện song song phân tán.
- Thay đổi động masking pattern: Trong kiến trúc BERT, việc che phủ (masking) được thực hiện một lần trong quá trình tiền xử lý dữ liệu, dẫn đến một mẫu mặt nạ tĩnh duy nhất. Để tránh sử dụng mẫu mặt nạ tĩnh duy nhất, dữ liệu huấn luyện được nhân đôi và mask 10 lần, mỗi lần với một chiến lược che phủ khác nhau trong 40 epochs, do đó có 4 epochs với cùng một mask. Chiến lược này được so sánh với việc dynamic masking, trong đó mỗi lần dữ liệu được đưa vào mô hình, một mask khác nhau được tạo ra.

PhoBERT là một pre-trained được huấn luyện monolingual language, tức là chỉ huấn luyện dành riêng cho tiếng Việt.

PhoBERT được train trên khoảng 20GB dữ liệu bao gồm khoảng 1GB Vietnamese Wikipedia corpus và 19GB còn lại lấy từ Vietnamese news corpus. Đây là một lượng dữ liệu khả ổn để train một mô hình như BERT.

PhoBERT base v2 được train trên 20GB of Wikipedia and News texts + 120GB text từ bộ dữ liệu OSCAR-2301 và là mô hình được chúng em quyết định sử dụng làm đồ án môn học này.

PhoBERT sử dụng RDRSegmenter của VnCoreNLP để tách từ cho dữ liệu đầu vào trước khi qua BPE encoder.

Như đã nói ở trên, do tiếp cận theo tư tưởng của RoBERTa, PhoBERT chỉ sử dụng task Masked Language Model để train, bỏ đi task Next Sentence Prediction.

Nhóm sử dụng pretrain PhoBERT và ghép thêm một layer fully connected để làm nhiệm vụ phân loại.

Giả sử có một câu như sau: “tôi báo lễ_tân điều_hoà hỏng mà không thấy có bất_kỳ nhân_viên nào xuất_hiện trong cả 1 đêm gia_đình tôi ở đó cả .” Mô hình sẽ xử lý như sau:

- Đầu tiên câu tiếng Việt trên sẽ được tiền xử lý input và trở thành 1 tensor có cấu trúc như hình bên dưới gồm có các trường: ‘text’ để lưu câu văn gốc chưa được thêm các token đặc biệt, ‘input_ids’ câu văn gốc đã được thêm các token đặc biệt sau đó được mã hóa thành số thứ tự trong bộ từ điển, ‘attention_masks’ dùng để chỉ ra đâu là nội dung, đâu là padding trong ‘input_ids’, và ‘target’ là nhãn của câu văn gốc.

```
{'text': 'tôi báo lễ_tân điều_hoà hỏng mà không thấy có bất_kỳ nhân_viên nào xuất_hiện trong cả 1 đêm gia_đình
tôi ở đó cả .',
 'input_ids': tensor([    0,    70,   441, 17112,  4496,  3054,    64,    17,   108,    10,
         1403,   650,   142,   395,    12,    94,    99,   396,   161,    70,
           25,    37,    94,     5,     2,     1,     1,     1,     1,     1,
            1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
            1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
            1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
            1,     1,     1,     1,     1,     1,     1,     1,     1,     1,
            1,     1,     1,     1,     1,     1,     1,     1,     1,     1]),
 'attention_masks': tensor([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0]),
 'targets': tensor(1)}
```

- Dữ liệu sau đó sẽ được truyền vào PhoBERT trước, input đầu vào của PhoBERT là thành phần ‘input_ids’ và ‘attention_masks’, output của PhoBERT là một mảng gồm 2 đối tượng. Đối tượng thứ 0 là mảng chứa embedding của từng token trong câu. Đối tượng thứ 1 là một tensor có chiều dài là 768 là embedding của cả câu. Có 2 cách để tiến hành phân loại: hoặc là sử dụng vector embedding trong token đầu tiên của đối tượng 0 để phân loại, hoặc là sử dụng vector embedding của cả câu (chính là tensor có chiều dài 768 của đối tượng 1). Nhóm sử dụng cách thứ hai và thu được tensor như sau:

```
tensor([[-0.1415,  0.2382, -0.1153,  ..., -0.2068,  0.1148,  0.1516],
```

- Tensor này có độ dài 768 xuất ra từ PhoBERT sẽ được đưa vào một mạng FC, sau đó đầu ra của nó là một tensor có chiều dài là 3, đúng bằng số class của bài toán.

```
tensor([-0.0189, -0.1820,  0.0408], grad_fn=<ViewBackward0>)
```

- Mô hình sẽ dự đoán bằng cách lấy ra giá trị index của phần tử lớn nhất trong tensor trên tức là 2 (Positive).

3.5. Các metric đánh giá mô hình:

Hàm loss: **Cross Entropy Loss**

Đối với bài toán phân loại thành C lớp (trong trường hợp của đồ án là 3 lớp: negative, neutral, positive), hàm CrossEntropyLoss tính toán loss cho mỗi mẫu dữ liệu dựa trên đầu ra của mô hình (score của từng lớp) và nhãn thực tế của mẫu đó.
Cho một mẫu dữ liệu với vector đầu ra $y = (y_1, y_2, ..., y_c)$và nhãn thực tế là t (nhãn này được biểu diễn dưới dạng chỉ số của lớp, chẳng hạn 0 cho negative, 1 cho neutral, 2 cho positive), hàm loss CrossEntropy được tính bằng công thức sau:

$$\text{Loss}(\mathbf{y}, \mathbf{t}) = -\log\left(\frac{\exp(y_t)}{\sum_{j=1}^{C}\exp(y_j)}\right)$$

Trong đó:

- yt là phần tử của vector đầu ra y ứng với lớp mà mẫu dữ liệu thực tế thuộc vào.
- y_j là giá trị đầu ra của mô hình tương ứng với lớp j.

Để tính loss cho toàn bộ một batch dữ liệu, hàm CrossEntropyLoss lấy trung bình của các loss đối với từng mẫu trong batch với N là số lượng mẫu trong batch.

$$\text{Loss} = \frac{1}{N} \sum_{i=1}^{N} \text{Loss}(\mathbf{y}_i, \mathbf{t}_i)$$

Confusion matrix: Confusion matrix là một bảng dùng để mô tả hiệu quả của một mô hình phân loại, đặc biệt hữu ích cho các bài toán phân loại nhiều nhãn (multiclass). Một confusion matrix cho bài toán phân loại ba nhãn (negative, neutral, positive) sẽ trông như sau:

	Predicted Negative	**Predicted Neutral**	**Predicted Positive**
Actual Negative	TN (True Negative)	FP (False Positive)	FP (False Positive)
Actual Neutral	FN (False Negative)	TN (True Neutral)	FP (False Positive)
Actual Positive	FN (False Negative)	FN (False Negative)	TP (True Positive)

Bảng 3.1. Cấu trúc của confusion matrix

Trong đó:

- True Positive (TP): Số lượng mẫu thuộc lớp Positive được dự đoán đúng.
- False Negative (FN): Số lượng mẫu thuộc lớp Positive nhưng bị dự đoán nhầm thành lớp khác (Negative hoặc Neutral), Số lượng mẫu thuộc lớp Neutral nhưng bị dự đoán nhầm thành lớp khác Negative.
- False Positive (FP): Số lượng mẫu thuộc lớp khác (Neutral hoặc Negative) nhưng bị dự đoán nhầm thành Positive, Số lượng mẫu thuộc lớp Negative nhưng bị dự đoán nhầm thành Neutral.
- True Neutral (TN): Số lượng mẫu trung tính Neutral được dự đoán đúng.
- True Negative (TN): Số lượng mẫu thuộc lớp Negative được dự đoán đúng.

Accuracy (độ chính xác): là một trong những độ đo cơ bản để đánh giá hiệu suất của mô hình phân loại. Nó được định nghĩa là tỷ lệ giữa số lượng dự đoán đúng và tổng số lượng dự đoán. Công thức tính Accuracy là:

$$\text{Accuracy} = \frac{\text{Số lượng dự đoán đúng}}{\text{Tổng số lượng dự đoán}}$$

Với bài toán phân loại ba nhãn (Negative, Neutral, Positive), chúng ta sẽ tính toán số lượng dự đoán đúng bằng cách cộng các giá trị True Positive, True Neutral, và True Negative, sau đó chia cho tổng số lượng mẫu.

Precision (Độ chính xác): Precision là tỷ lệ mẫu dự đoán đúng trong số các mẫu được dự đoán là thuộc một lớp nhất định. Precision cho từng lớp được tính bằng công thức:

$$\text{Precision} = \frac{TP}{TP+FP}$$

Recall (Độ nhạy): Recall là tỷ lệ mẫu dự đoán đúng trong số các mẫu thực tế thuộc một lớp nhất định. Recall cho từng lớp được tính bằng công thức:

$$\text{Recall} = \frac{TP}{TP+FN}$$

F1-score: F1-score là trung bình điều hòa của Precision và Recall. F1-score cho từng lớp được tính bằng công thức:

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1-score Weighted (F1 trọng số): F1-score weighted tính toán F1-score tổng thể cho toàn bộ mô hình bằng cách tính trung bình F1-score của từng lớp, có trọng số theo số lượng mẫu của từng lớp. Công thức tính F1-score weighted là:

$$\text{F1-score Weighted} = \sum_i \left(\frac{N_i}{N} \cdot \text{F1-score}_i \right)$$

Trong đó:

- Ni là số lượng mẫu thuộc lớp i
- N là tổng số mẫu trong tập dữ liệu.

Chương 4:
CÀI ĐẶT VÀ THỬ NGHIỆM

4.1. Cài đặt phương pháp so sánh – SVM

4.1.1. Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là một bước quan trọng trong quá trình phân tích dữ liệu, bao gồm việc chuẩn bị dữ liệu để nó có thể được sử dụng cho các thuật toán học máy hoặc các mô hình thống kê. Đây là một phần rất quan trọng trong việc xử lý các bài toán phân loại như Sentiment Analysis.
Thông qua EDA nhóm nhận thấy bộ dữ liệu mà nhóm đang thực hiện đã tách sẵn các kí tự theo khoảng trắng, dữ liệu cũng không có những kí hiệu emoji, nó chỉ có một vấn đề lớn cần phải tiền xử lý đó là có nhiều kí tự viết tắt trong câu bình luận, từ đó khi vector hóa chúng có thể gây nhiễu, khó khăn cho mô hình.

a. Xử lý teencode, viết tắt

Viết tắt là hiện tượng sử dụng các ký tự viết tắt để thay thế cho một cụm từ hoặc từ ngữ nào đó. Việc sử dụng từ viết tắt phổ biến trong nhiều lĩnh vực, bao gồm cả tiếng Việt. Trong bài toán Sentiment Analysis (Phân tích cảm xúc) tiếng Việt, việc xử lý từ viết tắt là một bước quan trọng trong giai đoạn tiền xử lý dữ liệu.

Ví dụ: "khi m nhận phòng đc nv xếp cho phòng ở tầng 1 là phòng twin nhưng p rất xấu , chỉ giống nhà nghỉ ."
Trong câu trên có rất nhiều từ viết tắt như:
"m" → "mình", "đc" → "được", "nv" → "nhân viên", "p" → "phòng".

Xử lý viết tắt và teencode giúp đưa các từ viết tắt về dạng chuẩn, thống nhất để mô hình học máy có thể dễ dàng phân tích và hiểu được ý nghĩa của văn bản. Bên cạnh đó, việc sử dụng từ viết tắt có thể gây nhiễu cho mô hình học máy, dẫn đến kết quả phân tích không chính xác. Việc xử lý từ viết tắt giúp loại bỏ nhiễu này, từ đó nâng cao hiệu quả phân tích.

Để xử lý các từ viết tắt trong văn bản nhóm sử dụng một từ điển bởi vì đây là phương pháp đơn giản và hiệu quả nhất. Từ điển từ viết tắt có thể được xây dựng thủ công hoặc tự động từ các nguồn dữ liệu trực tuyến. Và từ điển này thậm chí có thể tái sử dụng lại khi áp dụng sang những bài toán tiếng Việt khác. Cụ thể là nhóm xây dựng một từ điển gồm những từ viết tắt thường xuyên xuất hiện trong bộ ngữ liệu và sau đó mapping các từ thành các câu chuẩn, không viết tắt.

Ví dụ: "khi m nhận phòng đc nv xếp cho phòng ở tầng 1 là phòng twin nhưng p rất xấu , chỉ giống nhà nghỉ ."
→ "khi mình nhận phòng được nhân viên xếp cho phòng ở tầng 1 là phòng twin nhưng phòng rất xấu , chỉ giống nhà nghỉ ."

b. Sử dụng VNCoreNLP để phân đoạn, tách từ (word segmentation)

Phân đoạn từ (hay còn gọi là tách từ, phân chia từ, word segmentation, tokenization) là quá trình chia một chuỗi văn bản thành các đơn vị nhỏ hơn, được gọi là **token**. Các

token này có thể là từ đơn, cụm từ, ký hiệu, hoặc bất kỳ đơn vị ngôn ngữ nào khác có ý nghĩa.

Tiếng Việt là ngôn ngữ đơn lập, nghĩa là mỗi từ có một âm tiết và không có hình thái biến đổi. Do đó, việc tách từ tiếng Việt tương đối đơn giản hơn so với các ngôn ngữ kết dính như tiếng Anh, tiếng Trung… trong đó các từ có thể được ghép từ nhiều âm tiết và có hình thái biến đổi.
Tuy nhiên, tiếng Việt cũng chính đặc điểm này khiến cho việc tách từ phức tạp hơn bởi vì tiếng Việt không sử dụng dấu cách để phân biệt các từ, đó là chưa kể tiếng Việt có nhiều từ ghép được tạo thành từ hai hoặc nhiều từ đơn, ví dụ như "nhà cửa", "bàn ghế". Việc xác định ranh giới giữa các từ có thể gặp khó khăn.

Để giải quyết khó khăn trên, nhóm sử dụng phương pháp tách từ bằng công cụ **VNCoreNLP.** VNCoreNLP là một bộ công cụ xử lý ngôn ngữ tự nhiên (NLP) mã nguồn mở dành cho tiếng Việt. Nó được phát triển bởi nhóm nghiên cứu NLP tại Đại học Quốc gia Hà Nội và được công bố lần đầu tiên vào năm 2018. VNCoreNLP cung cấp nhiều tính năng hữu ích cho các tác vụ NLP tiếng Việt, bao gồm:

- **Phân đoạn từ:** Chia văn bản tiếng Việt thành các từ riêng lẻ.
- **Gán thẻ POS:** Gán nhãn cho từng từ theo loại từ (danh từ, động từ, tính từ, v.v.).
- **Nhận diện thực thể tên riêng (NER):** Xác định và phân loại các thực thể tên riêng trong văn bản (nhân vật, địa điểm, tổ chức, v.v.).
- **Phân tích cú pháp:** Xác định cấu trúc ngữ pháp của câu, bao gồm mối quan hệ giữa các từ.
- **Phân tích ngữ nghĩa:** Xác định ý nghĩa của văn bản, bao gồm các khái niệm và mối quan hệ giữa các khái niệm.

Nhóm chỉ sử dụng tính năng phân đoạn từ của VNCoreNLP để phân đoạn các từ trong ngữ liệu. Từng từ trong ngữ liệu sẽ được phân tách với nhau bởi khoảng trắng, đối với từ ghép thì các tiếng của từ ghép sẽ được tách nhau bởi dấu "_".
Ví dụ: "mình thực sự thất vọng ." → "mình thực_sự thất_vọng ."

c. Loại bỏ stopword

Stopword (hay còn gọi là từ dừng) là những từ thường gặp trong một ngôn ngữ nhưng mang ít ý nghĩa và ít giá trị thông tin. Do đó, trong xử lý ngôn ngữ tự nhiên (NLP), stopword thường được loại bỏ khỏi dữ liệu văn bản trước khi thực hiện các tác vụ như phân tích cú pháp, phân tích ngữ nghĩa, phân loại văn bản…

Dựa vào bảng thống kê các từ thường xuyên xuất hiện trong văn bản (Hình 2.3) ta có thể thấy có những từ xuất hiện rất thường xuyên và dày đặc trong văn bản, nhưng bản thân chúng chỉ mang tính chất liên kết, làm cho câu dễ hiểu và đầy đủ ý nghĩa chứ không ảnh hưởng gì đến cảm xúc của câu.
Một số từ mà nhóm xem như là stopword trong bộ dữ liệu gồm: "có", "rất", "tôi", "ở", "của", "là", "với", "cho", "được", "thì", "đã", "trong", "sẽ", "này", "đến" …
Đối với các mô hình máy học cơ bản việc loại bỏ stopword có thể giúp khử nhiễu, giảm kích thước dữ liệu đầu vào của mô hình, từ đó giúp mô hình hoạt động nhanh hơn và chính xác hơn. Phương pháp tiền xử lý này khá phù hợp với bài toán Sentiment Analysis khi giải quyết bằng mô hình máy học cơ bản.

d. Vector hóa câu tiếng Việt bằng phương pháp TF-IDF

TF-IDF (viết tắt từ Term Frequency – Inverse Document Frequency) là một phương pháp thống kê được sử dụng trong xử lý ngôn ngữ tự nhiên (NLP) để đánh giá mức độ quan trọng của một từ đối với một văn bản trong một tập hợp các văn bản. Nói cách khác, TF-IDF giúp xác định những từ nào là đặc trưng nhất cho một văn bản cụ thể so với các văn bản khác trong cùng tập hợp.

Về cách thức hoạt động, TF-IDF được tính toán dựa trên hai yếu tố chính:

- Tần suất xuất hiện của từ (TF): Số lần xuất hiện của một từ trong một văn bản.
- Độ hiếm gặp của từ (IDF): Mức độ phổ biến của một từ trong tập hợp các văn bản.

Công thức tính toán:

TF-IDF(t, d) = TF(t, d) * IDF(t, D)

- TF(t, d): Tần suất xuất hiện của từ t trong văn bản d, được tính toán bằng cách chia số lần xuất hiện của từ t trong văn bản d cho tổng số từ trong văn bản d.
- IDF(t, D): Độ hiếm gặp của từ t trong tập hợp văn bản D, được tính toán bằng cách lấy logarit của nghịch đảo tỷ lệ phần trăm tài liệu chứa từ t.

Giả sử có bộ dữ liệu gồm 5 câu đã được tách từ bằng VNCoreNLP, tính TF-IDF của từ "khách_sạn" (nếu không coi các dấu câu là một từ):

Dữ liệu	TF("khách_sạn")	TF-IDF("khách_sạn")
khách_sạn đã cho tôi có 1 kỳ nghỉ tuyệt_vời , tôi rất hài_lòng !	1/12	1/12 x 0,223 = 0,0186
nhân_viên ở khách_sạn rất thân_thiện , hoà_đồng .	1/6	1/6 x 0,223 = 0,0372
nhân_viên nhiệt_tình hỗ_trợ khách trong thời_gian lưu_trú , tôi rất hài_lòng .	0/10	0/10 x 0,223 = 0
Khách_sạn sạch_sẽ , nhân_viên thân_thiện , phục_vụ nhanh , gần bãi biển , đi vào trung_tâm thành_phố rất tiện	1/15	1/15 x 0,223 = 0,0149
khách_sạn gần sát biển , chỉ mất 2p để đi bộ .	1/10	1/10 x 0,223 = 0,0223
IDF("khách_sạn") = ln (5/4) = 0,223		

Bảng 4.1. Ví dụ về cách tính TF-IDF

Thay vì sử dụng CountVectorizer hay phương pháp tiên tiến hơn là CBOW, nhóm chọn phương pháp TF-IDF vì nó tiện lợi, dễ cài đặt, dễ sử dụng mà lại cho kết quả thực nghiệm cao hơn so với các phương pháp còn lại. Đặc biệt với khả năng có thể đánh giá được mức độ quan trọng của từ trong văn bản cho thấy nó vô cùng phù hợp với đặc điểm của bài toán Sentiment Analysis.

Những từ ngữ hay features có chỉ số TF cao chứng tỏ nó xuất hiện nhiều trong câu văn bản, nó thế hiện tính liên quan cao đối với câu văn bản đó. Những từ ngữ có IDF cao chứng tỏ nó ít xuất hiện trong các văn bản khác, thể hiện tính

đặc trưng đối với văn bản hiện hữu. Như vậy chỉ số TF-IDF càng cao thì từ ngữ hay feature đó càng mang tính đặc trưng cho văn bản và càng có **ảnh hưởng lớn tới mô hình**.

	thời_tiết nha_trang dễ_chịu mát_mẻ và thoáng_đãng	
và	(0, 2467)	0.16762189168081051
thời_tiết	(0, 2129)	0.4549001234131668
thoáng_đãng	(0, 2020)	0.517174090842856
nha_trang	(0, 1487)	0.30197865338057805
mới_mẻ	(0, 1313)	0.4837417254702786
dễ_chịu	(0, 702)	0.4150934531528868

Hình 4.1. Ví dụ TF-IDF của một câu trong bộ dữ liệu

Có thể thấy những từ "thời_tiết", "thoáng_đãng", "mới_mẻ", "dễ_chịu" có chỉ số TF-IDF cao hơn hẳn những từ khác, điều này cho thấy nó sẽ có tác động lớn hơn khi train mô hình và nó cũng là những features quan trọng, mang tính phân loại.

4.1.2. Huấn luyện mô hình SVM (Support Vector Machine)

Sau khi đã vector hóa văn bản tiếng Việt thành dạng số, nhóm tiến hành chia dữ liệu thành 2 tập train và test với tỉ lệ **80:20** rồi sau đó huấn luyện và tinh chỉnh mô hình SVM.

SVM là viết tắt của Support Vector Machine, là một thuật toán học máy có giám sát được sử dụng phổ biến trong các bài toán phân loại và hồi quy. SVM được phát triển vào đầu những năm 1990 và nhanh chóng trở thành một trong những thuật toán phân loại mạnh mẽ và linh hoạt nhất.

Cách thức hoạt động của SVM:

- **Biểu diễn dữ liệu:** Mỗi điểm dữ liệu được biểu diễn dưới dạng một vectơ trong không gian đa chiều, với mỗi chiều tương ứng với một thuộc tính của dữ liệu.
- **Tìm siêu phẳng phân chia:** SVM tìm một siêu phẳng trong không gian đa chiều có thể phân chia tốt nhất các điểm dữ liệu thành hai hoặc nhiều lớp. Siêu phẳng này được gọi là mặt phân cách tối ưu.
- **Tính toán khoảng cách**: Khoảng cách từ mỗi điểm dữ liệu đến mặt phân cách tối ưu được tính toán.
- **Xác định vectơ hỗ trợ:** Các điểm dữ liệu nằm gần mặt phân cách tối ưu nhất được gọi là vectơ hỗ trợ. Vectơ hỗ trợ đóng vai trò quan trọng trong việc xác định vị trí của mặt phân cách.
- **Dự đoán:** Dựa vào vị trí của điểm dữ liệu mới so với mặt phân cách tối ưu, SVM có thể dự đoán lớp của điểm dữ liệu mới.

Nhóm thực hiện cài đặt mô hình SVC được cung cấp sẵn bởi thư viện scikit-learn hai siêu tham số quan trọng của mô hình này đó là:

- kernel: là một hàm xác định cách thức so sánh các điểm dữ liệu trong không gian đa chiều. Scikit-learn cung cấp một số kernel khác nhau, bao gồm:

+ 'linear': Đây là kernel mặc định, sử dụng một đường thẳng để phân chia các điểm dữ liệu.

+ 'poly': Sử dụng một đa thức để phân chia các điểm dữ liệu.

+ ‘rbf’: Sử dụng hàm kernel Radial Basis Function (RBF) để phân chia các điểm dữ liệu.

- C: là tham số điều chỉnh mức độ phạt đối với các điểm dữ liệu nằm ngoài lề tối ưu. Giá trị C cao hơn dẫn đến mô hình học chặt chẽ hơn, có thể dẫn đến hiện tượng quá học (overfitting) và giảm độ chính xác trên dữ liệu chưa nhìn thấy. Giá trị C thấp hơn dẫn đến mô hình học lỏng lẻo hơn, có thể dẫn đến hiện tượng thiếu học (underfitting) và giảm độ chính xác trên dữ liệu tập luyện. Mặc định của thư viện giá trị C = 1.

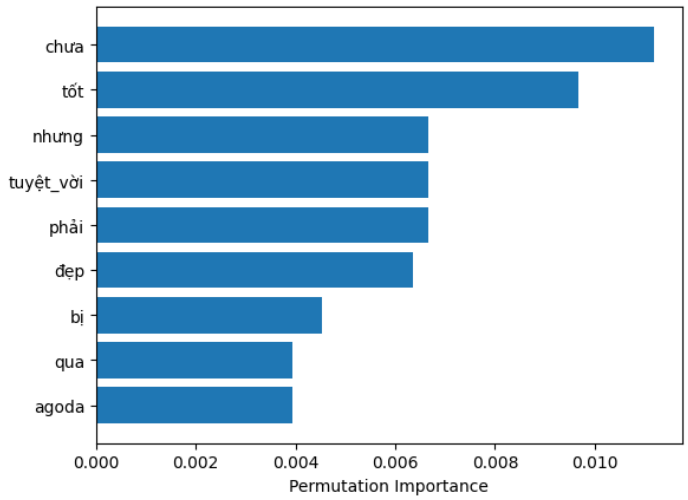
Nhóm thực hiện fine-tune mô hình bằng phương pháp Grid Search và đạt được chỉ số accuracy cao nhất đối với bộ siêu tham số là **kernel = ‘rbf’, C = 1.2**.

Các features được chọn của mô hình SVM:

Bản thân các phương pháp tiền xử lý như remove stopword nhóm đã thực hiện loại bỏ một cách thủ công những từ ngữ hay features xuất hiện nhiều nhưng không có ý nghĩa phân loại trong bộ dữ liệu.

Đối với những features còn lại thì khi vectorizer bằng phương pháp TF-IDF không loại bỏ bớt features nhưng nó thay đổi giá trị hay trọng số của các features, các features ít đặc trưng cho câu hơn sẽ có giá trị nhỏ hơn. Thông qua đó giúp chọn ra được những features có đặc trưng cao hơn và khi huấn luyện chúng cũng tác động đến mô hình lớn hơn.

Sau khi huấn luyện xong mô hình SVM, Nhóm cũng thực hiện tính toán chỉ số Permutation Importance. Đây là một chỉ số đại diện cho mức độ ảnh hưởng của từng features hay từ ngữ đến mô hình. Nó sẽ xáo trộn những giá trị trong từng feature của tập test rồi tiến hành dự đoán và đo các chỉ số đánh giá, sau đó đem so sánh với chỉ số đánh giá của mô hình khi dự đoán trên dữ liệu test gốc. Chỉ số này dương chứng tỏ feature đó khi bị xáo trộn sẽ làm cho mô hình tệ đi. Ngược lại khi âm thì khi xáo trộn mô hình sẽ tốt hơn. Dưới đây là chỉ số Permutaion Importance của 30 từ cao nhất:



Hình 4.2. Biểu đồ Permutation Importance của 30 từ cao nhất

Giải thích:

- Các từ “chưa”, “tốt”, “tuyệt_vời”, “đẹp” đều là các tính từ chỉ sắc thái cảm xúc tiêu biểu và xuất hiện với tần suất nhiều trong văn bản do đó chúng có tác động lớn đến việc phân loại của mô hình SVM.

- Đối với từ “nhưng” rất thường xuyên xuất hiện trên các câu neatral nhữ câu có từ này thường có 2 vế đối lập nhau. Từ này có thể gây đảo ngược trạng thái của câu do đó nó vô cùng quan trọng.
- Các từ “phải”, “bị”, “qua” thường là phó từ đi kèm với những từ chỉ sắc thái cảm xúc khác. Ví dụ: “phải đi bộ lên phòng”, “điều_hòa bị hư”,… tuy không có tính từ thể hiện sắc thái cảm xúc nhưng nó thường xuất hiện trong trường hợp Negative.
- Từ “agoda” là danh từ riêng chỉ một nền tảng đặt phòng khách sạn trực tuyến vô cùng nổi tiếng. Những bình luận có từ này thường là neutral vì nó sẽ liên quan tới sự đánh giá đối với dịch vụ của nền tảng này hơn là với khách sạn.

4.2. Cài đặt phương pháp chính – PhoBERT

4.2.1. Tiền xử lý dữ liệu

Đối với PhoBERT, trước tiên nhóm cũng thực hiện 2 bước đó là xử lý teencode, viết tắt và phân đoạn từ tiếng Việt bằng VNCoreNLP (do yêu cầu của mô hình) tương tự như phương pháp thử nghiệm bên trên. Sau đó nhóm chia dữ liệu thành 3 bộ train, val, test với tỉ lệ **60:20:20**.

PhoBERT có cung cấp sẵn một bộ Tokenizer, nhóm sử dụng công cụ AutoTokenizer được cung cấp bởi thư viện transformer. Bộ tokenizer này của PhoBERT có nhiệm vụ sau:

- **Phân token:** Bộ tokenizer chia văn bản đầu vào thành các đơn vị nhỏ hơn gọi là token. Đối với PhoBERT, điều này thường có nghĩa là chia văn bản thành từ và các từ con, tuân theo các nguyên tắc token hóa của BERT. Đồng thời, nó còn tự động thêm các token đặc biệt như [CLS] (token phân loại ở đầu chuỗi) và [SEP] (token phân tách ở cuối chuỗi hoặc giữa các cặp câu). Sử dụng token [UNK] để đại diện cho các từ không có trong từ vựng.
- **Chuẩn hóa độ dài của dữ liệu:** Đảm bảo tất cả các chuỗi đầu vào có cùng độ dài bằng cách thêm các token padding ([PAD]) vào các chuỗi ngắn hơn. Cắt ngắn các chuỗi vượt quá độ dài tối đa cho phép của mô hình.
- **Tạo attention mask:** Tạo ra các mặt nạ chú ý cho biết token nào là dữ liệu thực (1) và token nào là padding (0). Điều này giúp mô hình tập trung vào các phần liên quan của đầu vào.
- **Chuyển đổi token thành ID:** Ánh xạ mỗi token thành một ID số duy nhất theo từ vựng của PhoBERT.

Sử dụng **AutoTokenizer** cho 3 tập của bộ dữ liệu với các tham số:

- padding="max_length": tham số này chỉ định rằng tất cả các chuỗi đầu vào sẽ được điền thêm (padding) để đạt đến độ dài tối đa max_length.
- max_length=max_seq_length: xác định độ dài tối đa cho các chuỗi đầu vào sau khi đã được token hóa. Các chuỗi dài hơn sẽ bị cắt ngắn, các chuỗi ngắn hơn sẽ được điền thêm (padding).
- add_special_tokens=True: thêm các token đặc biệt như [CLS] ở đầu chuỗi và [SEP] ở cuối chuỗi hoặc giữa các cặp câu.
- return_attention_mask=True: trả về mặt nạ chú ý (attention mask), cho biết token nào là dữ liệu thực (1) và token nào là padding (0).

- return_tensors='pt': trả về các tensor của PyTorch (thay vì các định dạng khác như NumPy arrays).

Sau khi đã tokenize, dữ liệu sẽ được chuyển thành dạng tensor rồi đưa vào Dataloader với thông số **batch_size = 16, shuffle = True**. Việc chia dữ liệu ra 16 batch giúp giảm bớt yêu cầu về bộ nhớ và cải thiện tốc độ huấn luyện. Bên cạnh đó, việc xáo trộn dữ liệu trong mỗi epoch (khi đặt Shuffle = True) là rất quan trọng để tránh mô hình học theo thứ tự của dữ liệu và giúp cải thiện độ chính xác.

4.2.2. Cài đặt PhoBERT

Nhóm tiến hành cấu hình mô hình phân loại trước khi đưa vào huấn luyện. Mô hình phân loại này gồm 2 phần: **pre-trained PhoBERT** và **lớp fully connected**.
Pre-trained của mô hình PhoBERT đã được cung cấp sẵn bởi đối tượng AutoModel của thư viện transformer, chỉ cần sử dụng phương thức AutoModel.from_pretrained ("vinai/phobert-base-v2") là có thể sử dụng được như một lớp neural network. Nhóm ghép thêm một lớp fully connected vào sau lớp PhoBERT với số tín hiệu đầu vào là số tín hiệu đầu ra của PhoBERT, số tín hiệu đầu ra là số lượng class (numclasses = 3).

Sau khi đã cấu hình xong mô hình phân loại, Nhóm đưa Dataloader của dữ liệu train vào và train **3 epochs** với learning rate là **2.10^{-5}**, optimizer **Adam** và hàm loss là **Cross Entropy Loss**.

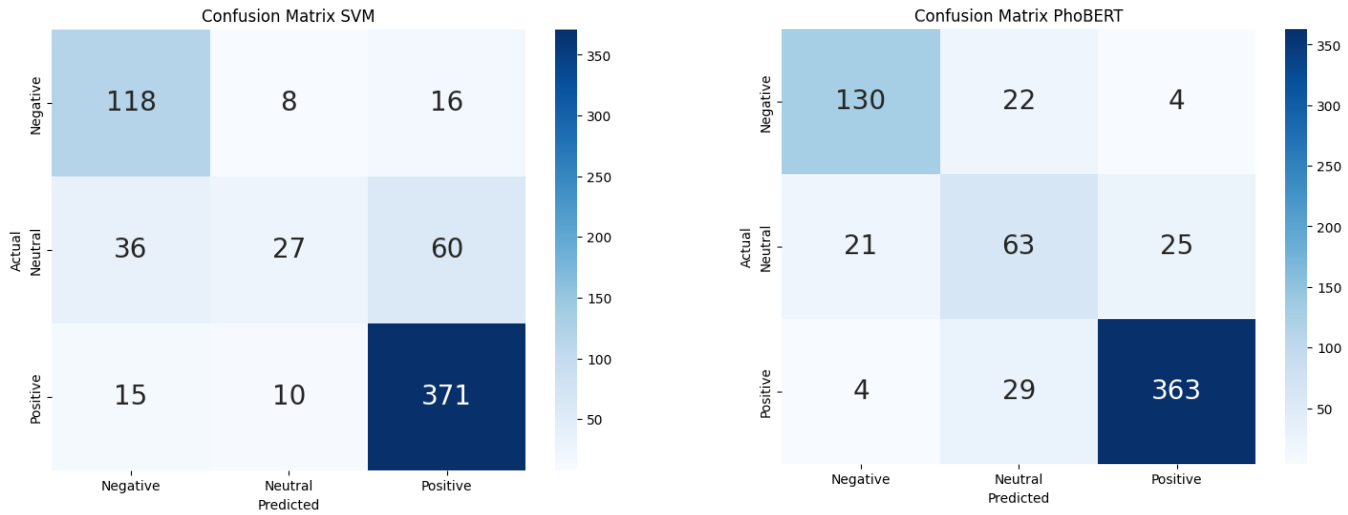
4.3. Kết quả thử nghiệm

4.3.1. Phân tích kết quả đạt được

Kết quả sau khi train của 2 phương pháp sẽ được đánh giá bằng độ đo accuracy, f1-score weighted và confusion matrix.

Mô hình	Accuracy	F1-weighted
SVM	78%	75%
PhoBERT	84%	84%

Bảng 4.2. Kết quả đánh giá của 2 phương pháp



Hình 4.3. Confusion Matrix của 2 mô hình SVM và PhoBert

Dựa vào bảng trên ta có thể thấy được kết quả trên cả 2 độ đo accuracy và f1-weighted của mô hình PhoBERT đều vô cùng vượt trội so với một mô hình máy học cơ bản là SVM.

Điều này đạt được là do mô hình PhoBERT đã được huấn luyện trước với lượng dữ liệu vô cùng lớn, hơn nữa nó cũng có khả năng hiểu được các cấu tạo từ và mối quan hệ giữa các từ trong tiếng Việt từ đó có thể hiểu được ngữ nghĩa của văn bản. Tuy nhiên việc sử dụng mô hình PhoBERT cũng sẽ có thời gian dự đoán lâu hơn so với mô hình SVM đơn giản, nó cũng đòi hỏi nhiều tài nguyên hơn.

Khi xét sâu hơn về kết quả đánh giá trên các lớp dữ liệu khác nhau, ta nhận thấy mô hình có xu hướng dự đoán thiên vị cho lớp positive khá nhiều, tiếp theo là lớp negative. Lớp neutral hầu như rất khó phân loại và ít giá trị dự đoán đúng trong khoảng này. Điều này là do sự nhập nhằng khi phân lớp cộng với việc dữ liệu bị mất cân bằng rất lớn, có quá ít mẫu dữ liệu thuộc về lớp neutral so với những lớp còn lại.

4.3.2. Phân tích một số trường hợp sai

STT	Bình luận	Truth	Predict	Nhận xét
1	để quên vài vật_dụng cá_nhân trị_giá khoảng $ 30 nhưng chẳng ai thông_báo hay trả lại	Negative	Neutral	Đây là câu bình luận mang hàm ý tiêu cực thông qua cụm "chẳng ai". Nhưng vì khi tách ra thành token thì từ "chẳng" và "ai" không tự thể hiện ý tiêu cực rõ ràng mà mô hình SVM lại không có khả năng nhận biết ngữ cảnh nên nó đã nhận nhầm.
2	một ngày nghỉ_ngơi tại	Neutral	Positive	Đây chỉ đơn thuần là cung cấp thông tin nên đã được đánh ground truth là Neutral. Tuy nhiên do dữ liệu bị lệch khá nhiều sang nhãn positive nên đối với những trường hợp hi hữu mô hình cũng dự đoán sang positive.
3	ăn sáng ngon tuy_nhiên nên mở nhạc_nhẹ nhàng vào thời_điểm ăn sáng	Positive	Negative	Câu nói mang hàm ý khen qua từ "ngon", phần phía sau chỉ mang tính góp ý và cũng không có từ ngữ tiêu cực nhưng có lẽ do lượng ngữ liệu còn hạn chế nên mô hình đã dự đoán sai.
4	nếu resort chịu đầu_tư thêm hạng_mục giải_trí bên khu_vực để du_khách có_thể vui_chơi thư_giãn sau tham_quan đà_lạt càng tuyệt_vời	Neutral	Positive	Câu bình luận chỉ mang tính góp ý nhưng phía sau lại có nhiều từ mang ý nghĩa khen nên mô hình bị nhầm lẫn, một phần cũng là do SVM chưa hiểu được ngữ cảnh.

5	khách_sạn nên đầu_tư thay_thế nệm mới	Neutral	Positive	Câu bình luận mang tính góp ý, nếu xét về hàm ý sâu xa còn có thể là đang tiêu cực vì nệm của khách sạn đã cũ. Nhưng thể hiện trên mặt chữ lại có từ "mới" mang ý tích cực nên mô hình đã dự đoán nhầm lẫn

Bảng 4.3. Một số trường hợp sai của SVM

Ta có thể thấy, do nhược điểm lớn của SVM là không thể hiểu được ngữ cảnh, do đó mô hình rất dễ bị đánh lừa khi câu bình luận có chứa các token mang ý nghĩa thể hiện cảm xúc nhưng xét về cả ngữ cảnh câu văn thì nó chỉ mang ý đóng góp. Thêm nữa là mô hình SVM cũng bị ảnh hưởng khá nhiều bởi dữ liệu khi nó bị lệch về phía nhãn Positive.

STT	Bình luận	Truth	Predict	Nhận xét
1	nói_chung lần sau tôi sẽ chọn 1 khách sạn khác với mức giá tương_tự .	Negative	Neutral	Câu bình luận mang hàm ý tiêu cực về chất lượng phòng của khách sạn có thể là không tốt so với mức giá của nó. Nhưng trong câu không có từ ngữ hay ngữ cảnh thể hiện sự tiêu cực rõ ràng làm cho mô hình đã bị nhầm lẫn khi dự đoán.
2	Text: khách_sạn nên thay tv màn_hình crt bằng lcd .	Negative	Neutral	Câu bình luận có hàm ý ẩn là đang không hài lòng về tivi ở khách sạn. Nhưng trong câu có từ "nên" đặt trong nhiều ngữ cảnh có thể là đang góp ý nên đã đánh lừa được mô hình PhoBERT
3	ks hoàng lộc có phong_cách vila , gia_đình quản_lý .	Neutral	Positive	Bình luận này đơn thuần cung cấp thông tin về khách sạn chứ không có ý tích cực hay tiêu cực. Dựa vào ngữ cảnh ở vế đầu tiên có cụm "phong_cách villa" có thể PhoBERT đã hiểu là đang khen về phong cách của khách sạn nhưng thực tế câu này thiên về cung cấp thông tin nhiều hơn.
4	thanks agoda !	Neutral	Positive	Bình luận có ý tích cực, nhưng là về Agoda – một trang web đặt phòng khách sạn, do đó nó được đánh nhãn Neutral. Để hiểu được ý nghĩa của câu còn cần cả những thông tin về Agoda nên có thể PhoBERT đã nhầm với

				những tên riêng của khách sạn hoặc địa danh.
5	chẳng_thể chê vào đâu được , có_lẽ diện_tích tổng_thể quá nhỏ nên không lên được hàng 5 sao .	Positive	Negative	Bình luận có hàm ý tích cực, về sau thể hiện sự tiếc nuối vì khách sạn tuy tốt nhưng chưa được lên 5 sao. Mô hình hiểu được ngữ cảnh nhưng ngữ liệu chưa đủ để nó có thể hiểu được hàm ý của câu nói.

Bảng 4.3. Một số trường hợp sai của PhoBERT

Hầu hết các trường hợp dự đoán sai của cả 2 mô hình đều liên quan tới sự nhập nhằng liên quan tới nhãn Neutral. PhoBERT tuy đã khác phục được nhược điểm lớn của SVM ở vấn đề ngữ cảnh nhưng hàm ý ẩn bên trong của câu tiếng Việt thì vô cùng phong phú, mô hình vẫn không thể đạt được kết quả chính xác hoàn toàn. Vấn đề về câu dự đoán thiếu ngữ cảnh cần thiết trong câu cũng gây khó khăn không nhỏ đối với mô hình PhoBERT để dữ đoán.

Chương 5:

KẾT LUẬN

Trong đề tài này, nhóm đã trình bày về bài toán phân tích văn bản tiếng việt, sử dụng **PhoBERT** làm phương pháp chính và so sánh với **SVM** (Supported Vector Machine).

Từ kết quả nhận được của các phương pháp, ta có thể thấy rằng PhoBERT đã chứng tỏ độ hiệu quả khi cho ra kết quả đánh giá vượt trội hơn SVM về nhiều mặt. Sự vượt trội này đến từ việc BERT nói chung, hay PhoBERT nói riêng là một mô hình học sâu, được thiết kế để dành riêng cho các tác vụ về NLP. BERT học các biểu diễn ngữ cảnh (contextual representations) nên có khả năng hiểu biết sâu về ngữ nghĩa. Cùng với đó, mô hình này được huấn luyện trên một tập dữ liệu rất lớn giúp mô hình nắm bắt được các cấu trúc ngôn ngữ phức tạp. Trong khi đó, SVM là mô hình học máy, chuyên về phân loại và hồi quy. SVM cũng không có khả năng hiểu ngữ cảnh tốt vì nó chủ yếu dựa vào các vector từ tĩnh (static word vectors). Tuy nhiên, nếu xét về mặt thời gian và tài nguyên, PhoBERT cần tiêu tốn nhiều hơn so với SVM bởi cấu trúc phức tạp của mô hình.

Bên cạnh đó nhóm cũng nhận thấy rằng, phân tích dữ liệu và tiền xử lý dữ liệu cũng là những bước rất quan trọng. Việc phân tích dữ liệu giúp nhóm có sự hiểu biết rõ ràng hơn về bộ dữ liệu đang sử dụng, từ đó xử lý dữ liệu một cách hiệu quả hơn. Tiền xử lý dữ liệu (phân đoạn, tách từ, loại bỏ teencode, TF-IDF, …) đúng cách sẽ giúp dữ liệu trở nên sạch hơn, cải thiện độ chính xác của bài toán một cách đáng kể.

Sau khi tìm hiểu đề tài, nhóm hiểu rõ được các bước để xây dựng một mô hình giải quyết bài toán phân tích văn bản tiếng việt, cách thức hoạt động và cách sử dụng PhoBERT cũng như đã tìm hiểu thêm một số cách làm sạch dữ liệu. Từ nền tảng trên, nhóm có thể tiếp tục tìm hiểu các hướng tiếp cận khác cho bài toán này, cũng như tiếp cận các bài toán nâng cao hơn.

TÀI LIỆU THAM KHẢO

[1] "Support Vector Machines," [Online]. Available: https://scikit-learn.org/stable/modules/svm.html.

[2] Dang Van Thin, Duong Ngoc Hao, Ngan Luu-Thuy Nguyen, "Sentiment Analysis in Code-Mixed Vietnamese-English Sentence-level," Association for Computational Linguistics, 2022.

[3] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," 2019.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2019.

[5] Dat Quoc Nguyen, Anh Tuan Nguyen, "PhoBERT: Pre-trained language models for Vietnamese," 2020.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, "Attention Is All You Need," 2017.

[7] "The A-Z guide to Support Vector Machine," [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/support-vector-machine-better-understanding/.

[8] "Grid Searching From Scratch using Python," 2024. [Online]. Available: https://www.geeksforgeeks.org/grid-searching-from-scratch-using-python/.

[9] "Optimizer- Hiểu sâu về các thuật toán tối ưu (GD,SGD,Adam,..)," [Online]. Available: https://viblo.asia/p/optimizer-hieu-sau-ve-cac-thuat-toan-toi-uu-gdsgdadam-Qbq5QQ9E5D8.

[10] N. C. Thắng, "Thử nhận diện cảm xúc văn bản Tiếng Việt với PhoBert," [Online]. Available: https://miai.vn/2020/12/29/bert-series-chuong-3-thu-nhan-dien-cam-xuc-van-ban-tieng-viet-voi-phobert-cach-1/.

[11] "Cách tách từ cho Tiếng Việt," [Online]. Available: https://streetcodevn.com/blog/vntok.

[12] L. T. Hương, "Tách từ tiếng Việt," [Online]. Available: https://users.soict.hust.edu.vn/huonglt/UNLP/3_wordsegmentation.pdf.

[13] "Simple Sentiment Analysis — Python," [Online]. Available: https://medium.com/analytics-vidhya/simple-sentiment-analysis-python-bf9de2d75d0.