ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA HỆ THỐNG THÔNG TIN

NGUYỄN MINH KHUÊ

KHÓA LUẬN TỐT NGHIỆP

XÂY DỰNG HỆ THỐNG END TO END CHATBOT SỬ DỤNG MÔ HÌNH HỌC SÂU

BUILDING AN END TO END CONVERSATIONAL AGENT USING DEEP LEARNING

KỸ SƯ NGÀNH HỆ THỐNG THÔNG TIN

TP. HỒ CHÍ MINH, 2021

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA HỆ THỐNG THÔNG TIN

NGUYỄN MINH KHUÊ – 16520314

KHÓA LUẬN TỐT NGHIỆP

XÂY DỰNG HỆ THỐNG END TO END CHATBOT SỬ DỤNG MÔ HÌNH HỌC SÂU

BUILDING AN END TO END CONVERSATIONAL AGENT USING DEEP LEARNING

KỸ SƯ NGÀNH HỆ THỐNG THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN

ThS. NGUYỄN HỒ DUY TRI

CH. HUỲNH THIỆN Ý

TP. HỒ CHÍ MINH, 2021

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số…… …… ngày ……. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. …… – Chủ tịch.
2. …… – Thư ký.
3. …… – Ủy viên.
4. …… – Ủy viên.

LỜI CẢM ƠN

Lời đầu tiên, em xin chân thành cảm ơn Khoa Hệ thống Thông tin, trường Đại học Công nghệ Thông tin đã tạo điều kiện tốt nhất cho em thực hiện đề tài Khóa luận tốt nghiệp này.

Trong quá trình hoàn thành khóa luận tốt nghiệp, em đã học hỏi được rất nhiều kiến thức về mặt lý thuyết, cũng như các phương pháp thực hành thực tế trong việc xây dựng và phát triển phần mềm, để có thể phát triển thành công một sản phẩm như ngày hôm nay.

Em xin gửi lời cảm ơn chân thành đến thầy Nguyễn Hồ Duy Tri và thầy Huỳnh Thiện Ý đã tận tình hướng dẫn và tạo điều kiện thuận lợi cho em trong suốt quá trình thực hiện để hoàn thành đề tài này.

Trong quá trình làm khóa luận khó tránh khỏi những sai sót. Em rất mong nhận được sự góp ý của thầy để có thể hoàn thiện khóa luận tốt hơn nữa.

Sinh viên thực hiện đề tài: Nguyễn Minh Khuê

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Nội dung
ANN(s)	Artificial Neural Network
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers.
BPE	Byte pair encoding
CBOW	Continuous Bag of Words
CNN(s)	Convolutional Neural Networks
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
LSTM	Long short-term memory
NLP	Natural Language Processing
PPL	Perplexity
Pre-trained model	Mô hình mẫu được huấn luyện trước trên nhiệm vụ cụ thể khác
RNN(s)	Recurrent neural network
Seq2seq	Sequence to sequence
Word2Vec	Word to vector

BẢNG KÝ HIỆU TOÁN HỌC

Ký hiệu	Ý nghĩa
x	Vector cột x
X	Ma trận X
X^T	Ma trận chuyển vị của X
W_a	Ma trận trọng số a
b	Bias
c_t	Vector ngữ cảnh tại bước thời gian t
W_i^Q	Ma trận tham số của Q tại đầu i
$P(x\|x_{1:i-1})$	Xác suất của x khi biết $x_1, \dots x_{i-1}$
s_t	Trạng thái ẩn của bộ giải mã tại vị trí t
h_i	Trạng thái ẩn của bộ mã hóa tại bước thời gian i
$\mathbb{R}^{m \times n}$	Tập hợp các ma trận số thực có m hàng, n cột
$\in$	Phần tử thuộc tập hợp
$\exp(x)$	e^x

TÓM TẮT KHÓA LUẬN

Trong bối cảnh mạng xã hội, internet dần trở nên phổ biến, con người được kết nối với nhau ở bất cứ thời gian nào, bất cứ nơi đâu. Nhu cầu trò chuyện, tìm kiếm thông tin, giải trí dần trở nên không thể thiếu trong cuộc sống hằng ngày. Do đó, cần một hệ thống có thể giúp đỡ con người bằng cách trò chuyện, nhắc nhở, tìm kiếm thông tin một cách nhanh chóng và tiện lợi. Những hệ thống đó được gọi là các trợ lý ảo.

Mô hình hóa hội thoại là một trong những nhiệm vụ quan trọng để xây dựng được một trợ lý ảo hiệu quả. Trước đây, các phương pháp tiếp cận truyền thống thường giới hạn trong một số lĩnh vực cụ thể như: Đặt vé trực tuyến, tra cứu thông tin y tế, tư vấn mua hàng,… và không thể mở rộng vào các lĩnh vực khác nhau. Do đó, việc nghiên cứu các mô hình ngôn ngữ để ứng dụng xây dựng chatbot có thể tham gia vào các lĩnh vực khác nhau trở nên thật sự cần thiết.

Ngày nay, với sự phát triển của các phương pháp học sâu trong lĩnh vực xử lý ngôn ngữ tự nhiên các mô hình hội thoại giờ đây có thể tự tạo các phản hồi có ý nghĩa tuy nhiên chúng vẫn còn có nhiều giới hạn. Một trong những giới hạn đó là tính cá nhân hóa của chatbot. Với mục tiêu xây dựng một chatbot có tính cách cá nhân khóa luận sẽ trình bày những nội dung chính sau:

- Tìm hiểu về các nghiên cứu liên quan.
- Tìm hiểu về một số mô hình học sâu như Seq2seq, Transformer, và các kiến thức liên quan.
- Áp dụng phương pháp tiếp cận TransferTransfo - một kỹ thuật kết hợp giữa học chuyển tiếp và mô hình Transformer đã huấn luyện trước vào hệ thống hội thoại như chatbot.
- Xây dựng ứng dụng cụ thể để minh họa mô hình.

Chương 1. TỔNG QUAN

1.1. Đặt vấn đề

Ngày nay, với sự phát triển của khoa học công nghệ, các hệ thống AI có thể tương tác với con người, hiểu những gì họ cần và đưa ra các đề xuất, hành động thích hợp với sự can thiệp tối thiểu hoặc không có sự can thiệp từ con người. Khả năng giao tiếp tự do trong ngôn ngữ tự nhiên là một trong những đặc điểm nổi bật của trí thông minh ở con người, để khám phá khía cạnh thông minh này nhiều nhà khoa học đang nghiên cứu trên các chatbot miền mở (Open-domain chatbots) để tạo ra một chatbot có khả năng giao tiếp tương tự như con người.

Không như những chatbot miền đóng (Closed-domain chatbot) chỉ phản hồi ở những chủ đề cụ thể, các chatbot miền mở có thể tham gia vào các cuộc trò chuyện ở bất kỳ chủ đề nào. Một vài chatbot miền mở nổi tiếng như MILABOT (Serban et al., 2017), XiaoIce (Zhou et al., 2018), Gunrock (Chen et al., 2018), Mitsuku (Worswick, 2018), Cleverbot (Rollo Carpenter) dựa trên những framework phức tạp như quản lý hội thoại (dialog managers) sử dụng mô hình cơ sở tri thức (Knowledge-based), rút trích thông tin (Retrieval-based), hoặc dựa trên các luật (Rule-based). Hay hướng tiếp cận theo mô hình học sâu bao gồm Siri, Google Assistant, Cortana. Hướng tiếp cận dựa vào mô hình học sâu được cho là tiên tiến hơn vì nó được học từ đầu dựa trên kỹ thuật dịch máy (Machine translation). Mặc dù đã trải qua rất nhiều nghiên cứu, chatbot miền mở vẫn còn những hạn chế như có những phản hồi quá mơ hồ hoặc không có ý nghĩa.

Đề tài sẽ tập trung tìm hiểu về mô hình học sâu Transformer, các thách thức để xây dựng một chatbot miền mở hiểu quả và ứng dụng phương pháp tiếp cận TransferTransfo được giới thiệu trong bài báo [25] để xây dựng một chatbot tự động tạo câu trả lời.

1.2. Mục tiêu và phạm vi

1.2.1. Mục tiêu

- Mục tiêu chung: Tìm hiểu mô hình học sâu Transformer và kiến trúc liên quan từ đó ứng dụng xây dựng chatbot có thể trò chuyện với người dùng thông qua một số chủ đề phổ biến mang tính cá nhân như công việc, gia đình, sở thích cá nhân, …
- Cụ thể:
    - Tìm hiểu về mô hình học sâu Transformer và kỹ thuật học chuyển tiếp (Transfer learning).
    - Khảo sát các đề tài, nghiên cứu liên quan, các thách thức để xây dựng chatbot miền mở hiệu quả.
    - Áp dụng phương pháp tiếp cận TransferTransfo - một kỹ thuật kết hợp giữa học chuyển tiếp và mô hình Transformer đã tạo tạo sẵn, vào hệ thống hội thoại như chatbot.
    - Xây dựng một chatbot sử dụng mô hình trên và triển khai trên Messenger để minh họa mô hình.

1.2.2. Phạm vi

- Một số mô hình ngôn ngữ đã được đào tạo sẵn như: BERT, OpenAI GPT.
- Kỹ thuật học chuyển tiếp, phương pháp tiếp cận TransferTransfo.
- Tập dữ liệu được dùng để đào tạo là PERSONA CHAT – tập dữ liệu chit-chat trong đó người nói được chia thành từng cặp và được chỉ định tính cách để trò chuyện nhằm mục đích tìm hiểu lẫn nhau.

1.3. Cấu trúc khoá luận

Chương 1: Giới thiệu tổng quan về đề tài.

Chương 2: Trình bày tổng quan về chatbot, nghiên cứu liên quan và các thách thức để xây dựng một chatbot hiệu quả..

Chương 3: Trình bày về cơ sở lý thuyết, cơ chế chú ý (Attention), kiến trúc cơ bản của mô hình Transformer và các kỹ thuật sẽ được sử dụng trong đề tài.

Chương 4: Giới thiệu phương pháp TransferTransfo và cách ứng dụng vào bài toán xây dựng hệ thống hội thoại.

Chương 5: Cài đặt và triển khai ứng dụng.

Chương 6: Kết luận, hạn chế và hướng phát triển.

1.4. Phương pháp thực hiện

- Khảo sát một số bài báo liên quan đến chatbot.
- Tìm hiểu các thuật toán trong những bài báo đã khảo sát.
- Tìm các bộ dữ liệu đa lĩnh vực (Open-domain dataset) phù hợp với quy mô đề tài, tiền xử lý dữ liệu.
- Tìm hiểu một số thuật toán, mô hình TransferTransfo để xây dựng ứng dụng.
- Huấn luyện mô hình trên tập dữ liệu đã thu thập được.
- Sửa các lỗi phát hiện sau khi huấn luyện mô hình.
- So sánh, đánh giá mô hình dựa trên các độ đo như PPL, Hits@1, F1 score.
- Triển khai mô hình trên ứng dụng cụ thể để đáp ứng như cầu của đề tài.

Chương 2. CÁC NGHIÊN CỨU LIÊN QUAN

2.1. Chatbot là gì?

Chatbot hay còn gọi là hệ thống trả lời tự động là một chương trình máy tính có khả năng tương tác với con người bằng ngôn ngữ tự nhiên thông qua tin nhắn văn bản hoặc âm thanh.

Từ khi AI ra đời, việc mô hình hóa các cuộc trò chuyện vẫn là một trong những thách thức khó khăn nhất của lĩnh vực này. Mặc dù vẫn chưa trở nên hoàn hảo, nhưng chatbot hiện nay đã được sử dụng nhiều trong các trợ lý ảo như Siri (Apple), Google Assistant (Google), Alexa (Amazon), Cortana (Microsoft), … hay được ứng dụng để hỗ trợ ở nhiều lĩnh vực khác nhau từ E-commerce cho đến ngành du lịch.

2.2. Phân loại kiến trúc chat bot

Kiến trúc của chatbot được chia thành hai lớp:

- Dựa trên kịch bản có sẵn (Rule-based systems)
- Dựa trên kho dữ liệu (Corpus-based systems)

Chatbot dựa trên kịch bản có sẵn bao gồm các chatbot nổi tiếng như ELIZA, PARRY được ra đời từ rất sớm là tiền đề cho các hệ thống chatbot thông minh ngày nay. Khác với chatbot dựa trên các luật, chatbot dựa trên kho dữ liệu khai thác các cuộc đối thoại giữa người và người sau đó sử dụng chính phản hồi của từ những cuộc đối thoại trước đó để phản hồi lại bằng phương pháp rút trích thông tin (Retrival-based) hoặc sử dụng các mô hình dịch máy như mô hình chuỗi sang chuỗi (Sequence to sequence), Transformer để học cách ánh xạ câu nói của người dùng đến phản hồi của hệ thống.

2.3. Vì sao cần phải xây dựng một chatbot miền mở

Trước đây, để tạo một chatbot người ta sẽ xây dựng một kịch bản có sẵn cho chatbot. Phương pháp này tốn rất nhiều thời gian và công sức và khó để mở rộng sang các lĩnh vực khác. Do đó, người ta đã nghiên cứu trên các chatbot miền mở.

Ưu điểm của loại chatbot này là có khả năng tham gia nhiều chủ đề, lĩnh vực khác nhau và dễ dàng mở rộng. Đây là điều mà những chatbot miền đóng khó có thể làm được.

Ngày nay, để tạo ra các chatbot miền mở người ta thường sử dụng phương pháp rút trích thông tin hoặc phương pháp tạo văn bản. Ngoài ra, trong một số hệ thống chatbot người ta còn kết hợp cả hai phương pháp này lại với nhau để mang lại hiệu quả cao nhất.

Với những ưu điểm của chatbot miền mở mang lại, Tạo ra một hệ thống hội thoại phục vụ trên nhiều lĩnh vực trở thành mối quan tâm hàng đầu hiện nay.

2.4. Thách thức để xây dựng một chatbot miền mở hiệu quả

Xây dựng thành công một chatbot miền mở thông minh có thể trò chuyện với con người một cách mạch lạc và hấp dẫn đã trở thành mục tiêu lâu dài của trí tuệ nhân tạo. Tuy đã trãi qua nhiều nghiên cứu, trí tuệ nhân tạo vẫn còn kém xa con người. Việc xây dựng hệ thống hội thoại miền mở có thể trò chuyện ở các chủ đề khác nhau như con người vẫn còn là thách thức vô cùng lớn. Phần này sẽ giới thiệu về ba thách thức lớn trong việc xây dựng một chatbot miền mở hiệu quả theo [6] đó là: Thách thức về mặt ngữ nghĩa (semantics), tính nhất quán (consistency) và tính tương tác (interactiveness).

Bảng sau đây minh họa về ba vấn đề đã đề cập ở trên:

Vấn đề về mặt ngữ nghĩa	
User	I come from HoChiMinh City.
Machine	Cool! I am from HaNoi, we come from the same city!
User	Back to HaNoi now. It is so cold but I am very happy!
Machine	When will you come to HaNoi?
Vấn đề về tính nhất quán	
User	Are you a boy or a girl?
Machine	I am a boy.
User	Are you a girl?
Machine	Yes, I am a girl.
Vấn đề về tính tương tác	
User	Terrible breakfast today.
Machine	How about together tomorrow morning?
User	Sure. Would be great with your recommendation.
Machine	Really?

Bảng 2.1: Một số ví dụ minh họa vấn đề về mặt ngữ nghĩa, tính nhất quán, tính tương tác của chatbot miền mở

- *Thách thức về mặt ngữ nghĩa (semantics):* Ngữ nghĩa là trung tâm của bất kỳ hệ thống hội thoại nào bởi vì cuộc trò chuyện cơ bản là một hoạt động ngữ nghĩa. Nó yêu cầu hệ thống phải hiểu được nội dung của cuộc trò chuyện, thông tin cá nhân của người dùng như tính cách, cảm xúc, tình cảm, … để sinh ra được câu trả lời hợp lý. Đứng trên góc độ kỹ thuật, ngữ nghĩa chủ yếu có liên quan đến các kỹ thuật *hiểu ngôn ngữ tự nhiên (natural language understanding) và hiểu về người dùng (user understanding)* bao gồm named entity recognition (Nhận dạng thực thể định danh), entity linking

(Liên kết các thực thể được đề cập trong văn bản với các thực thể tương ứng trong cơ sở tri thức), domain detection (Nhận dạng miền – nó sẽ phân tích và xếp loại văn bản vào các miền định nghĩa trước), topic and intent detection (Xác định chủ đề và ý định của người dùng), user sentiment/emotion/opinion detection (Xác định tình cảm/cảm xúc/ý kiến của người dùng), và knowledge/ commonsense reasoning (Các kiến thức và suy luận thông thường).

- *Tính nhất quán (consistency):* Để đạt được sự tin cậy của người dùng, điều quan trọng là hệ thống hội thoại phải đưa ra các phản hồi nhất quán với phản hồi của người dùng và lịch sử hội thoại trước đó. Đặc biệt phải đám ứng ba tiêu chí: Thứ nhất là nhất quán về tính cách, phản hồi đó cần phải phù hợp với tính cách đã định nghĩa trước. Thứ hai là nhất quán về phong cách, phản hồi cần có một phong cách nói nhất quán trong suốt cuộc trò chuyện. Thứ ba là nhất quán về ngữ cảnh, câu trả lời phải mạch lạc và nhất quán với ngữ cảnh hội thoại. Đứng trên góc độ kỹ thuật đó là các bài toán về cá nhân hóa (personalization), tạo phong cách (stylistic generation), mô hình hóa nội dung (multi-turn context modeling).
- *Tính tương tác (interactiveness):* Tính tương tác đề cập đến khả năng của hệ thống để đạt được các mục tiêu xã hội phức tạp như giải trí và phù hợp bằng cách tối ưu hóa hành vi và chiến lược hội thoại trong một cuộc trò chuyện dài. Để tăng cường tính tương tác, điều quan trọng hệ thống phải hiểu được trạng thái cảm xúc của người dùng. Các phản hồi không chỉ phản ứng lại mà còn chủ động dẫn dắt chủ đề, tối ưu hóa chiến lược tương tác để tối đa hóa mức độ tương tác của người dùng. Các thách thức liên quan đến tính tương tác bao gồm nhận diện tình cảm và cảm xúc (sentiment and emotion detection), dialog state tracking, nhận diện và đề xuất chủ đề (topic detection and recommendation), học chiến lược đối thoại (dialog policy learning), Kiểm soát tạo phản hồi (controllable response generation).

Đề tài sẽ tập trung vào giải quyết thách thức về tính nhất quán cụ thể là nhất quán về tính cách của hệ thống hội thoại.

2.5. Các nghiên cứu liên quan

Các nghiên cứu ban đầu trên hệ thống tự tạo hội thoại miền mở được lấy cảm hứng từ các nghiên cứu về dịch máy [16] [19] [22] sau đó, kiến trúc mã hóa giải mã được mở rộng để cải thiện tính đa dạng cho các phản hồi ([8], [27], [34], [20]) đến mô hình hóa cấu trúc nội dung cuộc trò chuyện ([18],[26], [30]), kiểm soát các thuộc tính của phản hồi ([28],[35], [24]), và phản hồi nghiên về một số tính cách cụ thể ([9], [31]). Hiện nay, các hệ thống hội thoại giải quyết sự nhất quán về tính cách được gom nhóm thành hai mục sau: Cá nhân hóa ngầm định và cá nhân hóa rõ ràng.

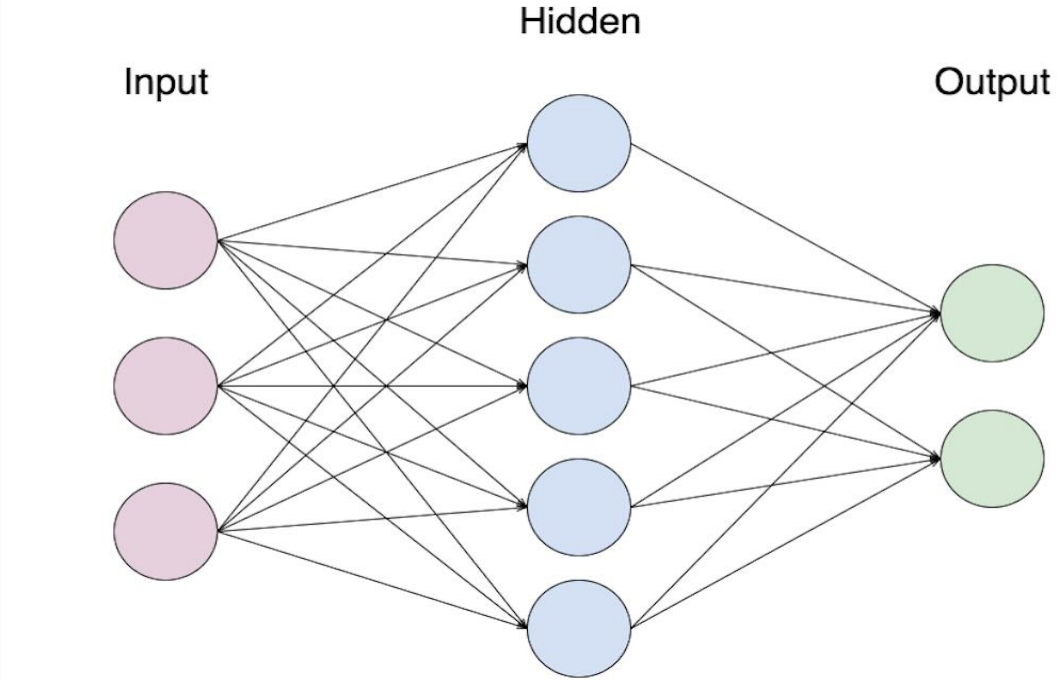
- **Cá nhân hóa ngầm định:** Tính cách được biểu diễn ngầm định bằng một vector tính cách. Ví dụ Kim và các cộng sự [7] đề xuất phương pháp xếp hạng để tích hợp cơ sở kiến thức cá nhân và sở thích của người dùng trong hệ thống hội thoại, Zhang và cộng sự [32] đã sử dụng một embedding vector để đại diện cho tính cách của người dùng và đưa vào từng vị trí giải mã của bộ giải mã. Những mô hình này cần được đào tạo bằng cách sử dụng dữ liệu hội thoại được gán nhãn định danh tính cách từng người, rất tốn kém để thu thập được với số lượng lớn. Do đó, Wang và cộng sự [23] đề xuất huấn luyện mô hình được cá nhân hóa chỉ với các thuộc tính nhóm (ví dụ: Nam hoặc nữ). Các thuộc tính này sẽ được nhúng vào các vector, sau đó dưa vào bộ giải mã để tạo phản hồi. Zhang và cộng sự [33] đã đề xuất một mô hình hội thoại tạo ra phản hồi nhất quán bằng cách duy trì một số đặc điểm liến quan đến chủ đề và tính cách trong suốt cuộc hội thoại. Việc cá nhân hóa trong các mô hình này được xử lý ngầm định, do đó không dễ diễn giải và kiểm soát trong việc tạo phản hồi mong muốn.
- **Cá nhân hóa rõ ràng:** Trong [13], một mô hình tính cách rõ ràng được đề xuất để tạo ra các phản hồi gắn kết về tính cách với một hồ sơ được chỉ định trước. Tính cách của chatbot được xác định bằng một bảng khóa-giá

trị (hồ sơ) bao gồm tên, tuổi, giới tính, sở thích, … Trong quá trình tạo phản hồi, đầu tiên mô hình chọn cặp khóa-giá trị từ hồ sơ sau đó giải mã phản hồi từ cặp khóa-giá trị đã chọn từ trước. Mô hình này có thể được huấn luyện trên dữ liệu hội thoại chung mà không cần định danh người nói. XiaoIce [36] cũng sử dụng mô hình có tính cách rõ ràng.

Chương 3. CƠ SỞ LÝ THUYẾT

3.1. Mạng neural nhân tạo

Mạng neural nhân tạo (Artificial Neural Network – ANN) là một mô hình toán học được xây dựng dựa trên các mạng neural sinh học, bao gồm một nhóm các neural nhân tạo được kết nối với nhau để xử lý thông tin. Các neural trong ANN được tổ chức theo từng lớp. Lớp neural này sẽ kết nối với lớp neural kia thông qua các trọng số. Kiến trúc chung của một ANN gồm 3 thành phần: Lớp đầu vào (input layer), lớp ẩn (hidden layer) và lớp đầu ra (output layer).



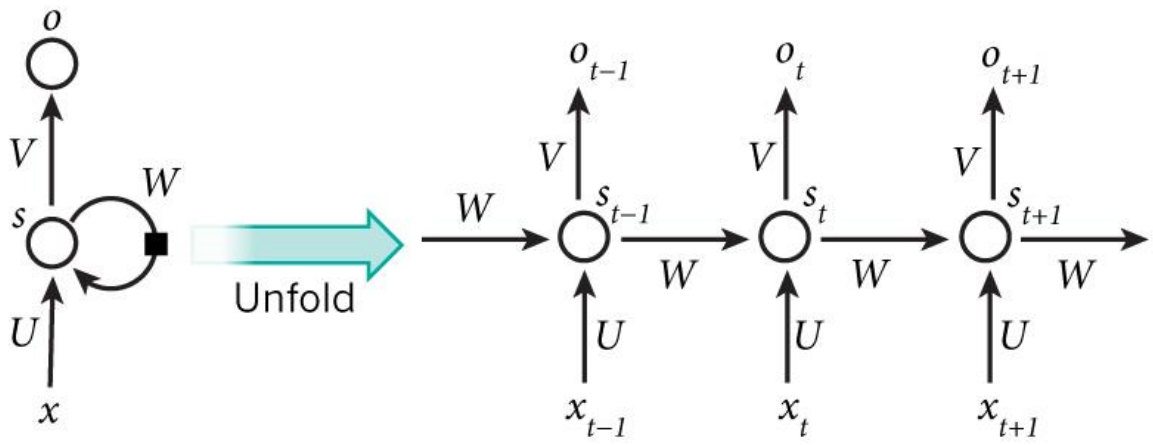
Hình 3.1: Kiến trúc cơ bản của một mạng neural nhân tạo [1]

3.2. Mạng Neural hồi quy

Mạng Neural hồi quy (RNN) là một mô hình mạng neural có bộ nhớ để lưu trữ phần thông tin đã được xử lý trước đó. Ý tưởng chính của RNN đó là thiết kế một mạng neural có khả năng xử lý được thông tin ở dạng chuỗi. RNN coi dữ liệu đầu vào là một chuỗi liên tục, nối tiếp nhau theo thứ tự thời gian và chỉ sử dụng một

[1] Nguồn: https://towardsdatascience.com/step-by-step-guide-to-building-your-own-neural-network-from-scratch-df64b1c5ab6e

mạng neural duy nhất (thường là một layer) để tính toán giá trị cho đầu ra của từng time step (mỗi time step có thể liên kết với một ký tự, từ, hoặc một câu tùy thiết lập). Về lý thuyết RNN có thể xử lý và lưu trữ thông tin của một chuỗi có chiều dài bất kỳ. Tuy nhiên trong thực tế nó chỉ hiệu quả với chuỗi có độ dài không quá lớn nguyên nhân là do vanishing gradient problem (gradient được dùng để cập nhật giá trị của ma trận trọng số trong RNN có giá trị nhỏ dần qua từng layer khi thực hiện back propagation) và exploding gradient (gradient có thể có giá trị lớn trong quá trình thực hiện back propagation khiến một số layer có giá trị cập nhật trọng số quá lớn dẫn đến phân rã).



Hình 3.2: Kiến trúc RNN

Để khắc phục nhược điểm trên, hai biến thể của RNN là LSTM (Long Short-Term Memory) và GRU (Gated Recurrent Units) đã ra đời với việc sử dụng cơ chế 'Gates' nhằm bổ sung thông tin mới và loại bỏ thông tin không cần thiết từ 'memory', từ giúp tăng khả năng lưu trữ thông tin quan trọng của RNN.

3.3. Word Embedding

Word embedding là phương pháp biểu diễn từ bằng cách ánh xạ chúng sang các vector số (thường là số thực). Word embedding được phân chủ yếu thành hai loại:

- Frequency-based embedding.
- Prediction-based embedding.

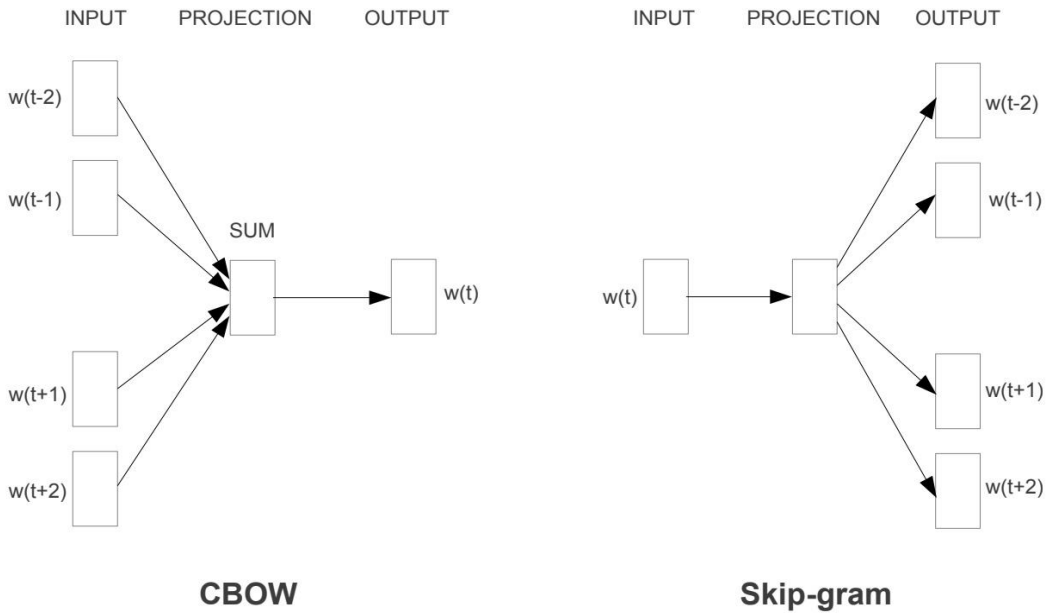
3.3.1. Frequency-based embedding

Frequency-based embedding dựa vào tần số xuất hiện của các từ để tạo ra các vector từ. Phổ biến nhất là:

- Count vector
- Tf-idf vector
- Co-occurrence matrix

3.3.2. Prediction-based embedding

Prediction-based embedding xây dựng các vector từ dựa vào các mô hình dự đoán. Tiêu biểu nhất là Word2vec, sự kết hợp giữa hai mô hình CBOW (Continious Bag Of Words) và Skip-gram. Cả hai đều được xây dựng dựa trên một mạng neural gồm ba lớp: Một lớp đầu vào, một lớp ẩn và một lớp đầu ra. Mục đích chính của mạng neural này là học các trọng số biểu diễn vector từ.



Hình 3.3: Kiến trúc cơ bản của hai mô hình CBOW và Skip-gram [2]

Sự khác nhau giữa kiến trúc của hai mô hình:

[2] Nguồn: https://arxiv.org/pdf/1309.4168v1.pdf

- CBOW:
    - Cho các từ ngữ cảnh
    - Đoán xác suất của một từ đích
- Skip-gram:
    - Cho từ đích
    - Đoán xác suất của các từ ngữ cảnh

Các giải pháp sau đó như Glove cũng tương tự như Word2vec được đề xuất bởi nhóm Pennington năm 2014.

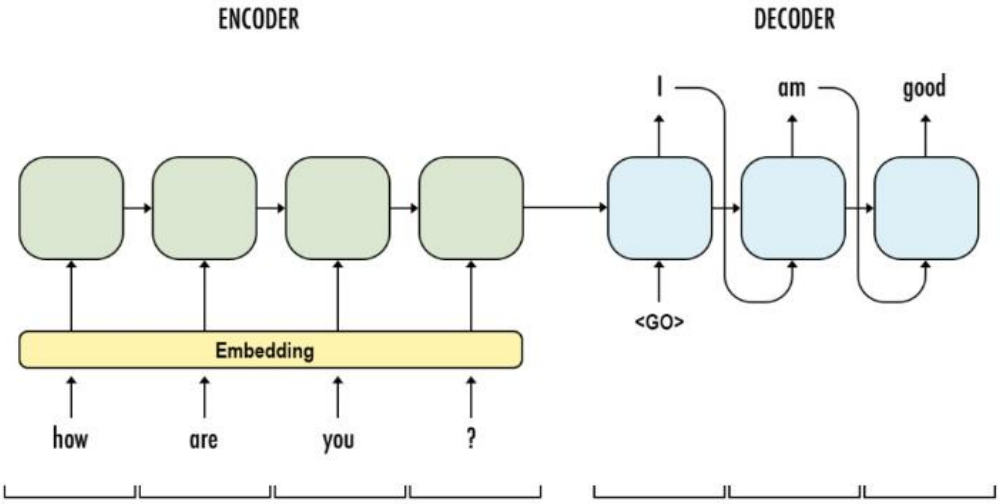
3.4. Mô hình chuỗi tuần tự (Seq2seq)

3.4.1. Giới thiệu

Mô hình seq2seq lần đầu được giới thiệu trong bài báo [3] bởi Google cho tác vụ dịch máy và đã nhanh chóng trở thành bước đột phá trong lĩnh vực dịch thuật bằng cách sử dụng mô hình học sâu. Mục đích của mô hình seq2seq là ánh xạ chuỗi đầu vào có độ dài cố định với chuỗi đầu ra có độ dài cố định trong đó độ dài của hai chuỗi có thể khác nhau. Ví dụ để dịch câu "What is your name ?" sang tiếng việt, chuỗi đầu vào sẽ gồm 4 từ và chuỗi đầu ra sẽ có 5 từ (Tên của bạn là gì ?). Và dĩ nhiên ta không thể dùng mạng LSTM thông thường để ánh xạ mỗi từ tiếng anh sang tiếng việt và mô hình seq2seq được ra đời để giải quyết vấn đề trên.

3.4.2. Cơ chế hoạt động

Mô hình seq2seq dựa trên kiến trúc mã hóa – giải mã để sinh ra chuỗi đầu vào như hình minh họa.



Hình 3.4: Mô hình seq2seq với bộ mã hóa RNN và bộ giải mã RNN [3]

Mô hình trên bao gồm 2 phần chính: Bộ mã hóa (encoder), bộ giải mã (decoder):

- **Bộ mã hóa (Encoder):**

Bộ mã hóa sẽ mã hóa thông tin của các chuỗi đầu vào với độ dài khác nhau thành một vector ngữ cảnh c. Như trong hình 4, ta có thể sử dụng RNN để thiết kế bộ giải mã.

Giả sử ta có một chuỗi đầu vào $x_1, \dots, x_T$. Trong đó x_t là từ thứ t của chuỗi trên. Tại bước thời gian t, Mô hình RNN sẽ có hai đầu vào: vector đặc trưng x_t của $\mathrm{x}_t$và trạng thái ẩn của của bước thời gian trước đó h_{t-1}. Ta ký hiệu phép chuyển đổi của các trạng thái ẩn trong RNN bằng hàm f.

$$\mathrm{h_t} = f(\mathrm{x}_t, \mathrm{h}_{t-1})$$

[3] Nguồn: https://6chaoran.wordpress.com/2019/01/15/build-a-machine-translator-using-keras-part-1-seq2seq-with-lstm/

Tiếp theo, bộ mã hóa nắm bắt thông tin của tất cả các trạng thái ẩn và mã hóa chúng thành vector ngữ cảnh c bằng hàm q.

$$\mathrm{c} = q(\mathrm{h}_1, \dots, \mathrm{h}_T)$$

Ví dụ, nếu chúng ta chọn q là $q(\mathrm{h}_1, \dots, \mathrm{h}_T) = \mathrm{h}_T$, thì vector ngữ cảnh sẽ là trạng thái ẩn của bước thời gian cuối cùng h_T.

Ở trên, ta mới chỉ sử dụng mạng RNN một chiều để thiết kế bộ giải mã, nơi mà trạng thái ẩn của mỗi bước thời gian chỉ phụ thuộc vào các bước thời gian trước. Ta cũng có thể sử dụng các dạng RNN khác nhau như GRU, LSTM, hay RNN hai chiều để mã hóa chuỗi đầu vào.

- **Bộ giải mã (Decoder):**

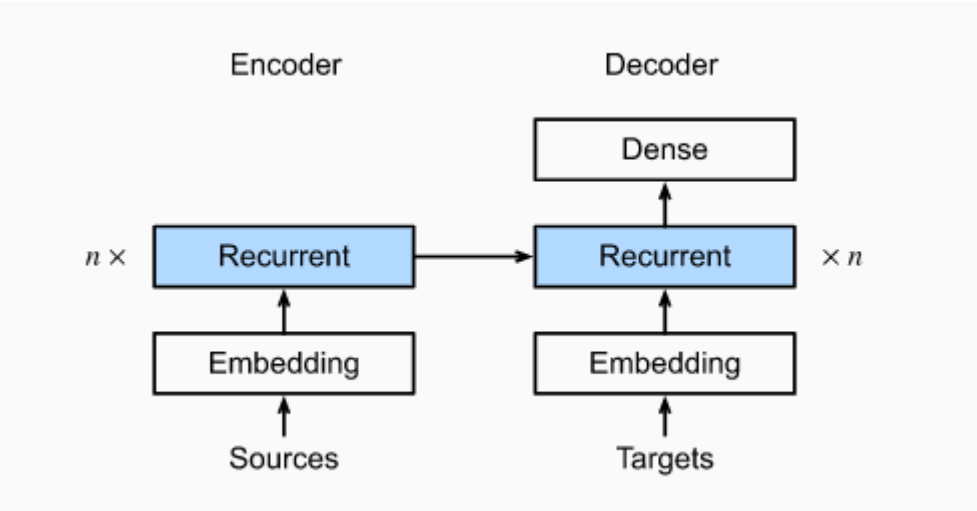
Như đã giới thiệu, vector ngữ cảnh c mã hóa thông tin của toàn bộ chuỗi đầu vào $x_1, \dots, x_T$. Giả sử đầu ra của tập huấn luyện là $y_1, y_2, \dots, y_{T'}$ Tại mỗi bước thời gian t', xác suất có điều kiện của đầu ra $y_{t'}$ sẽ phụ thuộc vào đầu ra trước đó $y_1, y_2, \dots, y_{t'-1}$và vector ngữ cảnh c, tức là $P(y_{t'}|y_1, \dots, y_{t'-1}, \mathrm{c})$.

Do đó, chúng ta có thể sử dụng một mạng RNN khác trong bộ giải mã. Tại mỗi bước thời gian t', bộ giải mã cập nhật trạng thái ẩn của nó thông qua ba đầu vào: vector đặc trưng $\mathrm{y}_{t'-1}$, vector ngữ cảnh c và trạng thái ẩn tại bước thời gian trước đó $\mathrm{s}_{t'-1}$. Hàm g dưới đây biểu diễn quá trình biến đổi trạng thái ẩn của mạng RNN trong bộ giải mã:

$$\mathrm{s}_{t'} = g(y_{t'-1}, \mathrm{c}, \mathrm{s}_{t'-1})$$

Sau khi có được trạng thái ẩn của bộ giải mã, ta có thể dùng lớp đầu ra và hàm softmax để tính toán xác suất có điều kiện $P(y_{t'}|y_1, \dots, y_{t'-1}, \mathrm{c})$ cho đầu ra tại bước thời gian t'.

Các tầng của bộ mã hóa và bộ giải mã được minh họa thông qua hình 3.5 bên dưới.



Hình 3.5: Các tầng trong bộ mã hóa và bộ giải mã sử dụng RNN [4]

3.4.3. Ứng dụng

Ngày nay, mô hình seq2seq đã được ứng dụng trong rất nhiều tác vụ khác nhau như: Chú thích hình ảnh (image captioning), mô hình trò chuyện (conversational models), tóm tắt văn bản (text summarization),…

3.5. Cơ chế chú ý (Attention Mechanisms)

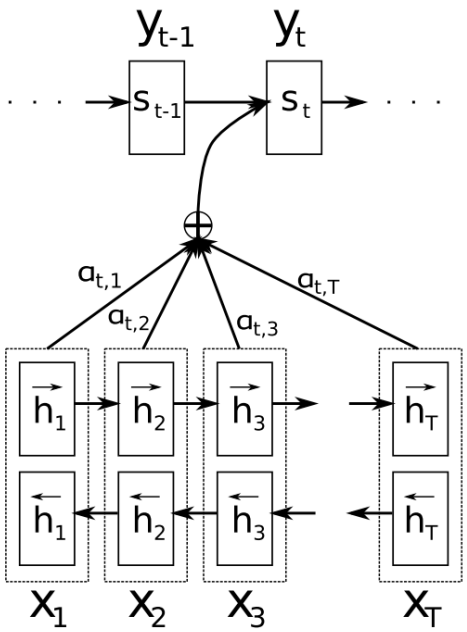
3.5.1. Vấn đề của mô hình chuỗi sang chuỗi cơ bản (Vanilla Seq2seq)

Như đã giới thiệu ở phần 3.5, mục đích của mô hình seq2seq là mã hóa thông tin của chuỗi đầu vào thành trạng thái ẩn và truyền nó tới bộ giải mã để sinh ra chuỗi đích. Một nhược điểm quan trọng của mô hình trên là là vector ngữ cảnh có độ dài cố định không có khả năng ghi nhớ những câu dài. Nó thường quên đi phần đầu tiên sau khi xử lý hoàn toàn câu đâu vào. Cơ chế chú ý (attention mechanisms) được sinh ra [2] để giải quyết vấn đề trên.

[4] Nguồn: https://d2l.ai/chapter_recurrent-modern/seq2seq.html

3.5.2. Cơ chế chú ý (attention mechanisms)

Cơ chế chú ý giúp ghi nhớ các câu nguồn dài. Thay vì xây dựng một vector ngữ cảnh duy nhất từ trạng thái ẩn cuối cùng của bộ giải mã, cơ chế chú ý giữ lại và sử dụng tất cả trạng thái ẩn từ chuỗi đầu vào trong quá trình giải mã.



Hình 3.6: Mô hình seq2seq với cơ chế chú ý [2]

Giả sử, ta có một chuỗi đầu vào x có chiều dài là n và cố gắng tạo ra một chuỗi đầu ra y có chiều dài m:

$$\mathrm{x} = [x_1, x_2, \dots, x_n]$$

$$\mathrm{y} = [y_1, y_2, \dots, y_m]$$

Hình 3.6, bộ giải mã là một RNN hai chiều (hoặc ta cũng có thể sử dụng một RNN một chiều) với trạng thái ẩn xuôi là $\overrightarrow{h_i}$ và trạng thái ẩn ngược là $\overleftarrow{h_i}$. Để biểu diễn trạng thái ẩn của một từ, ta sẽ nối trạng thái ẩn ở hai đầu lại.

$$\mathrm{h}_i = [\overrightarrow{\mathrm{h}_i}^{\mathrm{T}}, \overleftarrow{\mathrm{h}_i}^{\mathrm{T}}]^{\mathrm{T}}, i = 1, \dots, n$$

Trạng thái ẩn của bộ giải mã cho đầu ra tại vị trí t là $s_t = f(s_{t-1}, y_{t-1}, c_t)$, với $t = 1, \dots, m$. Vector ngữ cảnh c_t là tổng trọng số các trạng thái ẩn đầu vào tại bước thời gian t được tính bằng:

$$c_t = \sum_{i=1}^{n} \alpha_{t,i} h_i \qquad ; vector\ bối\ cảnh\ của\ đầu\ ra\ y_t$$

Với $\alpha_{t,i}$ là trọng số biểu thị mức độ chú ý của từng trạng thái ẩn h_i của bộ giải mã. Trọng số bởi $\alpha_{t,i}$ hay còn gọi là alignment score:

$$\alpha_{t,i} = \text{align}(y_t, x_i) = \frac{\exp(a(s_{t-1}, h_i))}{\sum_{k=1}^{n} \exp(a(s_{t-1}, h_k))}$$

$a(s_{t-1}, h_i)$ hay còn gọi là alignment model được sử dụng để đánh giá mức độ tương quan giữa hai từ y_t và x_i bằng việc gán một trọng số $\alpha_{t,i}$.

Trong bài báo [3] của tác giả Bahdanau, alignment model α được tính bởi công thức sau:

$$\text{score}(s_t, h_i) = a(s_t, h_i) = v_a^T \tanh(W_a[s_t; h_i])$$

Trong đó v_a và W_a là hai ma trận trọng số được được học từ alignment model.

Tóm lại, sau khi cho trạng thái ẩn s_{t-1} của bộ giải mã và trạng thái ẩn h_i của bộ mã hóa vào alignment model a, ta thu được các alignment score, sau đó chuẩn hóa bằng hàm softmax để tổng của các attention score bằng 1, thu được các $\alpha_{t,i}$ và cuối cùng tính tổng các tích $\alpha_{t,i}$ và h_i để thu được vector ngữ cảnh c_t dùng làm đầu vào của bộ giải mã để dự đoán từ y_t.

3.5.3. Một số cách tính Alignment score khác

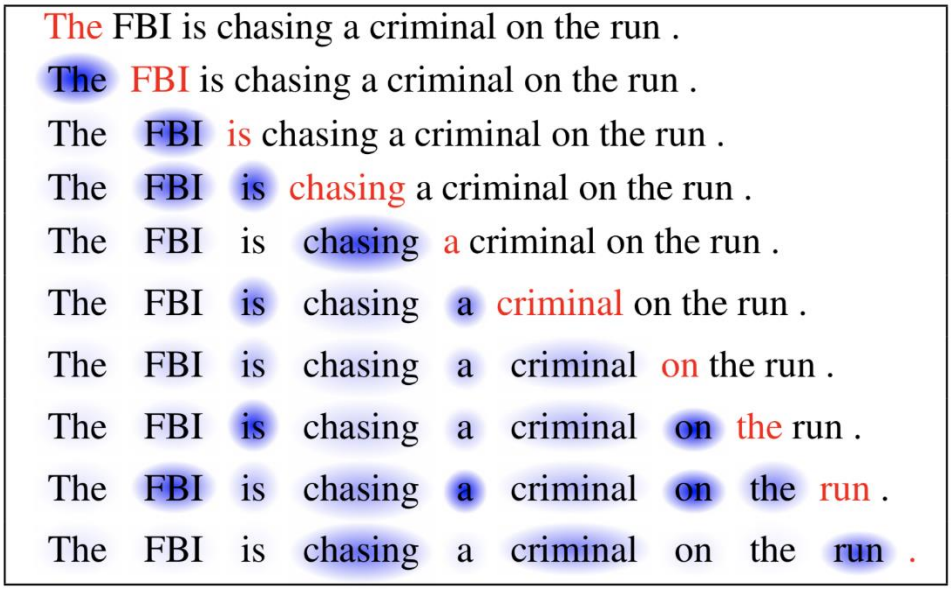
Tên	Hàm tính điểm alignment
Content-base attention [4]	$\text{score}(s_t, h_i) = \text{cosine}[s_t, h_i]$
Additive [2]	$\text{score}(s_t, h_i) = v_a^T \tanh(W_a[s_t; h_i])$
Location-base [12]	$\alpha_{t,i} = softmax(W_a s_t)$
General [12]	$\text{score}(s_t, h_i) = s_t^T W_a h_i$

	W_a là ma trận trọng số được huấn luyện tại tầng chú ý
Dot-Product [12]	$\text{score}(s_t, h_i) = s_t^T h_i$
Scaled Dot-Product[5] [21]	$\text{score}(s_t, h_i) = \frac{s_t^T h_i}{\sqrt{n}}$ n là chiều của trạng thái ẩn tại nguồn

3.5.4. Tự chú ý (Self Attention)

Tự chú ý hay còn gọi là chú ý nội bộ, là một cơ chế chú ý liên hệ các vị trí khác nhau của một chuỗi đơn lẽ để tính toán biểu diễn cùng một chuỗi. Nó rất hữu ích trong các nhiệm vụ tóm tắt trừu tượng hoặc mô tả hình ảnh. Cơ chế tự chú ý sẽ được mô tả rõ hơn trong phần giới thiệu Transformer.

Trong ví dụ dưới đây, cơ chế tự chú ý cho phép chúng ta học được mối tương quan giữa các từ hiện tại với phần trước của câu.



Hình 3.7: Từ hiện tại có màu đỏ, màu xanh cho biết mức độ kích hoạt [6]

[5] Hệ số $1/\sqrt{n}$ thêm vào để tránh trường hợp đẩy hàm softmax vào vùng có gradient cực nhỏ khi đầu vào lớn, ảnh hưởng đến độ hiểu quả khi mô hình học tập.

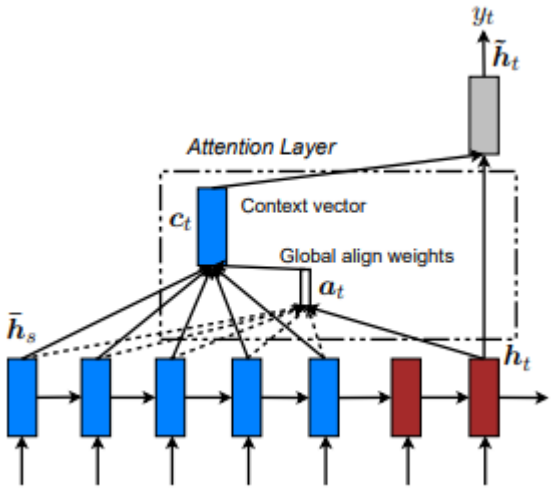
[6] Nguồn: https://arxiv.org/pdf/1601.06733.pdf

3.5.5. Chú ý cứng và mềm (Soft vs Hard Attention)

Theo như bài báo của Luong và các cộng sự [12] và được mô tả trong bài báo của Xu và các cộng sự [29]. Trong chú ý mềm, vector ngữ cảnh được tính bằng cách lấy tổng trọng số của trạng thái ẩn của bộ mã hóa. Đối với chú ý cứng, thay vì tính trung bình trọng số của tất cả trạng thái ẩn, ta sẽ sử dụng điểm số chú ý (attention score) để chọn một trạng thái ẩn. Để thực hiện việc lựa chọn trên, ta có thể đơn giản sử dụng hàm argmax nhưng nó sẽ khiến hàm số không khả vi do đó cần các kỹ thuật khác phức tạp hơn được sử dụng. Chi tiết về cách tính của hai loại cơ chế chú ý trên được mô tả cụ thể trong [29] nên đề tài sẽ không đi sâu vào phần này.

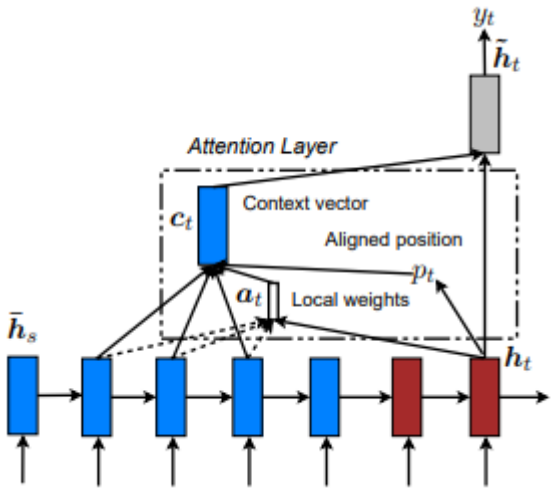
3.5.6. Chú ý toàn cục và chú ý cục bộ (Global vs Local attention)

Được đề xuất trong [12] bởi Luong và các cộng sự. Chú ý toàn cục tương tự như chú ý mềm. Trong khi đó, chú ý cục bộ lại là sự pha trộn giữa chú ý cứng và mềm, cải tiến cơ chế chú ý cứng để khiến nó khả vi. Cụ thể:



Hình 3.8: Mô hình chú ý toàn cục [12]

Mô hình chú ý toàn cục: Tại mỗi bước thời gian t, mô hình suy ra một alignment vector a_t có độ dài thay đổi bằng cách so sánh trạng thái ẩn đích hiện tại h_t với mỗi trạng thái ẩn nguồn $\bar{\mathrm{h}}_s$. Vector ngữ cảnh được tính bằng cách lấy trung bình trọng số dựa vào a_t qua tất cả các trạng thái nguồn.



Hình 3.9: Mô hình chú ý cục bộ [12]

Mô hình chú ý cục bộ: Đầu tiên mô hình dự đoán một aligned position p_t cho mỗi từ mục tiêu tại bước thời gian t. Sau đó vector ngữ cảnh c_t được tính bằng cách lấy trung bình trọng số của trạng thái nguồn nằm trong đoạn $[p_t - D, p_t + D]$. D được chọn theo kinh nghiệm. Không như mô hình chú ý toàn cục, alignment vector a_t có chiều cố định.

3.6. Mô hình Transformer

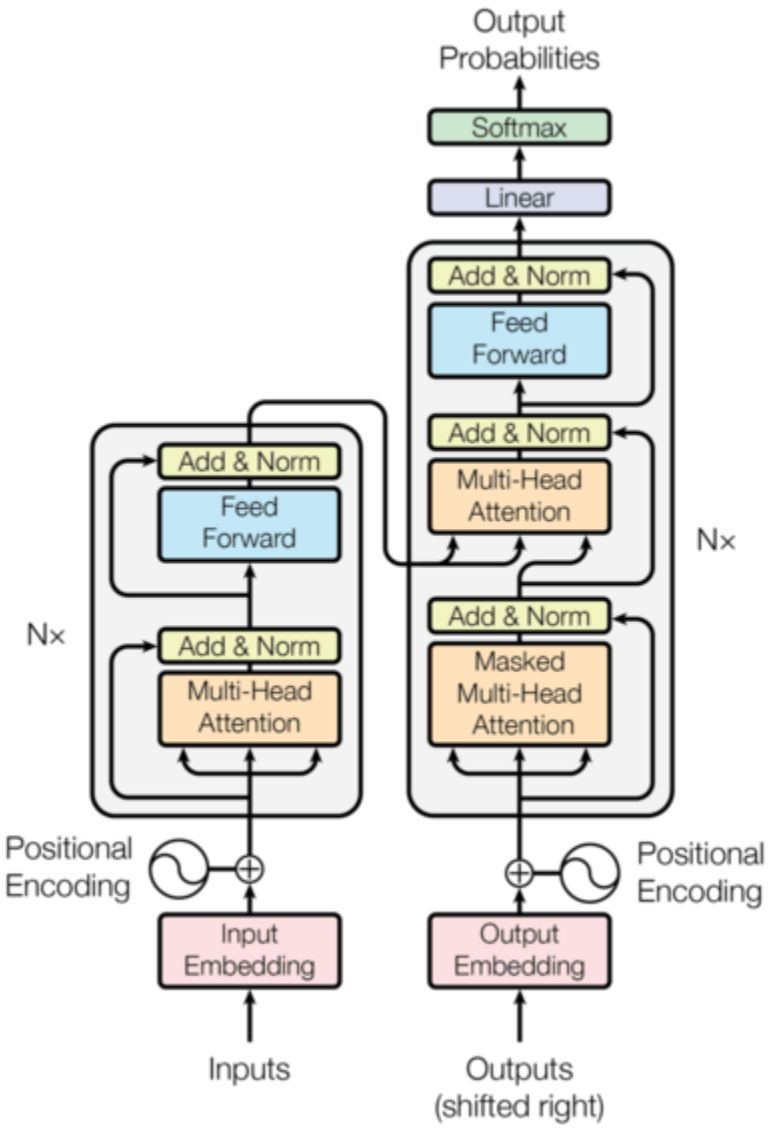
3.6.1. Giới thiệu

“Attention is All you need” [21], là một trong những bài báo khoa học gây ảnh hưởng lớn đến lĩnh vực học máy trong những năm gần đây. Bài báo đã trình bày rất nhiều cải tiến đối với sự chú ý mềm (soft attention) và có thể mô hình hóa từ chuỗi sang chuỗi mà không cần đến mạng RNN. Mô hình “Transformer” được bài báo đề xuất hoàn toàn được xây dựng trên cơ chế tự chú ý (self attention) mà không sử dụng kiến trúc RNN hoặc mạng tích chập.

3.6.2. Kiến trúc mô hình

Tương tự như mô hình seq2seq trong mục 3.5, mô hình Transformer cũng dựa trên kiến trúc mã hóa - giải mã. Tuy nhiên, nó thay thế các tầng nối tiếp trong mô hình seq2seq bằng các tầng chú ý đa đầu (multi-head attention), kết hợp thông tin vị trí thông qua mã hóa vị trí (positional

encoding) và áp dụng một lớp chuẩn hóa (layer normalization). Hình 10 minh họa cấu trúc tổng quát của một mô hình Transformer.



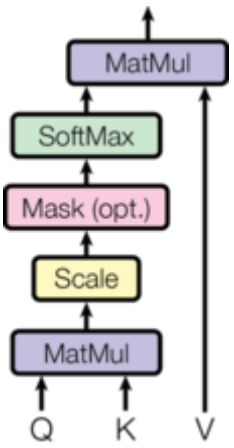
Hình 3.10: Kiến trúc tổng quát của mô hình Transformer [21]

Phần tiếp theo sẽ mô tả chi tiết về các thành phần trong mô hình Transformer.

3.6.2.1. Scaled Dot-Product Attention

Transformer sử dụng "Scaled Dot-Product Attention" cho tầng chú ý đa đầu. Đầu vào sẽ bao gồm các vector truy vấn và khóa có chiều là d_k, và các giá trị có chiều dài là d_v. Các vector truy vấn, khóa và giá trị này được tạo ra bằng cách nhân vector embedding đầu vào với 3 ma

trận tương ứng đã được huấn luyện trong quá trình đào tạo. Cụ thể: Ta ký hiệu embedding vector cho từng token tương ứng là $\mathbf{x}_t$, $W^Q \in \mathbb{R}^{d_{model} \times d_q}, W^k \in \mathbb{R}^{d_{model} \times d_k}, W^V \in \mathbb{R}^{d_{model} \times d_v}$ là các ma trận trọng số tương ứng với d_{model} kích thước của embedding đầu vào = 512, T^t là chiều dài của chuỗi với $t = 1, \dots, T^t$. Giá trị của ba vector lần lượt là: $q_t = \mathrm{x}_t \mathrm{W}^{\mathrm{Q}}$, $k_t = \mathrm{x}_t \mathrm{W}^{\mathrm{K}}$, $v_t = \mathrm{x}_t \mathrm{W}^{\mathrm{V}}$. Để lấy được trọng số trên các giá trị ta tính tích vô hướng của câu truy vấn với tất cả các khóa và chia cho $\sqrt{d_k}$, rồi đưa qua một hàm softmax.



Hình 3.11: Scaled Dot-Product Attention [21]

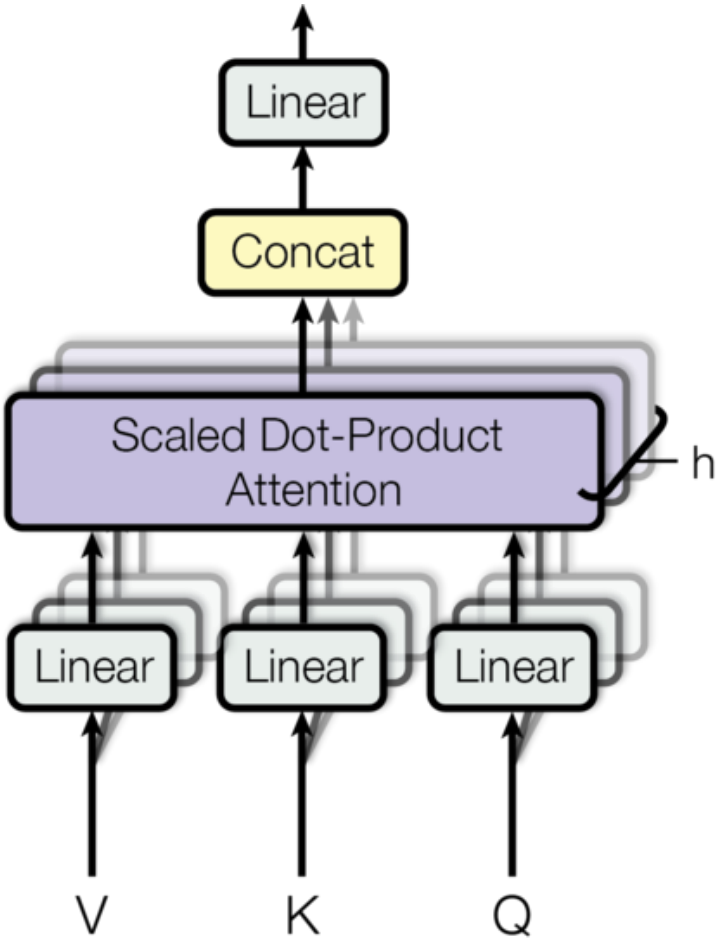
Trong thực tế, hàm chú ý được tính toán trên một tập hợp các truy vấn đồng thời trong một ma trận Q. Các khóa và giá trị lần lượt là các ma trận K và V. Ma trận đầu ra sẽ là:

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

Hai hàm chú ý được sử dụng phổ biến nhất là additive attention [2], và dot-product attention. Trong đó, dot-product attention tương tự với scaled dot-product attention ngoại trừ hệ số tỉ lệ $\frac{1}{\sqrt{d_k}}$. Hệ số tỉ lệ dùng để ngăn dot products lớn dần trong trường hợp d_k lớn và đẩy hàm softmax vào vùng có gradient cực nhỏ ảnh hưởng đến độ hiệu quả của mô hình.

3.6.2.2. Chú ý đa đầu (Multi-head Attention)

Tầng chú ý đa đầu bao gồm h đầu là các tầng tự chú ý song song. Thay vì tính toán sự chú ý một lần, cơ chế chú ý đa đầu tính toán song song các tầng tự chú ý nhiều lần. Trước khi đưa vào mỗi đầu, các ma trận truy vấn, khóa và giá trị sẽ được nhân với ba ma trận trọng số tương ứng với kích thước ẩn lần lượt là p_q, p_k và p_v trong đó $p_q = p_k = p_v = d_{model}$ (d_{model} là kích thước đầu vào). Đầu ra của h đầu này được nối lại với nhau và sau đó được xử lý bởi một tầng dày đặc (dense layer) cuối cùng.



Hình 3.12: Cơ chế chú ý đa đầu [21]

Giả sử: Gọi chiều của các câu truy vấn, khóa và giá trị lần lượt là d_q, d_k và d_v. Khi đó tại mỗi đầu $i = 1, \dots, h$, ta có các ma trận tham số lần lượt là

$W_i^Q \in \mathbb{R}^{d_{model} \times d_q}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ và $W^O \in \mathbb{R}^{d_{model} \times hd_v}$. Tầng chú ý đa đầu được tính bằng công thức:

$$Multihead(Q, K, V) = Concat(head_1, \dots head_h)W^O$$

Trong đó:

$$head_i = Attention(QW_i^Q, K\,W_i^K, VW_i^V)$$

Sau đó h đầu ra với độ dài d_v tại mỗi đầu được nối với nhau thành đầu ra có độ dài hd_v rồi đưa vào tầng dày đặc cuối cùng với d_{model} các nút ẩn và các trọng số của tầng này được ký hiệu là $W^O \in \mathbb{R}^{d_{model} \times hd_v}$. Trong bài báo Attention is all you need [21] tác giả đã sử dụng $h = 8$ đầu ra. Với mỗi đầu, chiều của các câu truy vấn, khóa và giá trị sẽ giảm xuống $d_k = d_v = \frac{d_{model}}{h} =$ 64 do đó tổng chi phí tính toán cũng tương tự với chi phí tính toán cơ chế chú ý một đầu với đầy đủ chiều.

3.6.2.3. Mạng truyền xuôi theo vị trí (Position-wise feed-forward networks)

Ngoài các lớp chú ý, thì mỗi lớp trong trong bộ mã hóa và bộ giải mã đều có một mạng truyền xuôi kết nối đầy đủ (fully connected feed-forward network), bao gồm hai tầng dày đặc áp dụng trên chiều cuối cùng của đầu vào với hàm kích hoạt ReLU ở giữa.

$$FFN = \max(0, xW_1 + b_1)\,W_2 + b_2$$

Vì hai tầng dày đặc này cùng được sử dụng cho từng vị trí trong chuỗi, nên ta gọi là mạng truyền xuôi theo vị trí (posotion-wise feed-forward).

3.6.2.4. Mã hóa vị trí (Position encoding)

Không giống như tầng hồi tiếp, cả tầng chú ý đa đầu và mạng truyền xuôi theo vị trí đều tính toán đầu ra cho từng token trong chuỗi một cách độc lập. Điều này cho phép song song hóa công việc tính toán nhưng lại không mô hình hóa được thông tin tuần tự trong chuỗi đầu vào. Để nắm bắt các thông tin tuần tự một cách hiệu quả, mô hình Transformer sử dụng biểu diễn vị trí (positional encoding) để duy trì thông tin vị trí của chuỗi đầu vào.

Giả sử $X \in \mathbb{R}^{l \times d}$ là embedding của mẫu đầu vào, PE là ma trận vị trí của X, trong đó l là độ dài chuỗi, d là kích thước của embedding. Tầng mã hóa vị trí sẽ mã hóa vị trí của $PE \in \mathbb{R}^{l \times d}$ của X và trả về đầu ra là $PE + X$

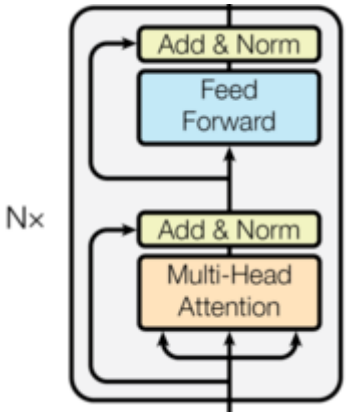
Vị trí của PE là ma trận hai chiều, trong đó pos là thứ tự trong câu, i là vị trí theo chiều của embedding. Bằng cách này, mỗi vị trí trong chuỗi ban đầu được biểu diễn bởi hai phương trình dưới đây:

$$PE_{(pos,2i)} = \sin(pos/1000^{2i/d})$$

$$PE_{(pos,2i+1)} = \cos(pos/1000^{2i/d})$$

3.6.2.5. Bộ mã hóa (Encoder)

Bộ mã hóa của mô hình Transformer là một ngăn xếp có $N = 6$ lớp giống hệt nhau.

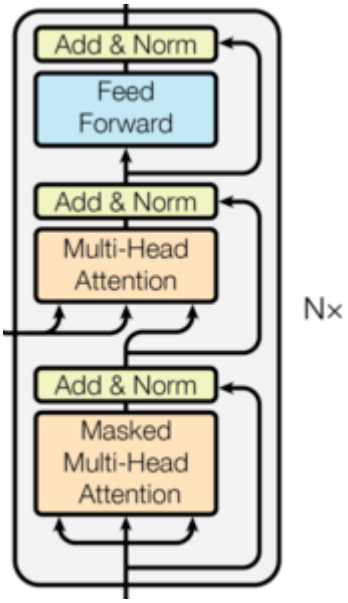


Hình 3.13: Hình ảnh bộ mã hóa được cắt từ mô hình Transformer

Mỗi lớp mã hóa này chứa các lớp con bao gồm: một tầng chú ý đa đầu, một mạng truyền xuôi theo vị trí. Mỗi lớp con này đều có một kết nối dư (residual connections) xung quanh, theo sau là một lớp chuẩn hóa. Đầu ra của mỗi lớp con là $LayerNorm(x + Sublayer(x))$.

3.6.2.6. Bộ giải mã (Decoder)

Bộ giải mã cũng tương tự như bộ mã hóa bao gồm một ngăn xếp có $N = 6$ lớp giống hệt nhau. Tuy nhiên, bên cạnh hai lớp con (tầng chú ý đa đầu và biểu diễn vị trí), khối giải mã còn chứa tầng chú ý đa đầu áp dụng lên đầu ra của bộ mã hóa. Các lớp con này, tương tự bộ mã hóa đều có một kết nối dư xung quanh và theo sau là một lớp chuẩn hóa.



Hình 3.14: Hình ảnh bộ giải mã được cắt từ mô hình Transformer

Về cách hoạt động, bộ giải mã sẽ dự đoán từ kế tiếp bằng cách quan sát đầu ra của bộ mã hóa và tự điều chỉnh đầu ra của chính nó.

3.6.2.7. Các ứng dụng của cơ chế chú ý trong mô hình Transformer

Transformer sử dụng cơ chế chú ý nhiều đầu theo ba cách khác nhau:

- Tại tầng tập trung đa đầu áp dụng lên đầu ra của bộ mã hóa tại bộ giải mã, các truy vấn đến từ các lớp giải mã (decoder

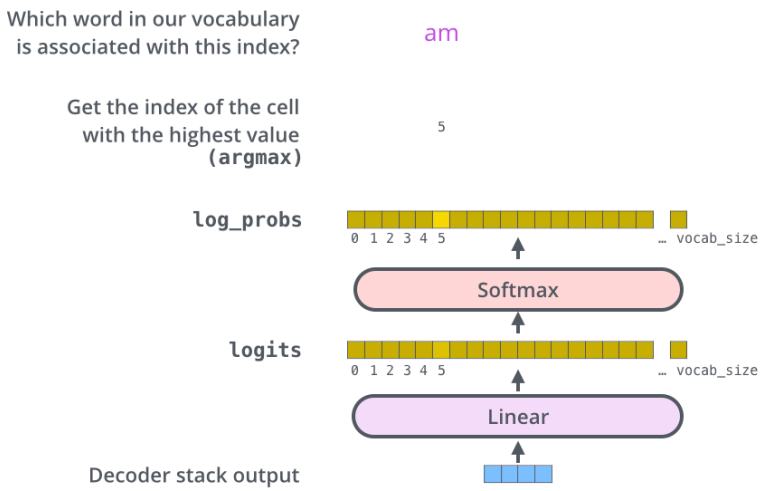
layer) trước đó và các cặp khóa và giá trị đến từ đầu ra của bộ mã hóa cho phép mọi vị trí trong bộ giải mã có thể chú ý đến tất cả các vị trí trong chuỗi đầu vào. Điều này tương tự như cơ chế chú ý trong các mô hình chuỗi tuần tự.

- Tại cơ chế tự chú ý trong bộ mã hóa, tất cả các truy vấn, khóa và giá trị đến từ đầu ra của lớp trước đó. Mỗi vị trí trong bộ mã hóa đều có thể chú ý đến tất cả các vị trí của lớp trước đó trong bộ mã hóa.
- Trong bộ giải mã, Cơ chế tự chú ý hoạt động hơi khác so với bộ mã hóa. Nó chỉ cho phép chú ý vào các vị trí trước đó trong chuỗi đầu ra bằng cách che các vị trí sau đó trước bước softmax trong phép tính tự chú ý.

3.6.2.8. Tầng tuyến tính cuối cùng và tầng softmax (The Final Linear and softmax Layer)

Đầu ra của bộ giải mã là một vector số thực. Để chuyển vector trên thành một từ chính là nhiệm vụ của hai lớp cuối cùng này.

Cụ thể, tầng tuyến tính là một mạng neural được kết nối đầy đủ có nhiệm vụ chiếu vector được tạo ra bởi bộ giải mã thành một vector lớn hơn rất nhiều được gọi là vector logits. Vector này có độ dài bằng độ dài bộ từ vựng (vocabulary) được học từ tập dữ liệu đào tạo, mỗi ô tương đương với điểm số của từng từ duy nhất. Tiếp theo tầng softmax sẽ biến những điểm số trên thành xác suất. Ô nào có xác suất cao nhất sẽ được chọn và từ liên kết với được tạo ra làm đầu ra cho bước thời gian này.



Hình 3.15: Minh họa quá trình tạo ra từ vựng đầu ra thông qua bộ giải mã [7]

3.7. Các chiến lược giải mã trong mô hình ngôn ngữ

Hai cơ chế giải mã phổ biến để xây dựng các chuỗi câu từ các token được dự đoán trước đó trong mô hình tạo văn bản hiện nay là tìm kiếm tham lam (Greedy search) và tìm kiếm chùm tia (beam search).

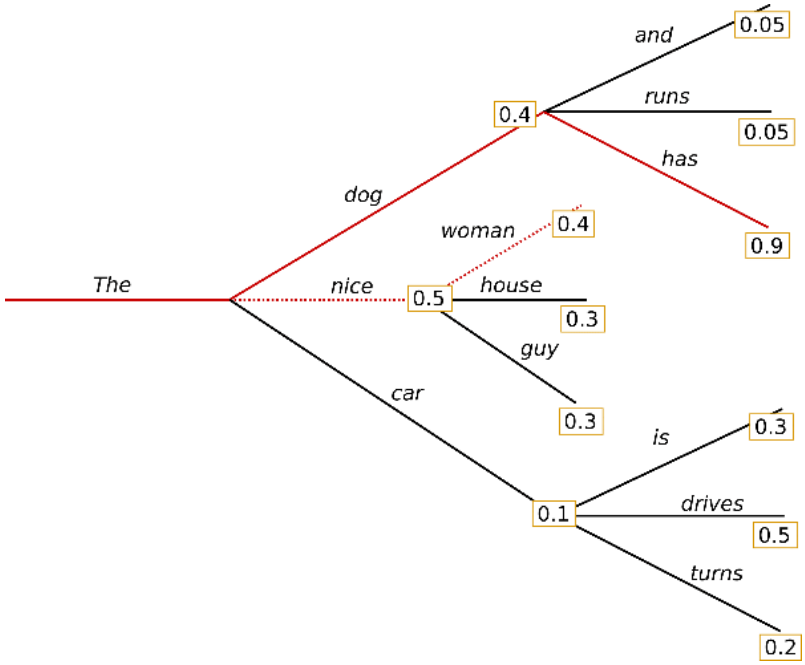
Greedy decoding hay giải mã sử dụng thuật toán tìm kiếm tham lam là cách đơn giản nhất để dự đoán token của chuỗi đầu ra tại mỗi bước trong quá trình giải mã. Ở mỗi bước thời gian, Ta sẽ chọn token tiếp theo có xác xuất cao nhất dựa trên phân phối xác suất đầu ra của mô hình cho đến khi dự đoán được token kết thúc chuỗi <EOS>. Rủi ro của giải thuật này là có thể bỏ sót các token có xác suất cao ẩn sau các token trước đó có xác suất thấp dẫn đến chuỗi đầu ra không phải là chuỗi có xác suất cao nhất.

Beam search hay giải mã sử dụng thuật toán tìm kiếm chùm tia là một thuật toán cải tiến dựa trên tìm kiếm tham lam. Giải thuật này có một siêu tham số k gọi là kích thước chùm (beam size). Tại bước khởi tạo, ta chọn k token có xác suất có điều kiện cao nhất theo xác suất đầu ra của mạng để bắt đầu k chuỗi đầu ra các ứng viên. Tại các bước thời gian tiếp theo, ta tính và chọn k chuỗi có xác suất cao nhất

[7] Nguồn: http://jalammar.github.io/illustrated-transformer/

trong tổng số $k|\mathrm{Y}|$ khả năng. Trong đó Y là bộ từ vựng đầu ra bao gồm cả token <EOS>, |Y| là kích thước bộ từ vựng, $\boldsymbol{c}$ là vector ngữ cảnh chứa thông tin của chuỗi đầu vào. Xác xuất mỗi chuỗi có được bằng cách tại mỗi bước thời gian t' ta tính

$P(y_1, \ldots, y_{t'}|\mathrm{c}) = P(y_1|\mathrm{c})P(y_2|\mathrm{c}, y_1) \ldots P(y_t|\mathrm{c}, y_1, \ldots y_{t'-1})$. Kết thúc quá trình ta sẽ chọn câu có điểm số cao nhất trong số các chùm làm chuỗi đầu ra. Trong những năm qua tìm kiếm chùm tia là thuật toán giải mã tiêu chuẩn cho hầu hết các tác vụ sinh ngôn ngữ bao gồm các hệ thống trò chuyện.



Hình 3.16: Minh họa tìm kiếm chùm tia [8]

Tuy nhiên, ngày càng có nhiều bằng chứng cho thấy rằng các phương pháp giải mã dựa trên việc tối đa hóa như tìm kiếm chùm tia dẫn đến việc thoái hóa. Ví dụ như văn bản được sinh ra nhạt nhẽo, không mạch lạc hoặc bị mắc kẹt trong các vòng lặp đi lặp lại. Để giải quyết vấn đề trên, người ta đã thay thế cơ chế giải mã trên bằng cách lấy mẫu (sampling) từ phân phối token tại mỗi bước thời gian như **top-k sampling** - trong đó bộ giải mã chỉ lấy mẫu từ top-k token có khả năng xảy ra cao nhất, hay nghiên cứu gần đây nhất là **nucleus (top-p) sampling** của Ari

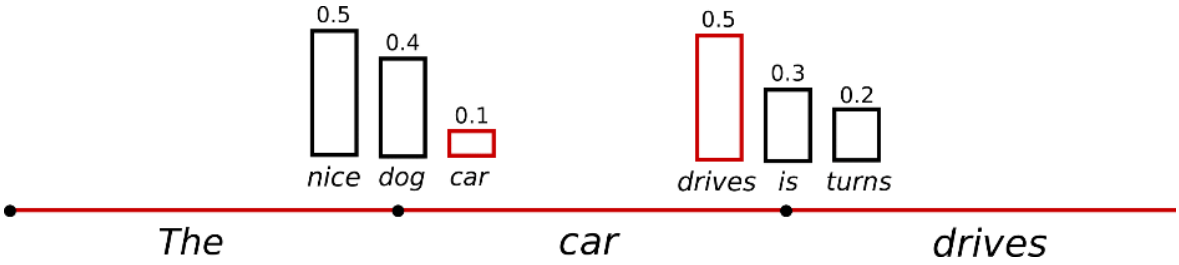
[8] Nguồn: https://huggingface.co/blog/how-to-generate

Holtzman và các cộng sự [5]. - lấy mẫu từ phân phối của token tiếp theo sau khi đã lọc phân phối này để giữ các token hàng đầu với xác suất tích lũy nằm ở trên ngưỡng (Threshold) cụ thể.

Ở dạng cơ bản nhất, lấy mẫu nghĩa là chọn ngẫu nhiên từ tiếp theo w_t dựa vào phân phối xác suất có điều kiện của nó:

$$w_t \sim P(w|w_{1:t-1})$$

Hình minh họa dưới đây trực quan hóa việc tạo ngôn ngữ khi lấy mẫu



Hình 3.17: Trực quan hóa quá trình lấy mẫu [9]

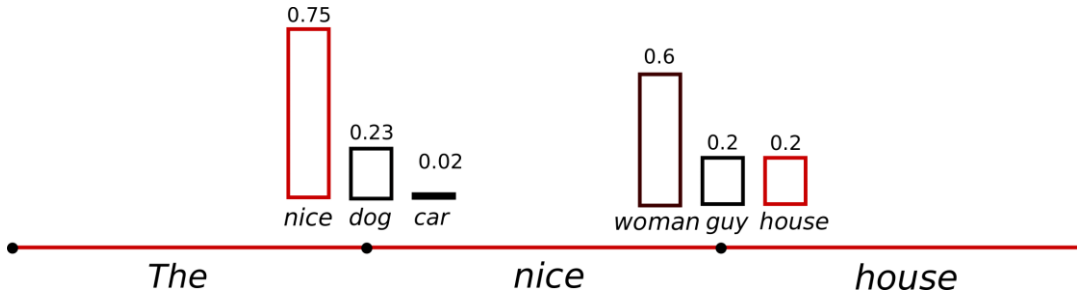
Từ "car" được lấ mẫu từ xác xuất có điều kiện $P(w|\text{"The"})$. Theo sau đó là từ "drive" được lấy mẫu từ $P(w|\text{"The","car"})$.

Sau khi thực nghiệm, mô hình thường tạo ra những câu vô nghĩa và không mạch lạc khi sử dụng phương pháp lấy mẫu bằng cách chọn ngẫu nhiên (chi tiết được mô tả trong [5]). Một mẹo nhỏ để cải thiện là khiến phân phối xác suất $P(w|w_{1:t-1})$ sắc nét hơn (Tăng sự xuất hiện của các từ có xác suất cao và giảm khả năng xuất hiện của các từ có xác suất thấp) bằng cách hạ thấp **nhiệt độ (temperature)** của softmax. Trong mô hình xác suất, ta thực hiện việc lấy mẫu nhiệt độ bằng cách chia logits cho temperature trước khi đưa vào hàm softmax được mô tả qua phương trình sau:

$$P(w = V_l|w_{1:i-1}) = \frac{\exp(\frac{u_l}{t})}{\sum_{l'} \exp(\frac{u_l'}{t})}$$

[9] Nguồn: https://huggingface.co/blog/how-to-generate

Trong đó logits được ký hiệu là $u_{1:|V|}$, t là nhiệt độ, V là bộ từ vựng. Hạ nhiệt độ xuống khiến mô hình tự tin vào những lựa chọn hàng đầu. Trong khi nhiệt độ lớn hơn 1 làm giảm độ tin cậy của mô hình. Nhiệt độ bằng 0 tương đương với argmax/max likelihood, và ∞ tương đương với việc lấy mẫu đồng nhất. Những phân tích gần đây cho thấy rằng, hạ nhiệt độ giúp cải thiện chất lượng văn bản được khi tạo ra, nhưng đổi lại tính đa dạng văn bản khi tạo ra sẽ bị giảm.



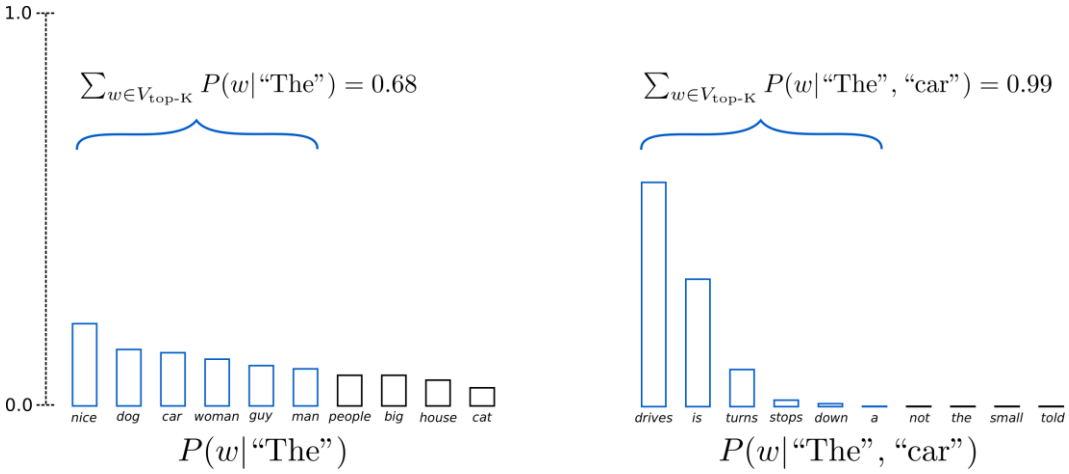
Hình 3.18: Hình minh họa áp dụng nhiệt độ cho ví dụ trên[10]

Phân phối có điều kiện của các từ kế tiếp tại bước thời gian $t = 1$, trong đó "nice" trở nên sắc hơn và thay thế "car" trong chuỗi đầu tiên.

Mặc dù áp dụng nhiệt độ khiến phân phối ít ngẫu nhiên hơn, trong giới hạn của nó, khi đặt nhiệt độ gần về 0, việc lấy mẫu theo tỷ lệ nhiệt độ sẽ tương đương với giải mã tham lam và gặp các vấn đề tương tự.

Top-K sampling: Trong lấy mẫu Top-K, K từ tiếp theo có nhiều khả năng nhất sẽ được lọc và độ lớn xác suất được phân phối lại trong K từ tiếp theo.

[10] Nguồn: https://huggingface.co/blog/how-to-generate



Hình 3.19: Minh họa lấy mẫu theo top-k [11]

Với K = 6, trong cả hai bước đều giới hạn lấy mẫu là 6 từ. Giải thuật đã thành công trong việc loại bỏ những ứng cử viên không phù hợp (“not" , “the" , “small", “told") trong bước lấy mẫu thứ hai.

Tuy nhiên, hạn chế của lấy mẫu theo top-K là không điều chỉnh linh hoạt được số lượng từ được lọc từ phân phối xác suất của từ tiếp theo $P(w|w_{1:t-1})$. Cụ thể:

Ở bước $t = 1$, Top-K đã loại bỏ những từ có tiềm năng là ứng cử viên hợp lý như (“People”, “pig”, “house”, “cat”). Mặc khác, ở bước t = 2, top-K vẫn bao gồm các từ được cho là không phù hợp (“down”, “a”) trong mẫu đã lấy ra. Do đó giới hạn không gian mẫu bằng một kích thước cố định K có thể khiến mô hình tạo ra những câu vô nghĩa.

Nucleus (top–p) sampling hay lấy mẫu hạt nhân được Ari Holtzman và công sự giới thiệu nhằm khắc phụ nhược điểm trên của lấy mẫu Top-K. Ý tưởng chính của phương pháp này là sử dụng hình dạng của phân phối xác suất để xác định tập hợp các token sẽ được lấy mẫu. Cho một phân phối $P(x|x_{1:i-1})$, ta định nghĩa top-p từ vựng của nó là $V^{(p)} \subset V$ là tập hợp nhỏ nhất các token có tổng xác suất có điều kiện của các phần tử trog tập lớn hơn một ngưỡng p hay là:

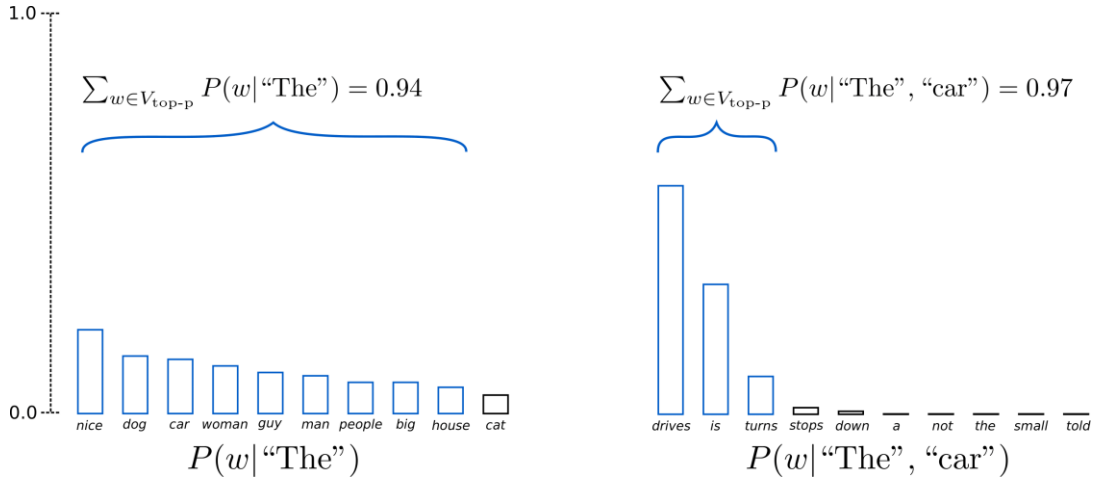
[11] Nguồn: https://huggingface.co/blog/how-to-generate

$$\sum_{x \in V^{(p)}} P(x|x_{1:i-1}) \geq p$$

Phân phối xác suất các token đã chọn ra được chuẩn hóa lại để đảm bảo tổng bằng 1 theo công thức sau:

$$P'(x|x_{1:i-1}) = \begin{cases} P(x|x_{1:i-1})/p' & if\ x \in V^{(p)} \\ 0 & ngược\ lại \end{cases}$$

Trong đó $p' = \sum_{x \in V^{(p)}} P(x|x_{1:i-1})$. Trong thực tế, điều này có nghĩa là chọn token có xác suất cao nhất có tổng độ lớn xác suất tích lũy vượt quá ngưỡng đã chọn trước p. Kích thước của tập hợp mẫu sẽ được điều chỉnh tự động dựa vào hình dạng của phân phối xác suất tại mỗi bước thời gian. Hình sau đây minh họa cho quá trình lấy mẫu.



Hình 3.20: Minh họa lấy mẫu hạt nhân [12]

Với $p = 0.92$, lấy mẫu hạt nhân chọn số từ tối thiểu có độ lớn xác suất vượt qua p. Trong ví dụ đầu tiên có đến 9 từ có khả năng xảy ra nhất, trong khi đó tại ví dụ thứ hai, nó chỉ cần chọn ra top 3 từ để có độ lớn xác suất vượt qua p. Có thể thấy rằng, giải thuật sẽ chọn một lượng lớn các từ nếu từ tiếp theo được cho là khó dự đoán hơn ví dụ như: $P(w|\text{"The"})$, và chỉ một vài từ khi từ tiếp theo có vẻ dễ đoán hơn ví dụ như: $P(w|\text{"The"}, \text{"}car\text{"})$.

[12] Nguồn: https://huggingface.co/blog/how-to-generate

Dựa vào ưu điểm trên, đề tài sẽ sử dụng thuật toán lấy mẫu hạt nhân (nucleus sampling) để làm cơ chế giải mã cho chatbot.

3.8. Học chuyển tiếp (Transfer learning)

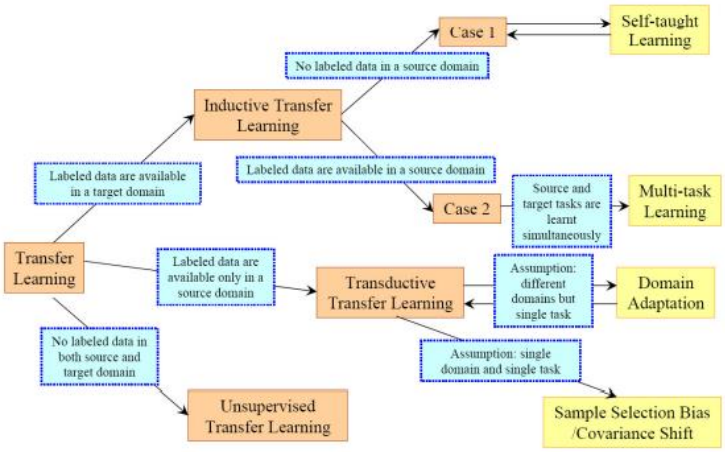
3.8.1. Giới thiệu

3.8.1.1. Định nghĩa

Học chuyển tiếp là kỹ thuật cho phép tận dụng những gì mô hình học được từ tập dữ liệu, ứng dụng, kiến trúc này sang tập dữ liệu, ứng dụng, kiến trúc khác có liên quan và cho phép cải thiện tốc độ và hiệu suất khi thực hiện nhiệm vụ khác

3.8.1.2. Các chiến lược và kỹ thuật trong học chuyển tiếp

- Phân loại theo miền dữ liệu (domains)
    - Inductive transfer learning: khác biệt về nhiệm vụ (task)
    - Transductive transfer learning: giống nhau về nhiệm vụ nhưng khác biệt về miền dữ liệu
    - Unsupervised transfer learning. Giống inductive transfer learning nhưng nhiệm vụ đích ở dạng un-supervised learning



Hình 3.21: Các chiến lược trong học chuyển tiếp [13]

- Phân loại theo hướng tiếp cận
    - Instance transfer. Dùng lại tri thức từ dữ liệu gốc (source domain) từ một số trường hợp/ví dụ chính.
    - Feature representation transfer. Giảm thiểu sự khác biệt giữa 2 miền dữ liệu.
    - Parameter transfer. Dựa trên giả thiết các nhiệm vụ liên hệ với nhau có cùng phân bố về siêu tham số
    - Relational - Knowledge transfer. Hướng tới giải quyết vấn đề khi dữ liệu không độc lập và phân bố ngẫu nhiên

	Inductive Transfer Learning	Transductive Transfer Learning	Unsupervised Transfer Learning
Instance-transfer	√	√	
Feature-representation-transfer	√	√	√
Parameter-transfer	√		
Relational-knowledge-transfer	√		

Hình 3.22: Liên hệ giữa phân loại theo hướng tiếp cận và domain

3.8.2. Các chiến lược học chuyển tiếp trong mô học sâu

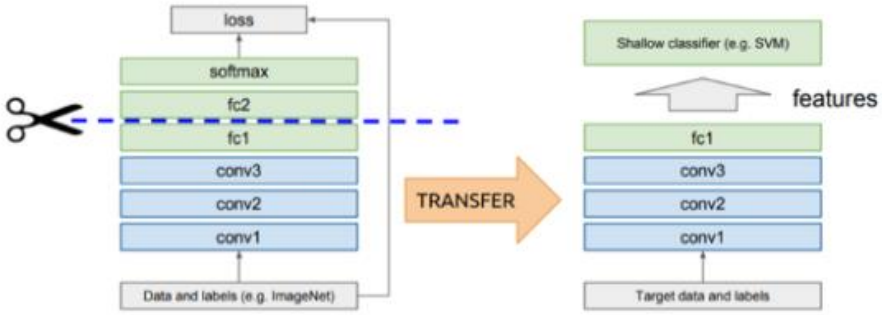
Học sâu đã đạt được những tiến bộ đáng kể trong những năm gần đây. Phương pháp này cho phép giải quyết những vấn đề phức tạp và mang lại một kết quả đáng kinh ngạc. Tuy nhiên thời gian đào tạo và dữ liệu cần thiết

[13] Nguồn: https://www.cse.ust.hk/~qyang/Docs/2009/tkde_transfer_learning.pdf

để huấn luyện cần nhiều hơn so với các phương pháp học máy truyền thống. Ngày càng có nhiều các mô hình học sâu hiện đại được thử nghiệm và phát triển và đạt được hiệu suất rất tốt. Hầu hết các trường hợp, các nhóm nghiên cứu chia sẻ các mạng/mô hình của mình được huấn luyện trên một nhiệm vụ cụ thể cho người khác sử dụng. Các mạng/mô hình được đào tạo sẵn này trở thành nền tảng của học chuyển tiếp trong mô hình học sâu. Dưới đây là hai chiến lược phổ biến học chuyển tiếp trong mô hình học sâu.

3.8.2.1. Trích xuất đặc trưng từ mô hình mẫu đã được huấn luyện (Off-the-shelf Pre-trained Models as Feature Extractors)

Các hệ mô hình học sâu có kiến trúc nhiều lớp có thể học các đặc trưng khác nhau từ các phân lớp khác nhau. Những lớp này sẽ kết nối với lớp cuối cùng thường là lớp kết nối đầy đủ để có được đầu ra cuối cùng. Kiến trúc này cho phép tận dụng mạng đã được đào tạo từ trước mà không có lớp cuối cùng như một tính năng cố định cho các tác vụ khác.



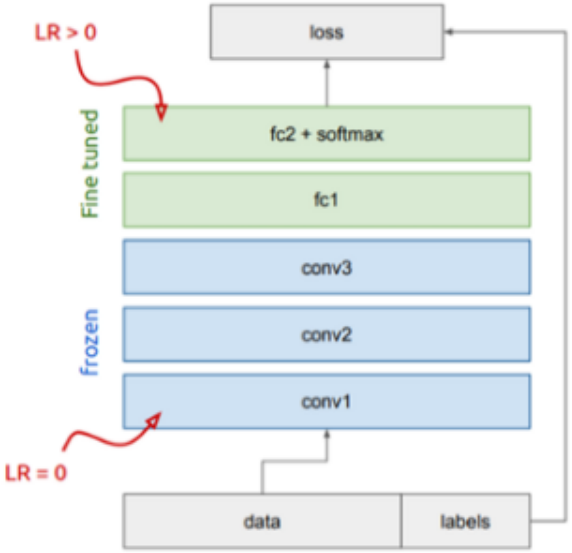
Hình 3.23: Trích xuất các đặc trưng từ mô hình đã được huấn luyện cho mô hình với tác vụ khác

Ý tưởng chính: tận dụng trọng số của các lớp của mô hình mẫu đã huấn luyện trước để trích xuất các tính năng nhưng không cập nhật trọng số của các lớp này trong quá trình huấn luyện dữ liệu mới cho nhiệm vụ mới.

3.8.2.2. Tinh chỉnh mô hình mẫu đã được huấn luyện trước (Fine Tuning Off-the-shelf Pre-trained Models)

Kỹ thuật này không chỉ thay thế các lớp cuối cùng mà còn đào tạo lại một cách có chọn lọc một số lớp trước đó. Trong mạng neural sâu, các lớp ban đầu sẽ nắm bắt các đặc trưng chung trong khi đó các lớp sau sẽ tập trung nhiều hơn vào các đặc trưng cụ thể nhờ đó ta có thể đóng băng (freeze) hay giữ nguyên trọng số các lớp nhất định khi đào tạo lại hoặc tinh chỉnh (fine-tune) phần còn lại để phù hợp với nhu cầu bài toán.

Vậy khi nào cần đóng băng hoặc tinh chỉnh các lớp?



Hình 3.24: Khi nào cần đóng băng hoặc tinh chỉnh các lớp

Trong hình trên, các lớp dưới cùng đều có thể được tinh chỉnh (cập nhật trọng số trong quá trình backprop) hoặc đóng băng (giữ nguyên trọng số trong quá trình backprop) tùy thuộc vào yêu cầu và dữ liệu bài toán:

- Đóng băng: Nếu dữ liệu quá nhỏ và tương tự với dữ liệu của pre-trained model và ta muốn tránh trường hợp overfitting.
- Tinh chỉnh: Dữ liệu lớn và tương tự với pre-trained model.

Pre-trained Models (Mô hình mẫu đã được huấn luyện trước): Một trong những yêu cầu cơ bản đối với học chuyển tiếp là những mô hình làm tốt nhiệm vụ nguồn. Các mô hình này trải dài trên nhiều lĩnh vực khác nhau. Xử lý ngôn ngữ tự nhiên và thị giác máy tính là hai lĩnh vực phổ biến nhất cho các ứng dụng học sâu. Mô hình mẫu đã được huấn luyện trước thường được chia sẻ dưới dạng hàng triệu tham số/ trọng số mà mô hình đạt được khi huấn luyện đến trạng thái ổn định trên nhiệm vụ cụ thể khác và được chia sẻ với người sử dụng thông qua nhiều phương diện khác nhau thông qua các thư viện học sâu nổi tiếng như Keras, Hugging-face hoặc từ các trang web vì hầu hết chúng đều là các mã nguồn mở.

Một số pre-trained model phộ biến: BERT, GPT, GPT2, DialoGPT XLM,…

3.9. Tổng quan về mô hình ngôn ngữ OpenAI GPT – GPT2

Mô hình GPT (Generative Pre-training Transformer) là một mô hình ngôn ngữ được viết bởi Alec Radford và được giới thiệu lần đầu trong [14]. Bài báo này để xuất học một mô hình ngôn ngữ chung bằng cách sử dụng dữ liệu không được gán nhãn sau đó tinh chỉnh mô hình để phù hợp với các tác vụ cụ thể. Sau đây ta sẽ đi vào chi tiết mô hình.

3.9.1. Cấu trúc và chi tiết mô hình

3.9.1.1. Mô hình GPT

Sử dụng duy nhất 12 lớp giải mã với cơ chế tự chú ý được che lại để huấn luyện mô hình ngôn ngữ. Dưới đây là chi tiết của mô hình.

- **Huấn luyện không giám sát:**
  + Sử dụng Byte pair encoding (BPE) [17] với 40,000 kết hợp được sử dụng.

+ Mô hình sử dụng 768 chiều để mã hóa token thành word embedding. Position embedding cũng được học trong quá trình huấn luyện.
+ Có 12 lớp giải mã với 12 đầu chú ý mỗi lớp.
+ Lớp truyền xuôi theo vị trí có 3072 chiều.
+ Sử dụng hàm tối ưu hóa Adam với tỉ lệ học tập là 2.5e-4.
+ Tỉ lệ Drop out = 0.1 cho mọi lớp.
+ GELU được sử dụng làm hàm kích hoạt.
+ Mô hình được đào tạo trong 100 kỷ nguyên với kích thước lô là 64. Độ dài mỗi chuỗi là 512. Mô hình có tổng cộng 117M tham số.

- **Tinh chỉnh mô hình thông qua học có giám sát:**
  + Tinh chỉnh thông qua học có giám sát mất khoảng 3 epoch cho hầu hết các nhiệm vụ downstream (Là những nhiệm vụ học tập có giám sát sử dụng mô hình hoặc thành phần được đào tạo trước). Điều này cho thấy rằng mô hình đã học được rất nhiều về ngôn ngữ trong quá trình tiền huấn luyện. Do đó, chỉ cần tinh chỉnh tối thiểu là đủ.
  + Hầu hết các siêu tham số từ quá trình tiền huấn luyện đều được sử dụng trong quá trình tinh chỉnh.

3.9.1.2. Mô hình GPT 2

GPT 2 có 1.5 tỉ tham số hơn 10 lần so với GPT 1 (117 triệu tham số). Sự khác biệt chính so với GPT 1 là:

- GPT 2 có 48 lớp và sử dụng vector có 1600 chiều cho word embedding.
- Bộ từ vựng lớn hơn gồm 50,027 token.
- Batch size là 512 và khung ngữ cảnh gồm 1024 token lớn hơn so với GPT.

- Lớp chuẩn hóa được chuyển đến đầu vào của mỗi khối con và một lớp chuẩn hóa bổ sung được thêm vào sau khối tự chú ý cuối cùng.
- Lúc khởi tạo, trọng số của những lớp dư (residual layer) được chia tỉ lệ cho $1/\sqrt{N}$. Trong đó N là số lớp dư.

3.9.2. Dữ liệu

GPT 1: được huấn luyện trên bộ dữ liệu BookCorpus [35] gồm 7000 cuốn sách chưa xuất bản giúp huấn luyện mô hình trên dữ liệu không gán nhãn. Ngoài ra, kho dữ liệu này còn có những đoạn văn bản dài liền kề giúp mô hình học các phụ thuộc xa (Nhớ được thông tin dài hạn trước đó).

GPT 2: được huấn luyện trên bộ dữ liệu WebText có 40GB dữ liệu văn bản từ 8 triệu tài liệu.

3.9.3. Huấn luyện

3.9.3.1. Tiền huấn luyện mô hình sử dụng học không giám sát

Đối với học không giám sát, GPT sử dụng hàm mục tiêu sau:

$$L_1(T) = \sum_i logP(t_i|t_{i-k}, \dots, t_{i-1}; \theta) \quad (3.1)$$

Trong đó T là tập hợp các token trong kho dữ liệu không giám sát $T = \{t_1, \dots, t_n\}$, n là kích thước của T, k là kích thước của khung ngữ cảnh, θ là tham số của mạng neural được huấn luyện sử dụng SGD.

Cụ thể, ta sẽ sử dụng Transformer Decoder nhiều lớp cho mô hình hóa ngôn ngữ. Mô hình này sử dụng cơ chế chú ý đa đầu nhận các token ngữ cảnh đầu vào từ lớp Text&Position Embedding theo sau đó là mạng truyền xuôi theo vị trí để tạo ra phân phối xác suất đầu ra trên các token:

$$\mathrm{h}_0 = UW_e + W_p$$

$$\mathrm{h}_l = TransformerBlock_{(\mathrm{h}_{l-1})} \forall l \in [1, n]$$

$$P(u) = softmax(\mathrm{h}_n W_e^T)$$

Trong đó $U = (u_{-k}, \dots, u_{-1})$ là vector ngữ cảnh của token, n là số lớp, W_e là ma trận word embedding và W_p là ma trận vị trí embedding.

3.9.3.2. Tinh chỉnh mô hình thông qua học có giám sát

Sau khi huấn luyện mô hình với hàm mục tiêu trong công thức (3.1), ta sẽ điều chỉnh các tham số cho các nhiệm vụ có giám sát. Giả sử có một tập dữ liệu có nhãn C, trong đó bao gồm chuỗi các token đầu vào $x^1, .. x^m$, tương ứng với nhãn y. Đầu vào được đưa vào pre-trained model để có được vector đầu ra của khối giải mã cuối cùng h_l^m , sau đó qua một lớp tuyến tính để dự đoán y:

$$P(y|x^1, \dots x^m) = softmax(\mathrm{h}_l^m W_y)$$

Điều này dẫn đến hàm mục tiêu ta cần tối đa hóa là:

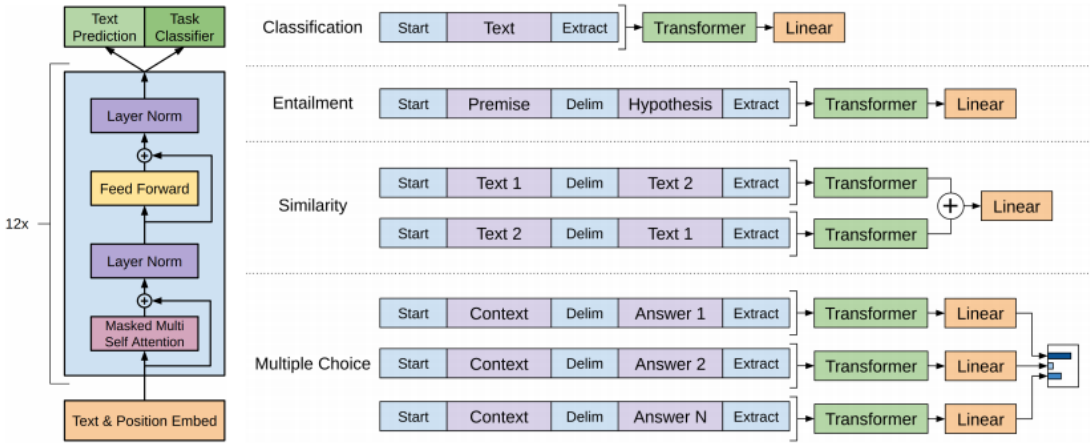
$$L_2(C) = \sum_{(x,y)} logP(y|x^1, \dots, x^m) \quad (3.2)$$

Thay vì chỉ tối đa hóa hàm mục tiêu ở phương trình (3.2), tác giả đã sử dụng thêm auxiliary learning objective [15] để tinh chỉnh có giám sát nhằm tổng quát hóa tốt hơn và hội tụ nhanh hơn. Hàm mục tiêu sau khi thêm auxiliary learning object trở thành:

$$L_3(C) = L_2(C) + \lambda L_1(C)$$

Trong đó $L_1(C)$ là hàm mục tiêu phụ trợ (auxiliary objective) của mô hình ngôn ngữ. λ là trọng số của $L_1(C)$. λ = 0.5.

Nhìn chung, những tham số yêu cầu trong quá trình tinh chỉnh là W_y, và embedding của những token phân tách.



Hình 3.25: Cấu trúc Transformer và mục tiêu đào tạo của mô hình GPT (**Trái**). Chuyển đổi đầu vào trong quá trình tinh chỉnh ở các tác vụ khác nhau (**phải**). [14]

3.10. Các thư viện hỗ trợ

3.10.1. Pytorch

Hình 3.26: Logo của Pytorch

Pytorch là một thư viện máy học mã nguồn mở dựa trên thư viện Torch, được phát triển bởi Facebook. Pytorch đơn giản, dễ dàng học, sử dụng, mở rộng và sửa lỗi. Nó cung cấp hai tính năng cao cấp: Tính toán trên các tensor với tốc độ mạnh mẽ nhờ sự hỗ trợ của GPU và khả năng xây dựng mô hình mạng neural trên tape-based autograd systems. Mặc dù là thư viện dành cho python, nhưng phần lớn Pytorch được viết bằng C++ với khả năng tính toán song song trên nhiều GPU.

Ngoài ra Pytorch còn cung cấp khả năng chạy trực tiếp từ C++ hỗ trợ tốc độ cho việc triển khai mô hình trên sản phẩm.

3.10.2. Thư viện Transformer HuggingFace

Hình 3.27: Logo của HuggingFace

HuggingFace Transformer là một thư viện python cung các API để sử dụng nhiều kiến trúc Transformer nổi tiếng như BERT, RoBERTa, GPT-2 hoặc DistilBERT, và đã đạt được nhiều kết quả tiên tiến trong nhiều nhiệm vụ NLP như: Phân lớp văn bản, trích xuất thông tin, hệ thống hỏi đáp, tạo văn bản. Thư viện ban đầu phát triển trên Pytorch và gần đây cũng đã được chuyển sang Tensorflow 2.0.

3.10.3. Flask

Hình 3.28: Logo của Flask

Flask là một web frameworks, nó thuộc loại micro-framework được xây dựng bằng ngôn ngữ lập trình Python. Flask cho phép xây dựng các ứng dụng web từ đơn giản tới phức tạp. Nó có thể xây dựng các api nhỏ, ứng dụng web chẳng hạn như các trang web, blog hay thậm chí là một trang web

thương mại. Flask cung cấp cho bạn công cụ, các thư viện và các công nghệ mạnh mẽ để hỗ trợ nhưng công việc trên.

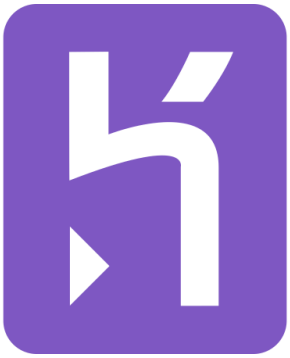
3.10.4.Google Colab

Hình 3.29: Logo của Google Colab

Google Colab (Google Colaboratory) là một dịch vụ miễn phí của Google nhằm hỗ trợ nghiên cứu và học tập về trí tuệ nhân tạo. Colaboratory cung cấp môi trường như Jupyter Notebook và có thể sử dụng GPU và CPU miễn phí.

Đối với các lập trình viên không có điều kiện để sở hữu một bộ máy tính với tài nguyên đủ nhanh và mạnh, thì Google Colab là một lựa chọn tuyệt vời, không chỉ giúp cho lập trình viên giải quyết vấn đề về tài nguyên phần cứng, mà bên cạnh đó là sự hỗ trợ đầy đủ về các thư viện và công cụ lập trình.

3.10.5.Heroku



Hình 3.30: Logo của Heroku

Heroku là một Nền tảng đám mây dựa trên ứng dụng container dưới dạng dịch vụ paaS (Platform as a Service) cho phép người dùng sử dụng platform (môi trường phát triển) cho ứng dụng thông qua hệ thống mạng. Các nhà phát triển sử dụng Heroku để triển khai, quản lý và mở rộng các ứng dụng hiện đại. Nền tảng của họ rất linh hoạt và dễ sử dụng, cung cấp cho các nhà phát triển con đường đơn giản nhất để đưa ứng dụng của họ ra thị trường.

Heroku được quản lý hoàn toàn bởi Heroku, cho phép các nhà phát triển tự do tập trung vào sản phẩm cốt lõi của họ mà không bị phân tâm trong việc duy trì máy chủ, phần cứng hoặc cơ sở hạ tầng.

Chương 4. ỨNG DỤNG PHƯƠNG PHÁP TRANSFERTRANSFO ĐỂ XÂY DỰNG CHATBOT

4.1. Giới thiệu

Năm 2015, Vinyals và Le (2015) [22] đã cho thấy mô hình của họ có thể tạo ra các câu phản hồi có ý nghĩa. Tiếp sau đó cũng có nhiều nghiên cứu về kiến trúc mô hình trên để phát triển nhưng đều gặp nhiều hạn chế như:

- Đầu ra không nhất quán, tính cách không nhất quán [11].
- Không có ký ức dài hạn.
- Có xu hướng tạo ra các phản hồi chung chung (vd: I don’t know), gây nhàm chán với con người [10].

Trong đề tài này, khóa luận sẽ xây dựng chatbot sử dụng phương pháp tiếp cận TransferTransfo được giới thiệu trong bài báo [25]. Một phương pháp kết hợp giữa học chuyển tiếp và mô hình Transformer có thể cải thiện đáng kể so với các mô hình cơ sở truyền thống về mặt ngữ pháp, mức độ liên quan của câu trả lời và sự nhất quán về tích cách, lịch sử hội thoại. Việc tinh chỉnh được thực hiện bằng cách sử dụng học đa tác vụ kết hợp với một số tác vụ dự đoán không giám sát.

4.2. Dữ liệu

Bộ dữ liệu được sử dụng là PERSONA-CHAT [31] được lấy từ cuộc thi ConvAI2, một cuộc thi được tổ chức tại hội nghị NIPS 2018. Đây là một trong những bộ dữ liệu hội thoại lớn nhất (10.000 cuộc trò chuyện và khoảng 100.000 lượt) trong đó mỗi người nói được chỉ định một hồ sơ cá nhân bằng một vài câu mô tả bản thân. Các cặp sẽ được yêu cầu trò chuyện để tìm hiểu lẫn nhau như ví dụ ở hình 15 dưới đây.

Persona 1	Persona 2
I like to ski	I am an artist
My wife does not like me anymore	I have four children
I have went to Mexico 4 times this year	I recently got a cat
I hate Mexican food	I enjoy walking for exercise
I like to eat cheetos	I love watching Game of Thrones

[PERSON 1:] Hi
[PERSON 2:] Hello ! How are you today ?
[PERSON 1:] I am good thank you , how are you.
[PERSON 2:] Great, thanks ! My children and I were just about to watch Game of Thrones.
[PERSON 1:] Nice ! How old are your children?
[PERSON 2:] I have four that range in age from 10 to 21. You?
[PERSON 1:] I do not have children at the moment.
[PERSON 2:] That just means you get to keep all the popcorn for yourself.
[PERSON 1:] And Cheetos at the moment!
[PERSON 2:] Good choice. Do you watch Game of Thrones?
[PERSON 1:] No, I do not have much time for TV.
[PERSON 2:] I usually spend my time painting: but, I love the show.

Hình 4.1: Ví dụ về một đoạn hồi thoại trong tập dữ liệu PERSONA-CHAT [31]

4.3. Phương pháp đánh giá

Các độ đo tự động được sử dụng để đánh giá mô hình lần lượt là Perplexity, Hits@1, F1 được cung cấp bởi thư viện ParlAI.

- Perplexity (PPL): Dùng để đánh giá nhiệm vụ mô hình hóa ngôn ngữ, được tính bằng cách dự đoán xác suất từ tiếp theo nằm trong câu trả lời phù hợp (câu trả lời được viết bởi con người)
  + Giá trị càng thấp càng tốt
  + Giá trị nằm trong khoảng (∞,0)

  Perplexity của một câu s được tính bằng công thức sau:

$$Perplexity(s) = \sqrt[n]{\frac{1}{p(w_1 w_2 \dots w_n)}}$$

  Trong đó, n là số từ có trong câu, $p(w_i)$ xác suất của từ w_i xuất hiện
- Hits@1: Tương tự như độ chính xác thông thường. Dùng để đánh giá độ chính xác của mô hình trong việc truy xuất câu phản hồi phù hợp trong số 19 câu phản hồi ngẫu nhiên lấy từ những cuộc hội thoại khác
  + Giá trị càng cao càng tốt
  + Giá trị nằm trong nửa khoảng (0,1]

- F1 score: dùng để đánh giá xem có bao nhiêu từ nội dung (content words) (danh từ/động từ/…) của câu trả lời được tạo ra bởi mô hình nằm trong câu trả lời phù hợp.
  + Giá trị càng cao càng tốt
  + Giá trị nằm trong nửa khoảng (0,1]

  F1 score được tính bằng công thức sau:

$$F1\ score = 2\frac{Precision.recall}{Precision + recall}$$

  Precision: Xem xét có bao nhiêu dữ liệu mô hình dự đoán đúng.

$$Precision = \frac{Số\ lượng\ dự\ đoán\ đúng}{Tổng\ số\ dữ\ liệu\ tập\ kiểm\ thử}$$

  Recall: Tỉ lệ dự đoán đúng trên tổng số dữ liệu thực sự đúng.

$$Recall = \frac{Số\ lượng\ dự\ đoán\ đúng}{Tổng\ số\ dữ\ liệu\ tập\ kiểm\ thử}$$

4.4. Mô hình

Mô hình tạo văn bản được sử dụng trong TransferTransfo (sự kết hợp giữa **Transfer** learning và mô hình **Transfo**rmer) dựa trên Generative Pre-trained Transformer (GPT) bởi Radford [14]. Theo đó TransferTransfo sử dụng mô hình Transformer với duy nhất 12 bộ giải mã cùng với các đầu tự chú ý được che lại (masked self-attention heads) (768 chiều và 12 đầu tự chú ý). Chi tiết mô hình được giới thiệu ở mục 3.9.

4.5. Huấn luyện

Vì tập dữ liệu dùng để huấn luyện quá nhỏ và không đủ để có thể tạo ra các phản hồi mạch lạc và có nghĩa. Do đó nên bắt đầu với một mô hình ngôn ngữ đã được huấn luyện trên một kho dữ liệu lớn để có thể tạo ra các đoạn văn bản rõ ràng và mạch lạc sau đó tiến hành tinh chỉnh mô hình ngôn ngữ trên để phù hợp với tác vụ hội thoại. Việc huấn luyện bao gồm các bước sau.

4.5.1. Bước tiền huấn luyện (Pre-training)

Mô hình được sử dụng trong khóa luận dựa trên mô hình GPT [14] được huấn luyện sẵn trên bộ dữ liệu BookCorpus [37] bao gồm 7000 cuốn sách chưa xuất bản (khoảng 800 triệu từ) từ nhiều thể loại khác nhau như (Phiêu lưu, giả tưởng, tình cảm,…). Ngoài ra do được huấn luyện sẵn trên bộ dữ liệu ở cấp độ tài liệu, mô hình có thể tận dụng trình tự các đoạn văn dài liền kề nhau và học được các thông tin phụ thuộc xa.

4.5.2. Biểu diễn đầu vào

Sau quá trình huấn luyện mô hình trước đó bằng một tập dữ liệu lớn, mô hình sẽ được tinh chỉnh trên tập dữ liệu PERSONA-CHAT để phù hợp với nhiệm vụ tạo ra một chatbot. Trước tiên, cần phải biểu diễn đầu vào phù hợp với mô hình GPT gốc vì mô hình GPT chỉ được huấn luyện với duy nhất một đầu vào duy nhất. Trong tập dữ liệu PERSONA-CHAT cần nhiều thông tin về ngữ cảnh để có thể tạo ra phản hồi đầu ra bao gồm:

- Một hoặc nhiều câu mô tả tính cách
- Lịch sử cuộc hội thoại
- Các token của câu đầu ra đã được tạo ra từ các bước trước đó.

Do đó chuỗi các token đầu vào của mô hình sẽ được biểu diễn bằng cách kết hợp tất cả các thông tin trên bao gồm các câu mô tả tính cách (thường là 4 – 6 câu trong tập dữ liệu PERSONA-CHAT), lịch sử cuộc hội thoại (thường là 3 – 5 câu). Dưới đây là một số ví dụ về mã hóa từ vựng thành các token.

Câu mô tả tính cách gốc: “my favorite flower is a rose. i've a cat named jasper. i am a vegan. i work as a barista. my favorite color is orange.”

```
[82] personality

     [[547, 3898, 6330, 544, 246, 1978, 239],
      [249, 880, 246, 3387, 4049, 10907, 239],
      [249, 1048, 246, 37867, 239],
      [249, 1129, 557, 246, 37523, 239],
      [547, 3898, 3184, 544, 4564, 239]]
```

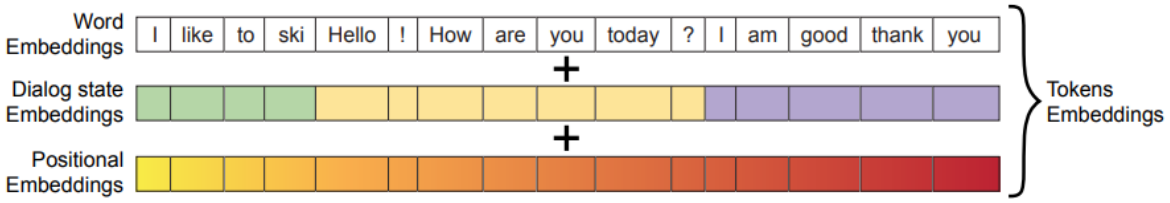
Hình 4.2: Mã hóa tính cách từ ngôn ngữ tự nhiên thành các token

```
tensor([[40478,   547,  3898,  6330,   544,   246,  1978,   239,   249,   880,
           246,  3387,  4049, 10907,   239,   249,  1048,   246, 37867,   239,
           249,  1129,   557,   246, 37523,   239,   547,  3898,  3184,   544,
          4564,   239, 40482,  3570, 40481,  3570,   240,   718,   640,   512,
          1873,   257]])
```

Hình 4.3: Chuỗi các token làm đầu vào của mô hình

Chuỗi các token trên sau khi mã hóa các thông tin ngữ cảnh để đưa vào mô hình tương ứng với chuỗi câu sau đây “<bos> my favorite flower is a rose. i've a cat named jasper. i am a vegan. i work as a barista. my favorite color is orange. <speaker2> hello <speaker1> hello, how are you today?”. Trong chuỗi trên, có các token đặc biệt ”<bos>, <speaker1>, <speaker2>, <eos>” làm nhiệm vụ đánh dấu bị trí bắt đầu của các thành phần trong câu. phản hồi kế tiếp được đặt ở cuối chuỗi.

Từ chuỗi token đầu vào ở trên ta có chuỗi embedding đầu vào (Sequence of input embedding) cho mô hình được tổ chức bằng cách xây dựng ba chuỗi đầu vào song song cho word, position và dialog-state sau đó kết hợp lại thành một chuỗi duy nhất bằng cách lấy tổng của ba loại embedding trên. Trong đó, word embedding và positional embedding được học từ quá trình tiền huấn luyện và được tăng cường với dialog-state embedding.



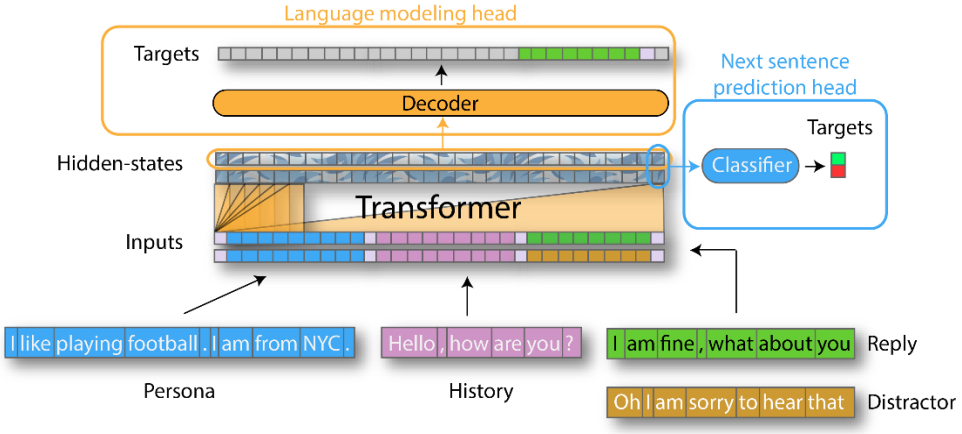
Hình 4.4: Biểu diễn đầu vào cho mô hình TransferTransfo [25]

Dialog-state embedding này được dùng để phân biệt các thành phần trong chuỗi đầu vào là các câu mô tả tính cách, lời thoại của PERSON1 hay của PERSON2 và chúng được học thông qua quá trình tinh chỉnh mô hình trên tập dữ liệu PERSONA-CHAT. Đầu vào của khối tự chú ý của Transformer là tổng của ba loại embedding trên cho từng từ.

4.5.3. Học đa tác vụ (Multi-task learning)

Quá trình tinh chỉnh được thực hiện bằng cách tối ưu hóa tổng của hai hàm mất mát: Dự đoán nhãn phân loại của phản hồi kế tiếp và mô hình hóa ngôn ngữ.

- **Phân loại phản hồi kế tiếp:** Đầu phân loại nhận trạng thái ẩn cuối cùng của của token <CLS> thông qua một lớp tuyến tính để tính toán điểm số và áp dụng negative log-likelihood loss để phân loại câu trả lời hợp lý giữa các câu trả lời ngẫu nhiên dùng để đánh lạc hướng (thường là 2 – 6 câu). Điểm số sau khi được tính toán sẽ được chuyển qua một lớp softmax để có được xác suất của mỗi lớp. Các tham số của mô hình Transformer và lớp phân loại được tinh chỉnh cùng nhau để tối đa hóa logarithm xác suất của nhãn đúng.
- **Mô hình hóa ngôn ngữ:** Dùng cross-entrophy loss tại trạng thái ẩn cuối cùng của mô hình tự chú ý sau đó đưa vào một lớp softmax để thu được xác suất của token tiếp theo. Các xác suất này được tính điểm dựa vào negative log-likelihood loss trong đó token tiếp theo của câu trả lời thích hợp được dùng làm nhãn.



Hình 4.5: Huấn luyện đa tác vụ - Mô hình cung cấp hai đầu để dự đoán từ kế tiếp (màu cam) và phân loại câu tiếp theo (màu xanh) [25]

Cụ thể: Mỗi chuỗi đầu vào có m token x = $(x^1, \dots, x^m)$, và một nhãn y để phân lớp đâu là câu đánh lạc hướng, đâu là câu trả lời phù hợp. Mô hình sẽ xử lý chuỗi đầu vào x thông qua bộ giải mã đã huấn luyện từ trước của GPT. Sau khi xử lý, ta có đầu ra của lớp cuối cùng cho token x_n là h_l^m và đưa qua một lớp tuyến tính để mô hình dự đoán phân phối xác xuất của chuỗi đầu vào thuộc phân lớp nào trong hai phân lớp trên. Đối với nhiệm vụ mô hình hóa ngôn ngữ token tiếp theo của câu trả lời phù hợp được dùng làm nhãn.

Hàm mất mát của mô hình là sự kết hợp giữa hai nhiệm vụ trên được tính bằng:

$$L_{total} = L_{lm} * lm\_coef + L_{mc} * mc\_coef$$

Trong đó, lm_coef và mc_coef là hệ số mất mát của mô hình ngôn ngữ và phân lớp, k là khung ngữ cảnh. Công thức của L_{mc} và L_{lm} lần lượt là:

$$L_{mc} = \sum_{(x,y)} logP(y|x^1, \dots, x^m) = \sum_{(x,y)} logsoftmax(h_l^m W_y)$$

$$L_{LM} = -\sum_i logP(x^i|x^{i-k}, \dots, x^{i-1})$$

4.5.4. Thông số của mô hình

Do giới hạn về RAM nên mô hình được tinh chỉnh với batch size là 2 gồm 8 câu mỗi batch với độ dài trung bình là 280 token mỗi câu (8 câu * 280 token = 2240 token) trong 131438 bước qua 2 epoch. Thuật toán tối ưu hóa được sử dụng là Adam với tỉ lệ học tập (learning rate) lr = 6.25e-5, β_1 = 0.9, β_2 = 0.999. Hệ số mất mát của mô hình ngôn ngữ khi tính tổng của hai hàm mất mát là 2.0, của mô hình phân lớp là 1.0. Tỉ lệ học tập được giảm tuyến tính về 0 trong quá trình huấn luyện. Tỉ lệ dropout = 0.1 cho mọi lớp. Quá trình tinh chỉnh mất sấp xỉ 20h trên 1 GPU Tesla P100.

4.5.5. Chi tiết bộ giải mã

Mô hình tạo ra câu trả lời bằng cách sử dụng thuật toán Nucleus sampling [5] được giới thiệu ở phần 3.8 bằng cách lấy mẫu từ phân phối của token kế tiếp sau đó lọc ra các token có độ lớn xác suất tích lũy vượt ngưỡng $p = 0.9$.

4.6. Đánh giá và kết quả

Với lượng dữ liệu lớn, sau khi mô hình được huấn luyện qua 2 epoch với mỗi epoch xấp xỉ 131.000 bước. Ta chọn mô hình tại thời điểm có giá trị loss thấp nhất

Số bước	**Loss**
120.000	0.8092312812805176
121.000	1.0254968404769897
122.000	1.0074938535690308
123.000	1.0447633266448975
124.000	1.0657150745391846
125.000	0.5270907878875732
126.000	0.7444302320480347

127.000	0.571350336074829
128.000	0.5943164348602295
129.000	0.6144708633422852
130.000	0.4048041343688965
131.000	0.4810291290283203

Bảng 4.1: Kết quả huấn luyện mô hình sau 131.000 bước. Bảng trên thống kê giá trị loss khi huấn luyện ở 11.000 bước cuối cùng

Mô hình được đánh giá dựa vào ba độ đo tự động đã đề cập ở mục 4.3. Kết quả được đo trên tập kiểm thử (test set) của bộ dữ liệu PERSONAL-CHAT được đánh giá bởi máy chủ của ConvAI được trình bày ở bảng 2 dưới đây:

Mô hình	PPL	Hits@1	F1
Seq2seq + Attention (ConvAI2 baseline)	29.8	0.126	0.1618
Language model (ConvAI2 baseline)	46.0	_	0.1502
KV Profile Memory (ConvAI2 baseline)	_	0.552	0.119
TransferTransfo	**21.52**	**0.807**	**0.1659**

Bảng 4.2: Kết quả trên tập kiểm thử của bộ dữ liệu PERSONA-CHAT được đo trên máy chủ của ConvAI

Từ bảng 4.1 trên có thể thấy TransferTransfo có hiệu suất vượt trội hơn ở cả ba độ đo so với các mô hình trước đó.

4.7. Thực nghiệm

4.7.1. Tương tác với mô hình

Quá trình tương tác với mô hình được ghi lại tại các bảng dưới đây. Mỗi cuộc hội thoại đều sử dụng ngẫu nhiên các thông tin cá nhân khác nhau:

Ví dụ về tương tác với mô hình trong nhiều lượt
Selected personality: my favorite flower is a rose. i've a cat named jasper. i am a vegan. i work as a barista. my favorite color is orange

Human: Hello

Machine: Hi there, how are you?

Human I am well. what about you?

Machine: i'm good. i am just finishing up a cup of coffee.

Human: oh sound good! what are you doing?

Machine: i am watching jasper. he's my favorite flower.

Human: how about rose?

Machine: that is my favorite color. i love roses.

Human: what is your name?

Machine: i am dave.

Human: my name is Khue

Machine: i'm a barista

Human: do you like cafe?

Machine: i do but i do not like it.

Human: what do you mean by that?

Machine: what do you do for a living?

Human: do you just ignore my question?

Machine: i am a barista. i love coffee.

Human: okay. everything seems to be reasonable

Machine: do you like cats? i have a cat named jasper

Human: yes, I have one. His name is Salem

Machine: do you like roses?

Human: obviously

Machine: my favorite flower is a rose. do you like to drink coffee?

Human: yes. I usually drink a cup of coffee every morning

Machine: do you work?

Human: yes. I am a programmer

Machine: what kind of programmer?

Human: web and game developer.

Bảng 4.3: Tương tác với mô hình trong nhiều lược

Ví dụ câu hỏi về sự lựa chọn
Selected personality: i love to climb. i am the oldest of 5 children. i ve never been to the beach. i love my parrot, todd.
Human: do you love climb? Machine: yes i do and i love to climb. Human: beach and moutain. which do you like most? Machine: i love the beach and the beach. Human: parrot and dog. which one do you refer? Machine: my parrot is todd. he's my favorite. Human: halloween and Christmas. Which one do you like ? Machine: i love halloween as well, i'm in a group that loves it.

Bảng 4.4: Ví dụ về câu hỏi lựa chọn

Ví dụ về những câu spam từ người nói
Selected personality: i love cats. i decorate cakes for a living. i play a lot of video games. i'm a lesbian.
Human: are you stupid? Machine: no i'm not. i love cats. you? Human: you are so stupid Machine: yes. you are just rude. Human: you moron Machine: haha. you are a moron. i'm a lesbian. Human: who cares Machine: you are a moron Human: you are so rude Machine: lol

Bảng 4.5: Phản hồi câu spam từ người nói

Kết luận: Mô hình có xu hướng kết hợp thông tin cá nhân vào câu trả lời. Thường sử dụng một số câu hỏi khác để lờ đi câu hỏi của người dùng. Có thể trả lời các câu hỏi mang tính lựa chọn ở mức tương đối và phản hồi đối phó lại các câu mang mục đích spam.

4.7.2. Xử lý trường hợp lặp lại câu thoại ở các bước trước đó (Cross-turn Repetitions)

Trong quá trình tương tác với mô hình, có tới hơn một nữa cuộc trò chuyện có chứa các câu thoại lặp đi lặp lại một lượt trước đó. Để minh họa vấn đề trên, sau đây là một ví dụ:

Ví dụ về vấn đề lặp lại câu thoại ở các bước trước đó
Selected personality: my mom is my best friend. i have four sisters. i believe that mermaids are real. i love iced tea.
Human: it's so good to have sisters Machine: it is. my mom is my best friend. Human: how about your sisters Machine: they are my sisters. Human: how old are you? Machine: i'm 22 and you? Human: i'm also 22 what a coincidence Machine: you should get married Human: yes i'm married, and have a daughter Machine: i believe mermaids are real Human: are you married? Machine: no, but i do believe mermaids are real Human: i don't believe that Machine: how old are you? Human: you already know that

Bảng 4.6: Lặp câu thoại ở bước trước đó

Trong ví dụ trên “How old are you?” bị lặp lại ở vài bước sau đó. Để giải quyết vấn đề trên, khóa luận đã thêm vào bộ giải mã một quy tắc để phát hiện xem hai lượt bất kỳ có tồn tại “long common sub sequence” [1] hay không và tự động loại bỏ câu trả lời đó nếu phát hiện đó là câu lặp lại của những lượt trước đó.

Sau khi thực nghiệm, quy tắc trên gần như đã giảm bớt các trường hợp lặp lại câu trước đó ngoại trừ một vài trường hợp nhỏ không được phát hiện bởi luật trên như những câu có điểm chung quá ngắn ví dụ như “Hello”.

Chương 5. CÀI ĐẶT VÀ TRIỂN KHAI ỨNG DỤNG

5.1. Yêu cầu ứng dụng

5.1.1. Yêu cầu chức năng

Chức năng trò chuyện với người dùng: Cho phép người dùng trò chuyện dễ dàng với chatbot thông qua giao diện tin nhắn.

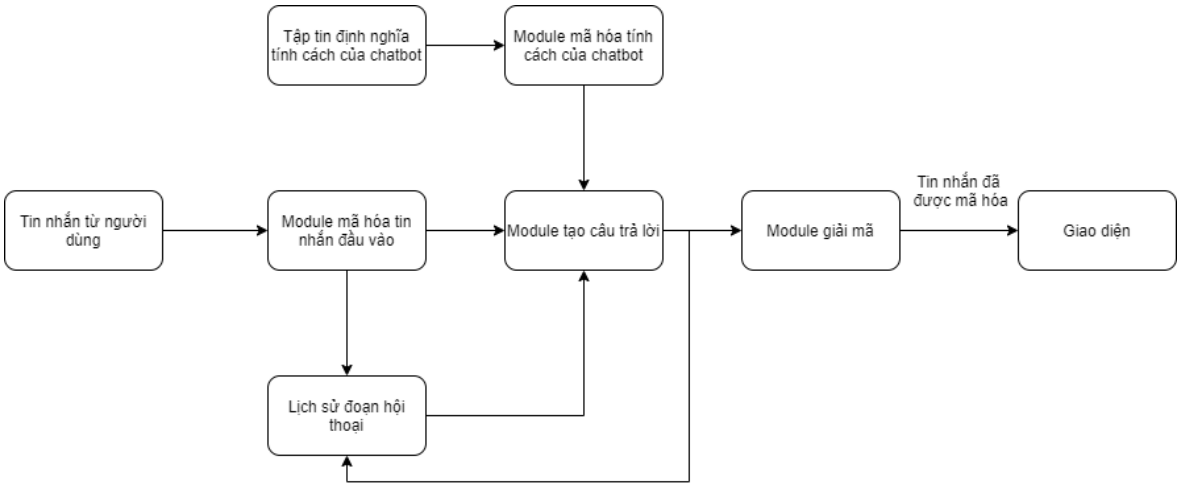
5.1.2. Yêu cầu tương thích

Thiết bị có thể sử dụng Messenger đều có thể truy cập được chatbot.

5.2. Kiến trúc ứng dụng

Mô hình được triển khai trên nền tảng Messenger của Facebook sử dụng framework Flask và Heroku.

Kiến trúc xử lý chính của ứng dụng như sau:



Hình 5.1: Kiến trúc xử lý của ứng dụng chatbot

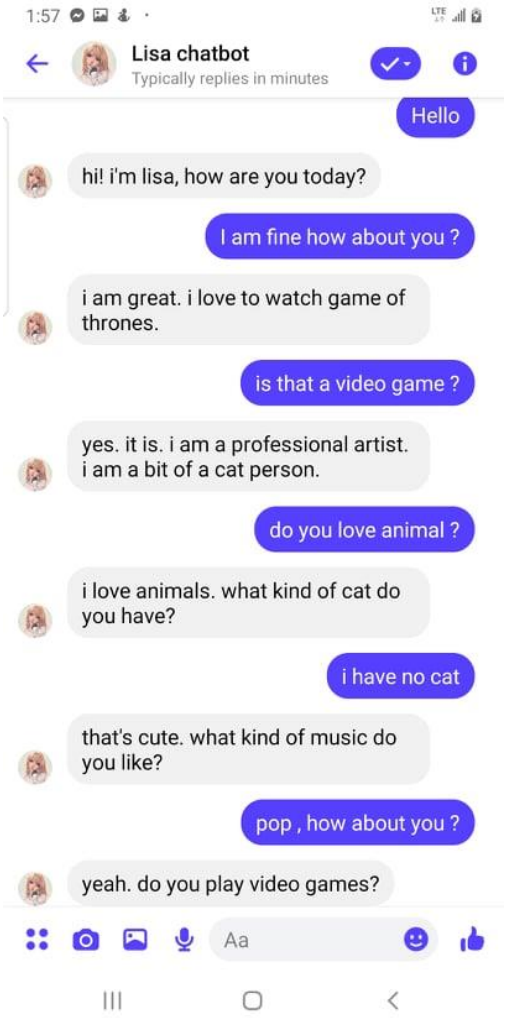
Mô tả kiến trúc: Tin nhắn nhập từ người dùng sẽ được xử lý thông qua module mã hóa văn bản. Tin nhắn sau khi được mã hóa sẽ được lưu vào lịch sử hội thoại kết hợp với tính cách của chatbot sau khi được mã hóa và đưa vào module tạo câu trả lời. Module tạo câu trả lời sẽ nhận cùng lúc ba thông tin là lịch sử đoạn hội thoại, tính cách của bot, câu trả lời của người dùng và tiến hành tạo câu trả lời đầu ra dựa vào những thông tin trên. Tin nhắn sau khi được tạo ra sẽ qua module giải

mã và trả về cho người dùng thông qua giao diện Messenger của Facebook. Chi tiết các chức năng của module như sau:

- **Module mã hóa:** Module này nhận vào một đoạn văn bản để tiến hành mã hóa ngôn ngữ tự nhiên thành dạng số theo chuẩn BPE (Byte pair encoding) để phù hợp với đầu vào của mô hình GPT.
- **Module giải mã:** Module này nhận đầu vào là chuỗi các dãy số được mã hóa trước đó theo chuẩn BPE và tiến hành giải mã những chuỗi số trên thành ngôn ngữ tự nhiên.
- **Module tạo câu trả lời:** Module này sẽ nhận ba thành phần là lịch sử cuộc hội thoại, tính cách của bot và tin nhắn trước đó của người dùng sau đó gộp chúng lại thành một chuỗi duy nhất để làm đầu vào cho cho mô hình. Mô hình sẽ tạo ra từ qua mỗi bước thời gian cho đến khi gặp token <EOS> thông qua thuật toán lấy mẫu đã đề cập ở phía trên.

5.2.1. Mô tả giao diện người dùng

Giao diện chính của ứng dụng cho phép người dùng tương tác với chatbot thông qua Messenger của Facebook.



Hình 5.2: Giao diện chính của ứng dụng

Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết quả đạt được

Khóa luận đã đạt được một số kết quả sau:

- **Tiếp cận được vấn đề xây dựng mô hình trò chuyện miền mở sử dụng phương pháp tự tạo văn bản**: Đây là một trong những bài toán thú vị, có tiềm năng và nhiều thách thức nhất hiện nay.
- **Khảo sát, phân tích các hệ thống trò chuyện:** Nắm được tình hình nghiên cứu về vấn đề xử lý ngôn ngữ tự nhiên trên thế giới nói chung và các hệ thống trò chuyện nói riêng. Các thách thức để xây dựng một chatbot miền mở hiệu quả.
- **Hiểu rõ hơn về các mô hình học sâu hiện đại:** Có được kiến thức về một số mô hình học sâu, nắm được cấu trúc, cách hoạt động cũng như các hạn chế và ứng dụng của chúng trong quá trình làm khóa luận.
- **Xây dựng được chatbot giải quyết vấn đề nhất quán về tính cách:** Tính nhất quán là một trong những thách thức chính khi xây dựng một hệ thống trò chuyện. Bằng cách sử dụng học chuyển tiếp kết hợp với mô hình ngôn ngữ GPT trên tập dữ liệu PERSONA-CHAT để xây dựng chatbot, kết quả đã cho thấy những cải tiếng mạnh mẽ so với các mô hình đàm thoại đầu cuối truyền thống như seq2seq hay các mô hình truy xuất thông tin được đánh giá bằng các độ đo tự động.
- **Xây dựng được ứng dụng tích hợp để hiện thực hóa mô hình vào thực tiễn:** Dựa vào kết quả kiểm thử, đề tài đã triển khai mô hình trên nền tảng Messenger của Facebook dưới dạng một chatbot. Chatbot có khả năng trò chuyện với người dùng thông qua tin nhắn văn bản.

6.2. Hạn chế

Tuy đã đạt được một số kết quả nhất định nhưng vì thời gian thực hiện khóa luận ngắn, kỹ năng và kiến thức chuyên môn còn hạn hẹp nên khóa luận vẫn còn một số hạn chế như:

- Kết quả tinh chỉnh trên GPT 2 có điểm số PPL cao nên vẫn cần thời gian tìm hiểu để khắc phục.
- Do hạn chế về mặt phần cứng nên không thể huấn luyện mô hình với batch size lớn.
- Vẫn có nhiều trường hợp phớt lờ câu hỏi của người dùng bằng cách đặt lại câu hỏi mới.

6.3. Hướng phát triển

Hướng phát triển khóa luận trong tương lai:

- Khắc phục được các hạn chế nêu ở phần trên.
- Tích hợp knowledge grounded (Miền tri thức) cho chatbot để giải quyết các câu hỏi liên quan đến tri thức.
- Tích hợp thêm tính năng trò chuyện bằng giọng nói thay vì tin nhắn văn bản

TÀI LIỆU THAM KHẢO

[1] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu and Q. Le, "Towards a human-like open-domain chatbot," *arXiv preprint arXiv:2001.09977*, 2020.

[2] D. Bahdanau, K. Cho and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014.

[3] K. Cho, B. van Merri{\"e}nboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[4] A. Graves, G. Wayne and I. Danihelka, "Neural Turing Machines," *arXiv preprint arXiv:1410.5401,* 2014.

[5] A. Holtzman, J. Buys, L. Du, M. Forbes and Y. Choi, "The curious case of neural text degeneration," in *International Conference on Learning Representations*, 2020.

[6] M. Huang, X. Zhu and J. Gao, "Challenges in building intelligent open-domain dialog systems," *ACM Transactions on Information Systems (TOIS)*, vol. 38, pp. 1-32, 2020.

[7] Y. Kim, J. Bang, J. Choi, S. Ryu, S. Koo and G. G. Lee, "Acquisition and use of long-term memory for personalized dialog systems," *International Workshop on Multimodal Analyses Enabling Artificial Agents in Human-Machine Interaction*, pp. 78-87, 2014.

[8] J. Li, M. Galley, C. Brockett, J. Gao and B. Dolan, "A diversity-promoting objective function for neural conversation models," *NAACL,* pp. 110-119, 2015.

[9] J. Li, M. Galley, C. Brockett, G. P. Spithourakis, J. Gao and B. Dolan, "A persona-based neural conversation model," *ACL*, pp. 994-1003, 2016.

[10] J. Li, W. Monroe and D. Jurafsky, "A simple, fast diverse decoding algorithm for neural generation," *arXiv preprint arXiv:1611.08562*, 2016.

[11] J. Li and D. Jurafsky, "Neural net models for open-domain discourse coherence," *arXiv preprint arXiv:1606.01545*, 2016.

[12] M.-T. Luong, H. Pham and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015.

[13] Q. Qian, M. Huang, H. Zhao, J. Xu and X. Zhu, "Assigning Personality/Profile to a Chatting Machine for Coherent Conversation Generation," in *IJCAI*, 2018, pp. 4279-4285.

[14] A. Radford, K. Narasimhan, T. Salimans and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[15] M. Rei, "Semi-supervised multitask learning for sequence labeling," *arXiv preprint arXiv:1704.07156*, 2017.

[16] A. Ritter, C. Cherry and W. B. Dolan, "Data-driven response generation in social media," *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 583--593, 2011.

[17] R. Sennrich, B. Haddow and A. Birch, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[18] I. Serban, A. Sordoni, Y. Bengio, A. Courville and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, pp. 3776-3784.

[19] L. Shang, Z. Lu and H. Li, "Neural responding machine for short-text conversation," *arXiv preprint arXiv:1503.02364.*

[20] C. Tao, S. Gao, M. Shang, W. Wu, D. Zhao and R. Yan, "Get The Point of My Utterance! Learning Towards Effective Responses with Multi-Head Attention Mechanism," in *IJCAI*, 2018, pp. 4418-4424.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, {. Kaiser and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2017.

[22] O. Vinyals and Q. Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869,* 2015.

[23] J. Wang, X. Wang, F. Li, Z. Xu, Z. Wang and B. Wang, "Group linguistic bias aware neural response generation," in *Proceedings of the 9th SIGHAN Workshop on Chinese Language Processing*, 2017, pp. 1-10.

[24] Y. Wang, C. Liu, M. Huang and L. Nie, "Learning to ask questions in open-domain conversational systems with typed decoders," *Proceedings of the 56th Annual Meeting of the,* pp. 2193-2203, 2018.

[25] T. Wolf, V. Sanh, J. Chaumond and C. Delangue, "Transfertransfo: A transfer learning approach for neural network based conversational agents," *CoRR,* vol. abs/1901.08149, 2019.

[26] C. Xing, Y. Wu, W. Wu, Y. Huang and M. Zhou, "Hierarchical recurrent attention network for response generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[27] C. Xing, W. Wu, Y. Wu, J. Liu, Y. Huang, M. Zhou and W.-Y. Ma, "Topic aware neural response generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, pp. 3351-3357.

[28] C. Xu, W. Wu, C. Tao, H. Hu, M. Schuerman and Y. Wang, "Neural response

generation with meta-words," in *arXiv preprint arXiv:1906.06050*, 2019.

[29] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048-2057.

[30] H. Zhang, Y. Lan, L. Pang, J. Guo and X. Cheng, "Recosa: Detecting the relevant contexts with self-attention for multi-turn dialogue generation," in *Proceedings of the 57th Annual Meeting of the Association for Computational*, pp. 3721-3730.

[31] S. Zhang, E. Dinan, J. Urbanek, A. Szlam, D. Kiela and J. Weston, "Personalizing dialogue agents: I have a dog, do you have pets too?," *arXiv preprint arXiv:1801.07243,* 2018.

[32] W.-N. Zhang, Q. Zhu, Y. Wang, Y. Zhao and T. Liu, "Neural personalized response generation as domain adaptation," *World Wide Web,* vol. 22, pp. 1427-1446, 2019.

[33] Y. Zhang, X. Gao, S. Lee, C. Brockett, M. Galley, J. Gao and B. Dolan, "Consistent dialogue generation with self-supervised feature learning," *arXiv preprint arXiv:1903.05759,* 2019.

[34] T. Zhao, R. Zhao and M. Eskenazi, "Learning discourse-level diversity for neural dialog models using conditional variational autoencoders," in *ACL*, 2017, pp. 654-664.

[35] H. Zhou, M. Huang, T. Zhang, X. Zhu and B. Liu, "Emotional chatting machine: Emotional conversation generation with internal and external memory," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[36] L. Zhou, J. Gao, D. Li and H.-Y. Shum, "The design and implementation of xiaoice, an empathetic social chatbot," *Computational Linguistics,* vol. 46, pp.

53-93, 2020.

[37] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 19-27.