

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

LÊ PHAN DUY TÙNG
NGUYỄN LÊ DUY

MỘT PHƯƠNG PHÁP CẢI TIẾN MÔ HÌNH NHÂN TỐ ĂN
CHO BÀI TOÁN TƯ VẤN NHÓM

KHÓA LUẬN TỐT NGHIỆP CỦ NHÂN CNTT
CHƯƠNG TRÌNH CHẤT LƯỢNG CAO

TP. HCM, 07/2024

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

LÊ PHAN DUY TÙNG - 20127661
NGUYỄN LÊ DUY - 20127482

MỘT PHƯƠNG PHÁP CẢI TIẾN MÔ HÌNH NHÂN TỐ ĂN
CHO BÀI TOÁN TƯ VẤN NHÓM

KHÓA LUẬN TỐT NGHIỆP CỦ NHÂN CNTT
CHƯƠNG TRÌNH CHẤT LƯỢNG CAO

GIẢNG VIÊN HƯỚNG DẪN
ThS. TIẾT GIA HỒNG

TP. HCM, 07/2024

Lời cảm ơn

Lời đầu tiên, chúng em xin chân thành cảm ơn cô Tiết Gia Hồng, người đã tận tình chỉ dẫn chúng em trong suốt quá trình tìm hiểu để tài cung như xây dựng đề cương và hoàn thành khóa luận. Nhờ sự dẫn dắt và những lời khuyên quý báu của cô đã giúp chúng em rất nhiều trong quá trình hoàn thiện khóa luận.

Chúng em xin gửi lời cảm ơn tới Trường Đại học Khoa Học Tự Nhiên và các thầy cô bạn bè đã truyền đạt những kiến thức và đã rèn luyện kỹ năng cho chúng em trong suốt quá trình học tập tại trường.

Cuối cùng chúng em xin cảm ơn gia đình và mọi người xung quanh đã tạo điều kiện để em được học tập và hoàn thành Khóa Luận Tốt Nghiệp. Sự chăm sóc, dạy bảo và động viên của mọi người là nguồn động lực rất lớn để cho chúng em có thể hoàn thiện bản thân và trưởng thành cho tới ngày hôm nay.

Chúng em đã cố gắng hoàn thành khóa luận chỉ trong phạm vi kiến thức của bản thân nên không thể tránh khỏi những thiếu sót và hạn chế. Xin kính mong nhận được sự thông cảm, góp ý và chỉ bảo của quý thầy cô.

Mục lục

Lời cảm ơn	i
Mục lục	ii
Danh mục hình ảnh	iv
Bảng thuật ngữ	vii
Tóm tắt	ix
Chương 1. Giới thiệu.....	1
1.1 Đặt vấn đề	1
1.2 Mục tiêu	2
1.3 Phạm vi đề tài	3
1.4 Cách tiếp cận dự kiến.....	3
1.5 Cấu trúc cuốn luận	3
Chương 2. Các nghiên cứu liên quan	4
2.1 Phát biểu về bài toán gợi ý.....	4
2.2 Các mô hình gợi ý cho người dùng cá nhân	5
2.3 Các chiến lược gợi ý cho nhóm người dùng.....	11
2.3.1 Chiến lược tổng hợp gợi ý.....	12
2.3.2 Chiến lược tổng hợp profile	12
2.4 Mô hình nhân tố ẩn cho bài toán gợi ý nhóm theo chiến lược tổng hợp profile	13
2.5 Giới thiệu về mô hình BERT	18
2.5.1 Thế nào là một mô hình ngôn ngữ :	18
2.5.2 Làm thế nào để huấn luyện một mô hình ngôn ngữ:.....	18
2.5.3 Cách một mô hình ngôn ngữ dự đoán	19
2.5.4 Giới thiệu mô hình BERT.....	21
2.5.6 Sự quan trọng của ngữ cảnh trái và ngữ cảnh phải	24
2.5.7 Quy trình huấn luyện trước của mô hình BERT	25
Chương 3. Hướng tiếp cận	32

3.1 Áp dụng chiến lược tổng hợp profile dựa trên trung bình tất cả đánh giá quan sát được kết hợp trọng số chi tiết.....	32
3.2 Tinh chỉnh mô hình BERT cho bài toán phân loại văn bản	35
3.3 Hệ thống được tích hợp cải tiến.....	39
Chương 4. Thực nghiệm	41
4.1 Thiết lập thực nghiệm	41
4.2 Bộ dữ liệu.....	42
4.3 Độ đo.....	43
4.4 Kết quả thực nghiệm.....	43
Chương 5. Kết luận và hướng phát triển	46
5.1 Kết quả đạt được	46
5.1.1 Kết quả đạt được của cá nhân.....	46
5.1.2 Kết quả đạt được của khóa luận	46
5.2 Kết luận mô hình thực nghiệm	47
5.3 Hướng phát triển	47
Tài liệu tham khảo	49
Phụ lục	55
A. Kiến thức về mạng nơ-ron trong lĩnh vực học sâu	55
A1. Tổng quan về mạng neural	55
A2. Huấn luyện mạng nơ-ron	56
A3. Tính toán đầu ra của nơ-ron được diễn ra như thế nào	59
A4. Lan truyền, lan truyền ngược, hàm chi phí và hàm mất mát.....	60
A5. Véc-tơ hóa và ma trận hóa trong mạng nơ-ron.....	68
B. Kiến trúc Transformer (Encoder)	71
1. Lớp nhúng đầu vào	72
2. Lớp mã hoá vị trí	74
3. Cơ chế tự chú ý đa đầu	75
4. Mạng truyền thẳng theo vị trí	79
5.Thành phần thêm và chuẩn hóa	80
6. Lớp tuyến tính và Softmax	81

Danh mục hình ảnh

Hình 2.1: Hệ thống gợi ý.....	4
Hình 2.2: Ví dụ ma trận đánh giá user-item.....	5
Hình 2.3: Minh họa hướng lọc cộng tác và hướng lọc dựa trên nội dung	7
Hình 2.4: Minh họa cách hoạt động của hệ thống lọc cộng tác dựa trên bộ nhớ (memory-based Collaborative Filtering).....	8
Hình 2.5: Ý tưởng của mô hình nhân tố ẩn.....	10
Hình 2.6: Xấp xỉ ma trận mô hình nhân tố ẩn.....	11
Hình 2.7: Quy trình tư vấn theo chiến lược tổng hợp gợi ý	12
Hình 2.8: Quy trình tư vấn theo chiến lược tổng hợp profile	13
Hình 2.9: Quy trình phát sinh gợi ý nhóm của mô hình nhân tố ẩn theo chiến lược tổng hợp profile	17
Hình 2.10: Ví dụ giá trị mà mô hình ngôn ngữ cần tính toán	18
Hình 2.11: Đoạn thơ mà mô hình cần học	18
Hình 2.12: Minh họa quá trình huấn luyện mô hình ngôn ngữ	19
Hình 2.13: Minh họa đầu vào để mô hình bắt đầu dự đoán	20
Hình 2.14: Minh họa quá trình mô hình dự đoán từng từ tiếp theo trong bài thơ	20
Hình 2.15: Kiến trúc tổng quan của mô hình BERT.....	23
Hình 2.16: Sự khác nhau giữa mô hình BERT và GPT/ Llama.....	24
Hình 2.17: Minh họa ngữ cảnh bên trái và ngữ cảnh bên phải trong ngôn ngữ tự nhiên	25
Hình 2.18: Minh họa mô hình cho tác vụ dự đoán từ bị ẩn trong câu	26
Hình 2.19: Minh họa ngữ cảnh trái và phải trong BERT	26
Hình 2.20: Minh họa chiến thuật masking	27
Hình 2.21: Minh họa quá trình huấn luyện mô hình BERT..... cho tác vụ dự đoán từ bị ẩn trong câu	28
Hình 2.22: Minh họa mô hình cho tác vụ dự đoán câu tiếp theo	29
Hình 2.23: Minh họa đầu vào của BERT cho tác vụ dự đoán câu tiếp theo	29

Hình 2.24: Minh họa quá trình huấn luyện mô hình BERT.....	30
cho tác vụ dự đoán câu tiếp theo	30
Hình 2.25: Minh họa token [CLS] trong BERT	31
Hình 3.1: Minh họa bài toán phân loại khiếu nại của khách hàng	35
Hình 3.2: Minh họa quá trình tinh chỉnh mô hình BERT	36
cho bài toán phân loại khiếu nại của khách hàng.....	36
Hình 3.3: Minh họa kiến trúc của Linear and Softmax Layer cho bài toán dự đoán điểm đánh giá từ nhận xét của người dùng	37
Hình 3.4: Minh họa quá trình tinh chỉnh mô hình BERT cho bài toán dự đoán điểm đánh giá từ nhận xét của người dùng	38
Hình 3.5: Quy trình triển khai mô hình cài tiền dựa trên chiến lược tổng hợp profile và sử dụng đánh giá của BERT	40
Hình 4.1: Kết quả F-score trên bộ dữ liệu Digital Music	44
Hình 4.2: Kết quả F-score trên bộ dữ liệu Musical Instrument	45
Hình A.1: Mạng nơ-ron nhân tạo và mạng nơ-ron thực tế	55
Hình A.2: Các lớp trong mạng nơ-ron	56
Hình A.3: Ví dụ tập ví dụ được sử dụng cho huấn luyện mạng nơ-ron	57
Hình A.4: Minh họa quá trình huấn luyện mạng nơ-ron.....	58
Hình A.5: Ví dụ kết quả dự đoán thực tế của mô hình	59
sau khi kết thúc quá trình huấn luyện	59
Hình A.6: Minh họa quy trình tính toán đầu ra của một nơ-ron	60
Hình A.7: Kiến trúc mạng nơ-ron để thực hiện toán tử XOR.....	62
Hình A.8: Đồ thị hàm sigmoid	62
Hình A.9: Mô tả quá trình lan truyền	63
Hình A.10: Kết quả của quá trình lan truyền đầu tiên	64
Hình A.11: Đồ thị biểu diễn mối quan hệ	65
giữa hàm mất mát và các tham số trong mạng	65
Hình A.12: Sự thay đổi của giá trị loss sau 100,000 lần lặp	68
Hình A.13: Kết quả dự đoán của mô hình đã huấn luyện	68
Hình A.14: Kiến trúc của mạng nơ-ron cần áp dụng ma trận hoá, véc-tơ hoá	69
Hình A.15: Các tham số trong mạng nơ-ron được biểu diễn	70
bằng các ma trận và véc-tơ.....	70

Hình B.1: Kiến trúc Transformer (Encoders)	71
Hình B.2: Minh họa quá trình chuyển câu đầu vào sang tập các token/input ID	72
Hình B.3: Minh họa quá trình chuyển từng token/input ID sang véc-tơ nhúng (embedding vector)	73
Hình B.4: Các véc-to nhúng tương ứng với từng token đầu vào	73
Hình B.5: Minh họa quá trình thêm vị trí được mã hoá..... vào véc-tơ nhúng đầu vào	74
Hình B.6: Quá trình tính giá trị mã hoá của từng vị trí trong câu đầu vào	75
Hình B.7: Minh họa ma trận nhúng của câu đầu vào.....	76
Hình B.8: Ma trận Q, K, V tương ứng với câu đầu vào.....	76
Hình B.9: Tổng điểm trên một dòng trong trận luôn bằng 1 nhờ sử dụng hàm Softmax	77
Hình B.10: Minh họa ngữ cảnh trái và phải trong câu.....	78
Hình B.11: Minh họa causal masking trong quá trình tính điểm chú ý	78
Hình B.12: Ma trận chú ý đầu ra của cơ chế tự chú ý.....	79
Hình B.13: Minh họa lớp chuẩn hoá trong khối mã hoá.....	80
Hình B.14: Ví dụ sử dụng hàm Softmax trong bài toán nhận diện động vật.....	82

Bảng thuật ngữ

Thuật ngữ tiếng Anh	Từ được sử dụng trong khóa luận
Latent Factor model	Mô hình nhân tố ẩn
Rating	Đánh giá
Text review	Nhận xét, Văn bản đánh giá
Item	Mục, sản phẩm
Group recommender system	Hệ thống gợi ý nhóm
Language model	Mô hình ngôn ngữ
Neural network	Mạng nơ-ron
Input – output	Đầu vào – đầu ra
Backpropagation	Lan truyền ngược
Loss Function	Hàm mất mát
Fine-tuning	Tinh chỉnh
Encoder	Bộ mã hóa
Feed-forward network	Mạng nơ-ron truyền thẳng
Attention Head	Đầu chú ý
Layer	Lớp / tầng
Embedding	nhúng
Context	Ngữ cảnh

Pre-trained	Huấn luyện trước
Masked	Ẩn
Block	Khối
Linear	Tuyến tính
Weight	Trọng số
Min-max normalization	Chuẩn hóa tối thiểu – tối đa
Bias	Độ thiên vị, thiên kiến
Mapping	Ánh xạ
Epoch	Giai đoạn
Self-attention	Tự chú ý
Learning rate	Tốc độ học
Training	Huấn luyện
Error rate	Độ lỗi
Supervised learning	Học giám sát
Actual output	Đầu ra thực tế
Expected output	Đầu ra mong đợi
Activation Function	Hàm kích hoạt
Vector	Véc-tơ

Tóm tắt

Trong bối cảnh công nghệ ngày càng phát triển và người tiêu dùng ngày càng đòi hỏi cao hơn, các doanh nghiệp đã nhận ra tầm quan trọng của việc cung cấp những trải nghiệm cá nhân hóa cho khách hàng. Hệ thống gợi ý đã trở thành công cụ quan trọng trong việc giúp người dùng tìm kiếm và chọn lọc thông tin, sản phẩm phù hợp. Tuy nhiên, sự phát triển không ngừng của nhu cầu và xu hướng tiêu dùng đã thúc đẩy các doanh nghiệp chuyển hướng từ việc gợi ý cho từng cá nhân sang việc gợi ý cho nhóm người dùng.

Tuy nhiên, các hệ thống gợi ý cho nhóm không chỉ nên dựa vào điểm đánh giá từ người dùng. Thay vào đó, việc sử dụng thêm các thông tin như nhận xét của người dùng cung cấp một cái nhìn chi tiết hơn về trải nghiệm của người dùng, giúp hệ thống hiểu rõ hơn về các yếu tố cụ thể mà người dùng quan tâm. Từ đó có thể nhận diện các xu hướng và sở thích tiềm ẩn từ nhận xét, giúp đề xuất những sản phẩm, dịch vụ phù hợp hơn với từng nhóm người dùng.

Trong đề tài khóa luận này nhóm chúng em sẽ đề xuất một phương pháp cải tiến dành cho mô hình nhân tố ẩn trong bài toán gợi ý nhóm thông qua việc tích hợp thêm nhận xét của người dùng vào trong hệ thống gợi ý nhóm. Sau đó tiến hành cài đặt, so sánh tính hiệu quả giữa phương pháp trước khi cải tiến và sau khi cải tiến từ các số liệu thu thập được trong quá trình thực nghiệm và đưa ra kết luận.

Chương 1

Giới thiệu

1.1 Đặt vấn đề

Hiện nay, với sự phát triển vượt bậc của công nghệ, con người dễ dàng tiếp cận được với một lượng lớn thông tin đến từ nhiều nguồn khác nhau dẫn đến vấn đề quá tải thông tin. Người dùng ngày càng khắt khe hơn và đòi hỏi cao hơn khi lựa chọn sản phẩm. Điều này thúc đẩy các doanh nghiệp phát triển và nâng cấp hệ thống của mình giúp người dùng dễ dàng tìm được sản phẩm phù hợp, giúp nâng cao trải nghiệm người dùng khi sử dụng sản phẩm.

Nhiều doanh nghiệp lớn hiện nay không chỉ dừng lại ở việc phát triển và tích hợp các mô hình gợi ý cho từng cá nhân mà còn hướng tới xây dựng các hệ thống tư vấn dành cho nhóm người dùng, vì xu hướng mua hàng theo nhóm ngày càng tăng cao. Tuy nhiên, việc tư vấn nhóm đối diện với một số thách thức, bao gồm:

- **Sự đa dạng của nhóm:** Mỗi người dùng trong nhóm có thể có sở thích, yêu cầu và hành vi mua hàng khác nhau. Do đó, việc tìm ra các sản phẩm phù hợp để gợi ý cho một nhóm với nhiều thành viên và đa dạng sở thích như vậy đòi hỏi các phương pháp và mô hình gợi ý phức tạp hơn.
- **Sự tương tác phức tạp:** Sự tương tác giữa các thành viên trong nhóm, cũng như giữa nhóm và sản phẩm có thể phức tạp và khó dự đoán. Cần có các phương pháp hiệu quả để mô hình hóa và dự đoán các mẫu tương tác này.

Nhóm thực hiện đề tài này nhằm mục tiêu giải quyết các thách thức trên bằng cách đề xuất một phương pháp cải tiến hệ thống tư vấn nhóm trở nên chính xác và đáng tin cậy hơn khi đối mặt với **vấn đề dữ liệu thưa thớt**, giúp các doanh nghiệp cung cấp các gợi ý sản phẩm phù hợp nhất cho các nhóm người dùng. Điều này có thể giúp

tăng cường trải nghiệm người dùng, nâng cao lợi nhuận và củng cố vị thế cạnh tranh của các doanh nghiệp.

1.2 Mục tiêu

Trong thời đại phát triển của công nghệ, việc đáp ứng đầy đủ và chính xác nhu cầu của người dùng có thể mang lại lợi ích to lớn cho các doanh nghiệp. Một trong những vấn đề được quan tâm chính là cải tiến các mô hình tư vấn sản phẩm cho không chỉ một người dùng mà còn có thể tư vấn cho một nhóm người dùng.

Hiện nay, các mô hình tư vấn dựa trên lọc cộng tác gồm lọc dựa trên láng giềng và lọc dựa trên mô hình sử dụng ma trận đánh giá (rating) của người dùng trên sản phẩm cho kết quả khá tốt. Tuy nhiên, đặc trưng của ma trận đánh giá **khá thưa và độ chính xác chưa cao**, nên phương pháp dựa trên mô hình, cụ thể là **lọc dựa trên nhân tố ẩn** (latent factor) [1] cho kết quả tốt hơn đối với trường hợp dữ liệu đánh giá thưa. **Mô hình nhân tố ẩn vẫn chưa giải quyết được vấn đề độ chính xác của ma trận đánh giá, đặc biệt là trên nhóm.** Nhóm đã đề xuất một phương pháp mới để cải thiện hệ thống tư vấn nhóm bằng cách **tích hợp thông tin đánh giá (review)** sau khi người dùng trải nghiệm sản phẩm với ma trận rating để tăng cường tính chính xác của ma trận rating khi dự đoán trong mô hình nhân tố ẩn khi được áp dụng cho các hệ tư vấn trên nhóm.

Đề tài có thể mang lại nhiều ảnh hưởng tích cực đối với cả vấn đề cụ thể về tư vấn nhóm và lĩnh vực nghiên cứu hệ thống tư vấn trong tổng thể, như sau:

- Đối với vấn đề cụ thể, một hệ thống tư vấn nhóm hiệu quả có thể giúp tăng cường trải nghiệm mua sắm của người dùng, tăng doanh số bán hàng cho các doanh nghiệp và củng cố vị thế cạnh tranh trên thị trường.
- Trong lĩnh vực nghiên cứu, phương pháp cải tiến có thể cung cấp một bước tiến quan trọng trong việc tăng cường hiệu suất của các hệ thống tư vấn thông qua việc áp dụng các kỹ thuật tiên tiến và phức tạp hơn trong mạng neural và học sâu. Điều này có thể mở ra cánh cửa cho nhiều nghiên cứu và ứng dụng các kiến thức về trí tuệ nhân tạo trong hệ thống tư vấn.

1.3 Phạm vi đề tài

Đề tài tập trung vào phương pháp cải tiến mô hình tư vấn nhóm bằng phương pháp tích hợp rating dự đoán từ mô hình ngôn ngữ lớn đã được huấn luyện để dự đoán rating sản phẩm từ bình luận dạng văn bản của người dùng (text review) vào trong mô hình nhân tố ẩn dành cho nhóm để dự đoán, đề xuất sản phẩm cho nhóm người dùng

1.4 Cách tiếp cận dự kiến

Hướng tiếp cận dự kiến của đề tài chính là sử dụng **mô hình ngôn ngữ lớn** để phát sinh ma trận user-item rating từ text-review của người dùng, kết hợp với phương pháp **tổng hợp profile nhóm trong mô hình nhân tố ẩn** trong bài báo [2]. Sau đó tiến hành thực nghiệm để đánh giá kết quả gợi ý của mô hình. Về thông tin và nội dung chi tiết sẽ được trình bày ở các chương tiếp theo.

1.5 Cấu trúc cuộn luận

Chương 1: “Giới thiệu” giới thiệu các thông tin cơ bản về khóa luận

Chương 2: “Các nghiên cứu liên quan” chương này cho biết các thông tin về một số nghiên cứu liên quan trong đề tài

Chương 3: “Hướng tiếp cận” trình bày hướng tiếp cận của đề tài

Chương 4: “Thực nghiệm” trình bày quá trình thực nghiệm của đề tài và nhận xét kết quả thực nghiệm

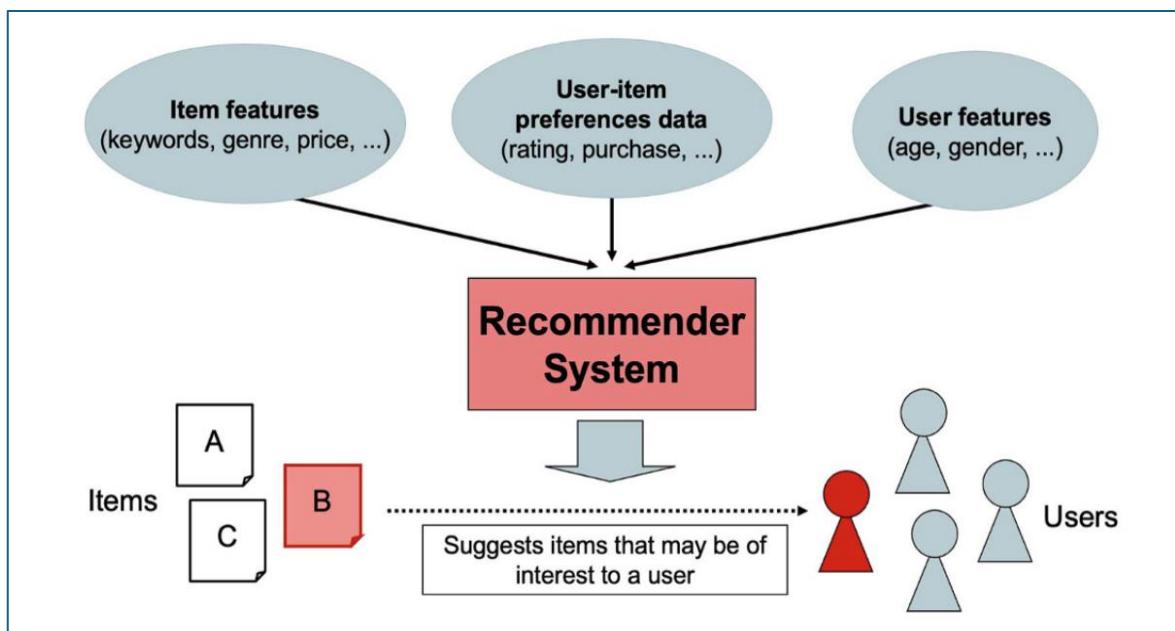
Chương 5: “Kết luận” kết luận sau quá trình thực nghiệm

Chương 2

Các nghiên cứu liên quan

2.1 Phát biểu về bài toán gợi ý

Bài toán gợi ý là bài toán liên quan đến việc mô hình hóa sở thích của người dùng dựa trên profile của họ. Hệ thống sẽ dựa vào trên các hoạt động tương tác của người dùng trên hệ thống như các từ khóa đã tìm kiếm, các sản phẩm đã mua, các đánh giá của sản phẩm đã trải nghiệm,... để tiến hành tổng hợp thành profile của người dùng, điều này giúp kết quả gợi ý sản phẩm cho họ phù hợp hơn như Hình 2.1.



Hình 2.1: Hệ thống gợi ý [4]

Một hình thức thích hợp để biểu diễn profile của người dùng trong bài toán gợi ý là sử dụng ma trận đánh giá user-item $R = [r_{u,i}]$ [3] trong đó $u=1, \dots, m$ là đại diện cho người dùng $i=1, \dots, n$ là đại diện cho sản phẩm trong hệ thống. Mỗi phần tử trong ma

trận đánh giá user-item thể hiện đánh giá (rating) sản phẩm sau khi trải nghiệm. Đánh giá này thường biểu diễn theo thang điểm 1-5. Trong đó:

- Hoàn toàn không đồng ý / hài lòng
- Không đồng ý / hài lòng
- Trung lập / bình thường
- Đồng ý / hài lòng
- Hoàn toàn đồng ý / hài lòng.

So với các bài toán khác, đặc trưng của bài toán gợi ý nằm ở việc **ma trận rating** của người dùng trên các sản phẩm thường thưa thớt, điều này có nghĩa là có một lượng lớn sản phẩm người dùng chưa trải nghiệm thường được biểu diễn $r_{u,i} = *$ trong ma trận

	i_1	i_2	i_3	i_4
u_1	*	1	5	
u_2	2	3	*	*
u_3	4	*	*	4
u_4	2	2	5	2

Hình 2.2: Ví dụ ma trận đánh giá user-item

Những dữ liệu bị thiếu này sẽ được **dự đoán dựa trên mô hình đã được huấn luyện** trên **tập dữ liệu thu thập được trước đó** (các thành phần dữ liệu không khuyết trong ma trận). Từ đó hệ thống tư vấn có thể dự đoán rating và đề xuất các sản phẩm phù hợp với người dùng.

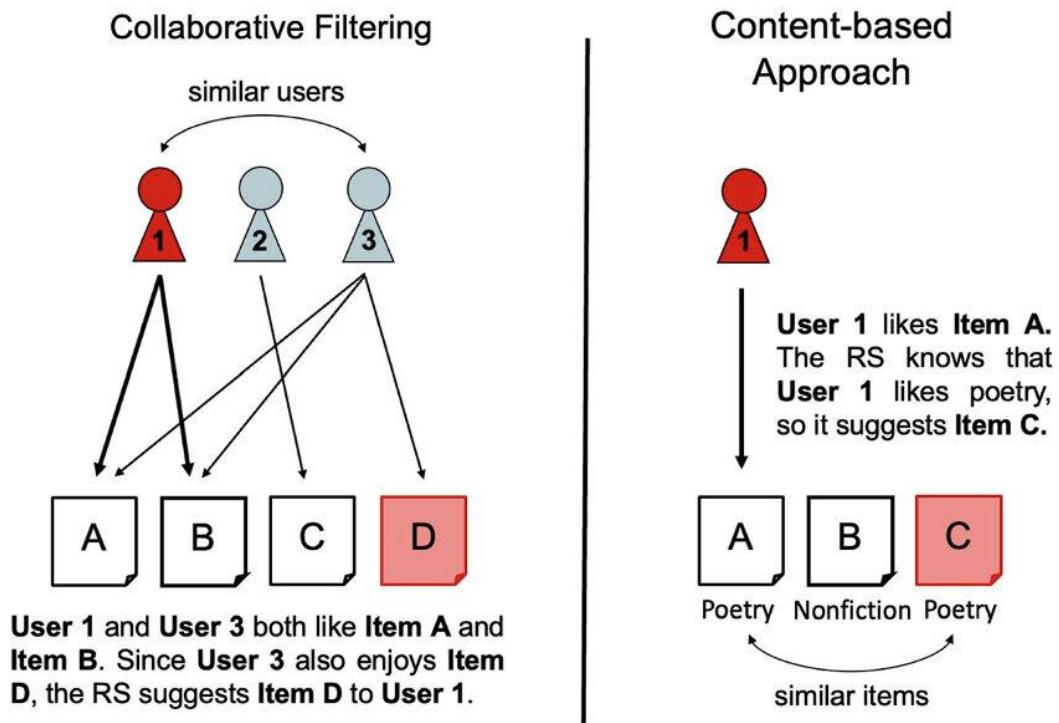
2.2 Các mô hình gợi ý cho người dùng cá nhân

Bài toán gợi ý sản phẩm còn được biết đến như là bài toán điền các giá trị khuyết trong ma trận đánh giá user-item, ma trận đánh giá sản phẩm của người dùng cho biết sở thích của người dùng trên sản phẩm mà họ đã trải nghiệm. Theo khảo sát của nhóm, hiện tại có hai hướng cơ bản để giải quyết bài toán này là **lọc dựa trên nội dung**

(content-based filtering) - là phương pháp **gợi ý** dựa trên **đặc điểm** của các sản phẩm và sở thích của người dùng để đưa ra kết quả phù hợp và **lọc cộng tác** (collaborative filtering) - là phương pháp **đề xuất** các item dựa trên hành vi và sở thích của **nhiều người dùng** tương đồng. Mặc dù hai hướng này cho thấy sự khác biệt nhau ở nhiều khía cạnh nhưng chúng hoàn toàn **có thể** kết hợp lại với nhau để **tạo ra** **những thuật toán** **tốt hơn**, được gọi là các **thuật toán lai** (hybrid systems) [4].

Cụ thể, các thuật toán **lọc theo nội dung** hoạt động dựa trên việc **phân tích** và **đánh giá** **nội dung** của các sản phẩm (items) mà **người dùng** **đã tương tác** hoặc **đánh giá cao** trong quá khứ, từ đó **đề xuất** các mục **tương tự** trong tương lai. Mô hình lọc theo nội dung **sử dụng** các **thuộc tính** của sản phẩm (như từ khóa, thể loại, tác giả, v.v.) để **xác định** **mức độ** **tương đồng** **giữa** **các sản phẩm**. Khi một người dùng thể hiện sự yêu thích đối với một số sản phẩm cụ thể, hệ thống sẽ **phân tích** các thuộc tính của những sản phẩm đó để **xây dựng** **hồ sơ** **sở thích** của người dùng (user profile). Sau đó, **dựa** **trên** **hồ sơ** **này**, hệ thống sẽ **tìm kiếm** và **đề xuất** các sản phẩm mới **có** **các** **thuộc tính** **tương đồng** **với** **các** **sản** **phẩm** **mà** **người** **dùng** **đã** **yêu** **thích** **trước** **đó**.

Ưu điểm của Content-Based Filtering bao gồm khả năng **đưa** **ra** **các** **đề** **xuất** **cá nhân hóa** **dựa** **trên** **sở** **thích** **cụ** **thể** **của** **từng** **người** **dùng**, **không** **phụ** **thuộc** **vào** **ý** **kiến** **hay** **hành** **vi** **của** **người** **dùng** **khác**. Điều này giúp mô hình **hoạt** **động** **hiệu** **quả** **ngay** **cả** **khi** **có** **ít** **dữ** **liệu** **người** **dùng** (cold start problem). Tuy nhiên, nhược điểm của mô hình này là dễ bị **giới** **hạn** **bởi** **những** **sở** **thích** **hiện** **tại** **của** **người** **dùng** **và** **khó** **khăn** **trong** **việc** **khám** **phá** **các** **mục** **mới** **ngoài** **sở** **thích** **đã** **biết** [5].



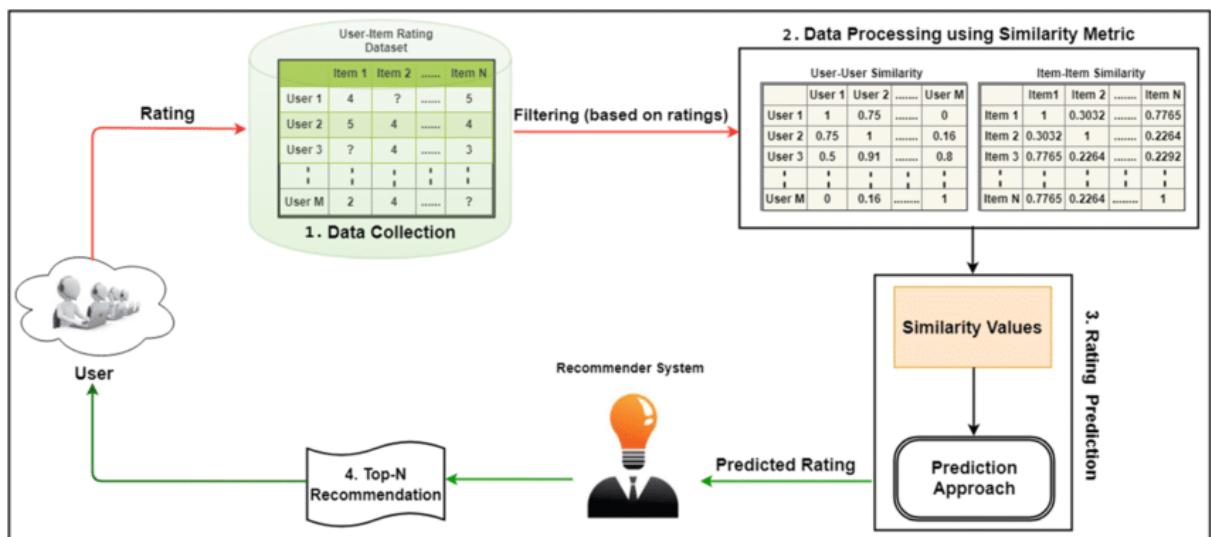
Hình 2.3: Minh họa hướng lọc cộng tác và hướng lọc dựa trên nội dung [6]

Những nhược điểm trên của các thuật toán lọc nội dung được khắc phục bằng các thuật toán lọc cộng tác. Thuật toán **lọc cộng tác** sử dụng sức mạnh cộng tác của các đánh giá được cung cấp bởi nhiều người dùng để đưa ra các gợi ý, vì các đánh giá đã quan sát thường biểu diễn sự tương quan cao giữa người dùng và giữa các sản phẩm [4]. Sự tương quan này được nhận diện bằng thuật toán nền tảng để từ đó sử dụng làm căn cứ gợi ý sản phẩm cho người dùng. Chính vì vậy nên dữ liệu đầu vào của thuật toán lọc cộng tác thường là ma trận đánh giá user-item. Các thuật toán lọc cộng tác được chia thành **dựa trên bộ nhớ** (memory-based) và **dựa trên mô hình** (model-based).

Các thuật toán dựa trên bộ nhớ (memory-based) còn được gọi là các thuật toán lọc cộng tác dựa trên lân cận (neighborhood-based collaborative filtering). Đây là một trong những thuật toán lọc cộng tác đầu tiên, trong đó các **đánh giá của người dùng-sản phẩm** được dự đoán dựa trên tập láng giềng của chúng. Các tập láng giềng này có thể được xác định theo một trong hai cách:

- Lọc cộng tác dựa trên người dùng (**User-based collaborative filtering**): Trong trường hợp này, các đánh giá được cung cấp bởi những người dùng có cùng sở thích với người dùng mục tiêu A được sử dụng để đưa ra các đề xuất cho A. Ý tưởng cơ bản là xác định những người dùng có sự tương đồng với người dùng mục tiêu A, và đề xuất các đánh giá cho những mục chưa được đánh giá của A bằng cách tính trung bình có trọng số các đánh giá của nhóm người dùng này.
- Lọc cộng tác dựa trên sản phẩm (**Item-based collaborative filtering**): Để đưa ra dự đoán đánh giá cho sản phẩm B cho người dùng A, bước đầu tiên là xác định một tập hợp S gồm các sản phẩm tương tự nhất với sản phẩm B. Các đánh giá trong tập hợp S, được đánh giá bởi người dùng A, được sử dụng để dự đoán liệu người dùng A có thích sản phẩm B hay không. Phương pháp này dựa trên giả định rằng nếu người dùng A đã đánh giá cao các sản phẩm tương tự trong quá khứ, thì có khả năng cao là họ cũng sẽ thích sản phẩm B.

Để xác định mức độ tương đồng giữa người dùng hoặc giữa các sản phẩm, các chỉ số như hệ số tương quan Pearson [8] hoặc khoảng cách cosine [7] thường được sử dụng.



Hình 2.4: Minh họa cách hoạt động của hệ thống lọc cộng tác dựa trên bộ nhớ (memory-based Collaborative Filtering) [9]

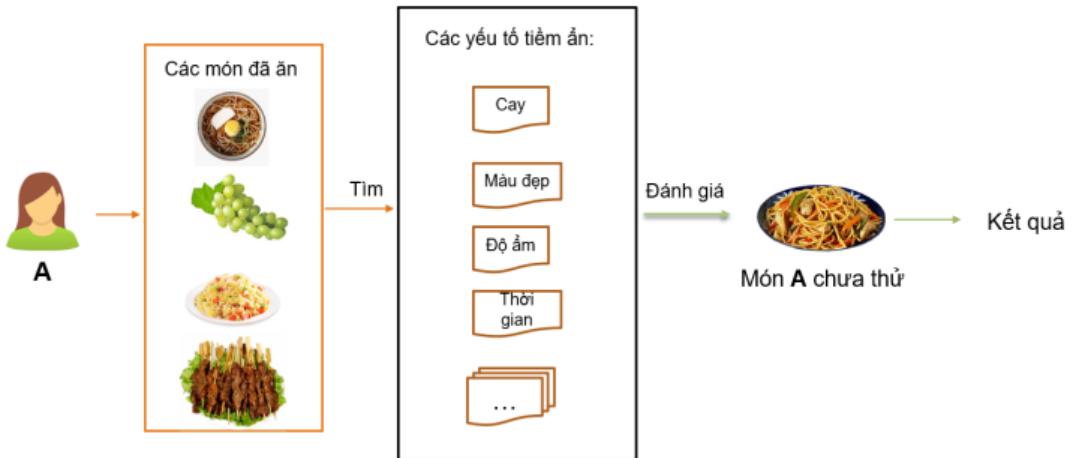
Ưu điểm của các thuật toán dựa trên bộ nhớ đó là chúng đơn giản để triển khai và các đề xuất kết quả thường dễ giải thích. Tuy nhiên, các thuật toán lọc dựa trên bộ nhớ không hoạt động tốt với các ma trận đánh giá thưa thớt [4] và hiệu suất kém với nguồn dữ liệu lớn [10].

Trong các thuật toán dựa trên mô hình, các thuật toán học máy và khai thác dữ liệu được sử dụng trong ngữ cảnh là các mô hình dự đoán. Trong các mô hình này, sẽ có các tham số cần được tối ưu hóa để cải thiện hiệu suất của mô hình. Quá trình học của các tham số này diễn ra trong một khung tối ưu hóa, nhằm tìm ra các giá trị tối ưu cho các tham số dựa trên dữ liệu hiện có.

Một số ví dụ về các phương pháp dựa trên mô hình bao gồm cây quyết định (decision trees) [12], phương pháp Bayes (Bayesian methods) [11] và mô hình nhân tố ẩn (latent factor models) [13]. Nhiều phương pháp trong số này, đặc biệt là mô hình nhân tố ẩn, có thể đạt được mức độ bao phủ cao ngay cả khi đối mặt với các ma trận đánh giá thưa thớt [4]. Điều này có nghĩa là các mô hình này có khả năng cung cấp các dự đoán chính xác và hiệu quả ngay cả khi dữ liệu đánh giá không đầy đủ hoặc không phong phú.

Mô hình nhân tố ẩn (Latent Factor Model) là một kỹ thuật phổ biến và hiệu quả trong lĩnh vực học máy, đặc biệt là trong các hệ thống gợi ý. Mô hình này hoạt động bằng cách biểu diễn tập người dùng và sản phẩm bằng những nhân tố ẩn (Latent Factors) không thể quan sát trực tiếp được nhưng có tác động mạnh mẽ đến dữ liệu. Các nhân tố này có thể đại diện cho những đặc điểm hoặc khía cạnh sâu bên trong dữ liệu, chẳng hạn như sở thích của người dùng hoặc đặc trưng của các sản phẩm. Hệ thống sau đó sử dụng các nhân tố ẩn đã học được để đưa ra gợi ý cho người dùng.

▪ Mô hình nhân tố ẩn (latent factor model)



Hình 2.5: Ý tưởng của mô hình nhân tố ẩn [3]

Các nhân tố ẩn có thể được học bằng cách xấp xỉ của ma trận đánh giá user-item R như sau:

$$R \approx H * V^T \quad (1)$$

Trong đó:

- $R (m \times n)$ là ma trận chứa đánh giá của m người dùng trên tập n sản phẩm
- $H (m \times s)$ là ma trận user-factor biểu diễn đánh giá của người dùng trong không gian s nhân tố ẩn.
- $V (n \times s)$ là ma trận item-factor chứa biểu diễn của các item trong không gian s nhân tố ẩn

Sau khi học được các ma trận H , V mô hình nhân tố ẩn sẽ dự đoán một đánh giá khuyết của một người dùng u đối với một sản phẩm i ($r_{u,i} = *$) bằng cách thực hiện **xấp xỉ tích vô hướng** của véc-tơ user-factor **ứng với** người dùng u trong ma trận H (h_u) và **véc-tơ item-factor tương ứng** với item i trong ma trận V (v_i) như sau:

$$\hat{r}_{u,i} \approx o_u + p_i + \mu + H \cdot V_{row}^T \approx o_u + p_i + \mu + \sum_{z=1}^s h_{u,z} * v_{i,z} \quad (2)$$

Trong đó:

- $r_{u,i}$ là đánh giá mà mô hình dự đoán cho một đánh giá khuyết.

- o_u là đại diện cho bias của người dùng u khi thực hiện đánh giá các sản phẩm (người dùng khó tính sẽ có xu hướng đánh giá sản phẩm thấp hơn so với mặt bằng chung và người dùng dễ tính sẽ thường đánh giá sản phẩm cao hơn mặt bằng chung)
- p_i là bias trong đánh giá sản phẩm i từ những người dùng trong hệ thống (với những sản phẩm đang được ưa chuộng sẽ được đánh giá cao hơn và các sản phẩm dù tốt nhưng lỗi thời có thể bị đánh giá thấp).
- μ là giá trị trung bình của toàn bộ đánh giá từ người dùng quan sát được trong hệ thống.

The diagram illustrates the decomposition of a user-item rating matrix H into its components. On the left, the original matrix H is shown as a 7x7 grid of user u_1 to u_7 and item i_1 to i_7 . An orange arrow points from H to the right side, which shows the factorization process:

- Ma trận user-factor V :** A 7x3 matrix where rows represent users and columns represent items f_1, \dots, f_x . The matrix contains numerical values for each rating entry.
- Ma trận item-factor U :** A 3x7 matrix where columns represent items and rows represent users u_1, \dots, u_7 . The matrix contains numerical values for each rating entry.
- user bias o_u :** A column vector of size 7 containing user-specific bias values o_{u_1}, \dots, o_{u_7} .
- item bias p_i :** A column vector of size 7 containing item-specific bias values p_{i_1}, \dots, p_{i_7} .

Below the matrices, the formula for predicting a rating $\hat{r}_{u,i}$ is given:

$$\hat{r}_{u,i} \approx o_u + p_i + \mu + \mathbf{h}_{\text{row}_u} * \mathbf{v}_{\text{row}_i}^T \approx o_u + p_i + \mu + \sum_{z=1}^s \mathbf{h}_{u,z} * \mathbf{v}_{i,z}$$

Hình 2.6: Xây dựng ma trận mô hình nhân tố ẩn [3]

2.3 Các chiến lược gợi ý cho nhóm người dùng

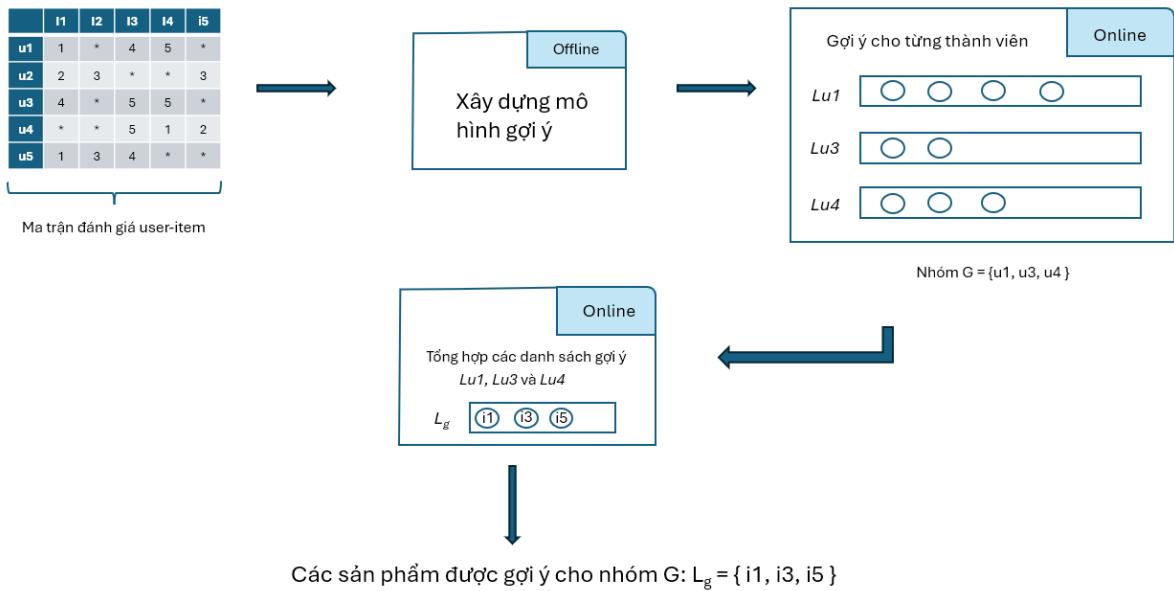
Hiện nay, con người có xu hướng gắn kết hơn trong nhiều hoạt động vui chơi, giải trí, v.v. Kéo theo đó là nhu cầu gợi ý các sản phẩm có thể giành cho nhiều người (nhóm) cùng trải nghiệm ví dụ như xem phim, du lịch, ăn uống, v.v. Từ đó nhu cầu về hệ thống gợi ý nhóm được phát triển.

So với các hệ thống gợi ý người dùng cá nhân, các mô hình hệ thống gợi ý nhóm phải đổi mới với nhiều vấn đề hơn, khi chúng cần phải cân bằng sở thích đa dạng của tất cả thành viên trong nhóm để đảm bảo các kết quả gợi ý có thể làm hài lòng nhiều

thành viên nhất có thẻ. Trong bài toán gợi ý nhóm, việc có 2 chiến lược phổ biến là **tổng hợp gợi ý** và **tổng hợp profile** [14]. Đối với cả 2 phương pháp này việc **lựa chọn chiến lược xây dựng mô hình gợi ý** là rất quan trọng có thể ảnh hưởng trực tiếp đến kết quả gợi ý.

2.3.1 Chiến lược tổng hợp gợi ý

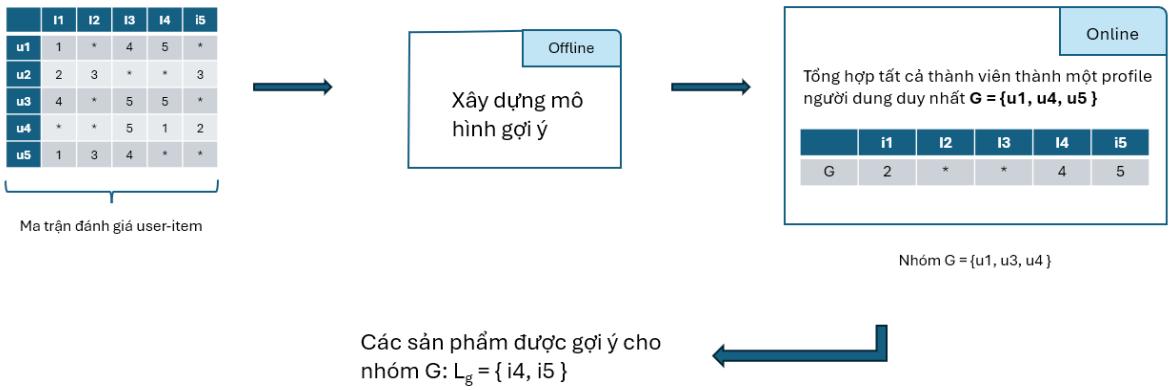
Theo đó, chiến lược **tổng hợp gợi ý** sẽ sử dụng **danh sách các gợi ý** cho từng thành viên trong nhóm để thực hiện **tổng hợp** các danh sách này thành một danh sách các sản phẩm **đề xuất** cho cả nhóm [14].



Hình 2.7: Quy trình tư vấn theo chiến lược tổng hợp gợi ý

2.3.2 Chiến lược tổng hợp profile

Khác với chiến lược **tổng hợp gợi ý**, chiến lược **tổng hợp profile** thực hiện **tổng hợp sở thích** của các thành viên trong nhóm (các đánh giá quan sát được của các thành viên) để **tạo ra một profile người dùng ảo** đại diện cho nhóm. Các **đề xuất** cho người dùng ảo này được coi là **các gợi ý** cho chính nhóm người dùng [14].



Hình 2.8: Quy trình tư vấn theo chiến lược tổng hợp profile

2.4 Mô hình nhân tố ẩn cho bài toán gợi ý nhóm theo chiến lược tổng hợp profile

Mô hình nhân tố ẩn đã mang lại kết quả đáng kể trong các hệ thống tư vấn dành cho người dùng đơn lẻ [15]. Ngoài ra, nhiều nghiên cứu trước đây về gợi ý nhóm cũng chỉ ra rằng việc **tổng hợp profile** thường mang lại hiệu suất tốt hơn so với việc **tổng hợp gợi ý** [16][17]. Do đó **mô hình nhân tố ẩn** dựa trên chiến lược **tổng hợp profile** là một giải pháp hiệu quả cho bài toán gợi ý nhóm [2].

Một phương pháp phổ biến để áp dụng mô hình nhân tố ẩn trong chiến lược **tổng hợp profile** đó là phát sinh người dùng ảo (virtual user) trong hệ thống tư vấn nhóm [18] bằng cách sử dụng **Stochastic Gradient Descent (SGD)** để phân rã trực tiếp ma trận đánh giá người dùng, cụ thể quy trình của phương pháp diễn ra như sau:

Bước 1[Giai đoạn Offline]: Sử dụng Stochastic Gradient Descent (SGD) để phân rã ma trận đánh giá người dùng user-item R thành 2 ma trận nhân tố ẩn H, V^T . Quá trình tối ưu hàm mục tiêu liên quan đến độ lỗi sẽ **xấp xỉ giữa R và $H * V^T$** . Bởi vì R là ma trận khuyết nên **hàm mục tiêu** chỉ có thể được **định nghĩa** trên các đánh giá **không khuyết** quan sát được từ người dùng. Cụ thể là **bình phương chênh lệch giữa** các đánh giá không khuyết ($r_{u,i} \neq *$) và các đánh giá được dự đoán:

$$\min_{H, V, o_u, p_i} \frac{1}{2} * \sum_{\{(u,i) \mid u = 1 \dots m \wedge i = 1 \dots n \wedge r_{u,i} \neq *\}} (r_{u,i} - \hat{r}_{u,i})^2 = \quad (3)$$

$$\min_{H, V, o_u, p_i} \frac{1}{2} * \sum_{\{(u,i) \mid u = 1 \dots m \wedge i = 1 \dots n \wedge r_{u,i} \neq *\}} (r_{u,i} - o_u - p_i - \mu - h_{v,:} * v_{i,:}^T)^2$$

Tuy nhiên khi số lượng đánh giá quan sát quá ít, quá trình xấp xỉ có thể rơi vào trường hợp over-fitting. Regulation [19] là một cách giải quyết phổ biến cho vấn đề này. Đó là thêm các thành phần $\|\mathbf{H}\|^2, \|\mathbf{V}\|^2$ vào hàm mục tiêu để ngăn các giá trị học quá lớn. Do đó hàm mục tiêu được hiệu chỉnh thành:

$$\begin{aligned} \min_{H, V, o_u, p_i} & \frac{1}{2} * \sum_{\{(u,i) \mid u = 1 \dots m \wedge i = 1 \dots n \wedge r_{u,i} \neq *\}} (r_{u,i} - o_u - p_i - \mu - \mathbf{H}_{u,:} * \mathbf{V}_{i,:}^T)^2 + \\ & \frac{\lambda}{2} \|\mathbf{H}\|^2 + \frac{\lambda}{2} \|\mathbf{V}\|^2 + \frac{\lambda}{2} \sum_{u=1}^m o_u^2 + \frac{\lambda}{2} \sum_{i=1}^n p_i^2 \end{aligned} \quad (4)$$

Với λ là trọng số của các thành phần regularization trong hàm mục tiêu. Sau khi hình thành các ma trận \mathbf{H}, \mathbf{V} và tham số bias người dùng o_u và bias sản phẩm p_i , ta sẽ thực hiện ước lượng các đánh giá khuyết của người dùng theo công thức:

$$\hat{r}_{u,i} \approx o_u + p_i + \mu + \mathbf{H}_{u,:} * \mathbf{V}_{i,:}^T \quad (5)$$

Bước 2 [Giai đoạn Online]: Đối với nhóm người dùng \mathbf{G} gồm \mathbf{m} thành viên $\{u_i \mid i = 1 \dots m\}$, việc tổng hợp hồ sơ của các thành viên trong nhóm trong không gian \mathbf{n} sản phẩm được kí hiệu bằng một véc-tơ group-item $\mathbf{V}_g = [r_{g,1}, r_{g,2}, \dots, r_{g,n}]$ được thực hiện để tạo người dùng ảo \mathbf{v} . Trong toàn bộ sản phẩm, các sản phẩm mà chưa được bất kì thành viên nào trong nhóm trải nghiệm sẽ được coi là những đe xuất tiềm năng cho nhóm đó. Do đó, một đánh giá $r_{g,i}$ với $i = 1 \dots m$ trong véc-tơ profile người dùng ảo \mathbf{v} sẽ bị khuyết ($r_{g,i} = *$) nếu không có bất kì thành viên nào trong nhóm từng trải nghiệm và đánh giá sản phẩm ($r_{u_1,i} = r_{u_2,i} = \dots = r_{u_m,i} = *$). Ngược lại, nếu có ít nhất một thành viên đánh giá cho sản phẩm i , các đánh giá khuyết của các thành viên còn lại trong nhóm sẽ được ước lượng bằng mô hình nhân tố ẩn theo Eq.(2):

$$s_{u,i} = \begin{cases} r_{u,i} & \text{if } r_{u,i} \neq * \\ \hat{r}_{u,i} = o_u + p_i + \mu + \mathbf{H}_{u,:} * \mathbf{V}_{i,:}^T & \text{if } r_{u,i} = * \end{cases} \quad (6)$$

Sau đó, các đánh giá của người dùng ảo đối với từng sản phẩm sẽ được tính bằng trung bình tất cả đánh giá của các thành viên trong nhóm như sau :

$$r_{v,i} = \frac{\sum_{u \in G} s_{u,i}}{\sum_{u \in G} 1} \quad (7)$$

Bước 3 [Giai đoạn Online]: Uớc lượng véc-tơ group-factor \mathbf{h}_g đại diện cho profile nhóm trong không gian nhân tố ẩn tương tự như ma trận user-factor, với mỗi dòng đại diện một người dùng ảo v . Cùng với đó giá trị bias của nhóm \mathbf{o}_g bằng cách tối ưu độ lỗi xấp xỉ chỉ trên các đánh giá không khuyết của \mathbf{g} cùng với các thành phần regularization ($\|\mathbf{h}_g\|^2, \|\mathbf{o}_g\|^2$) như sau:

$$\begin{aligned} & \min_{H, V, o_u, p_i} \frac{1}{2} * \sum_{\{i | i = 1 \dots n \wedge r_{u,i} \neq *\}} (r_{u,i} - o_g - p_i - \mu - \hat{r}_{g,i})^2 + \frac{\lambda}{2} \|\mathbf{h}_g\|^2 + \frac{\lambda}{2} \|\mathbf{o}_g\|^2 + \frac{\lambda}{2} \|p_i\|^2 \\ &= \min_{H, V, o_u, p_i} \frac{1}{2} * \sum_{\{i | i = 1 \dots n \wedge r_{u,i} \neq *\}} (r_{u,i} - o_g - p_i - \mu - \mathbf{h}_{g,:} * \mathbf{V}_{i,:}^T)^2 + \frac{\lambda}{2} \|\mathbf{h}_g\|^2 + \frac{\lambda}{2} \|\mathbf{o}_g\|^2 + \frac{\lambda}{2} \|p_i\|^2 \end{aligned} \quad (8)$$

Để tối ưu hàm mục tiêu trên, ta có thể dùng Ridge Regression như sau :

$$(\mathbf{h}_g, o_g) = (\bar{\mathbf{g}} - \bar{\mathbf{p}} - \mu * \mathbf{z}) * \bar{\mathbf{V}} * (\bar{\mathbf{V}} * \bar{\mathbf{V}}^T + \lambda * \mathbf{I})^{-1} \quad (9)$$

Trong đó:

- $\bar{\mathbf{g}}$ được hình thành từ \mathbf{g} sau khi loại bỏ các thành phần khuyết
- $\bar{\mathbf{p}}$ là các giá trị bias của sản phẩm tương ứng với các thành phần trong \mathbf{g}
- \mathbf{z} là một véc-tơ hàng có các thành phần bằng 1

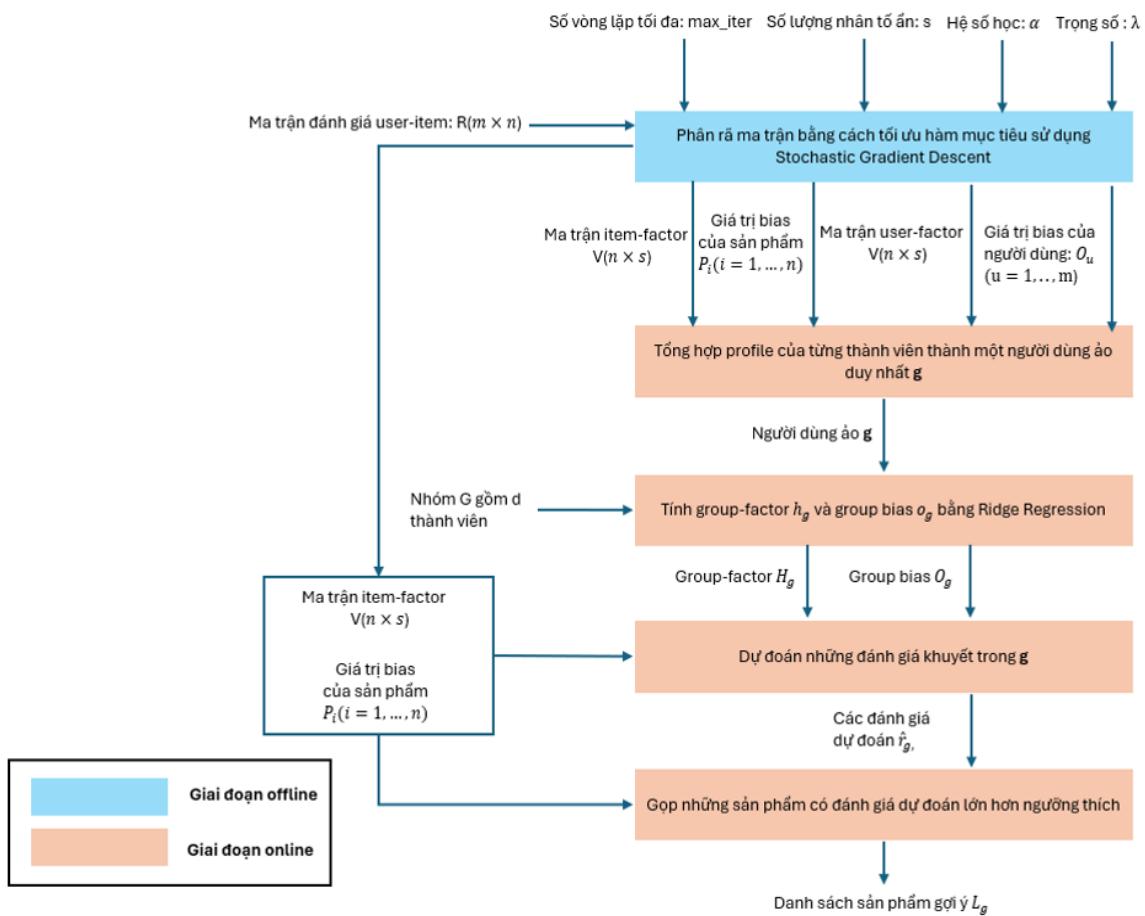
- $\bar{\mathbf{V}}$ được hình thành từ việc chỉ giữ lại các véc-tơ item-factor của các sản phẩm tương ứng với các thành phần trong \mathbf{g} của ma trận \mathbf{V} . Sau đó, tạo cột mới ở vị trí cuối cùng và chuyển nó thành cột mà tất cả giá trị đều bằng 1.
- \mathbf{I} là ma trận đơn vị

Bước 4 [Giai đoạn Online]: Tương tự với quá trình dự đoán đánh giá của một người dùng bằng mô hình nhân tố ẩn như Eq.(2), các đánh giá khuyết (tức các sản phẩm tiềm năng của nhóm \mathbf{g} đại diện bởi người dùng ảo v , $r_{v,i} = *$, $i = 1 \dots m$) như sau:

$$r_{g,i} \approx o_g + p_i + \mu + \mathbf{h}_{g,:} * \mathbf{V}_{i,:}^T \quad (10)$$

Bước 5 [Giai đoạn Online]: Sau khi đã lập đầy các giá trị dự đoán của tập sản phẩm tiềm năng của nhóm \mathbf{G} , ta sẽ lọc ra các sản phẩm có đánh giá được dự đoán lớn hơn ngưỡng “thích” δ vào tập sản phẩm có thể đề xuất cho nhóm.

$$\mathbf{Cg} = \{ r_{g,i} > \delta \mid i = 1 \dots n \} \quad (11)$$

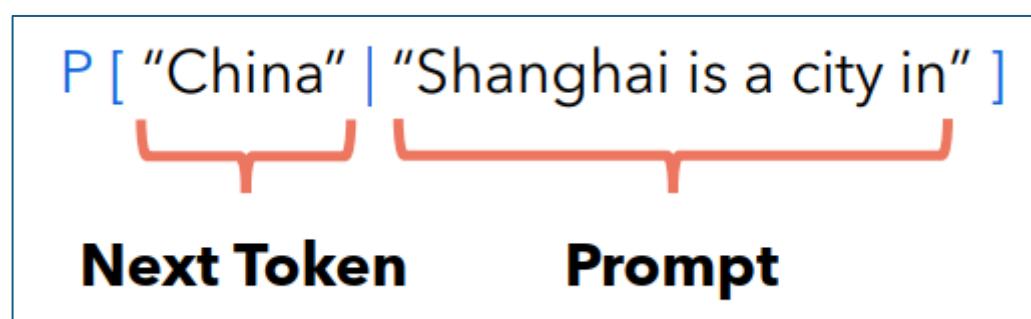


Hình 2.9: Quy trình phát sinh gợi ý nhóm của mô hình nhân tố ẩn theo chiến lược tổng hợp profile [3]

2.5 Giới thiệu về mô hình BERT

2.5.1 Thế nào là một mô hình ngôn ngữ:

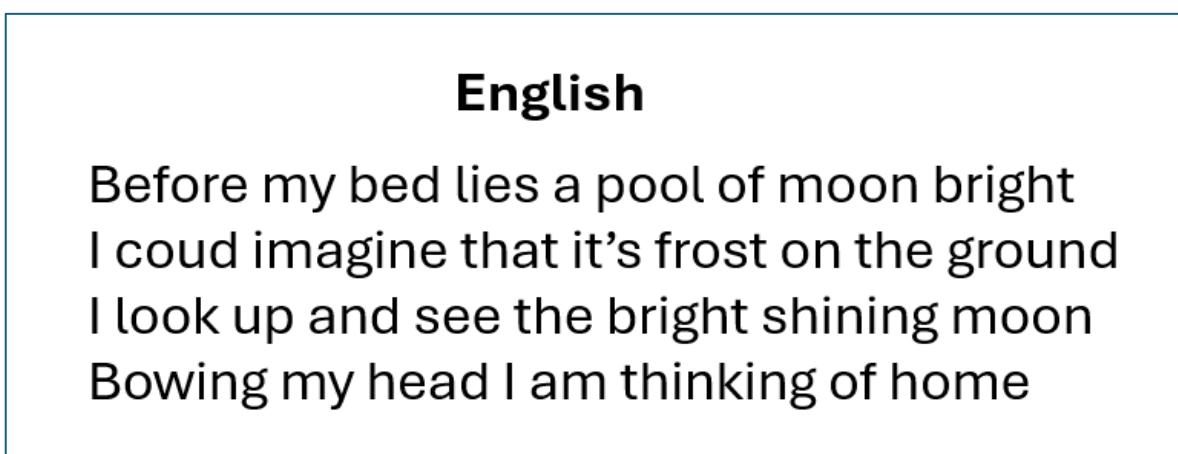
Một mô hình ngôn ngữ (language model) là một mô hình xác suất (probabilistic model) có khả năng gán xác suất cho một chuỗi từ. Trong thực tế, mô hình ngôn ngữ thường được huấn luyện để giúp chúng ta tính toán xác suất xuất hiện của một từ A có phải là từ tiếp theo của một chuỗi từ B hay không như trong hình 2.10. Điều này có nghĩa là, mô hình ngôn ngữ có thể dự đoán từ tiếp theo trong một câu dựa trên ngữ cảnh của các từ trước đó trong câu. Một mạng nơron (neural network) được train với tập dữ liệu text lớn thường được gọi mô hình ngôn ngữ lớn (Large Language Model)



Hình 2.10: Ví dụ giá trị mà mô hình ngôn ngữ cần tính toán [20]

2.5.2 Làm thế nào để huấn luyện một mô hình ngôn ngữ:

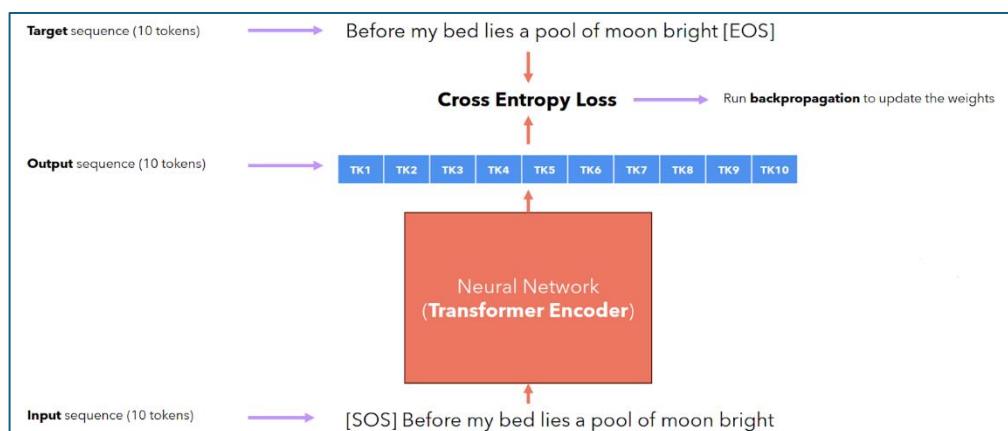
Xét ví dụ ta cần huấn luyện một mô hình để học đoạn thơ sau:



Hình 2.11: Đoạn thơ mà mô hình cần học [20]

Để huấn luyện mô hình (model) học được đoạn thơ trên, ta cần đầu vào (input) là các câu trong đoạn thơ và đầu ra (output) mà chúng ta mong muốn mô hình sẽ trả về cũng chính là các đoạn thơ đó. Với mỗi câu trong đoạn thơ, ta sẽ thực hiện tiền xử lý để nó chuyển câu đầu vào thành tập hợp các token (Token là một đơn vị cơ bản của văn bản, có thể là từ, ký tự hoặc các thành phần khác trong một đoạn văn bản được xử lý trong ngôn ngữ tự nhiên và các mô hình học máy). Sau đó thêm vào trước mỗi câu đầu vào một token [SOS] (Start Of Sentence) để đánh dấu vị trí bắt đầu của câu. Ta cũng cần có đầu ra mà chúng ta mong muốn mô hình sẽ trả về, giả sử ở đây là tập token giống như câu đầu vào nhưng thay vào đó sẽ thêm ở cuối câu token [EOS] (End Of Sentence) để đánh dấu vị trí kết thúc của câu.

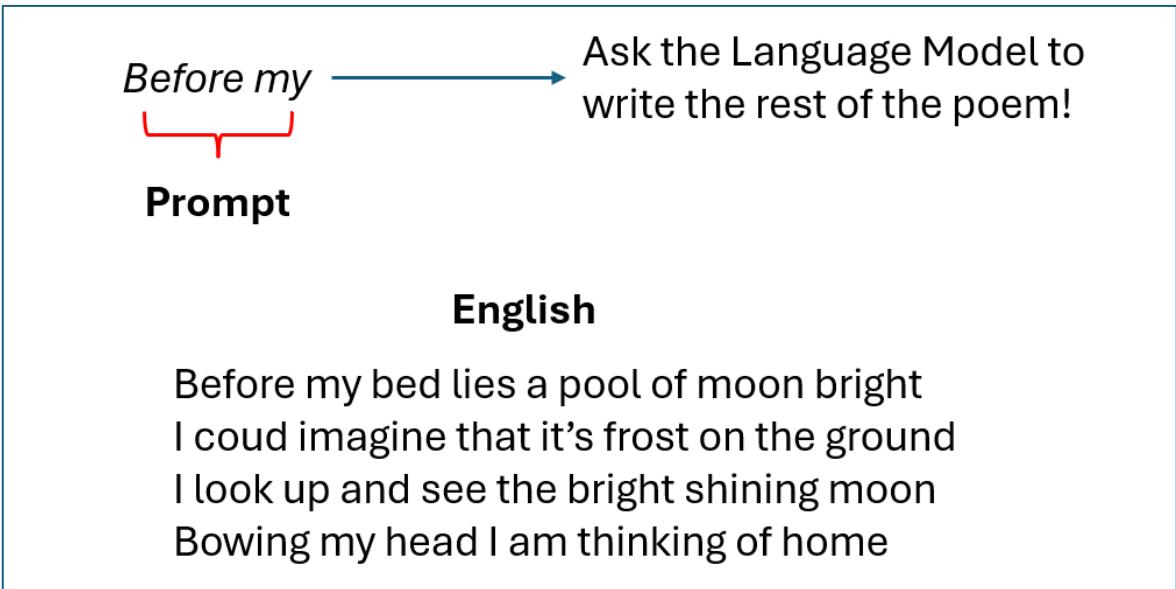
Mô hình sau khi nhận câu đầu vào sẽ bắt đầu học và trả về đầu ra cũng là một tập token có độ dài giống như câu đầu vào. Để điều chỉnh lại tham số của mô hình trong quá trình học, ta sẽ sử dụng hàm mất mát chéo (Cross Entropy Loss Function) (**phụ lục A4**) để kiểm tra sự sai lệch giữa dự đoán của mô hình và đầu ra mong muốn, sau đó chạy lan truyền ngược (backpropagation) (**phụ lục A4**) để cập nhật lại các tham số của mô hình.



Hình 2.12: Minh họa quá trình huấn luyện mô hình ngôn ngữ [20]

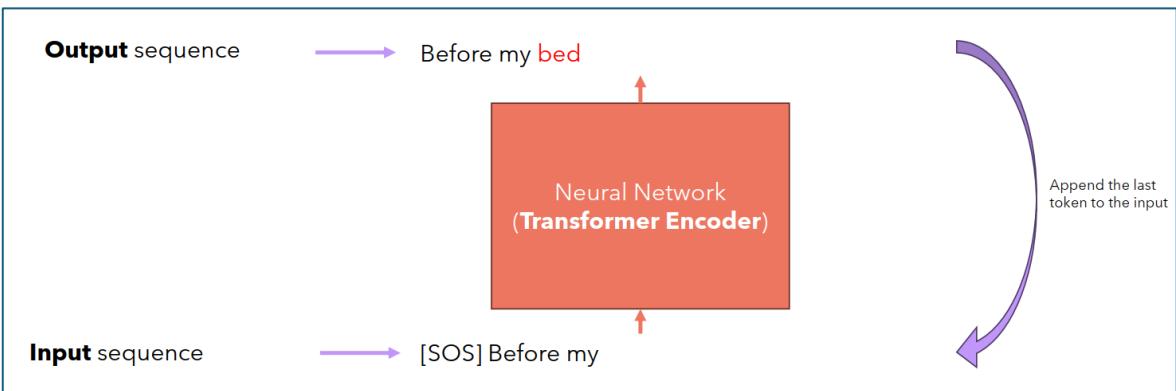
2.5.3 Cách một mô hình ngôn ngữ dự đoán

Lúc này ta sẽ sử dụng mô hình đã học ở trên để viết lại đoạn thơ trên chỉ với 2 từ đầu của bài thơ được trình bày trong hình 2.13.



Hình 2.13: Minh họa đầu vào để mô hình bắt đầu dự đoán [20]

Khi ta đã truyền câu đầu vào gồm 2 từ của bài thơ và token [SOS] vào mô hình, mô hình lúc này sẽ trả về kết quả dự đoán có kích thước bằng câu đầu vào, ta có thể nhận thấy là mô hình đã dự đoán từ tiếp theo sau 2 từ đầu là từ “bed”. Lúc này ta sẽ thêm từ “bed” vào input và tiếp tục quy trình trước đó đến khi mô hình đã dự đoán đến cuối bài thơ được đánh dấu bởi token [EOS].



Hình 2.14: Minh họa quá trình mô hình dự đoán từng từ tiếp theo trong bài thơ [20]

2.5.4 Giới thiệu mô hình BERT

Kiến trúc của mô hình BERT (Bidirectional Encoder Representations from Transformers) được tạo thành từ các lớp mã hóa (Encoder layer) của kiến trúc Transformer [21].

Trong bài báo gốc về BERT [22], nhóm tác giả đã giới thiệu mô hình với 2 cấu hình sau:

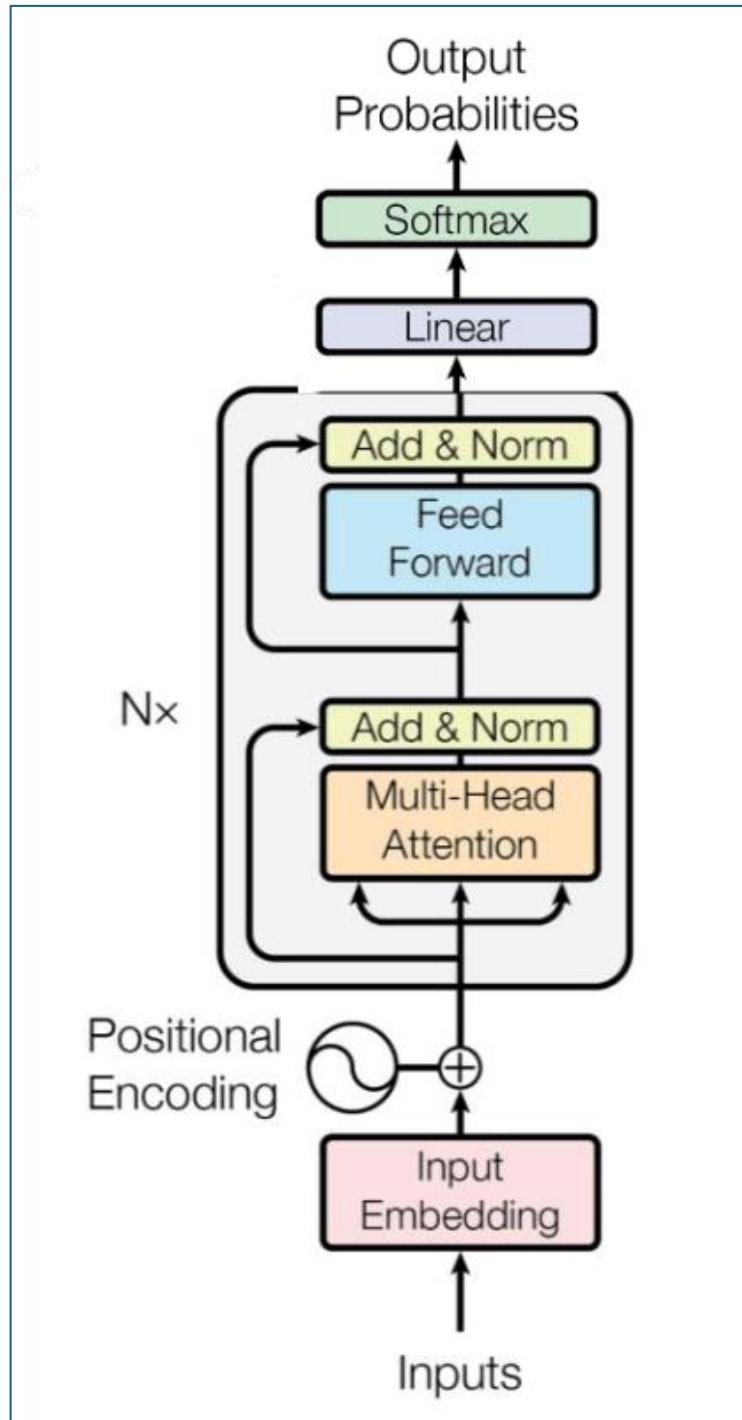
- **BERT-base:** bao gồm 12 lớp mã hóa (encoder layers), mỗi lớp được xếp chồng lên nhau. Tất cả bộ mã hóa (encoder) đều sử dụng 12 đầu chú ý (attention heads). Mạng nơ-ron truyền thẳng (Feedforward network) trong các bộ mã hóa (encoder) bao gồm 768 đơn vị ẩn (hidden units). Do đó, kích thước của biểu diễn thu được từ BERT-base sẽ là 768. Tổng số lượng tham số BERT-base là 110 triệu.
- **BERT-large:** bao gồm 24 lớp mã hóa (encoder layers), mỗi lớp được xếp chồng lên nhau. Tất cả bộ mã hóa (encoder) đều sử dụng 16 đầu chú ý (attention heads). Mạng nơ-ron truyền thẳng (Feedforward network) trong từng bộ mã hóa (encoder) bao gồm 1024 đơn vị ẩn (hidden units). Do đó, kích thước của biểu diễn thu được từ BERT-base sẽ là 1024. Tổng số lượng tham số BERT-base là 340 triệu.

Sự khác biệt của BERT so với kiến trúc Transformer [21]:

- Trong BERT, kích thước của véc-tơ nhúng (embedding vector) được sử dụng thường là 768 hoặc 1024, tùy thuộc vào phiên bản của mô hình (BERT-base hoặc BERT-large). Trong kiến trúc Transformer, kích thước của véc-tơ nhúng (embedding vector) thường là 512. Các véc-tơ nhúng vị trí (Positional embeddings) là tương đối và sẽ được học trong quá trình huấn luyện mô hình, giới hạn chỉ có 512 vị trí
- Trong BERT, các véc-tơ nhúng vị trí (positional embeddings) không cố định và sẽ được học trong quá trình huấn luyện mô hình. Điều này có nghĩa là BERT

không sử dụng mã hóa vị trí (positional encodings) kiểu sin-cos như kiến trúc Transformer, mà thay vào đó, các vị trí của các token trong câu sẽ được biểu diễn bằng các véc-tơ nhóm (vector embeddings) và sẽ được cập nhật trong quá trình huấn luyện.

- BERT sử dụng bộ tách từ WordPiece (WordPiece tokenizer [23]), cho phép sub-word tokens thay vì mỗi token trong câu được biểu diễn bằng một từ trong bộ từ vựng như ở Transformer gốc. Kích thước của bộ từ vựng trong BERT xấp xỉ 30,000 token.

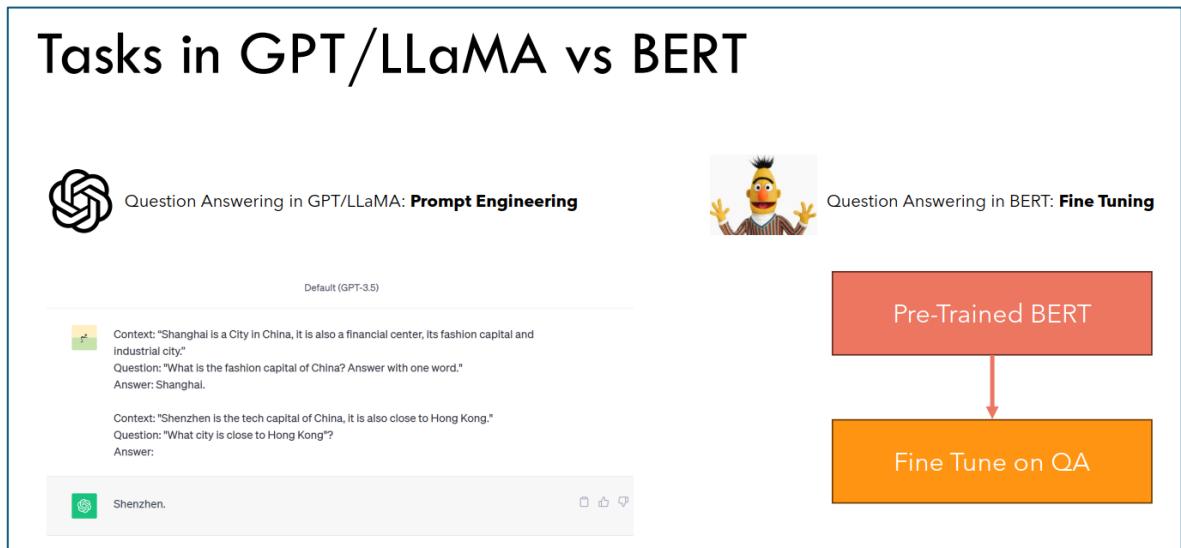


Hình 2.15: Kiến trúc tổng quan của mô hình BERT [20]

2.5.5 Sự khác nhau giữa mô hình BERT so với mô hình ngôn ngữ thông thường

BERT viết tắt của Bidirectional Encoder Representations from Transformers dịch sang tiếng Việt là Biểu diễn mã hóa hai chiều từ Transformers.

- Khác với các mô hình ngôn ngữ thông thường như GPT hay Llama, BERT không xử lý các "nhiệm vụ cụ thể" bằng cách sử dụng gợi ý hay lời nhắc (prompts). Thay vào đó, nó có thể được tinh chỉnh (fine-tuning) để đạt hiệu suất tốt hơn trên các nhiệm vụ cụ thể sau khi được huấn luyện trước trên một lượng lớn dữ liệu.
- Không giống như các mô hình ngôn ngữ thông thường, BERT đã được huấn luyện bằng cách sử dụng cả ngữ cảnh bên trái (left context) và ngữ cảnh bên phải (right context).
- Không giống như các mô hình ngôn ngữ thông thường, BERT không được xây dựng một cách đặc biệt cho việc tạo văn bản. Vì BERT không được huấn luyện để dự đoán token tiếp theo (Next Token Prediction). Thay vào đó, BERT được huấn luyện trên hai tác vụ chính: Mô hình ngôn ngữ ẩn (Masked Language Model) và dự đoán câu tiếp theo (Next Sentence Prediction).



Hình 2.16: Sự khác nhau giữa mô hình BERT và GPT/ Llama [20]

2.5.6 Sự quan trọng của ngữ cảnh trái và ngữ cảnh phải

Trước khi tìm hiểu quy trình huấn luyện trước (pre-trained) của BERT, ta phải cần hiểu tại sao khác với kiến trúc Transformer gốc [21] chỉ tập trung vào ngữ cảnh trái

(left context) trong câu thì BERT lại tập trung vào cả ngữ cảnh phải (right context) và ngữ cảnh trái (left context) trong câu thông qua ví dụ sau:

Imagine there's a kid who just broke his mom's favorite necklace. The kid doesn't want to tell the truth to his mom, so he decides to make up a lie.

So, instead of saying directly: "Your favorite necklace has broken"

The kid may say: "Mom, I just saw the cat playing in your room and your favorite necklace has broken."

Or it may say: "Mom, aliens came through your window with laser guns and your favorite necklace has broken."

*Hình 2.17: Minh họa ngữ cảnh bên trái và ngữ cảnh bên phải
trong ngôn ngữ tự nhiên [20]*

Như chúng ta quan sát được ở ví dụ trên, đứa trẻ có thể nói bất kỳ lời nói dối nào, nhưng với mỗi một lời nói dối khác nhau sẽ luôn có sự ảnh hưởng khác nhau đến khả năng đưa ra kết luận cuối mà chúng ta muốn là (your favorite necklace has broken) nếu chúng ta chỉ tập trung vào ngữ cảnh phải (right context.)

2.5.7 Quy trình huấn luyện trước của mô hình BERT

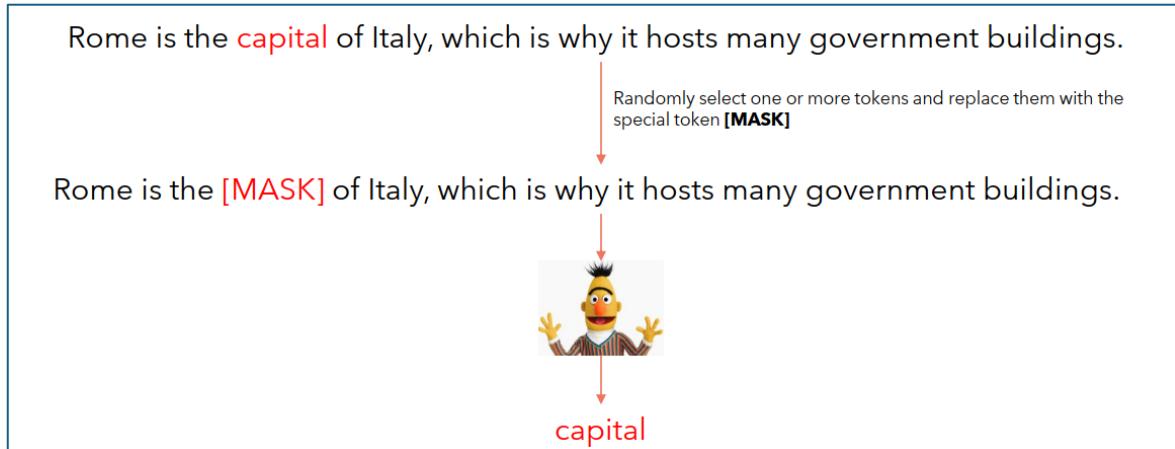
Huấn luyện trước (pre-trained) ở đây nghĩa là gì? Giả sử chúng ta có một mô hình là m , đầu tiên ta sẽ huấn luyện mô hình m trên một tập dữ liệu lớn về một tác vụ cụ thể và lưu lại mô hình đã huấn luyện, mô hình đó được xem là một mô hình đã được huấn luyện trước (pre-trained model). Lúc này, để giải quyết một bài toán khác, thay vì phải khởi tạo lại mô hình và huấn luyện mô hình lại từ đầu thì ta chỉ cần sử dụng mô hình đã được huấn luyện trước (pre-trained model) và tinh chỉnh (fine-tune) bộ trọng số của nó cho tác vụ mới.

Trong bài báo gốc [22], nhóm tác giả đã huấn luyện trước (pre-trained) mô hình BERT trên hai tác vụ chính đó là mô hình ngôn ngữ ẩn (Masked Language Model) (MLM) và dự đoán câu tiếp theo (Next Sentence Prediction) (NSP).

a. Huấn luyện mô hình BERT cho tác vụ mô hình ngôn ngữ ẩn

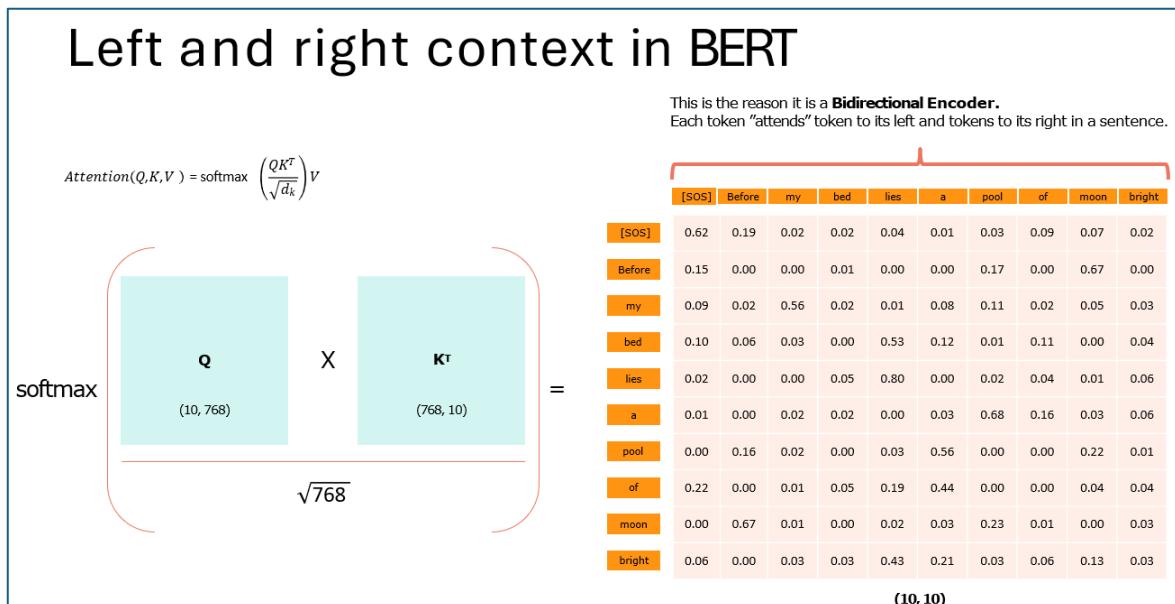
Huấn luyện mô hình cho tác vụ mô hình ngôn ngữ ẩn tương tự như việc dạy cho mô hình cách làm một bài tập điền khuyết. Cụ thể, chúng ta sẽ lựa chọn ngẫu nhiên một từ trong câu để ẩn đi (masked), và mô hình phải dự đoán chính xác từ bị ẩn đó

dựa trên ngữ cảnh trái và phải của câu đã cho.



Hình 2.18: Minh họa mô hình cho tác vụ dự đoán từ bị ẩn
trong câu [20]

Quy trình tính giá trị chú ý (Attention value) (phục lục B3) trong các khối mã hóa (Encoder) của BERT được diễn ra tương tự như trong kiến trúc Transformer [21]. Nhưng nhóm tác giả đã loại bỏ sử dụng ẩn tuân tự (casual masking), nhằm cho phép sử dụng cả ngữ cảnh trái và phải tham gia vào quá trình tính giá trị chú ý (Attention value). Đó là lý do mà các khối mã hóa (Encoder) của BERT được gọi là Bộ Mã hóa Hai chiều (Bidirectional Encoder)



Hình 2.19: Minh họa ngữ cảnh trái và phải trong BERT [20]

Để tránh mô hình có thể dễ dàng học bằng cách ghi nhớ các từ đã được ẩn (masked) thay vì học được khả năng tổng quát hóa và xác định được ngữ cảnh. Nhóm tác giả đã sử dụng chiến thuật masking như sau:

- Tỷ lệ ẩn: Khoảng 15% các từ trong câu sẽ được chọn để ẩn. Đây là một phần của dữ liệu đầu vào được xử lý để mô hình học cách dự đoán các từ bị ẩn.
- Phương pháp ẩn:
 - Thay thế bằng [MASK] Token: Khoảng 80% số từ được chọn để ẩn sẽ được thay thế bằng ký hiệu đặc biệt [MASK].
 - Thay thế bằng từ ngẫu nhiên: Khoảng 10% số từ bị ẩn sẽ được thay thế bằng một từ ngẫu nhiên khác trong từ điển. Việc này giúp mô hình học cách nhận diện từ chính xác trong trường hợp từ bị ẩn không phải là [MASK].
 - Giữ Nguyên: Khoảng 10% số từ bị ẩn sẽ không được thay đổi, mà vẫn giữ nguyên từ gốc. Điều này giúp mô hình học cách dự đoán từ mà không bị ảnh hưởng bởi sự thay đổi.

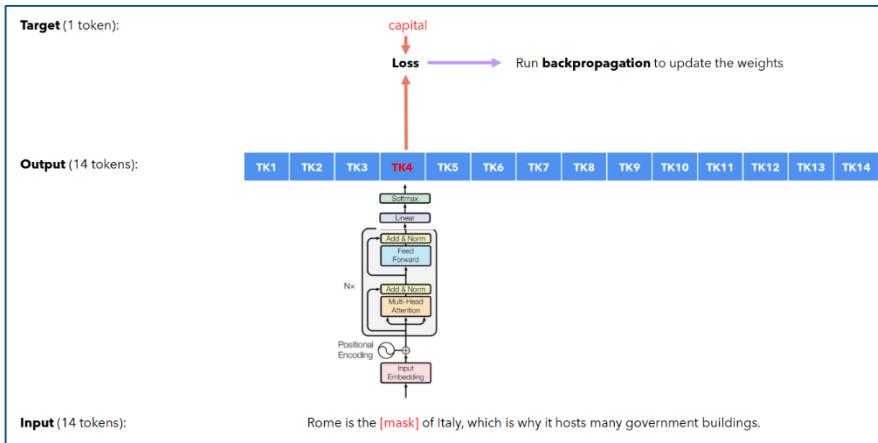
Rome is the **capital** of Italy, which is why it hosts many government buildings.

The pre-training procedure selects 15% of the tokens from the sentence to be masked. When a token is selected to be masked (suppose the word “capital” is selected):

- 80% of the time it is replaced with the [MASK] token → Rome is the [MASK] of Italy, which is why it hosts many government buildings.
- 10% of the time it is replaced with a random token → Rome is the **zebra** of Italy, which is why it hosts many government buildings.
- 10% of the time it is not replaced → Rome is the **capital** of Italy, which is why it hosts many government buildings.

Hình 2.20: Minh họa chiến thuật masking [20]

Quy trình huấn luyện mô hình BERT dành cho tác vụ mô hình ngôn ngữ ẩn (Masked Language Model) được mô tả bằng hình dưới đây:



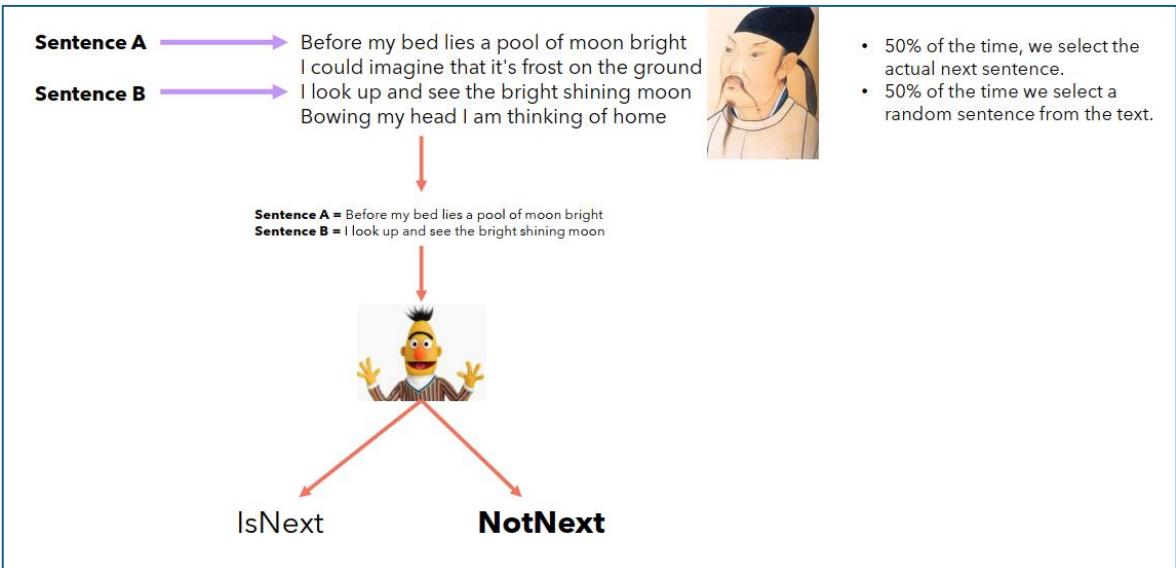
Hình 2.21: Minh họa quá trình huấn luyện mô hình BERT cho tác vụ dự đoán từ bị ẩn trong câu [20]

Sau khi thu được biểu diễn của các token trong câu đầu vào qua các lớp mã hóa (Encoder layer), ta sẽ lấy biểu diễn của token đã bị ẩn (masked) để truyền vào lớp Tuyến tính và Softmax (Linear & Softmax layer) (**phụ lục B6**) để thực hiện dự đoán từ đã bị mask. Sau đó tính độ mất mát (loss) và chạy lan truyền ngược (backpropagation) (**phụ lục A4**) để cập nhật lại các trọng số của mô hình.

Việc huấn luyện theo tác vụ mô hình ngôn ngữ ẩn (Masked Language Model) (MLM) đã giúp mô hình học được biểu diễn của từ trong các ngữ cảnh của câu và làm tăng cường khả năng hiểu ngôn ngữ tự nhiên.

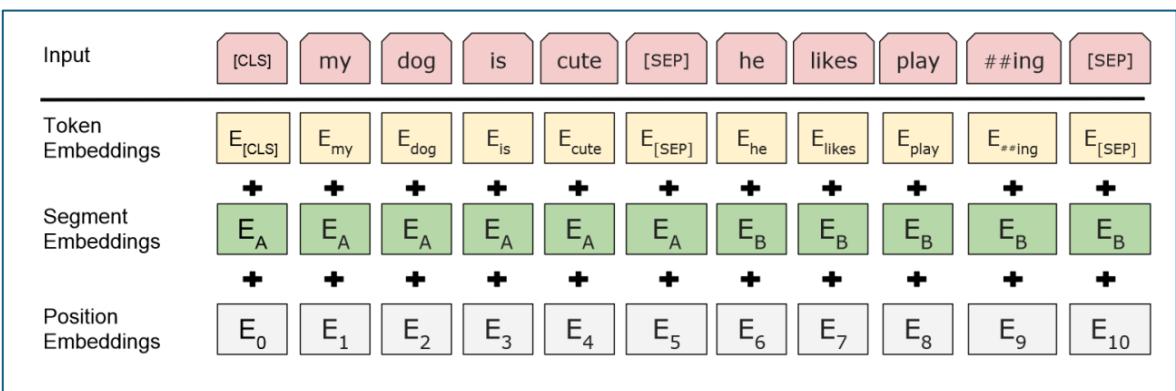
b. Huấn luyện mô hình BERT cho tác vụ dự đoán câu tiếp theo

Trong thực tế, một số tác vụ hạ nguồn (downstream task) yêu cầu mô hình phải học mối quan hệ giữa các câu hơn là giữa các token trong cùng một câu. Ví dụ, xem xét một tác vụ hạ nguồn (downstream task) như được minh họa dưới đây: mô hình nhận vào hai câu thơ A và B được lấy từ cùng một bài thơ. Nhiệm vụ của mô hình là dự đoán xem câu B có phải là câu tiếp theo của câu A hay không.



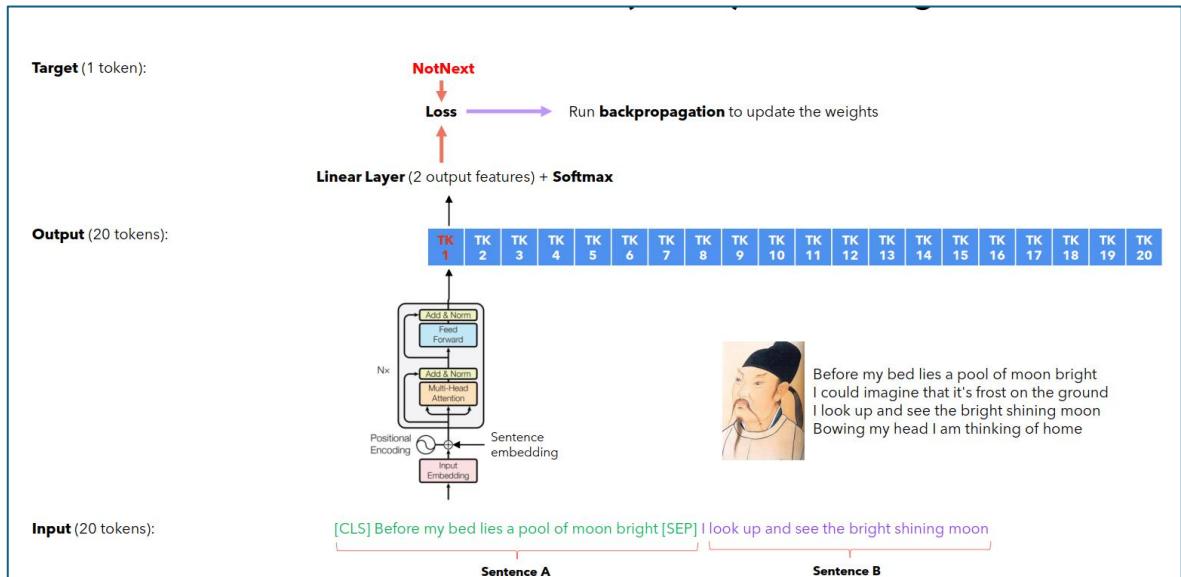
Hình 2.22: Minh họa mô hình cho tác vụ dự đoán câu tiếp theo [20]

Đó là lý do BERT được pre-trained trên tác vụ dự đoán câu tiếp theo (Next Sentence Prediction) (NSP). Để chuyển 2 câu A, B trên thành đầu vào cho BERT. Ta sẽ cần tạo ra một đầu vào có hai câu liên tiếp A, B. Mỗi câu được phân tách bằng token [SEP]. Sau đó, ta sẽ thêm token [CLS] vào đầu câu đầu tiên và một token [SEP] vào cuối câu thứ hai. Bên cạnh thông tin về token embedding (**phụ lục B1**) và position embedding (**phụ lục B2**) thì nhóm tác giả còn cung cấp thêm segment embedding [22] vào đầu vào của BERT để mô hình hiểu được token nào sẽ thuộc về câu A hoặc B



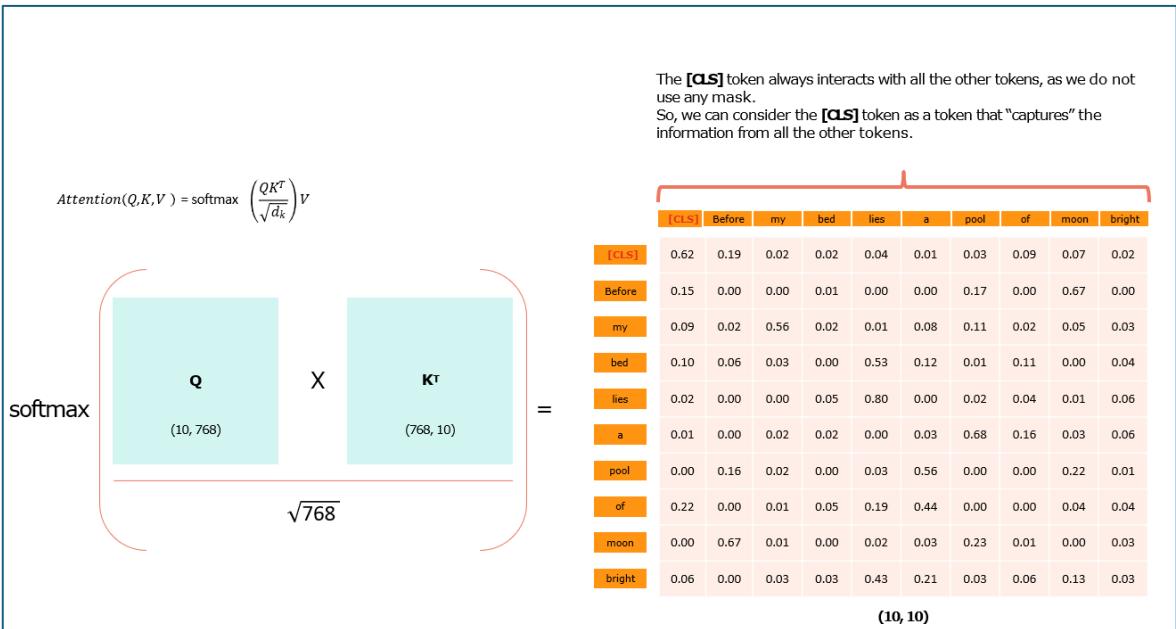
Hình 2.23: Minh họa đầu vào của BERT cho tác vụ dự đoán câu tiếp theo [21]

Tương tự như khi huấn luyện BERT trên tác vụ mô hình ngôn ngữ ẩn, quy trình huấn luyện BERT trên tác vụ dự đoán câu tiếp được minh họa như sau:



*Hình 2.24: Minh họa quá trình huấn luyện mô hình BERT
cho tác vụ dự đoán câu tiếp theo [20]*

Khác với khi train theo mô hình ngôn ngữ ẩn (Masked Language Model) (MLM). Vì Token [CLS] luôn tương tác với tất cả các token khác trong câu, hay nói cách khác là token [CLS] là một token tổng hợp tất cả các thông tin của các token khác trong câu. Vì thế nên trong quá trình huấn luyện mô hình BERT cho vụ dự đoán câu tiếp theo. Ta sẽ lấy biểu diễn của token [CLS] ở đầu ra để đưa vào lớp tuyến tính và Softmax (Linear & Softmax layer) (**phụ lục B6**) để thực hiện dự đoán.



Hình 2.25: Minh họa token [CLS] trong BERT [20]

Chương 3

Hướng tiếp cận

3.1 Áp dụng chiến lược tổng hợp profile dựa trên trung bình tất cả đánh giá quan sát được kết hợp trọng số chi tiết

Như đã trình bày ở mục 2.4, việc áp dụng mô hình nhân tố ẩn cho gợi ý nhóm có thể áp dụng ở chiến lược tổng hợp gợi ý và tổng hợp profile. Để hệ thống gợi ý nhóm đạt được kết quả tốt hơn, các chiến lược tổng hợp profile tiên tiến hơn nên được áp dụng.

Việc tổng hợp profile trong hệ thống tư vấn nhóm thường được thực hiện bằng trung bình có trọng số đánh giá của các thành viên trong nhóm [24] [25]. Mỗi trọng số thể hiện mức độ ảnh hưởng của thành viên tương ứng trong việc ra quyết định của nhóm. Do đó, có thể thấy rằng hiệu quả của việc tổng hợp profile được xác định bởi 2 yếu tố: cách ước lượng các đánh giá của người dùng ảo cho các sản phẩm và cách xác định trọng số của các thành viên trong nhóm [2].

- Việc ước lượng đánh giá của người dùng ảo trong profile về một sản phẩm cụ thể chỉ có thể được thực hiện dựa trên các đánh giá cho sản phẩm được quan sát từ các thành viên trong nhóm [26] [27]. Tuy nhiên, với tính chất rất thưa thớt của dữ liệu đánh giá của người dùng, rất khó để tận dụng nguồn dữ liệu này để thực hiện tổng hợp profile một cách chính xác và hiệu quả vì sự đồng thuận của nhóm sẽ không cao khi sở thích của nhóm về sản phẩm được xây dựng dựa trên sở thích của một số ít thành viên thay vì tất cả các thành viên [28] [15]. Trên thực tế, việc tổng hợp profile bằng các hàm tổng hợp có sự đóng góp của càng nhiều thành viên nhóm sẽ trở nên đáng tin cậy hơn [28].
- Đối với các trọng số thể hiện mức độ ảnh hưởng của các thành viên trong nhóm, các tính toán trước đó đã giả định chúng giống nhau đối với tất cả các sản phẩm

được xem xét [29]. Tuy nhiên, trên thực tế, đối với một sản phẩm, sức ảnh hưởng của một thành viên đối với nhóm có thể là rất đáng kể, nhưng đối với sản phẩm khác, sức ảnh hưởng của thành viên đó sẽ không còn nhiều như trước [30]. Hoặc đối với một sản phẩm cụ thể, một thành viên có thể có trọng số lớn nhất trong nhóm này nhưng khi ở một nhóm khác, thành viên đó không chắc chắn mang trọng số lớn nhất. Do đó, ta cần xem xét trọng số của mỗi người dùng trong từng nhóm phân biệt đối với từng sản phẩm khác nhau.

Từ những vấn đề trên, nhóm đề xuất sử dụng chiến lược “tổng hợp profile dựa trên tất cả đánh giá quan sát được tích hợp trọng số chi tiết” (AOFRAM&W) [2]. Đây là chiến lược nhằm khai thác tối đa sự đồng thuận giữa các thành viên trong nhóm và cho thấy hiệu suất tốt hơn đáng kể so với các chiến lược trước đó bằng việc tận dụng trọng số thông tin về tầm ảnh hưởng của mỗi thành viên và độ tin cậy khi đưa ra đánh giá của mỗi thành viên.

Cụ thể, đối với đánh giá của thành viên $u \in \mathbf{G}$ đối với sản phẩm i , trọng số $w_{u,i}$ là sự kết hợp giữa tầm ảnh hưởng của thành viên u , ký hiệu là k_u và độ tin cậy của xếp hạng $r_{u,i}$ ký hiệu là $c_{u,i}$ hàm ý rằng một thành viên chỉ đóng vai trò quan trọng trong quá trình ra quyết định của nhóm về một sản phẩm nếu thành viên đó đảm bảo cả 2 yếu tố: thành viên ấy có nhiều đánh giá trong hệ thống so với các thành viên khác trong nhóm và các đánh giá về sản phẩm do thành viên ấy đưa ra có độ chắc chắn cao trong mô hình tư vấn.

Ngược lại, mặc dù một thành viên có độ ảnh hưởng lớn trong nhóm, nhưng trong trường hợp đánh giá về một sản phẩm nào đó là không chắc chắn thì tầm quan trọng của đánh giá đó sẽ giảm đi trong quyết định của nhóm đối với sản phẩm đó. Cụ thể trọng số $w_{u,i}$ được tính như sau:

$$w_{u,i} = c_{u,i}^{[\beta,1]} * k_u^{[\beta,1]} \quad (12)$$

Nhằm cân bằng giữa độ ảnh hưởng của một thành viên và độ chắc chắn của đánh giá từ thành viên đó, ta cần chuẩn hóa để đưa về cùng phạm vi $[\beta,1]$ bằng

phương pháp min-max normalization [2] với β thường được mang giá trị 0.4 [31].

Cụ thể:

- Công thức chuẩn hóa một giá trị k_u về khoảng $[\beta, 1]$:

$$k_u^{[\beta,1]} = \beta + (1 - \beta) * \frac{k_u - \min_{a \in G} k_a}{\max_{a \in G} k_a - \min_{a \in G} k_a} \quad (13)$$

Với k_u là số lượng đánh giá quan sát được từ hệ thống của người dùng u .

- Công thức chuẩn hóa một giá trị $c_{u,i}$ về khoảng $[\beta, 1]$:

$$k_u^{[\beta,1]} = \beta + (1 - \beta) * \frac{k_u - \min_{a \in G} k_a}{\max_{a \in G} k_a - \min_{a \in G} k_a} \quad (14)$$

Với $c_{u,i}$ được xem xét trong mô hình nhân tố ẩn sẽ phụ thuộc vào độ lỗi xấp xỉ, với độ lỗi càng nhỏ thì độ chắc chắn của đánh giá sẽ càng cao.

Nếu giá trị $r_{u,i}$ là một đánh giá quan sát được ($r_{u,i} \neq *$) thì độ lỗi sẽ là độ chênh lệch giữa đánh giá dự đoán từ mô hình nhân tố ẩn với đánh giá quan sát được $|r_{u,i} - \hat{r}_{u,i}|$. Ngược lại, nếu đánh giá $r_{u,i}$ khuyết ($r_{u,i} = *$), ta đưa ra dự đoán về nó bằng cách sử dụng độ thiên vị bias được học từ ma trận nhân tố ẩn, tức là $|o_u + p_i + \mu|$. Sử dụng dự đoán đánh giá thiên vị làm cơ sở so sánh cho thấy độ tin cậy cao [32][33]. Sau đó, độ lỗi xấp xỉ của đánh giá khuyết được ước tính bằng độ lệch giữa đánh giá thiên vị $|o_u + p_i + \mu|$, và đánh giá dự đoán dựa trên nhân tố ẩn, tức là $|\mathbf{H}_{u,:} * \mathbf{V}_{i,:}^T|$. Cuối cùng, độ chắc chắn của đánh giá được tính bằng cách đảo ngược độ lỗi xấp xỉ được chuẩn hóa về $[0,1]$ như sau :

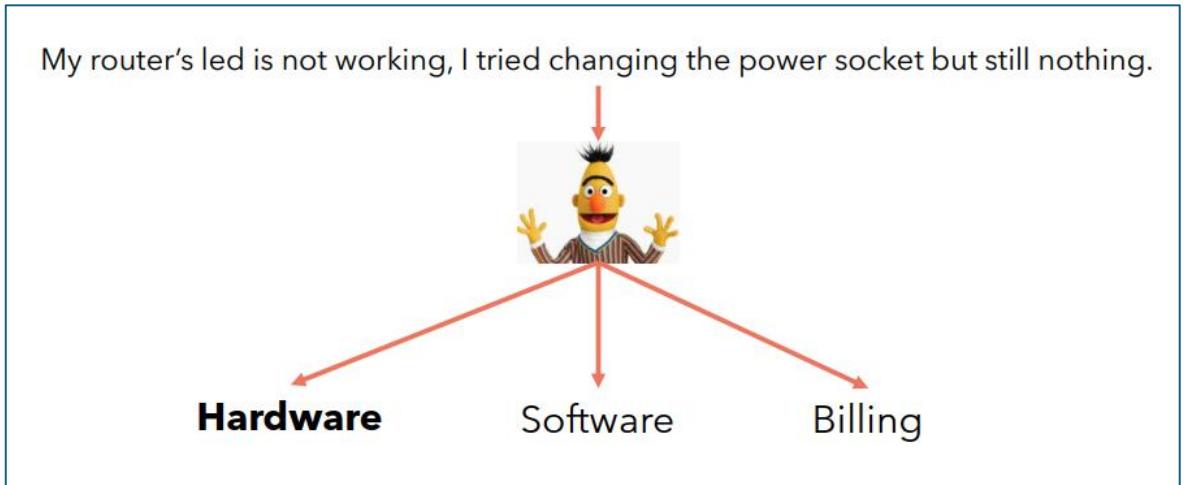
$$c_{u,i} = \begin{cases} 1 - |r_{u,i} - o_u - p_i - \mu - \mathbf{H}_{u,:} * \mathbf{V}_{i,:}^T|^{[0,1]} & \text{if } r_{u,i} \neq * \\ 1 - |o_u + p_i + \mu - \mathbf{H}_{u,:} * \mathbf{V}_{i,:}^T|^{[0,1]} & \text{if } r_{u,i} = * \end{cases} \quad (15)$$

Do đó, công thức Eq.(7) được trình bày tại mục 2.4 sẽ được cải tiến bằng việc thêm trọng số chi tiết như sau:

$$r_g, i = \frac{\sum_{u \in G} S_{u,i} * w_{u,i}}{\sum_{u \in G} w_{u,i}} \quad (16)$$

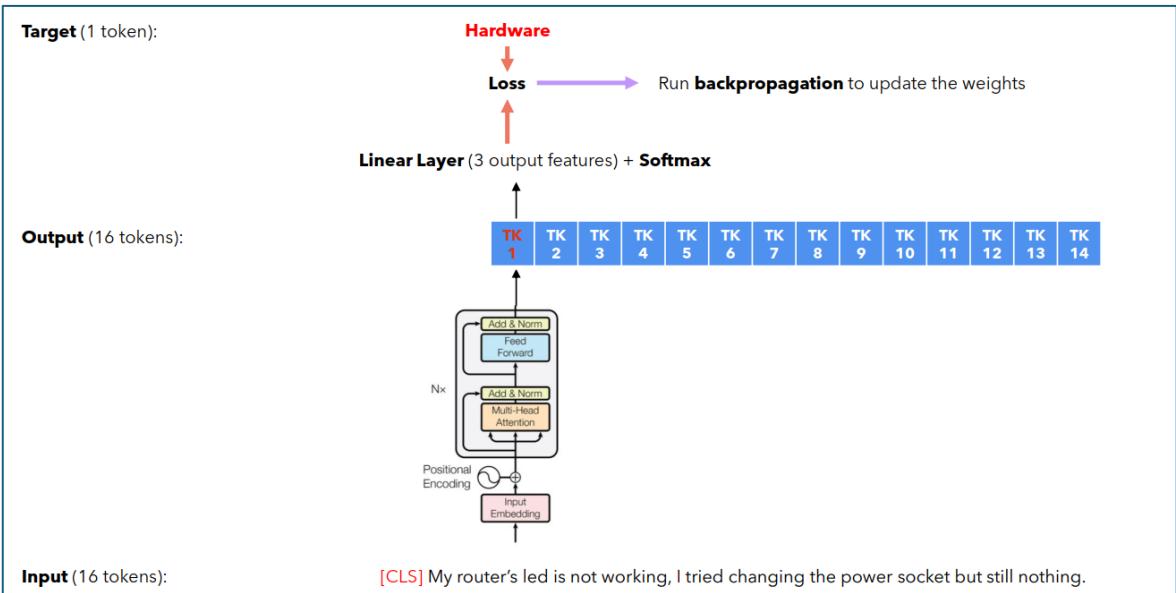
3.2 Tinh chỉnh mô hình BERT cho bài toán phân loại văn bản

Phân loại văn bản (Text classification) là bài toán mô hình gán nhãn cho một tập text đầu vào. Như ví dụ sau: Chúng ta cần một mô hình để nhận biết khiếu nại của khách hàng là về phần cứng, phần mềm hay vấn đề về thanh toán.



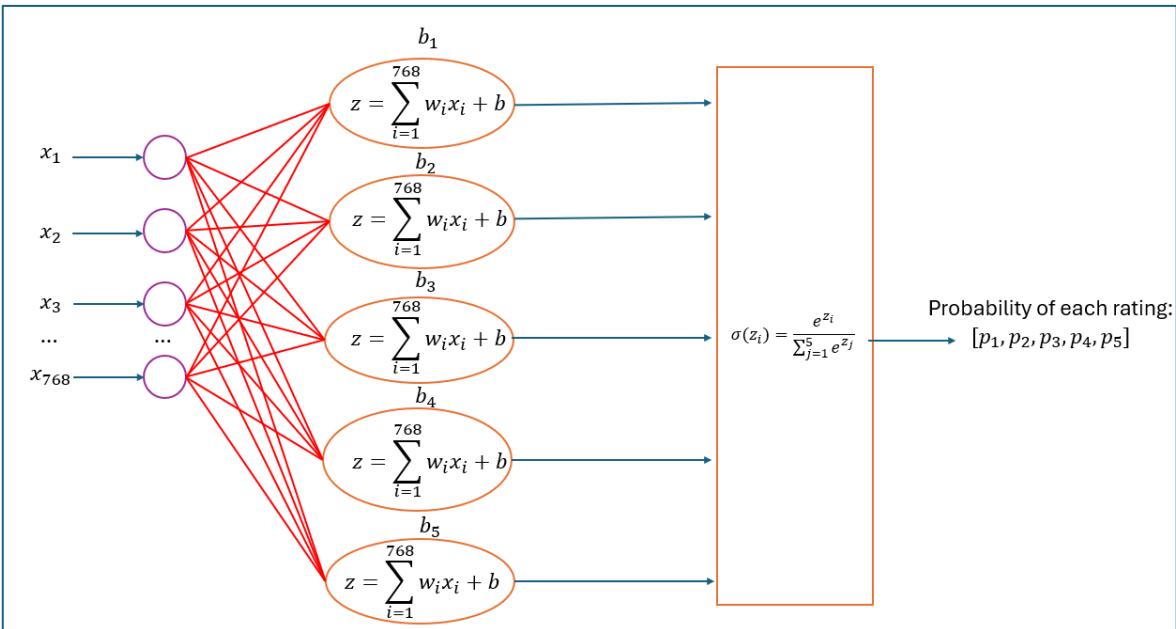
Hình 3.1: Minh họa bài toán phân loại khiếu nại của khách hàng [20]

Vì cấu tạo đặc biệt của mình nên mô hình BERT có thể được tinh chỉnh (fine-tune) để giải quyết các bài toán khác nhau bằng cách thay lớp ánh xạ đầu ra (output mapping layer) ở cuối mô hình BERT đã được huấn luyện trước (pre-trained BERT model) bằng một lớp ánh xạ đầu ra khác sau đó tinh chỉnh lại các trọng số của mô hình BERT đã được huấn luyện trước theo output của bài toán mà ta muốn giải quyết.



*Hình 3.2: Minh họa quá trình tinh chỉnh mô hình BERT
cho bài toán phân loại khiếu nại của khách hàng [20]*

Ở trong khóa luận này, chúng em sẽ tinh chỉnh mô hình BERT để dự đoán rating sản phẩm trên thang điểm từ 1 đến 5 dựa trên nhận xét (text review) của người dùng hay cụ thể hơn là đánh nhau nhận xét của người dùng. Chúng em sẽ tái sử dụng lại mô hình BERT-base đã được huấn luyện trước (pre-trained) và thay đổi kiến trúc của lớp tuyến tính và Softmax (Linear And Softmax Layer) (**phụ lục B6**) được minh họa dưới đây:



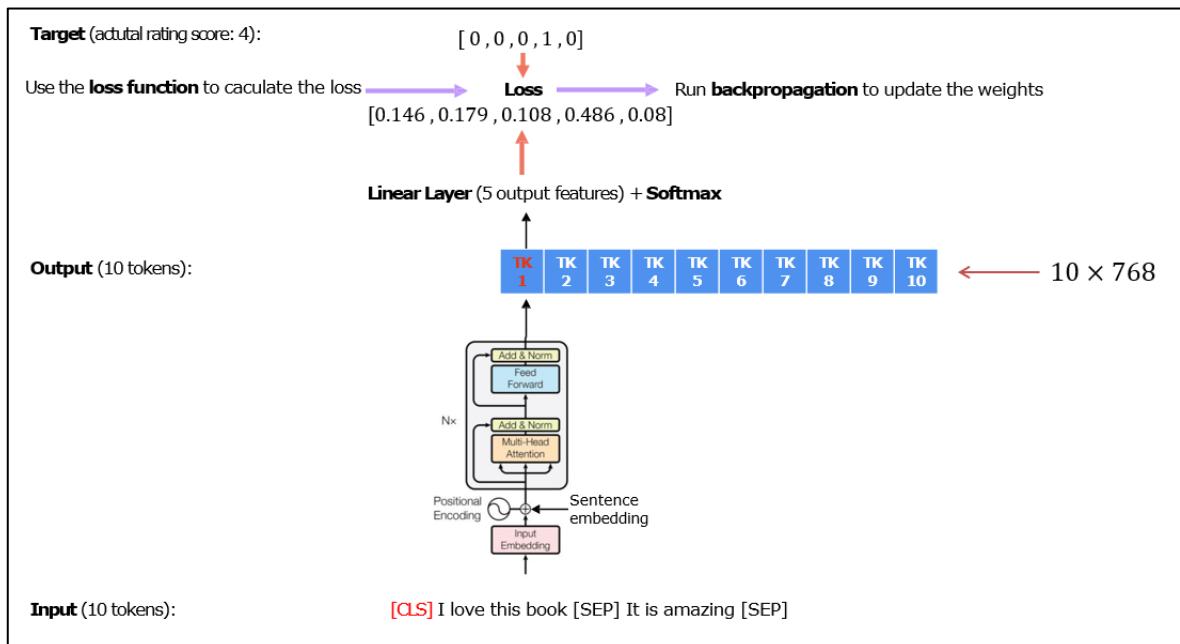
Hình 3.3: Minh họa kiến trúc của Linear and Softmax Layer cho bài toán dự đoán điểm đánh giá từ nhận xét của người dùng

Trong đó:

- Input là một véc-tơ $x = [x_1, x_2, \dots, x_{768}]$ có chiều dài là 768 tương ứng với kích thước biểu diễn của một token thu được sau quá trình lan truyền thông tin qua các khôi mã hóa (encoder).
- Mỗi đơn vị ẩn (neural) tính toán một tổng có trọng số của các đầu vào cộng một bias với công thức tổng quát như sau: $z = \sum_{i=1}^{768} w_i * x_i + b$
- Tập 5 giá trị đầu ra z_i của layer sau đó được truyền vào hàm softmax để chuyển đổi thành các phân phối xác suất của từng loại điểm đánh giá sao cho tổng các xác suất bằng 1. Công thức của hàm softmax: $\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^5 e^{z_j}}$
- Đầu ra cuối cùng của lớp chính là một véc-tơ chứa $p = [p_1, p_2, p_3, p_4, p_5]$ chứng tỏ xác suất xuất hiện của từng loại điểm đánh giá từ 1-5

Như đã đề cập trước đó, trong token [CLS] là token tổng hợp tất cả thông tin của các token khác trong câu nên khi nhận được output sau quá trình lan truyền qua các

khỏi mã hóa (Encoder block), chúng em sẽ lấy véc-tơ biểu diễn của token [CLS] để thực hiện phân loại văn bản diễn ra trong hình sau:



Hình 3.4: Minh họa quá trình tinh chỉnh mô hình BERT cho bài toán dự đoán điểm đánh giá từ nhận xét của người dùng

Để tính toán giá trị mất mát (loss) cho giai đoạn lan truyền ngược backpropagation (**phụ lục A4**) nhóm chúng em sử dụng hàm mất mát (loss function) được sử dụng cho bài toán phân loại có nhiều đầu ra với công thức như sau:

$$CE\ Loss = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^k y_{ij} \log(p_{ij})$$

Trong đó:

- n là số lượng mẫu trong tập dữ liệu
- k là số lượng lớp phân loại (ở đây là 5)
- y_{ij} là nhãn thực tế cho mẫu dữ liệu thứ i và lớp j
- p_{ij} là xác suất dự đoán cho mẫu thứ i lớp thứ j, được tính bằng hàm Softmax

Trong bài báo của mô hình BERT [21], nhóm tác giả đã đề xuất khoảng giá trị bộ tham số cho quá trình huấn luyện như sau:

- Tốc độ học (Learning rate): 5e-5, 3e-5, 2e-5
- Số lượng epoch: 2, 3, 4
- Batch size: 16, 32

Tuy nhiên để tìm một bộ tham số tối ưu cho quá trình huấn luyện còn phải tùy thuộc vào độ phức tạp của tác vụ cần tinh chỉnh (fine tune) và sự khác nhau giữa dữ liệu sử dụng để tinh chỉnh so với dữ liệu mà model BERT đã được huấn luyện trước (pre-trained). Đã có một số nghiên cứu được thực hiện nhằm tìm ra giá trị tham số tối ưu để fine tune BERT như trong các công trình [34, 35, 36, 37, 38, 39]. Trong khóa luận này nhóm chúng em sử dụng bộ tham số huấn luyện như sau: Learning rate: 5e-5, số lượng epoch: 10, Batch size: 32.

Bộ dữ liệu mà nhóm sử dụng để tinh chỉnh mô hình BERT là bộ dữ liệu Yelp reviews được sử dụng trong công trình [40] với thông tin như sau:

- Training set: 650,000 nhận xét (text review) đã được đánh nhãn sẵn với 130,000 nhận xét (text review) với mỗi điểm rating
- Test set: 50,000 nhận xét (text review) đã đã được đánh nhãn sẵn với mỗi loại rating sẽ có 10,000 nhận xét (text review)

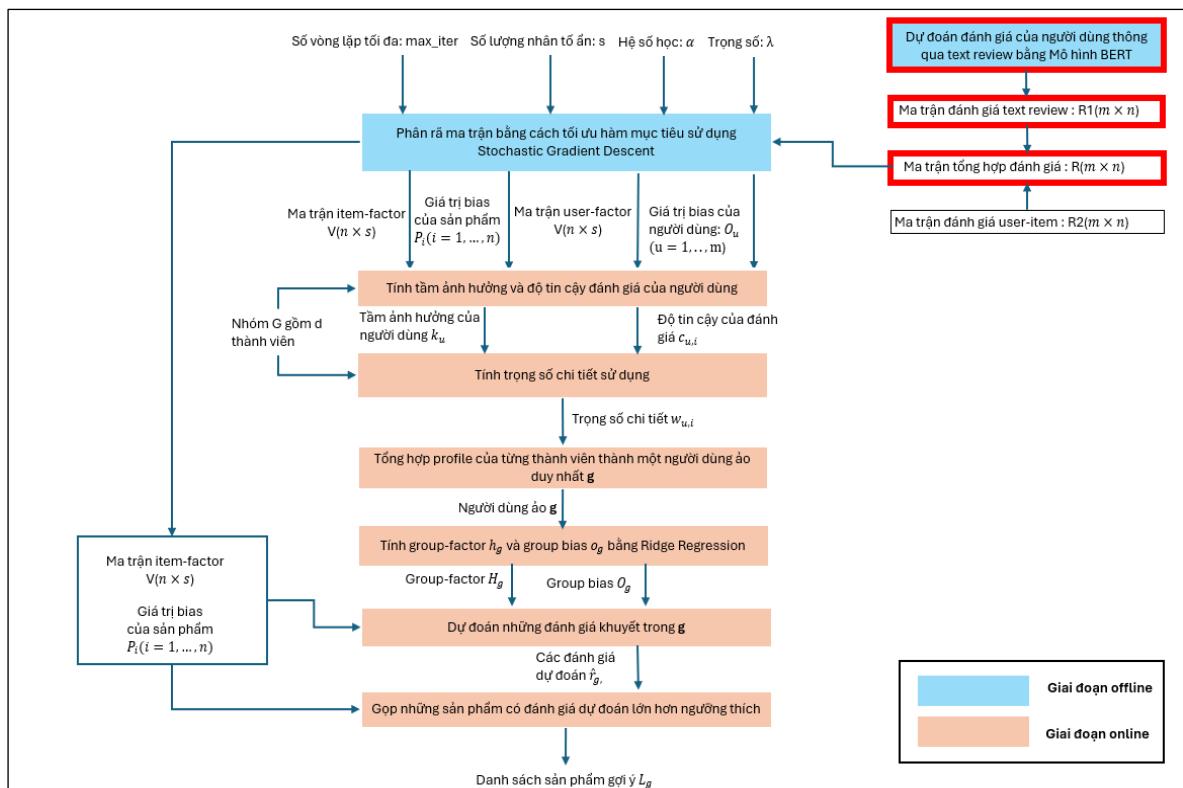
Mô hình sau khi đã tinh chỉnh được đánh giá trên trên tập dữ liệu kiểm thử và cho ra độ chính xác là 68,516%

3.3 Hệ thống được tích hợp cải tiến

Mặc dù các đánh giá được quan sát từ người dùng là nguồn thông tin hữu ích cho mô hình gợi ý trong việc tìm hiểu sở thích của người dùng. Tuy nhiên, không phải tất cả người dùng đều đưa ra đánh giá thật sự khách quan và phù hợp với ý kiến của họ vì ảnh hưởng của nhiều yếu tố như cảm xúc, thiên kiến,... Hiện nay các đánh giá của người dùng được thể hiện qua văn bản (text review) có thể mang lại hiệu quả tích cực

cho hệ thống khi nó cung cấp các nhận xét cụ thể trên nhiều khía cạnh khác nhau của sản phẩm mà không thể nhận biết thông qua các đánh giá đơn thuần từ việc đánh giá xếp hạng từ 1-5.

Do đó, nhóm đề xuất cải tiến mô hình tư vấn nhóm bằng cách sử dụng mô hình nhân tố ẩn dựa trên phương pháp tổng hợp profile được trình bày tại mục 2.4 với chiến lược tổng hợp được trình bày tại mục 3.1, đồng thời tích hợp thêm các điểm đánh giá được trích xuất thông qua nhận xét người dùng (text review) bằng mô hình BERT được trình bày tại mục 3.2 bằng cách tính trung bình đánh giá của 2 ma trận đầu vào tức ma trận đánh giá user-item và ma trận “text-review” user-item. Dưới đây là hình ảnh minh họa quy trình triển khai mô hình cải tiến.



Hình 3.5: Quy trình triển khai mô hình cải tiến dựa trên chiến lược tổng hợp profile và sử dụng đánh giá của BERT

Chương 4

Thực nghiệm

4.1 Thiết lập thực nghiệm

Trong khóa luận này chúng em tiến hành thực nghiệm trên mô hình áp dụng chiến lược tổng hợp profile dựa trên tất cả đánh giá quan sát được tích hợp trọng số chi tiết (mục 3.1) và mô hình được đề xuất (mục 3.3). Chúng em sử dụng ngôn ngữ Python để cài đặt và thực thi những mô hình trên.

Hầu hết các hệ thống gợi ý nhóm đều được đánh giá bằng các tập dữ liệu của hệ thống gợi ý cho người dùng đơn lẻ. Vì dữ liệu nhóm người dùng chưa được công bố, nên để đánh giá hệ thống tư vấn nhóm chúng ta cần giả lập nhóm trước khi thực hiện các thực nghiệm trên hệ thống gợi ý nhóm. Trong khóa luận này nhóm chúng em sẽ tiến hành đánh giá cả hai mô hình gợi ý trên 1000 nhóm được tạo ngẫu nhiên với từng kích thước nhóm là 2, 3 và 4 thành viên. Nhằm đánh giá các hệ thống gợi ý nhóm trong mọi tình huống, từ điều kiện tốt nhất đến điều kiện tệ nhất về kích thước lẫn mâu thuẫn sở thích giữa các thành viên trong nhóm

Để đánh giá độ chính xác của một tập sản phẩm (item set) được đề xuất cho một nhóm, ta cần xác định tập sản phẩm mong đợi T của nhóm. Vì không có đánh giá nào được thu thập từ nhóm trong tập dữ liệu, nên tập mong đợi T này cũng cần được giả lập. Cụ thể, một sản phẩm i được gợi ý cho nhóm G nếu sản phẩm i chưa được tất cả thành viên trong nhóm trải nghiệm (các đánh giá cho training của tất cả các thành viên nhóm đều bị khuyết : $\forall u \in G | r_{u,i} = *$) và được thích bởi tất cả thành viên (các đánh giá cho testing của các thành viên trong nhóm đều lớn hơn ngưỡng thích: $\forall u \in G | r_{u,i} > \delta$) chính là sản phẩm mà cả nhóm thực sự mong đợi (danh sách mong đợi T)

Liên quan đến quá trình xấp xỉ ma trận bằng Stochastic Gradient Descent. Chúng em sử dụng bộ tham số của Simon Funk [41] đặt ra cho hệ thống của anh ấy và đạt giải thưởng cao trong cuộc thi Netflix Challenge. Bộ tham số của Simon được trình bày trong bảng sau:

Số lượng nhân tố ẩn (s)	60
Hệ số học (α)	0.001
Trọng số cho Regularization (λ)	0.02
Nguồn lỗi (θ)	10^{-6}
Số lần lặp tối đa (max_iter)	200

4.2 Bộ dữ liệu

Bộ dữ liệu nhóm sẽ sử dụng là bộ dữ liệu từ [Amazon dataset 2018](#) [42] chứa các đánh giá thu thập từ người dùng, bên cạnh điểm đánh giá cho sản phẩm thì mỗi đánh giá trong bộ dữ liệu còn đi kèm với các thông tin như hình ảnh, nhận xét văn bản(text review), ... Kích thước của 2 tập dữ liệu được sử dụng để đánh giá được trình bày dưới đây:

- Digital Music (phiên bản 5-core) chứa 169.781 đánh giá với 16.566 người dùng và 11.797 sản phẩm mà mỗi người dùng, sản phẩm trong tập đều có ít nhất 5 đánh giá.
- Musical Instrument (phiên bản 5-core) chứa 231.392 đánh giá với 27.530 người dùng và 10.620 sản phẩm mà mỗi người dùng, sản phẩm trong tập đều có ít nhất 5 đánh giá.

4.3 Độ đo

Chúng em sử dụng độ đo F-Score để đánh giá hiệu suất gợi ý cho nhóm. F-score đánh giá độ chính xác của các sản phẩm được đề xuất cho nhóm. Do đó nó là sự kết hợp giữa độ chính xác (precision) và độ bao phủ (recall) của một danh sách gợi ý C so với một danh sách mong đợi T như sau:

$$\text{precision}_c = \frac{|T \cap C|}{|C|} \quad \text{recall}_c = \frac{|T \cap C|}{|T|}$$
$$F_{score} = \frac{2 \times \text{precision}_c \times \text{recall}_c}{\text{precision}_c + \text{recall}_c}$$

Trong đó tập mong đợi của nhóm được phát sinh dựa trên điều kiện đã được đề cập ở mục 4.1. Tuy nhiên, điều này quá nghiêm ngặt dẫn đến tập T gần như không thể tồn tại, trong khi kích thước của tập hợp T cho mỗi nhóm yêu cầu phải lớn hơn 2. Do đó, chúng em đã sử dụng nhiều đánh giá cho testing hơn bình thường. Cụ thể, tỷ lệ giữa số lượng đánh giá cho training và số lượng đánh giá cho testing là 50:50.

Ngoài ra, đối với các nhóm có kích thước 3 và 4 thành viên, chúng em đã nới lỏng các tiêu chí trên để mô phỏng thành công tập hợp item mong đợi T của nhóm. Cụ thể là một sản phẩm được nhóm mong đợi nếu trên 50% thành viên trong nhóm thích thay vì 100% ở nhóm 2 thành viên.

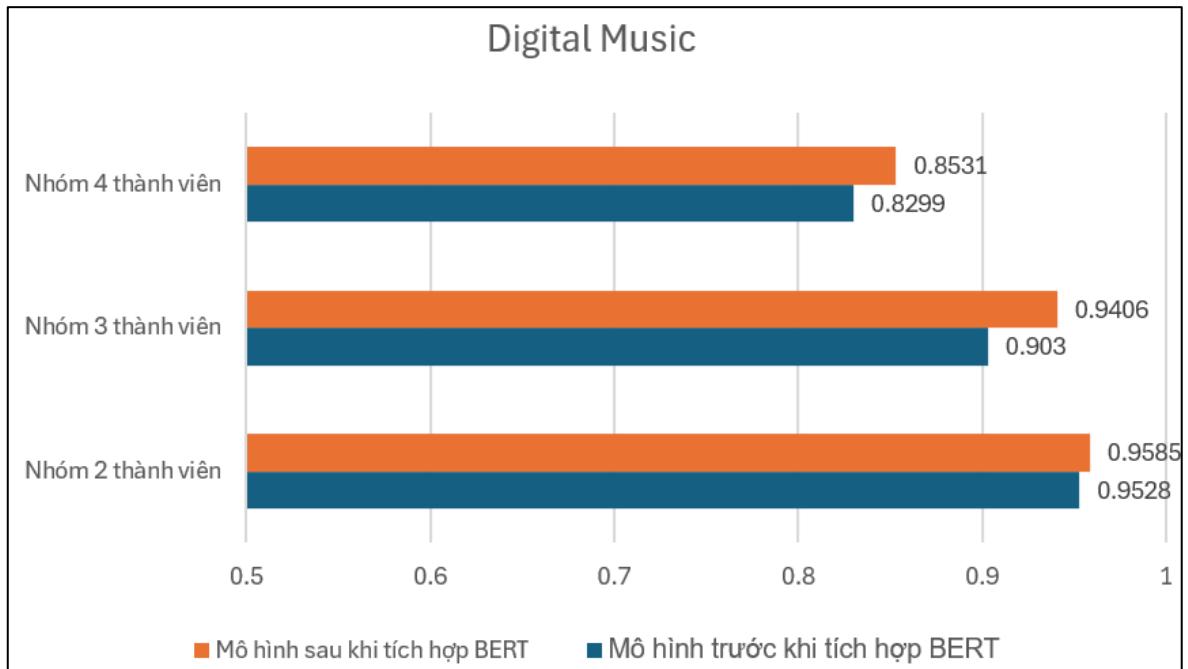
4.4 Kết quả thực nghiệm

Bộ dữ liệu 1

Với bộ dữ liệu Digital Music (phiên bản 5-core, 2018) chứa 169.781 đánh giá với 16.566 người dùng và 11.797 sản phẩm. Ta chia thành các tập ngẫu nhiên 1000 nhóm người dùng với mỗi nhóm gồm 2 thành viên, 3 thành viên và 4 thành viên. Kết quả thực nghiệm cho ra F-score như sau:

	Nhóm 2 thành viên	Nhóm 3 thành viên	Nhóm 4 thành viên
Mô hình trước khi tích hợp BERT	0.9528	0.9030	0.8299
Mô hình sau khi tích hợp BERT	0.9585	0.9406	0.8531

Bảng 4.1: Kết quả thực nghiệm trên bộ dữ liệu Digital Music



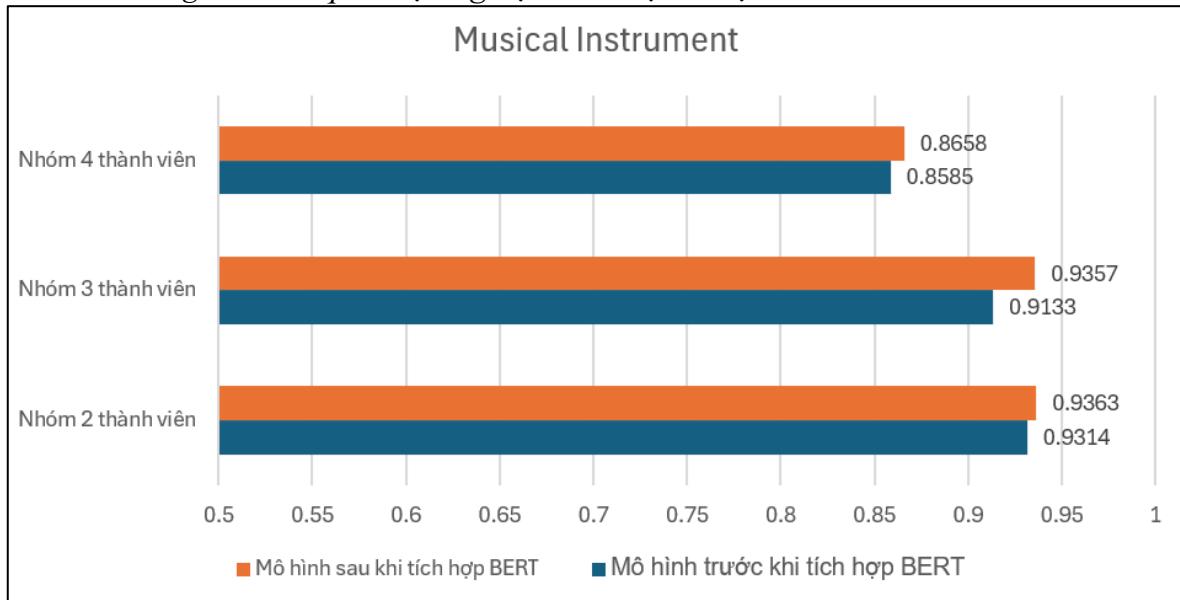
Hình 4.1: Kết quả F-score trên bộ dữ liệu Digital Music

Bộ dữ liệu 2

Với bộ dữ liệu Musical Instrument (phiên bản 5-core, 2018) chứa 231.392 đánh giá với 27.530 người dùng và 10.620 sản phẩm. Ta chia thành các tập ngẫu nhiên 1000 nhóm người dùng với mỗi nhóm gồm 2 thành viên, 3 thành viên và 4 thành viên. Kết quả thực nghiệm cho ra F-score như sau:

	Nhóm 2 thành viên	Nhóm 3 thành viên	Nhóm 4 thành viên
Mô hình trước khi tích hợp BERT	0.9314	0.9133	0.8585
Mô hình sau khi tích hợp BERT	0.9363	0.9357	0.8658

Bảng 4.2: Kết quả thực nghiệm trên bộ dữ liệu Musical Instrument



Hình 4.2: Kết quả F-score trên bộ dữ liệu Musical Instrument

Qua kết quả thực nghiệm trên chúng em đưa ra những nhận xét sau:

- Trên 2 bộ dữ liệu ta thấy rằng với nhóm 2 người dùng cho ra kết quả tốt nhất và giá trị F-score trên 2 bộ dữ liệu đều cho ra kết quả tốt > 0.85
- Mặc dù mức cải thiện không lớn, nhưng vẫn cho thấy rằng BERT đã giúp hệ thống gợi ý hiệu quả hơn khi đối mặt với nhóm có quy mô lớn và sự đa dạng trong sở thích giữa các thành viên trong nhóm

Chương 5

Kết luận và hướng phát triển

5.1 Kết quả đạt được

5.1.1 Kết quả đạt được của cá nhân

Sau khi hoàn thành những thành quả mà em đạt được là:

Về mặt lý thuyết:

- Hiểu được mô hình nhân tố ẩn
- Hiểu được các chiến lược tư vấn nhóm người dùng
- Hiểu được cách ứng dụng mô hình nhân tố ẩn vào hệ thống tư vấn nhóm
- Hiểu được cách huấn luyện và sử dụng mô hình ngôn ngữ lớn trong bài toán phân loại văn bản
- Hiểu được cách đánh giá một hệ thống tư vấn

Về mặt cài đặt:

- Cài đặt và thực nghiệm một hệ thống tư vấn nhóm bằng mô hình nhân tố ẩn với thư viện Numpy và Pandas của Python
- Cài đặt và sử dụng các thư viện phổ biến trong lĩnh vực deep learning như Transformer và TensorFlow để tinh chỉnh mô hình BERT

5.1.2 Kết quả đạt được của khóa luận

Sau khi kết thúc khóa luận đã được các kết quả sau:

- Hoàn thành mục tiêu đề ra ban đầu của khóa luận
- Kết quả thực nghiệm thu được cho ra kết quả gợi ý khá ổn

Về mặt hạn chế, khóa luận vẫn chưa đạt được một số vấn đề sau:

- Chỉ thực nghiệm mô hình tư vấn nhóm theo hệ số của Simon, tốt nhất ta vẫn nên thực nghiệm với nhiều bộ hệ số khác để tìm ra bộ hệ số phù hợp nhất. Tuy nhiên do thời gian và yếu tố phần cứng nên chúng em đã bỏ qua bước này và sử dụng hệ số có sẵn
- Nên thực nghiệm với những bộ dữ liệu lớn hơn và ít thưa hơn
- Thực nghiệm chỉ sử dụng một độ đo để đánh giá mô hình nên ta vẫn nên sử dụng thêm những độ đo khác để có thể so sánh tốt nhất giữa các mô hình

5.2 Kết luận mô hình thực nghiệm

Có thể nói rằng từ kết quả thực nghiệm cho thấy khi tích hợp thêm nhận xét của người dùng (text review) bằng mô hình BERT bên cạnh điểm đánh giá từ người dùng vào mô hình nhân tố ẩn cho nhóm đã mang lại kết quả gợi ý tốt hơn so với mô hình nhân tố ẩn cho nhóm chỉ dựa trên điểm đánh giá từ người dùng đã trải nghiệm. Do đó ta có thể mở rộng để nghiên cứu theo hướng này, cụ thể là khi áp dụng mô hình nhân tố ẩn cho nhóm thì không chỉ nên quan tâm cách tổng hợp profile nhóm mà còn phải quan tâm đến cách kết hợp nhận xét của người dùng (text review) và điểm đánh giá của người dùng để cải thiện khả năng hiểu sâu hơn về sở thích và nhu cầu của nhóm từ đó đưa ra gợi ý phù hợp hơn.

5.3 Hướng phát triển

Dù đạt được một số kết quả tích cực qua thực nghiệm, việc hạn chế về khả năng đa dạng các loại ngôn ngữ của mô hình dự đoán đánh giá thông qua văn bản có thể là một điểm trừ đối với hệ thống. Giải pháp tích hợp các mô hình ngôn ngữ lớn được huấn luyện để xử lý đa ngôn ngữ nên được xem xét.

Bên cạnh đó nên dành thời gian để chạy thực nghiệm hệ thống trên nhiều bộ tham số khác nhau để xác định được bộ tham số phù hợp nhất thay vì dùng bộ tham số có sẵn của Simon

Ngoài ra, nhóm thực hiện chỉ phát sinh các nhóm người dùng ngẫu nhiên và chưa có giải pháp để trích xuất các nhóm người dùng tiềm năng trong hệ thống. Điều này làm tăng đáng kể thời gian chạy và phản hồi của hệ thống vì hầu hết các bước trong quy trình đề xuất gợi ý của hệ thống đều chạy ở giai đoạn online. Do đó các giải pháp để phát sinh nhóm dựa trên lảng giềng hay xem xét độ tương đồng cũng cần được áp dụng để tối ưu quá trình phát sinh nhóm cũng như kết quả đầu ra được cải thiện hơn.

Tài liệu tham khảo

- [1] Le Nguyen Hoai Nam, Ho Thi Hoang Vy, Le Hoang My, Le Thi Tuyet Mai, Hong Tiet Gia, and Ho Le Thi Kim Nhung. 2019. An approach to improving group recommendation systems based on latent factor matrices. In Proceedings of the 10th International Symposium on Information and Communication Technology (SoICT '19). Association for Computing Machinery, New York, NY, USA, 98–105
- [2] Le Nguyen Hoai Nam. 2021. Towards comprehensive profile aggregation methods for group recommendation based on the latent factor model. Expert Syst. Appl. 185, C (Dec 2021).
- [3] TS Lê Nguyễn Hoài Nam, Nguyễn Thị Mỹ Tiên - 1612687.Nghiên cứu mô hình nhân tố ẩn cho bài toán tư vấn cho nhóm người dùng
- [4] Charu C. Aggarwal.Recommender Systems: The Textbook 1st ed. 2016 Edition
- [6] Peter Knees, Julia Neidhardt, and Irina Nalis. Recommender Systems: Techniques, Effects, and Measures Toward Pluralism and Fairness
- [5] Ana Belén Barragáns-Martínez, Enrique Costa-Montenegro, Juan C. Burguillo, Marta Rey-López, Fernando A. Mikic-Fonte, Ana Peleteiro,A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition,Information Sciences,Volume 180, Issue 22,2010,Pages 4290-4311,ISSN 0020-0255
- [7] Suja Cherukullapurath Mana and T. Sasiprappa. Research on Cosine Similarity and Pearson Correlation Based Recommendation Models
- [8] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen.Collaborative Filtering Recommender Systems

- [9] Choudhury, Prasenjit. (2020). Sharing Information for "Enhancing recommendation accuracy of item-based collaborative filtering using Bhattacharyya coefficient and most similar item <https://rdcu.be/b52qp>. Applied Intelligence. 50. 1-23. 10.1007/s10489-020-01775-4.
- [10] Ms. Tejashri Sharad Phalle, Prof. Shivendu Bhushan. Content Based Filtering And Collaborative Filtering: A Comparative Study
- [11] Shengbo Guo. Bayesian Recommender Systems: Models and Algorithms
- [12] Gershman, Amir; Meisels, Amnon; Lüke, Karl-Heinz; Rokach, Lior; Schclar, Alon; Sturm, Arnon (2010): A decision tree-based recommender system. 10th International Conference On Innovative Internet Community Systems (I2CS) – Jubilee Edition 2010 –. Bonn: Gesellschaft für Informatik e.V.. PISSN: 1617-5468. ISBN: 978-3-88579-259-8. pp. 170-179. Regular Research Papers. Bangkok, Thailand. June 3-5, 2010
- [13] D. Wu, X. Luo, M. Shang, Y. He, G. Wang and M. Zhou, "A Deep Latent Factor Model for High-Dimensional and Sparse Matrices in Recommender Systems," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 51, no. 7, pp. 4285-4296, July 2021, doi: 10.1109/TSMC.2019.2931393.
- [14] Alexander Felfernig, Ludovico Boratto, Martin Stettinger, Marko Tkalčič, Group Recommender Systems: An Introduction (Signals and Communication Technology) 2nd ed. 2024 Edition
- [15] Felfernig, A., Boratto, L., Stettinger, M.& Tkalcic (2018). Algorithms for group recommendation. In Group recommender systems (pp. 27–58).
- [16] Baltrunas, L., Makcinskas, T., & Ricci, F. (2010). Group recommendations with rank aggregation and collaborative filtering. In Proceedings of the fourth ACM conference on Recommender systems (pp. 119 126).

- [17] Van Deventer, O., De Wit, J., Vanattenhoven, J., & Gualbahar, M. (2013). Group recommendation in a hybrid broadcast broadband television context. *GroupRS 2013: Group Recommender Systems: Concepts Technology, Evaluation*, 997, 12–18.
- [18] Ortega, F., Hernando, A., Bobadilla, J., & Kang, J. H. (2016). Recommending items to group of users using matrix factorization based collaborative filtering. *Information Sciences*, 345, 313–324.
- [19] Hastie T., Tibshirani R. Friedman J. “The elements of statistical learning: data mining, inference, and prediction”. In: Springer Science usiness Media. 2009
- [20] Umar Jamil, BERT explained: Training, Inference, BERT vs GPT/LLamA, Fine tuning, [CLS] token
- [21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need
- [22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
- [23] Xinying Song, Alex Salcianu, Yang Song, Dave Dopson, Denny Zhou. Fast WordPiece Tokenization
- [24] Berkovsky, S., Freyne, J., Coombe, M., & Bhandari, D. (2010b, September): Recommender algorithms in activity motivating games. In *Proceedings of the fourth ACM conference on Recommender systems* (pp. 175–182).
- [25] Delic, A., Neidhardt, J., Nguyen, T. N., & Ricci, F. (2018). An observational user study for group recommender systems in the tourism domain. *Information Technology & Tourism*, p87–116.

- [26] Kagita, V. R., Pujari, A. K., & Padmanabhan, V. (2015). Virtual user approach for group recommender systems using precedence relations. *Information Sciences*, 294, 15–30.
- [27] Seo, Y.-D., Kim, Y.-G., Lee, E., Seol, K.-S., & Baik, D.-K. (2018). An enhanced aggregation method considering deviations for a group recommendation. *Expert Systems with Applications*, 93, 299–312.
- [28] Chen, J., Wang, C., Shi, Q., Feng, Y., & Chen, C. (2019). Social recommendation based on users' attention and preference. *Neurocomputing*, p341, 1–9.
- [29] Boratto, L., & Carta, S. (2015). The rating prediction task in a group recommender system that automatically detects groups: Architectures, algorithms, and performance evaluation. *Journal of Intelligent Information Systems*, 45(2), 221–245.
- [30] Guo, L., Yin, H., Wang, Q., Cui, B., Huang, Z., & Cui, L. (2020). In April). Group recommendation with latent voting mechanism (pp. 121–132).
- [31] Manning, C. D., Raghavan, P., & Schutze, H. (2008). In *Introduction to Information Retrieval* (pp. 100–123). Cambridge: Cambridge University Press.
- [32] Bell, R. M., & Koren, Y. (2007). Lessons from the Netflix prize challenge. *Acm Sigkdd Explorations Newsletter*, 9(2), 75–79.
- [33] Koren, Y., & Bell, R. (2011). In *Recommender Systems Handbook* (pp. 145–186). Boston, MA: Springer US.
- [34] Bilal, M., Almazroi, A.A. Effectiveness of Fine-tuned BERT Model in Classification of Helpful and Unhelpful Online Customer Reviews. *Electron Commer Res* 23, 2737–2757 (2023).

- [35] Marius Mosbach, Maksym Andriushchenko, Dietrich Klakow. On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines
- [36] X. Li, X. Wang and H. Liu, "Research on fine-tuning strategy of sentiment analysis model based on BERT," 2021 International Conference on Communications, Information System and Computer Engineering (CISCE), Beijing, China, 2021, pp. 798-802.
- [37] M.P. Geetha, D. Karthika Renuka. Improving the performance of aspect-based sentiment analysis using fine-tuned Bert Base Uncased model
- [38] Ashok Kumar Durairaj and Anandan Chinnalagu, "Transformer based Contextual Model for Sentiment Analysis of Customer Reviews: A Fine-tuned BERT" International Journal of Advanced Computer Science and Applications (IJACSA), 12(11), 2021.
- [39] Karabila, Ikram & Darraz, Nossayba & El-Ansari, Anas & Alami, Nabil & El Mallahi, Mostafa. (2023). BERT-enhanced sentiment analysis for personalized e-commerce recommendations. *Multimedia Tools and Applications*. 83. 1-26.
- [40] Xiang Zhang, Junbo Zhao, Yann LeCun. Character-level Convolutional Networks for Text Classification.
- [41] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. 2008. Matrix factorization and neighbor based algorithms for the netflix prize problem. In Proceedings of the 2008 ACM conference on Recommender systems (RecSys '08). Association for Computing Machinery, New York, NY, USA, 267–274.
- [42] Jianmo Ni, Jiacheng Li, Julian McAuley. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects
- [43] Rivas-Blanco, Irene & Perez-del-Pulgar, Carlos & Garcia-Morales, Isabel & Munoz, Victor. (2021). A Review on Deep Learning in Minimally Invasive Surgery. *IEEE Access*. PP. 1-1.

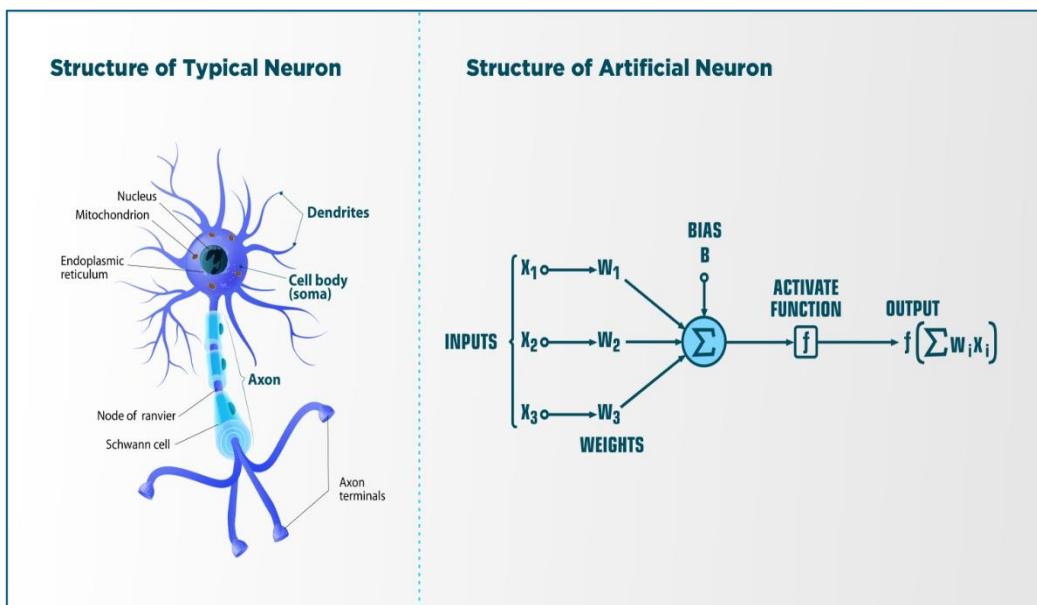
- [44] Sebastian Raschka. Build a Large Language Model (From Scratch)
- [45] Sudharsan Ravichandiran. Getting Started with Google BERT: Build and train state-of-the-art natural language processing models using BERT
- [46] Kiprono Elijah Koech, Towards Data Science. Cross-Entropy Loss Function

Phụ lục

A. Kiến thức về mạng nơ-ron trong lĩnh vực học sâu

A1. Tổng quan về mạng neural

Artificial neural networks (ANNs) - mạng nơ-ron nhân tạo thường được gọi là đơn giản là mạng nơ-ron (neural networks) (NNs) là hệ thống tính toán được lấy cảm hứng từ mạng nơ-ron sinh học của bộ não con người hay của sinh vật nói chung.



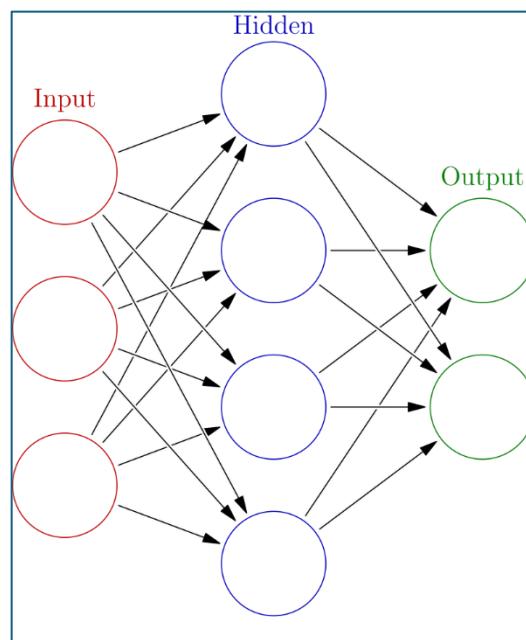
Hình A.1: Mạng nơ-ron nhân tạo và mạng nơ-ron thực tế

Một mạng nơ-ron nhân tạo được cấu tạo bởi tập hợp nút (node) được gọi là nơ-ron nhân tạo. Mỗi nơ-ron được xem như một nơ-ron trong bộ não sinh học, đầu vào của nơ-ron có thể là dữ liệu đầu vào (hình ảnh hoặc văn bản) hoặc là đầu ra của các nơ-ron khác. Những kết nối giữa các nút (node) hay được gọi là cạnh được xem giống như các khớp thần kinh trong bộ não sinh học. Giữa các nơ-ron và cạnh là các trọng số khác nhau thể hiện tầm quan trọng của từng kết nối giữa các nơ-ron, giúp điều chỉnh quá trình học.

Một nơ-ron nhân tạo sẽ nhận tín hiệu từ nơ-ron phía trước nó thông qua khớp thần kinh. Tín hiệu trong mạng nơ-ron là một số thực. Đầu ra của nơ-ron cuối cùng là kết

quả cuối cùng của mạng nơ-ron cho những tác vụ mà mạng đã được huấn luyện vd: nhận diện khuôn mặt, nhận diện văn bản, ...

Các nơ-ron thường được phân thành các nhóm hay được gọi là lớp (layer) để thực hiện các biến đổi khác nhau dựa trên đầu vào nhận được. Tín hiệu được truyền từ lớp đầu tiên (input layer) đến lớp cuối cùng (output layer), trong suốt quá trình lan truyền đó thì tín hiệu sẽ đi qua các lớp ở giữa nếu có được gọi là lớp ẩn (hidden layer).



Hình A.2: Các lớp trong mạng nơ-ron

A2. Huấn luyện mạng nơ-ron

Mạng nơ-ron sẽ học hay còn được gọi là training bằng tập hợp các ví dụ. Mỗi ví dụ bao gồm một đầu vào và đầu ra mong muốn (expected output)

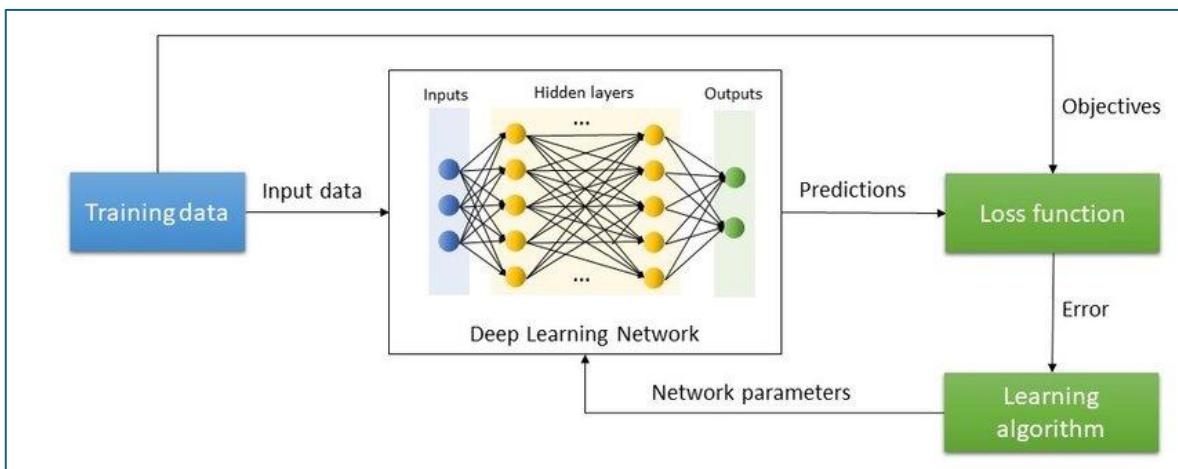
INPUT	OUTPUT
	1
	0
	1
	0

Hình A.3: Ví dụ tập ví dụ được sử dụng cho huấn luyện mạng nơ-ron

Quá trình huấn luyện (training) là tập hợp các bước lặp đi lặp lại với mỗi bước gồm các giai đoạn sau:

- Mạng nơ-ron nhân tạo sẽ tính toán đầu ra (propagation) dựa theo đầu vào trong tập ví dụ
- Tính toán sự khác biệt hay còn được gọi là độ lỗi (error) giữa từng đầu ra của mạng với từng đầu ra mục tiêu (target output)
- Điều chỉnh lại các tham số của mạng (backpropagation) để cải thiện độ chính xác của dự đoán ở các bước tiếp theo

Quá trình huấn luyện sẽ diễn ra liên tục đến khi độ lỗi (error) trên toàn bộ tập ví dụ huấn luyện được gọi là loss đạt đến giá trị nhỏ nhất có thể. Đó chính là mục đích của quá trình huấn luyện (training). Hình thức này còn được gọi là học có giám sát (supervised learning). Những hệ thống như mạng nơ-ron là một hệ thống học từ ví dụ chứ không phải là được lập trình theo quy tắc cụ thể như những câu lệnh if-else



Hình A.4: Minh họa quá trình huấn luyện mạng nơ-ron

Giá trị cuối cùng của các tham số trong mạng nơ-ron sau quá trình học chính là tri thức mà mạng đã học được sau quá trình training. Những giá trị đó giúp mô hình có thể tự động nhận diện được các đặc trưng của đối tượng đã học. Ngay cả khi quá trình huấn luyện đã kết thúc thì độ lỗi (error rate) không thể nào giảm về giá trị 0 hoàn chỉnh. Điều đó có nghĩa là những đầu ra thực tế (actual output) không thể giống 100% với đầu ra mục tiêu (target output). Dù mạng nơ-ron được thiết kế phức tạp đến đâu.

Nếu độ độ lỗi (error rate) quá cao ví dụ như trên 50% thì chúng ta là mạng neural chúng ta sử dụng chưa phù hợp với bài toán hoặc cần training thêm cho đến khi độ lỗi giảm đến giá trị mà chúng ta mong muốn ví dụ như 30%

INPUT	OUTPUT	ACTUAL
	1	0.55
	0	0.48
	1	0.26
	0	0.67

Hình A.5: Ví dụ kết quả dự đoán thực tế của mô hình sau khi kết thúc quá trình huấn luyện

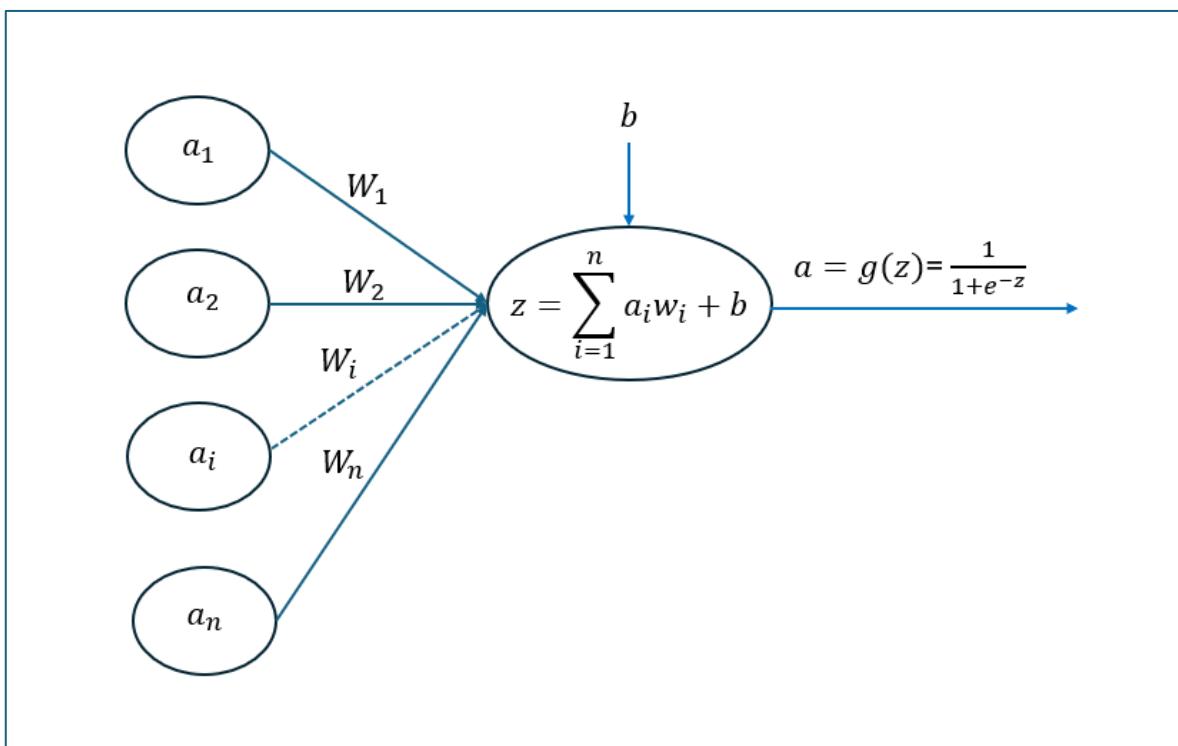
A3. Tính toán đầu ra của nơ-ron được diễn ra như thế nào

Giả sử chúng ta có một nơ-ron đang nhận n đầu vào $a_1 \rightarrow a_n$ từ n nơ-ron trước đó và trên các cạnh kết nối giữa nơ-ron đang xét với các nơ-ron trước đó sẽ có các trọng số lần lượt là $W_1 \rightarrow W_n$

Đầu tiên nơ-ron sẽ tính tổng $z = \sum_{i=0}^n a_i w_i + b$ với b là bias của nơ-ron. Sau đó z sẽ được chuyển vào một hàm kích hoạt (activation function) ký hiệu là $a = g(z)$ trong đó g là hàm kích hoạt. Hàm kích hoạt là một bước quan trọng trong việc tính toán đầu

ra của một nơ-ron. Nó có vai trò là giúp mạng nơ-ron giải quyết các bài toán phi tuyến tính và nếu không có hàm kích hoạt thì các tính toán trên toàn mạng nơ-ron sẽ đều là tuyến tính dẫn đến mạng chỉ giải quyết được bài toán tuyến tính. Nhưng trên thực tế có rất nhiều bài toán phi tuyến tính vì thế nên hàm kích hoạt thường là phi tuyến như trong ví dụ này là hàm sigmod $g(z) = \frac{1}{1+e^{-z}}$

Mỗi đầu ra của một nơ-ron này sẽ được sử dụng làm đầu vào cho các nơ-ron tiếp theo, lớp (layer) tiếp theo. Các trọng số w và bias b chính là các tham số của mạng nơ-ron. Và chính trong quá trình học thì các tham số này sẽ được điều chỉnh để đầu ra đưa ra gần với đầu ra mong muốn nhất



Hình A.6: Minh họa quy trình tính toán đầu ra của một nơ-ron

A4. Lan truyền, lan truyền ngược, hàm chi phí và hàm mất mát

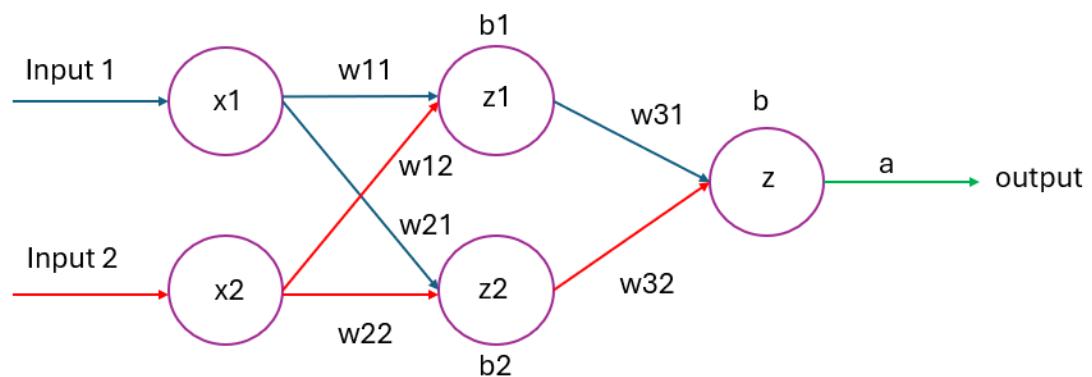
Giả sử chúng ta cần thiết kế một mạng nơ-ron để thực hiện toán tử logic XOR với đầu vào là 2 giá trị nhị phân và đầu ra là giá trị phép toán XOR giữa chúng. Ở đây chúng ta có 4 ví dụ để huấn luyện vì chúng ta chỉ có 4 trường hợp khả dĩ.

Input 1	Input 2	Output
0	0	0
0	1	1
1	1	0
1	0	1

Bảng A.1: Tập ví dụ huấn luyện cho bài toán

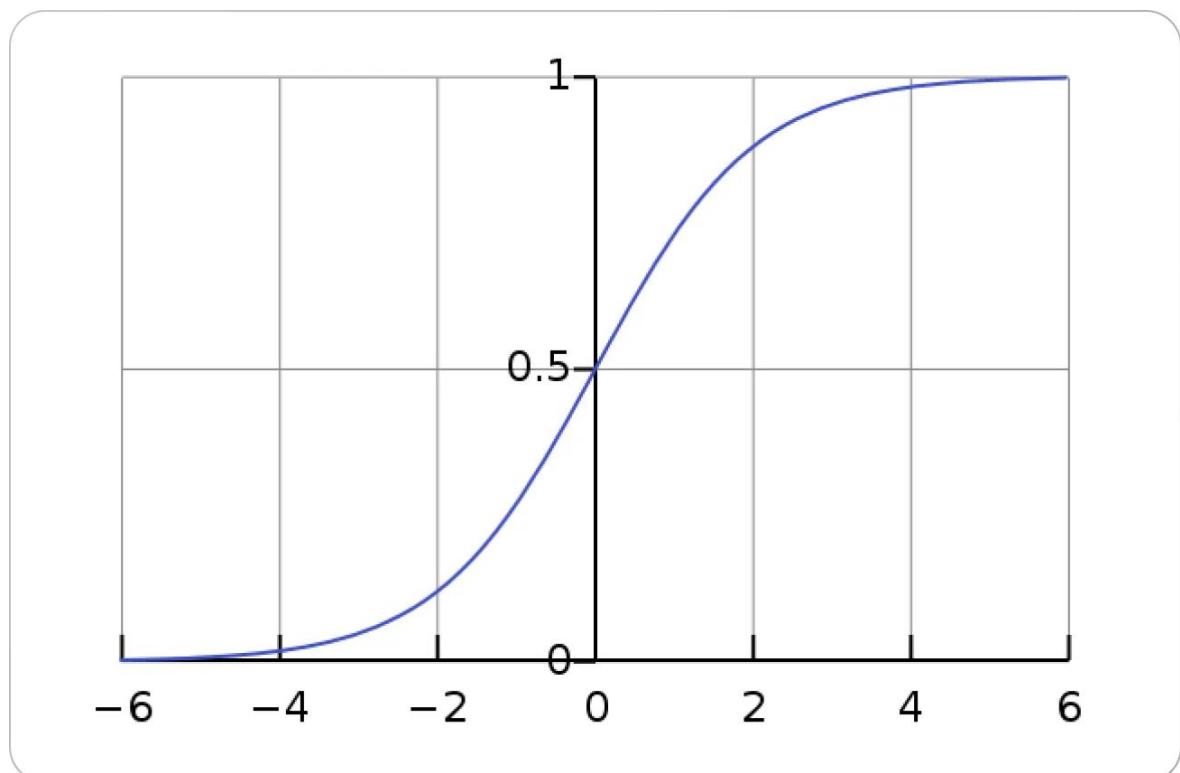
Dựa trên đề bài chúng ta có một mạng nơ-ron đơn giản với các thông tin như sau:

- 3 lớp (Layer): Lớp đầu vào (input layer) (2 nơron), lớp đầu ra (output layer) (1 nơron) và 1 lớp ẩn (hidden layer) (2 nơron)
- Bộ tham số (giá trị của bộ tham số thường được khởi tạo ngẫu nhiên)
 - Trọng số (weight): $w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}$
 - Bias: b_1, b_2, b
- Đầu vào/Đầu ra:
 - Đầu vào: x_1, x_2
 - Đầu ra: a



Hình A.7: Kiến trúc mạng nơ-ron để thực hiện toán tử XOR

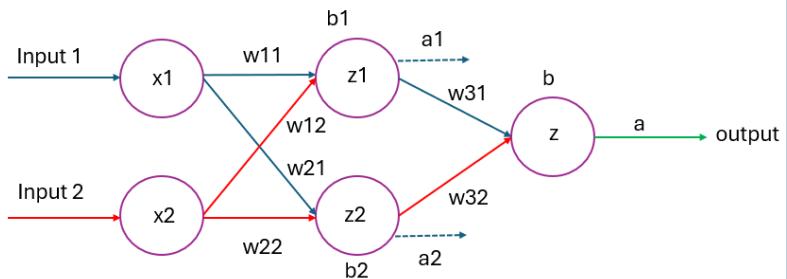
Hàm kích hoạt mà chúng ta sẽ dùng trong ví dụ này vẫn sẽ là hàm sigmoid. Chúng ta có thể thấy hàm số sigmoid trong hình.



Hình A.8: Đồ thị hàm sigmoid

Hàm sigmoid có miền giá trị từ 0 đến 1. Hàm sigmoid này có độ dốc lớn nhất hay đạo hàm lớn nhất khi giá trị z bằng 0 và đạo hàm nhỏ nhất khi z tiến về dương vô cực và âm vô cực.

$$\begin{aligned}
 & \Delta z_1 = w_{11} \times x_1 + w_{12} \times x_2 + b_1 \\
 & \Delta a_1 = \sigma(z_1) = \frac{1}{1 + e^{-z_1}} \\
 & \Delta z_2 = w_{21} \times x_1 + w_{22} \times x_2 + b_2 \\
 & \Delta z = w_{31} \times a_1 + w_{32} \times a_2 + b \\
 & \Delta a_2 = \sigma(z_2) = \frac{1}{1 + e^{-z_2}} \\
 & \Delta a = \sigma(z) = \frac{1}{1 + e^{-z}}
 \end{aligned}$$



Hình A.9: Mô tả quá trình lan truyền

Lan truyền (Propagation) là quá trình tính toán đầu ra của đầu vào thông qua mạng nơ-ron. Quá trình này sẽ diễn ra từ lớp đầu vào (input layer) đến lớp đầu ra (output layer). Giá trị đầu ra cuối cùng sẽ phụ thuộc vào đầu vào lẫn giá trị của các tham số trên mạng

Giả sử với ví dụ huấn luyện đầu tiên là $x_1=1$, $x_2=0$ thì ta có giá trị $a = 0.45$ trong khi đầu ra mong muốn ở đây là $y=1$ lúc này ta cần có một hàm chi phí (cost function) để xác định a và y khác nhau như thế nào. Chúng ta có thể sử dụng hàm chi phí(cost function) đơn giản như sau:

$$L(a, y) = |a - y|$$

Tuy nhiên hàm ở trên lại không có lợi cho ta ở bài toán này. Do đó ta có hàm chi phí (cost function) như sau:

$$L(a, y) = -(y \times \log(a) + (1 - y) \times \log(1 - a))$$

Hàm này tuy nhìn tính toán rất phức tạp so với hàm trước đó nhưng nó giúp ta đơn giản tính toán hơn ở những trường hợp đặc biệt sau:

- Nếu $y=1$:
 - $L(a, y) = -\log(a)$
 - Để giá trị của L giảm thì ta cần điều chỉnh tăng giá trị của a
- Nếu $y = 0$:
 - $L(a, y) = -\log(1 - a)$

- Để giá trị của L giảm thì ta cần điều chỉnh giảm giá trị của a

Chính nhờ hàm chi phí (cost function) này mà ta sẽ có các định hướng rõ ràng trong việc điều chỉnh các tham số trên mạng nơ-ron để thay đổi giá trị của a. Lúc này hàm chi phí (cost function) sẽ không phụ thuộc vào a và y nữa mà sẽ phụ thuộc vào các trọng số weight và bias của mạng

Nếu như hàm chi phí (cost function) $L(a, y)$ là tính độ khác nhau giữa a và y trên một ví dụ huấn luyện thì hàm mất mát (Loss function) $J(a, y)$ là hàm tính trung bình cộng của tất cả các hàm chi phí (cost function) trên toàn bộ tập dữ liệu huấn luyện:

$$J(a, y) = \frac{1}{m} \sum_{i=1}^m L(a_i y_i)$$

Vì là tính trung bình cộng của hàm chi phí (Cost Function) $L(a, y)$ nên hàm mất mát (Loss Function) $J(a, y)$ cũng phụ thuộc vào weight and bias bên cạnh đó hàm mất mát (Loss Function) $J(a, y)$ còn được gọi là hàm mất mát entropy chéo (cross-entropy loss function)

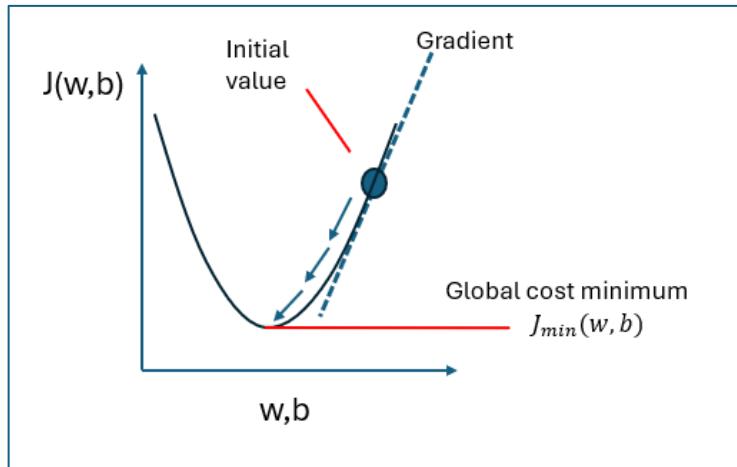
Sau lần lan truyền (propagation) đầu tiên ta có bảng kết quả như sau:

x1	x2	w11	w12	b1	w21	w22	b2	z1	a1	z2	a2	w31	w32	b	z	a	y	cost
0	0	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.37	0.59	-0.98	0.27	0.41	-0.9	0.35	0.35	0.59	0	0.8816
0	1	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.3	0.57	-0.35	0.41	0.41	-0.9	0.35	0.21	0.55	1	0.5921
1	0	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.53	0.63	-1.64	0.16	0.41	-0.9	0.35	0.46	0.61	1	0.4886
1	1	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.46	0.61	-1.01	0.27	0.41	-0.9	0.35	0.36	0.59	0	0.8899
																		Loss 0.713

Hình A.10: Kết quả của quá trình lan truyền đầu tiên

Mục tiêu chính của quá trình huấn luyện là điều chỉnh giá trị của các tham số qua mỗi bước lặp sao cho giá trị loss qua mỗi bước giảm đến một mức tối ưu nhất có thể. Quá trình được diễn ra trong giai đoạn lan truyền ngược (backpropagation)

Vì hàm mất mát (Loss function) phụ thuộc vào giá trị của bộ tham số trong mạng nơ-ron nên quan hệ giữa hàm mất mát và các tham số trong mạng có thể biểu diễn bằng đồ thị sau:



*Hình A.11: Đồ thị biểu diễn mối quan hệ
giữa hàm mất mát và các tham số trong mạng*

Như ta đã biết là trong một hàm số giá trị đạo hàm tại một điểm hay được gọi là gradient là tiếp tuyến của hàm số tại điểm đó và có ý nghĩa thể hiện xu hướng và tốc độ tăng giảm giá trị của hàm số. Trong đồ thị trên, để hàm số $J(w,b)$ có thể đạt giá cực trị thì hàm số ta cần điều chỉnh giá trị của các tham số w và b trong hàm số theo chiều ngược lại với đạo hàm của hàm mất mát $J(w,b)$. Việc cập nhật này được gọi là Gradient Descent

Trong giai đoạn lan truyền ngược (Backpropagation), thuật toán tối ưu Gradient Descent sẽ được sử dụng để cập nhật lại giá trị của từng tham số trong mạng nơ-ron theo công thức như sau:

$$w := w - \alpha \times \delta w$$

$$b := b - \alpha \times \delta b$$

Trong đó:

- w là các trọng số trong mạng ($w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}$)
- b là các bias trong mạng (b_1, b_2)
- α là tốc độ học

- δb và δw là đạo hàm từng phần (partial derivatives) của hàm mất mát (loss function) theo từng tham số

$$\delta b = \frac{dJ}{db} = \frac{1}{m} \sum_{i=1}^m \frac{dL(a_i y_i)}{db}$$

$$\delta w = \frac{dJ}{dw} = \frac{1}{m} \sum_{i=1}^m \frac{dL(a_i y_i)}{dw}$$

Để đơn giản hóa quá trình tính đạo hàm của hàm mất mát (loss function) theo từng giá trị weight và bias, quy tắc chuỗi (The chain rule) trong đạo hàm được sử dụng với công thức như sau:

Với hàm số $y = f(u)$ và $u = g(x)$. Khi đó y là hàm hợp của x thông qua u , tức là $y = f(g(x))$. Quy tắc chuỗi cho phép ta tính đạo hàm của y theo x :

$$\frac{dy}{dx} = \frac{dy}{du} \times \frac{du}{dx}$$

Dựa trên công thức được đề cập trong giai đoạn lan truyền (propagation):

- $L(a, y) = -(y \times \log(a) + (1 - y) \times \log(1 - a))$
- $a = \sigma(z) = \frac{1}{1+e^{-z}}$
- $z = w31 \times a_1 + w32 \times a_2 + b$

Đạo hàm của hàm mất mát (loss function) theo 3 tham số $w31$, $w32$ và b lần lượt được tính bằng quy tắc chuỗi như sau:

$$\begin{aligned}\delta w31 &= \frac{dj}{dw32} = \frac{dL}{dw31} = \frac{dL}{da} \times \frac{da}{dz} \times \frac{dz}{dw31} \\ &= -\left(\frac{y}{a} - \frac{1-y}{1-a}\right) \times a(1-a) \times a_1 \\ &= (a - y) \times a_1\end{aligned}$$

$$\delta w32 = \frac{dj}{dw32} = \frac{dL}{dw32} = \frac{dL}{da} \times \frac{da}{dz} \times \frac{dz}{dw32}$$

$$= -\left(\frac{y}{a} - \frac{1-y}{1-a}\right) \times a(1-a) \times a_2$$

$$= (a - y) \times a_2$$

$$\delta wb = \frac{dJ}{db} = \frac{dL}{db} = \frac{dL}{da} \times \frac{da}{dz} \times \frac{dz}{db}$$

$$= -\left(\frac{y}{a} - \frac{1-y}{1-a}\right) \times a(1-a) \times 1$$

$$= a - y$$

Áp dụng tương tự với những tham số còn lại w11, w12, w21, w22, b1 và b2. Ta sẽ tính được đạo hàm của hàm mất mát (loss function) theo từng tham số trong mạng nơ-ron dùng để cập nhật lại giá trị của từng tham số như sau:

$$w11 := w11 - \alpha \times \delta w11$$

$$w12 := w12 - \alpha \times \delta w12$$

$$w21 := w21 - \alpha \times \delta w21$$

$$w31 := w31 - \alpha \times \delta w31$$

$$w32 := w32 - \alpha \times \delta w32$$

$$b1 := b1 - \alpha \times \delta b1$$

$$b2 := b2 - \alpha \times \delta b2$$

$$b := b - \alpha \times \delta b$$

Với bộ dữ liệu gồm 4 ví dụ huấn luyện ban đầu với tham số $\alpha = 0.1$, qua nhiều bước lặp các giai đoạn lan truyền (propagation) và lan truyền ngược (backpropagation) chúng ta có thể quan sát được rằng giá trị loss thay đổi như sau:

Step	w11	w12	b1	w21	w22	b2	w31	w32	b	loss
Initial	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.41	-0.9	0.35	0.713
#1	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.4	-0.9	0.34	0.712
#2	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.4	-0.9	0.33	0.711
#3	0.16	-0.07	0.37	-0.66	0.63	-0.98	0.39	-0.9	0.33	0.71
#4	0.16	-0.07	0.37	-0.65	0.63	-0.97	0.39	-0.91	0.32	0.709
#5	0.16	-0.07	0.37	-0.65	0.63	-0.97	0.39	-0.91	0.31	0.708
...
#100,000	7.69	7.69	-3.55	6.34	6.34	-9.68	14.74	-15.39	-7.02	0.001

Hình A.12: Sự thay đổi của giá trị loss sau 100,000 lần lặp

Sau 100,000 lần lặp trong quá trình huấn luyện, giá trị loss đã giảm về gần bằng 0 và giá trị cuối cùng của các tham số chính là tri thức mà mạng nơ-ron đã học được.

Sau khi kết thúc quá trình huấn luyện, ta hãy thử truyền vào mạng tập đầu vào ở đê bài ban đầu và nhận được kết quả như sau:

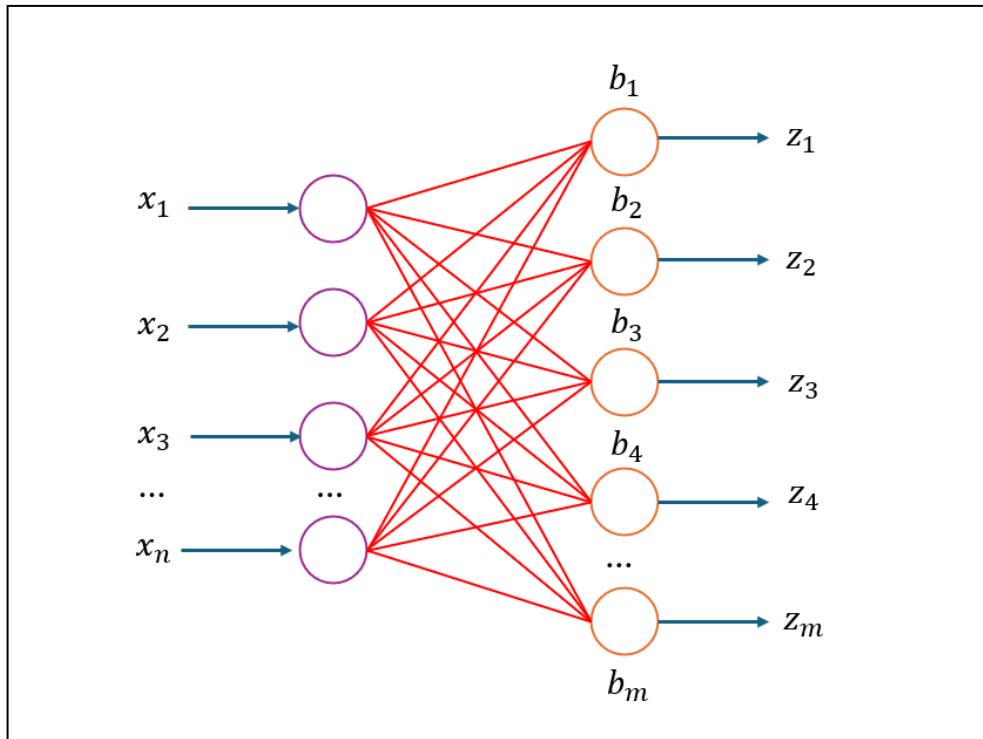
x1	x2	w11	w12	b1	w21	w22	b2	z1	a1	z2	a2	w31	w32	b	z	a	y	cost
0	0	7.96	7.96	-3.55	6.34	6.34	-9.68	-3.55	0.03	-9.68	0	14.74	-15.39	-7.02	-6.61	0.001346	0	0.0006
0	1	7.96	7.96	-3.55	6.34	6.34	-9.68	4.14	0.98	-3.34	0.03	14.74	-15.39	-7.02	6.96	0.999054	1	0.0004
1	0	7.96	7.96	-3.55	6.34	6.34	-9.68	4.14	0.98	-3.34	0.03	14.74	-15.39	-7.02	6.96	0.999054	1	0.0004
1	1	7.96	7.96	-3.55	6.34	6.34	-9.68	11.83	1	3	0.95	14.74	-15.39	-7.02	-6.94	0.000967	0	0.0004
																	Loss	0.0005

Hình A.13: Kết quả dự đoán của mô hình đã huấn luyện

Như ta đã quan sát được để tính đạo hàm của hàm mất mát (loss function) theo từng tham số thì cần phải kết hợp các đạo hàm một cách tuần tự từ đầu ra của mạng về phía đầu vào, hay nói cách khác là đạo hàm (gradient) được lan truyền ngược từ tầng đầu ra về tầng đầu vào giúp cập nhật các trọng số để cải thiện hiệu suất dự đoán của mạng. Đây chính là lý do quá trình này được gọi là lan truyền ngược (backpropagation)

A5. Véc-tơ hóa và ma trận hóa trong mạng nơ-ron

Trong quá trình huấn luyện mạng nơ-ron với mỗi lần thực hiện lan truyền (propagation) và lan truyền ngược (backpropagation) thì chúng ta cần phải thực hiện rất nhiều phép toán đặc biệt là trong các kiến trúc mạng nơ-ron có cấu tạo phức tạp thì con số các phép tính cần thực hiện có thể lên đến hàng triệu do đó bên cạnh tối ưu thuật toán thì ta cũng cần tối ưu cách triển khai đó chính là lý do chúng ta cần áp dụng kỹ thuật ma trận hóa, véc-tơ hóa trong huấn luyện mạng nơ-ron. Để có thể hiểu được khái niệm ma trận hóa và véc-tơ hóa, ta sẽ xét mạng nơ-ron có kiến trúc như sau:



Hình A.14: Kiến trúc của mạng nơ-ron càn áp dụng ma trận hoá, véc-to hoá

Trong mạng nơ-ron này có:

- n đầu vào $x_1, x_2, x_3, \dots, x_n$
- m đầu ra $z_1, z_2, z_3, \dots, z_n$
- Một lớp ẩn (hidden layer) có m weight và bias tương ứng

Như đã đề cập ở mục A3 thì để tính được một giá trị z_j thì ta sẽ thực hiện như sau:

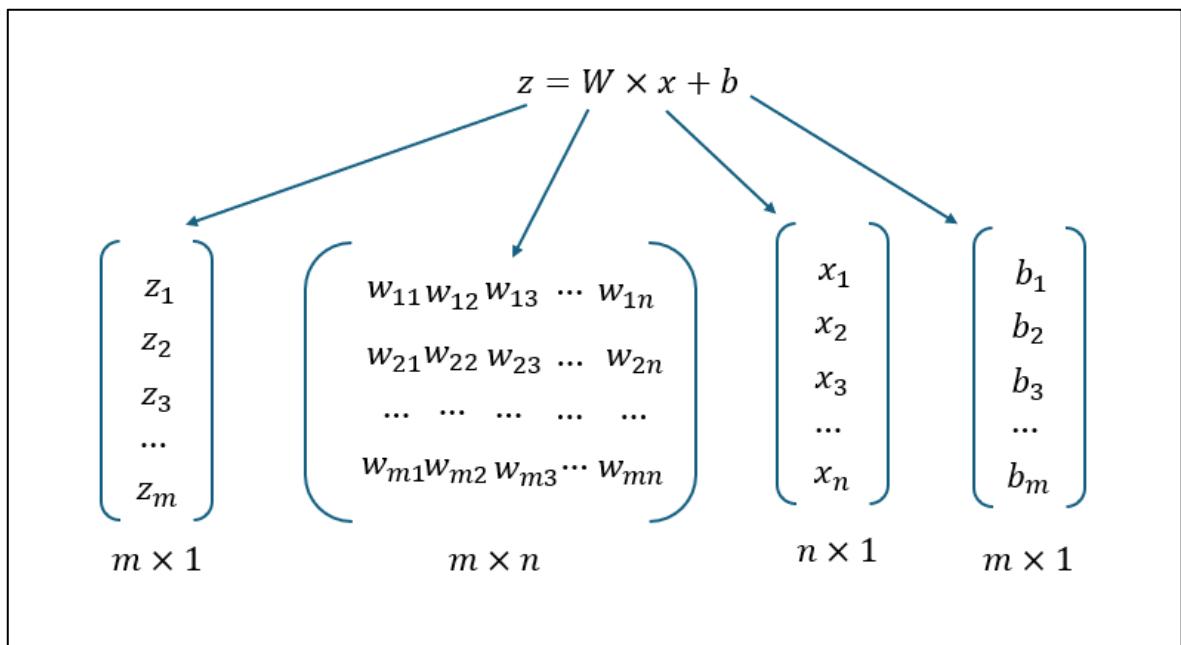
$$z_j = \sum_{i=1}^n w_{ji} \times x_i + b_j$$

Trong đó:

- w_{ij} là trọng số kết nối giữa từng x_i đến nơ-ron có đầu ra là z_j
- b_j là bias của nơ-ron có đầu ra là z_j

Để tận dụng sức mạnh tính toán được tối ưu hóa bằng phần cứng (ví dụ: GPU), đảm bảo tính rõ ràng, dễ dàng bảo trì và mở rộng mạng nơ-ron thì thay vì tính từng

z_j một cách đơn lẻ, ta có thể biểu diễn các phép tính này dưới dạng mảng ma trận và các véc-tơ như ảnh sau:

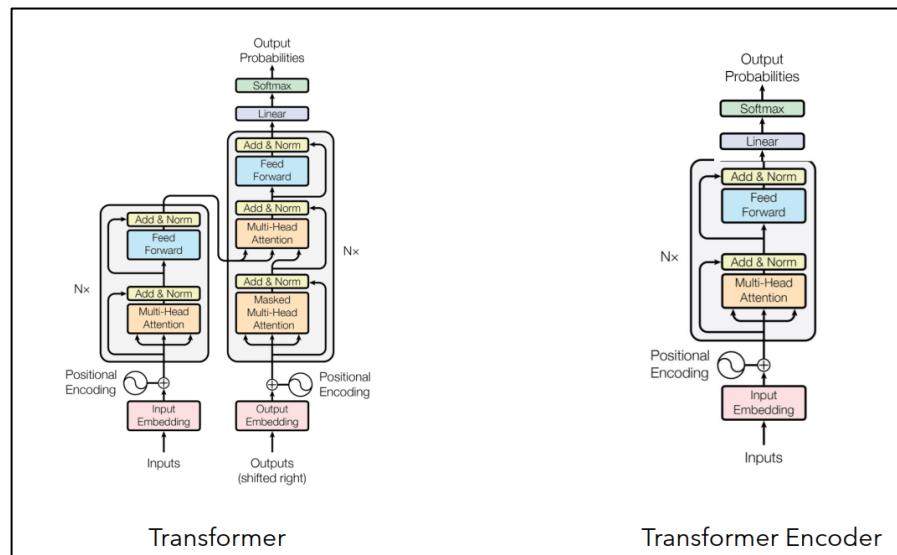


Hình A.15: Các tham số trong mạng nơ-ron được biểu diễn bằng các ma trận và véc-tơ

Trong đó các tập đầu vào, đầu vào và bias sẽ được biểu diễn bằng các vecto x , z , b tương ứng, còn tập trọng số trên mạng sẽ được biểu diễn bằng một ma trận W có kích thước ($m \times n$)

Lúc này công thức để tính đầu ra z sẽ được biểu diễn tổng quát hóa bằng một phép tính giữa các ma trận và véc-tơ như ở ảnh trên: $z = W * x + b$

B. Kiến trúc Transformer (Encoder)



Hình B.1: Kiến trúc Transformer (Encoders) [20]

Transformer là một kiến trúc được giới thiệu trong bài báo "Attention Is All You Need" [21]. Sự ra đời của *Transformer* tạo ra một bước đột phá lớn trong lĩnh vực xử lý ngôn ngữ tự nhiên và cũng mở ra con đường cho các kiến trúc mới mang tính cách mạng như BERT, GPT và nhiều mô hình khác.

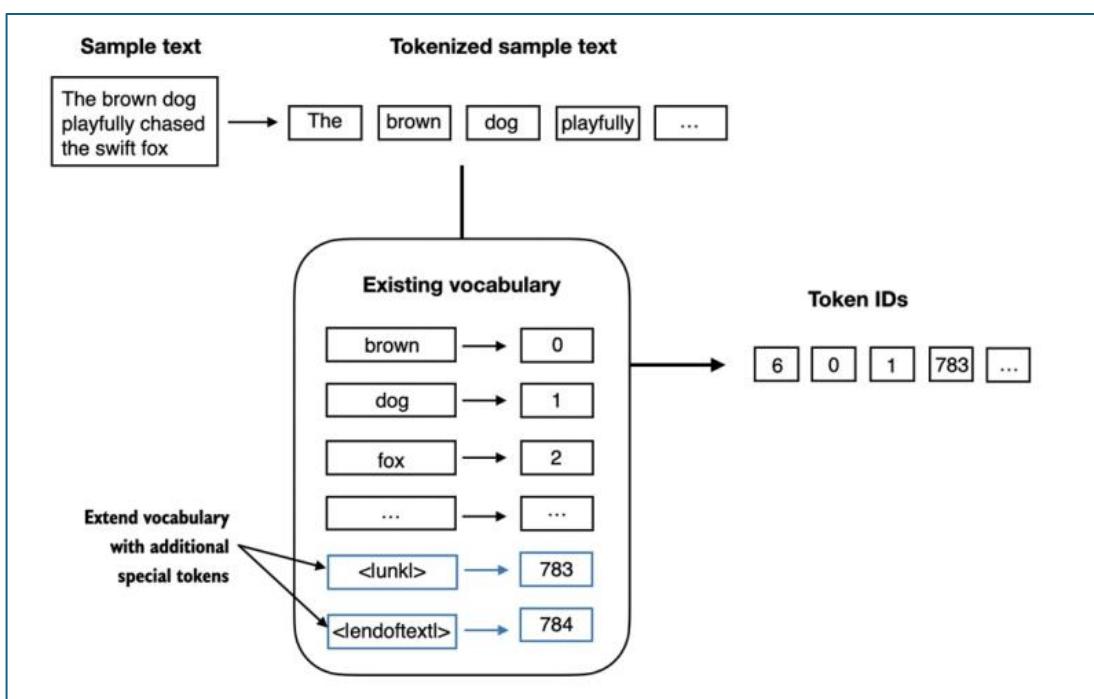
Transformer dựa hoàn toàn vào cơ chế chú ý (attention mechanism) và hoàn toàn loại bỏ sự lặp lại như ở RNN và LSTM. *Transformer* đã giải quyết các khó khăn khi mô hình làm việc với dữ liệu lớn đó chính là thời gian huấn luyện và khả năng nắm bắt các phụ thuộc dài hạn mà RNN và các biến thể của RNN chưa giải quyết được.

Khác với kiến trúc bao gồm bộ mã hóa – bộ giải mã (encoder – decoder) ở kiến trúc transformer gốc. *Transformer Encoder* chỉ sử dụng lại bộ mã hóa, các lớp nhúng đầu vào (input embedding) và lớp tuyến tính và Softmax (linear and Softmax layer) để thực hiện phân loại kết quả đầu ra. Mỗi lớp mã hóa bao gồm 2 lớp phụ (sub-layers) chính là lớp tự chú ý đa đầu (multi-head self-attention layer) và lớp truyền thẳng (feed-forward layer).

1. Lớp nhúng đầu vào

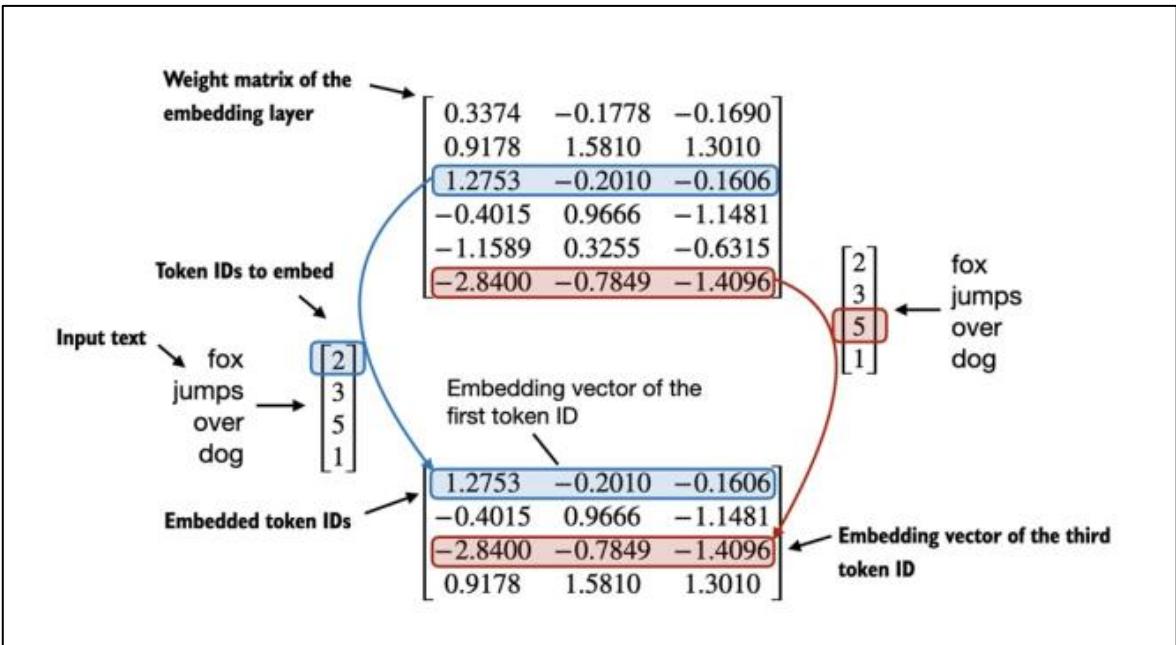
Mô hình không thể làm việc trực tiếp được với ngôn ngữ tự nhiên do đó chúng ta phải chuyển câu đầu vào sang véc-tơ. Đầu tiên ta chuyển câu thành một tập các token. Tùy vào loại mô hình mà một từ có thể được xem là một token hoặc từng chữ có thể được xem là một token. Ở đây ta xem các từ trong câu là một token

Sang bước tiếp theo, mỗi token sẽ được ánh xạ (mapping) với một con số chỉ vị trí của nó trong tập từ điển. Cụ thể ở đây là mỗi từ sẽ có nắm một vị trí trong tập từ vựng, các từ giống nhau sẽ được biểu diễn bởi cùng một con số.



Hình B.2: Minh họa quá trình chuyển câu đầu vào sang tập các token/input ID [44]

Và cuối cùng là chúng ta sẽ ánh xạ các Input ID hay Token ID sang các véc-tơ chứa trong ma trận trọng số của lớp nhúng (embedding layer) thường có kích thước là 512. Đây là véc-tơ lưu trữ nghĩa của các token trong câu.



Hình B.3: Minh họa quá trình chuyển từ token/input ID sang véc-tơ nhúng (embedding vector) [44]

Xuyên suốt quá trình chúng ta tìm hiểu về kiến trúc Transformer (Encoder) chúng ta sẽ sử dụng câu đầu vào sau “Before my bed lies a pool of moon bright”, để minh họa cách hoạt động của kiến trúc Transformer (Encoder). Trước tiên chúng ta sẽ chuyển câu đầu vào sang dưới dạng tập các véc-tơ nhúng (embedding vector) như đã đề cập trước đó.

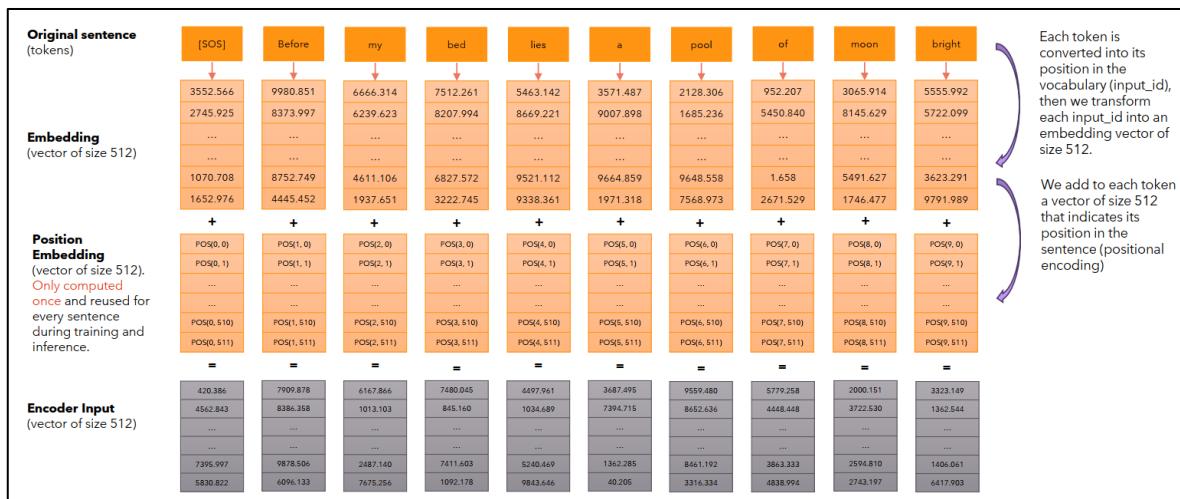
Original sentence (tokens)	[SOS]	Before	my	bed	lies	a	pool	of	moon	bright
Input IDs (position in the vocabulary)	1	90	231	413	559	952	421	7540	62	864
Embedding (vector of size 512)	3552.566 2745.925 ... 1070.708 1652.976	9980.851 8373.997 ... 8752.749 4445.452	6666.314 6239.623 ... 4611.106 1937.651	7512.261 8207.994 ... 6827.572 3222.745	5463.142 8669.221 ... 9521.112 9338.361	3571.487 9007.898 ... 9664.859 1971.318	2128.306 1685.236 ... 9648.558 7568.973	952.207 5450.840 ... 1.658 2671.529	3065.914 8145.629 ... 5491.627 1746.477	5555.992 5722.099 ... 3623.291 9791.989

Hình B.4: Các véc-tơ nhúng tương ứng với từng token đầu vào [20]

2. Lớp mã hóa vị trí

Để mô hình có thể hiểu rõ ngữ cảnh trong câu thì bên cạnh nghĩa của từ trong câu, ta cũng cần phải cung cấp thêm thông tin về vị trí của từ trong câu. Đó là lý do chúng ta cần “mã hóa vị trí” - Positional encoding.

Thay vì truyền trực tiếp véc-tơ nhúng ở phần trước vào mô hình, ta sẽ cộng từng véc-tơ nhúng với tọa độ nhúng (Position Embedding) để tạo ra một véc-tơ nhúng mới vừa chứa nghĩa của từ trong câu mà còn chứa thông tin về vị trí của từ trong câu. Đó chính là đầu vào sẽ được truyền vào bộ mã hóa (Encoders).

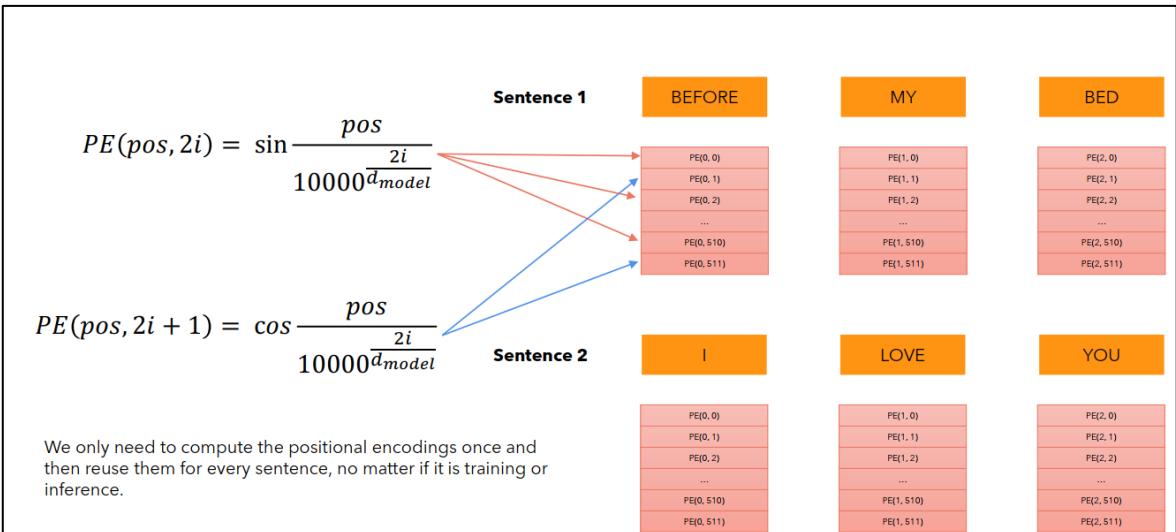


Hình B.5: Minh họa quá trình thêm vị trí được mã hóa vào véc-tơ nhúng đầu vào [20]

Mã hóa vị trí (Positional encoding) được tính theo công thức ở hình dưới đây với pos là vị trí của từ trong câu và i là vị trí trong véc-tơ nhúng (embedding vector).

Ta có thể thấy vị trí được mã hóa PE sẽ được tính bằng hàm \cos nếu i là chẵn và ngược lại nếu với i là lẻ

Vì các kích thước véc-tơ và dữ liệu đầu vào không đổi nên nhờ công thức trên mà chúng ta chỉ cần tính mã hóa vị trí (Positional encoding) một lần và sử dụng lại với nhiều câu.



Hình B.6: Quá trình tính giá trị mã hoá của từng vị trí trong câu đầu vào [20]

3. Cơ chế tự chú ý đa đầu

Ý tưởng chính của “cơ chế tự chú ý đa đầu” (multi-head self-attention) là thay vì sử dụng một đầu tự chú ý (self-attention head) thì ta sẽ sử dụng nhiều đầu tự chú ý (multi-head self-attention) để tăng độ chính xác của mô hình theo công thức sau:

$$\text{Multi-head attention} = \text{Concatenate}(Z_1, Z_2, \dots, Z_i, \dots, Z_8)W_0$$

Trong đó:

- Z_1, \dots, Z_8 là các “ma trận chú ý” của mỗi đầu tự chú ý (self-attention head)
- W_0 là ma trận trọng số
- $\text{Concatenate}()$ là quá trình nối các “ma trận chú ý” lại với nhau để tạo ra một ma trận tổng hợp

Ma trận tổng hợp sau khi nhân với ma trận trọng số W_0 sẽ cho ra ma trận chú ý (attention matrix) cuối. Vì vậy chúng ta chỉ cần tìm hiểu về cơ chế tự chưa trên một đầu (single-head self-attention)

Cơ chế tự chú ý

Để mô hình ngôn ngữ có thể học được ngữ nghĩa của một từ trong câu, các mô hình phải hiểu được mối liên hệ giữa từ đó và các từ xung quanh trong câu. Để giải quyết vấn đề trên, các mô hình dựa trên Transformer đã sử dụng cơ chế tự chú ý.

Câu đầu ra sau khi được đưa vào lớp nhúng (embedding layer) sẽ được biểu diễn dưới dạng một ma trận được gọi là ma trận nhúng (embedding matrix) với mỗi dòng là véc-tơ biểu diễn của từ trong câu.



Hình B.7: Minh họa ma trận nhúng của câu đầu vào [20]

Dựa trên ma trận nhúng và các ma trận trọng số của mô hình, ta được ba ma trận có kích thước giống với ma trận nhúng có lần lượt có tên gọi là ma trận truy vấn (Query matrix) (Q), ma trận khóa (Key matrix) (K) và ma trận giá trị (Value matrix) (V). Mỗi dòng của từng ma trận sẽ lần lượt là các véc-tơ truy vấn, véc-tơ khóa, véc-tơ giá trị của từng từ trong câu.

In a Large Language Models (LLM) we employ the Self-Attention mechanism, which means the Query (Q) , Key (K) and Value (V) are the same matrix .												
	Query					Key			Value			
[SOS]	420.386	4562.843	7395.997	5830.822	420.386	4562.843	7395.997	5830.822
Before	7909.878	8386.358	9878.506	6096.133	7909.878	8386.358	9878.506	6096.133
my	6167.866	1013.103	2487.140	7675.256	6167.866	1013.103	2487.140	7675.256
bed	7480.045	845.160	7411.603	1092.178	7480.045	845.160	7411.603	1092.178
lies	4497.961	1034.689	5240.469	9843.646	4497.961	1034.689	5240.469	9843.646
a	3687.495	7394.715	1362.285	40.205	3687.495	7394.715	1362.285	40.205
pool	9559.480	8652.636	8461.192	3316.334	9559.480	8652.636	8461.192	3316.334
of	5779.258	4448.448	3863.333	4838.994	5779.258	4448.448	3863.333	4838.994
moon	2000.151	3722.530	2594.810	2743.197	2000.151	3722.530	2594.810	2743.197
bright	3323.149	1362.544	1406.061	6417.903	3323.149	1362.544	1406.061	6417.903

Hình B.8: Ma trận Q, K, V tương ứng với câu đầu vào [20]

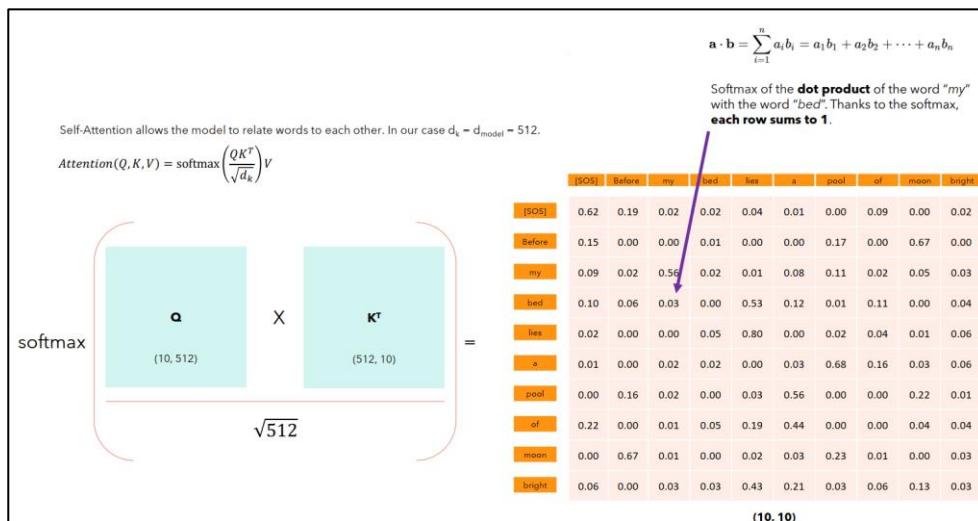
Sau khi đã có ba ma trận Q, K, V ta sẽ tính giá trị chú ý (attention value) của từng từ trong câu theo công thức như hình:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Trong đó:

- Q, K, V lần lượt là ma trận truy vấn, ma trận khóa và ma trận giá trị
- $d_k = 512$: Số chiều biểu diễn của véc-tơ nhúng trong kiến trúc Transformer

Ma trận mà ta nhận được sau khi áp dụng công thức có thể được gọi là ma trận điểm vì nó chứa các “điểm chú ý” (attention score) là giá trị về độ tương đồng của các thành phần nhúng biểu diễn của từ trong câu. Bên cạnh đó tổng điểm trên một dòng trong ma trận luôn bằng 1 nhờ sử dụng hàm Softmax.

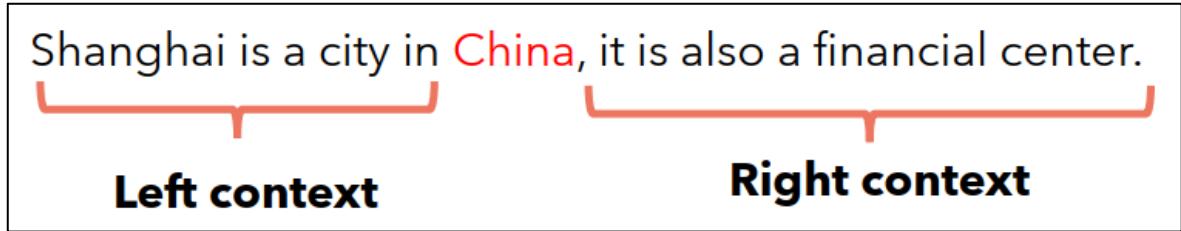


Hình B.9: Tổng điểm trên một dòng trong trận luôn bằng 1 nhờ sử dụng hàm Softmax [20]

Causal mask

Như chúng ta đã nói về khái niệm của mô hình ngôn ngữ, một mô hình ngôn ngữ có khả năng dự đoán xác suất xuất hiện của từ tiếp theo trong câu dựa trên ngữ cảnh của câu đầu vào. Nhưng để mô hình hóa xác xuất đó thì mỗi từ chỉ nên phụ thuộc vào

các từ ở trước nó, và không nên quan tâm đến các từ sau nó. Điều này làm nên cơ sở cho việc sử dụng causal softmax trong mô hình ngôn ngữ.



Hình B.10: Minh họa ngữ cảnh trái và phải trong câu [20]

Quy trình causal softmax có thể được minh họa như sau:

	[SOS]	Before	my	bed	lies	a	pool	of	moon	bright	$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$
[SOS]	5.45	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	
Before	4.28	2.46	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	
my	8.17	3.56	5.54	-∞	-∞	-∞	-∞	-∞	-∞	-∞	
bed	6.71	4.13	6.76	0.79	-∞	-∞	-∞	-∞	-∞	-∞	
lies	5.43	7.59	3.91	6.14	9.03	-∞	-∞	-∞	-∞	-∞	
a	4.42	4.35	7.55	3.14	1.35	7.57	-∞	-∞	-∞	-∞	
pool	8.36	6.00	4.56	0.52	3.13	6.78	9.00	-∞	-∞	-∞	
of	2.21	3.72	4.16	6.30	0.66	6.14	7.46	6.77	-∞	-∞	
moon	4.08	6.22	5.00	4.20	5.72	5.35	7.46	3.55	4.70	-∞	
bright	6.43	8.88	6.17	3.65	4.54	5.22	5.51	5.55	0.64	1.38	

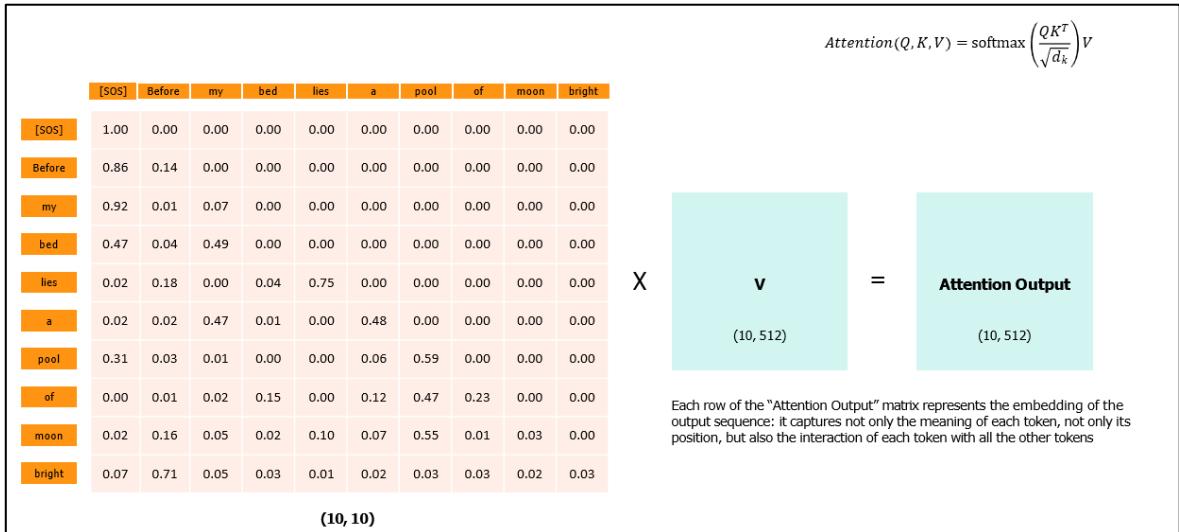
	[SOS]	Before	my	bed	lies	a	pool	of	moon	bright	$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$
[SOS]	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
Before	0.86	0.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
my	0.92	0.01	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
bed	0.47	0.04	0.49	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
lies	0.02	0.18	0.00	0.04	0.75	0.00	0.00	0.00	0.00	0.00	
a	0.02	0.02	0.47	0.01	0.00	0.48	0.00	0.00	0.00	0.00	
pool	0.31	0.03	0.01	0.00	0.00	0.06	0.59	0.00	0.00	0.00	
of	0.00	0.01	0.02	0.15	0.00	0.12	0.47	0.23	0.00	0.00	
moon	0.02	0.16	0.05	0.02	0.10	0.07	0.55	0.01	0.03	0.00	
bright	0.07	0.71	0.05	0.03	0.01	0.02	0.03	0.03	0.02	0.03	

Hình B.11: Minh họa causal masking trong quá trình tính điểm chú ý [20]

Trước khi áp dụng hàm softmax như phần trước, ta sẽ thay thế các giá trị điểm chú ý (attention score) từng từ với các từ nằm phía bên phải của chúng bằng âm vô cùng sau đó áp dụng hàm softmax như ở phần trước để chuẩn hóa các “giá trị chú ý”. Lúc này các giá trị âm vô cùng sẽ được chuẩn hóa về 0, ngăn chặn việc mô hình chú ý đến các từ trong tương lai.

Cuối cùng, ta nhân ma trận điểm với ma trận V để tạo ra ma trận chú ý (attention matrix). Mỗi dòng trong ma trận đại diện cho mỗi quan hệ hoặc mức độ quan trọng

giữa một token trong câu đầu vào với tất cả các token khác trong câu.



Hình B.12: Ma trận chú ý đầu ra của cơ chế tự chú ý [20]

4. Mạng truyền thẳng theo vị trí

Đầu ra của lớp tự chú ý đa đầu (multi-head self-attention) sau đó sẽ là đầu vào của mạng truyền thẳng (Feed-forward network) bao gồm hai lớp chuyển đổi tuyến tính (linear transformations) với hàm kích hoạt ReLU ở giữa chúng. Các tham số của mạng truyền thẳng (Feed-forward network) là giống nhau ở các vị trí khác nhau trong câu và khác nhau ở các khối mã hóa. Điều này giúp mô hình học được các biểu diễn đa dạng và phức tạp ở từng vị trí trong câu, trong khi vẫn giữ được tính chia sẻ và hiệu quả của các tham số.

Một đầu vào vec-tor x là biểu diễn của một từ trong câu sau khi truyền vào “mạng truyền thẳng theo vị trí” (Position-Wise Feed-Forward Network) được biểu diễn bằng công thức sau:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

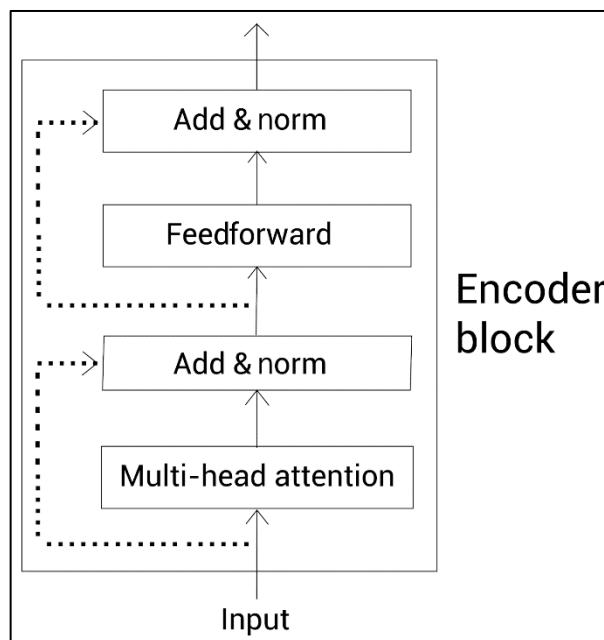
trong đó:

- W_1, W_2 là hai ma trận trọng số khác nhau của 2 lớp chuyển đổi tuyến tính (layer linear transformations)

- b_1, b_2 là bias véc-tơ của 2 lớp chuyển đổi tuyến tính (layer linear transformations)
- $\max(0, xW_1+b_1)$ là phần sử dụng hàm kích hoạt ReLU để tạo một đầu ra phi tuyến tính (nonlinear output) từ lớp đầu

5.Thành phần thêm và chuẩn hóa

Ở giữa mỗi lớp con của lớp tự chú ý đa đầu (multi-head self attention) và lớp truyền thẳng theo vị trí (Position-Wise Feed-Forward Network), nhóm tác giả đã triển khai thêm một thành phần được gọi là (Add and norm component) gồm một đường nét đứt - kết nối thừa (residual connection) sau đó là một lớp chuẩn hóa (normalization layer).



Hình B.13: Minh họa lớp chuẩn hóa trong khối mã hóa [45]

Cách hoạt động của thành phần thêm và chuẩn hóa (Add and norm component) có thể được tóm gọn lại theo công thức sau:

$$\text{LayerNorm}(x + \text{Sublayer}(x))$$

Trong đó

- “ $x + Sublayer(x)$ ” là kết nối thừa (residual connection) để thêm đầu vào của lớp con (sub-layer) vào đầu ra của chính lớp con (sub-layer) đó. Bước này đảm bảo rằng mô hình có thể học được những thông tin thừa từ đầu vào, giúp giảm thiểu vấn đề triệt tiêu của đạo hàm (the vanishing gradient problem) và tăng tốc quá trình hội tụ (convergence) trong quá trình huấn luyện. “”
- “ $LayerNorm()$ ” là lớp chuẩn hóa thực hiện việc chuẩn hóa giá trị đầu ra của lớp con sau khi được kết hợp với đầu vào ban đầu. Quá trình này giúp giảm biến động giữa các giá trị đặc trưng của mỗi token. Điều này làm cho quá trình học trở nên ổn định hơn.

6. Lớp tuyến tính và Softmax

Sau khi các khối mã hóa đã học được các biểu diễn ngữ nghĩa trong câu đầu vào, đầu ra thu được từ khối mã hóa cuối sẽ được truyền vào lớp tuyến tính và Softmax (linear layer and softmax layer) để biến đổi các biểu diễn đặc trưng của các token thành các dự đoán hoặc xác suất phù hợp với nhiệm vụ cụ thể mà mô hình đang được huấn luyện.

Lớp “Tuyến tính” thực hiện phép biến đổi tuyến tính trên các biểu diễn đặc trưng của các token trong câu bằng cách sử dụng một ma trận trọng số và véc-tơ bias để biến ma trận đầu vào x chứa các véc-tơ biểu diễn của các token trong câu, thành một véc-tơ đầu ra có kích thước giống như số lượng các nhãn mà chúng ta muốn mô hình phân loại. véc-tơ này thường được gọi là véc-tơ logit.

$$logits = xW + b$$

Trong đó:

- x : là ma trận đầu vào, chứa các véc-tơ biểu diễn của các token trong câu.
- W : Là ma trận trọng số của lớp tuyến tính
- b : Là véc-tơ chứa các bias của lớp tuyến tính

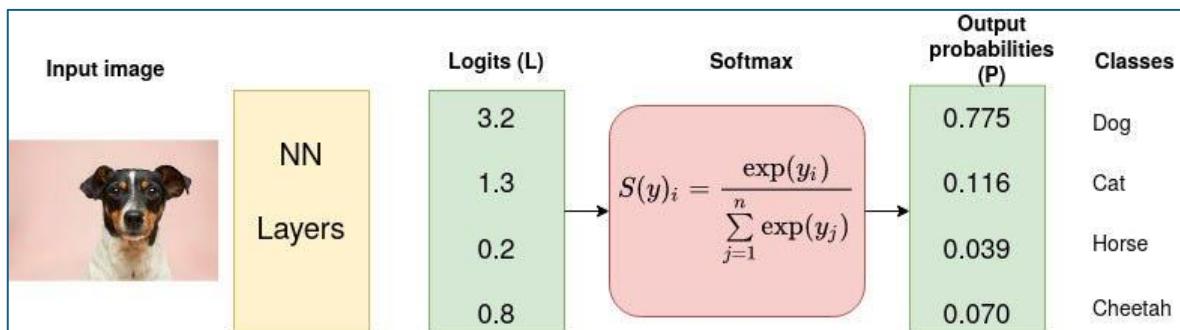
Sau đó, các giá trị logits này sẽ được đưa qua hàm softmax để chuyển thành các xác suất dự đoán của từng nhãn, sao cho tổng các xác suất bằng 1. Quy trình trên có thể được tóm gọn bằng công thức sau:

$$probabilities(logits_j) = softmax(logits_j) = \frac{e^{logits_j}}{\sum_{i=1}^k e^{logits_j}}$$

Trong đó:

- $logits_j$: Là một giá trị logits thứ j trong véc-tơ logit
- $\sum_{i=1}^k e^{logits_j}$: Là tổng của tất cả k giá trị logit trong véc-tơ logit

Chính vì khả năng chuẩn hóa các giá trị đầu ra của mạng nơ-ron thành các xác suất của từng nhãn mà mô hình muốn phân loại nên hàm softmax được sử dụng rất nhiều cho các bài toán phân loại, nhận diện như bài toán nhận diện động vật trong ảnh dưới đây :



Hình B.14: Ví dụ sử dụng hàm Softmax trong bài toán nhận diện động vật [46]