

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ



BÙI NGỌC QUANG

**NGHIÊN CỨU VÀ XÂY DỰNG CHATBOT TƯ VẤN
NGƯỜI DÙNG TRONG Y TẾ DA LIỄU**

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

HÀ NỘI – 2022

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ



BÙI NGỌC QUANG

**NGHIÊN CỨU VÀ XÂY DỰNG CHATBOT TƯ VẤN
NGƯỜI DÙNG TRONG Y TẾ DA LIỄU**

Ngành: Công nghệ thông tin

Chuyên ngành: Kỹ thuật phần mềm

Mã số: 18025038

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

HÀ NỘI – 2022

LỜI CAM ĐOAN

Tôi xin cam đoan nội dung được trình bày trong luận văn “Nghiên cứu và xây dựng chatbot tư vấn người dùng trong y tế da liễu” là do tôi nghiên cứu, tìm hiểu và phát triển dưới sự dẫn dắt của TS. Nguyễn Văn Vinh. Luận văn không sao chép từ các tài liệu, công trình nghiên cứu của người khác mà không ghi rõ trong tài liệu tham khảo. Tất cả những tham khảo từ các nghiên cứu liên quan đều được nêu nguồn gốc một cách rõ ràng từ danh mục tài liệu tham khảo trong luận văn. Tôi xin chịu trách nhiệm về lời cam đoan này.

Hà nội, ngày tháng năm 2022

LỜI CẢM ƠN

Tôi xin gửi lời cảm ơn tới các thầy cô trường đại học Công Nghệ, Đại học Quốc Gia Hà Nội đã tận tình giảng dạy và truyền đạt kiến thức trong suốt khóa cao học vừa qua. Tôi cũng xin được gửi lời cảm ơn đến các thầy cô trong bộ môn Kỹ Thuật Phần Mềm cũng như khoa Công Nghệ Thông Tin đã mang lại cho tôi những kiến thức vô cùng quý giá và bổ ích trong quá trình học tập tại trường.

Đặc biệt xin chân thành cảm ơn thầy giáo, TS. Nguyễn Văn Vinh, người đã định hướng, giúp đỡ, trực tiếp hướng dẫn và tận tình chỉ bảo tôi trong suốt quá trình nghiên cứu, xây dựng và hoàn thiện luận văn này.

Tôi cũng xin được cảm ơn tới gia đình, những người thân, các đồng nghiệp và bạn bè thường xuyên quan tâm, động viên, chia sẻ kinh nghiệm, cung cấp các tài liệu hữu ích trong thời gian học tập, nghiên cứu cũng như trong suốt quá trình thực hiện luận văn tốt nghiệp.

Hà Nội, ngày tháng năm 2022

MỤC LỤC

LỜI CẢM ƠN.....	2
DANH MỤC CÁC BẢNG.....	6
DANH MỤC HÌNH VẼ VÀ ĐỒ THỊ.....	7
MỞ ĐẦU.....	8
1. Động lực nghiên cứu	8
2. Mục tiêu luận văn	9
3. Cấu trúc luận văn.....	10
CHƯƠNG 1 : GIỚI THIỆU TỔNG QUAN HỆ THỐNG CHATBOT	11
1.1. Giới thiệu	11
1.2. Kiến trúc cơ bản hệ thống Chatbot theo hướng mục tiêu	13
1.2.1. Hiểu ngôn ngữ tự nhiên.....	14
1.2.2. Quản lý hội thoại.....	18
1.2.3. Sinh ngôn ngữ tự nhiên	19
CHƯƠNG 2 : CÁC KỸ THUẬT SỬ DỤNG TRONG CHATBOT	22
2.1. Mạng hồi quy RNN.....	22
2.1.1. RNN hai chiều.....	23
2.1.2. RNN (hai chiều) sâu.....	23
2.2. Mạng bộ nhớ dài-ngắn LSTM	24
2.2.1. Vấn đề phụ thuộc xa.....	24
2.2.2. Kiến trúc mạng LSTM	25
2.2.3. Ý tưởng cốt lõi của LSTM	27
2.3. Mô hình nhúng từ.....	30
2.3.1. Phương pháp Véc tơ hóa dựa trên tần số xuất hiện.....	30
2.3.2. Phương pháp Véc tơ hóa dựa vào dự đoán	34
2.4. Mô hình BERT.....	36
2.4.1. Mô hình Fine-tuning BERT	36
2.4.2. Mô hình ngôn ngữ được che giấu (Masked ML)	38
2.4.3. Dự đoán câu tiếp theo.....	39
2.4.4. Các kiến trúc mô hình BERT	41
2.5. Mô hình PhoBERT	41
2.6 Tổng quan về Rasa.....	42

2.6.1	Các thành phần trong Rasa.....	43
2.6.2	Kiến trúc của Rasa.....	44
CHƯƠNG 3 : XÂY DỰNG ỨNG DỤNG CHATBOT TƯ VẤN NGƯỜI DÙNG TRONG Y TẾ DA LIỄU.....		45
3.1.	Xây dựng dữ liệu.....	45
3.1.1.	Xây dựng các Ý Định.....	45
3.1.2.	Xây dựng các loại Entity và Slot.....	46
3.1.3.	Xây dựng Câu trả lời cho Bot	46
3.1.4.	Xây dựng Khung kịch bản và Quy tắc	47
3.2.	Cải tiến nhận dạng ý định và tên riêng sử dụng PhoBERT	48
3.3.	Áp dụng Rasa xây dựng chatbot trong ngành Y Tế da liễu	48
3.4.	Cài đặt Rasa	50
3.5.	Phân tích thiết kế hệ thống.....	52
3.6.	Kết quả thực nghiệm	54
3.6.1.	Môi trường thực nghiệm	55
3.6.2.	Phương pháp đánh giá.....	55
3.6.3.	Thực nghiệm	56
3.8.	Đánh giá	60
KẾT LUẬN.....		62
TÀI LIỆU THAM KHẢO.....		63
PHỤ LỤC.....		64

DANH MỤC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

Từ viết tắt	Từ chuẩn	Diễn giải
AI	Artificial Intelligence	Trí tuệ nhân tạo
NLG	Natural Language Generation	Thành phần sinh ngôn ngữ
NLP	Natural Language Processing	Xử lý ngôn ngữ tự nhiên
NLU	Natural Language Understanding	Hiểu ngôn ngữ tự nhiên
ML	Machine Learning	Học máy, máy có khả năng học tập
NER	Named Entity Recognition	Nhận dạng thực thể
ANN	Artificial Neural Network	Mạng Nơron nhân tạo
CBOW	Continuous Bag of Words	Mô hình dự đoán từ mục tiêu dựa vào các từ ngữ cảnh xung quanh
CNN	Convolution Neural Network	Mạng Nơron tích chập
CRF	Conditional Random Fields	Mô hình xác suất trường điều kiện ngẫu nhiên
DM	Dialogue Management	Quản lý hội thoại
DNN	Deep Neural Networks	Mô hình học máy
DTS	Dialogue State Tracking	Theo dõi trạng thái hội thoại
FSA	Finite State Automata	Mô hình dựa trên máy trạng thái hữu hạn
FSM	Finite State Machine	Máy trạng thái hữu hạn
GLAD	Global-Locally SelfAttentive Dialogue State Tracker	Trình theo dõi trạng thái đối thoại toàn cục-cục bộ
HMM	Hidden Markov Models	Mô hình Markov ẩn
LSTM	Long short-term memory	Mạng cải tiến để giải quyết vấn đề phụ thuộc quá dài
POS	Part Of Speech	Gán nhãn từ loại

RNN	Recurrent Neural Network	Mạng Nơon hồi quy
SVM	Véc tơ Support Machine	Máy véc tơ hỗ trợ
SVD	Singular Value Decomposition	Phương pháp phân tích suy biến
MLM	Masked Language Model	Mô hình ngôn ngữ được che giấu
BPE	Byte Pair Encoding	Cặp mã hóa Byte
Seq2Seq	Sequence to sequence	Từ chuỗi sang chuỗi
BERT	Bidirectional Encoder Representation from Transformer	Mô hình biểu diễn từ theo hai chiều

DANH MỤC CÁC BẢNG

Bảng 1.1 Bảng trích xuất thực thể trong một câu.....	17
Bảng 1.2 Ví dụ khung hội thoại với người dùng.....	19
Bảng 2.1 Ví dụ về ma trận đồng xuất hiện.....	32
Bảng 3.1 Bảng kết quả xác định ý định.....	56
Bảng 3.2 Bảng kết quả nhận dạng thực thể.....	56
Bảng 3.3 Bảng mô tả số lần thử nghiệm với người dùng.....	57
Bảng 3.4 Bảng hội thoại với chatbot sau lần thử nghiệm cuối.....	58

DANH MỤC HÌNH VẼ VÀ ĐỒ THỊ

Hình 1.1 Ảnh minh họa Chatbot [1].....	11
Hình 1.2 Các thành phần cơ bản của Chatbot [1].....	13
Hình 1.3 Kiến trúc của hệ thống phân lớp ý định.....	15
Hình 1.4 Phương pháp sinh ngôn ngữ class-based[2]	20
Hình 1.5 Phương pháp sinh ngôn ngữ Phrase-based[2]	21
Hình 2.1 Mạng Nơ-ron hồi quy RNN [2]	22
Hình 2.2 Mạng RNN hai chiều [16]	23
Hình 2.3 Mạng RNN (hai chiều) sâu [16]	24
Hình 2.4 RNN phụ thuộc short-term [5]	25
Hình 2.5 RNN phụ thuộc long-term [4]	25
Hình 2.6 Các mô-đun lặp của mạng RNN chứa một lớp [4].....	26
Hình 2.7 Các mô-đun lặp của mạng LSTM chứa bốn lớp [4].....	26
Hình 2.8 Tế bào trạng thái LSTM giống như một băng truyền [4]	27
Hình 2.9 Cổng trạng thái LSTM [4].....	28
Hình 2.10 LSTM focus f [4].....	28
Hình 2.11 LSTM focus i [4]	29
Hình 2.12 LSTM focus C [4]	29
Hình 2.13 LSTM focus O [4]	30
Hình 2.14 Mô hình CBOW với 1 đầu vào [3].....	34
Hình 2.15 Mô hình Skip-gram [3].....	35
Hình 2.16 Toàn bộ tiến trình huấn luyện trước và tinh chỉnh của BERT [10].....	37
Hình 2.17 Sơ đồ kiến trúc BERT cho tác vụ Masked ML [10].....	38
Hình 2.18 Sơ đồ kiến trúc mô hình BERT cho tác vụ NSP [10].....	40
Hình 2.19 Minh họa về PhoBERT[13].....	41
Hình 2.20 Các thành phần chính trong Rasa [12]	43
Hình 2.21 Kiến trúc của Rasa [12]	44
Hình 3.1 Kiến trúc mô hình PhoBERT	48
Hình 3.2 Cấu trúc hệ thống chatbot trong ngành Y Tế đa liễu.....	49
Hình 3.3 Biểu đồ cung cấp thông tin của chatbot.....	52
Hình 3.4 Sequence Diagram cho hành động hỏi thông tin bệnh.....	53
Hình 3.5 Sequence Diagram cho hành động hỏi thông tin thuốc.....	54
Hình 3.6 Mô tả cách các chỉ số đánh giá mô hình huấn luyện.....	55
Hình 3-7 Ma trận ước lượng nhầm lẫn xây dựng dữ liệu intent.....	57
Hình 3-8 Biểu đồ độ tin cậy cho các dự đoán	58

MỞ ĐẦU

Theo dòng chảy của cuộc cách mạng 4.0, trí tuệ nhân tạo ngày càng được phổ biến và ứng dụng rộng rãi trong mọi lĩnh vực của cuộc sống. Các công ty lớn đã phát triển các nền tảng có thể được sử dụng để thiết kế các tương tác AI đàm thoại như trợ lý giọng nói và trợ lý ảo, được tích hợp với các nền tảng trò chuyện khác nhau, như Trợ lý Google¹, Siri², IBM Watson Assistant³, v.v. và một trong những ứng dụng được phát triển mạnh mẽ hiện nay là chatbot.

Sự phát triển của tự động hóa thúc đẩy cuộc cách mạng chatbot toàn cầu. Những tiến bộ của AI thúc đẩy những đổi mới của chatbot. Các số liệu thống kê về chatbot cho thấy tình trạng hiện tại về tốc độ phát triển và sự thâm nhập toàn cầu của nó:

- ✓ Cụ thể, giá trị thị trường chatbot ở mức 17,17 tỷ đô la vào năm 2020. Con số này dự kiến sẽ tăng lên con số khổng lồ 102,29 tỷ đô la vào năm 2026, với tốc độ tăng là 34,75% trong giai đoạn 2021-2026. (Juniper Research 2021)
- ✓ Người ta ước tính rằng tiết kiệm chi phí từ việc sử dụng chatbots trong ngân hàng sẽ đạt 7,3 tỷ đô la trên toàn cầu vào năm 2023, tăng so với ước tính năm 2019 là 209 triệu đô la. (Nghiên cứu Juniper, 2021)
- ✓ Ứng dụng nhắn tin có hơn 5 tỷ người dùng hoạt động mỗi tháng. (HubSpot)
- ✓ Các quốc gia có số lượng người dùng chatbot nhiều nhất là Mỹ (36%), Ấn Độ (11%) và Đức (4%). (Collect.chat, 2021)
- ✓ 24% doanh nghiệp, 15% công ty quy mô vừa và 16% doanh nghiệp nhỏ hiện sử dụng chatbots. (Spiceworks, 2021)

Sự xuất hiện của chatbots hiện đang mang lại nhiều khả năng khác nhau, bao gồm cả việc áp dụng AI tiên tiến để hỗ trợ con người. Với việc công nghệ phát triển không ngừng thì chúng ta mong đợi hiệu quả mà chatbot đem lại sẽ tăng trưởng trong những năm tới và chatbot có thể trở thành kênh kỹ thuật số được lựa chọn trong tương lai.

1. Động lực nghiên cứu

Hiện nay trên thế giới đã có hệ thống Sensely⁴, ở Việt Nam thì có hệ thống Hedima, FPT.AI Conversation được ứng dụng rất phổ biến và mạnh mẽ hỗ trợ giải đáp người dùng trong ngành Y nhưng không tập trung về một lĩnh vực bệnh lý cụ thể, vì thế luận văn này sẽ giải đáp chuyên sâu hơn trong Y tế da liễu. Tìm hiểu các kỹ thuật xử lý ngôn

¹ <https://cloud.google.com/dialogflow>

² <https://www.apple.com/siri/>

³ <https://www.ibm.com/products/watson-assistant>

⁴ <https://sensely.com/>

ngữ tự nhiên, phân loại ý định, trích xuất thông tin, quản lý hội thoại... trong việc xây dựng chatbot hỗ trợ người dùng hỏi về các triệu chứng bệnh về da liễu, từ đó sẽ có phương hướng điều trị tốt hơn. Và Học sâu là một lĩnh vực mới của học máy và được ứng dụng rất nhiều trong cuộc sống, trong đó có xử lý dữ liệu tự động hoá quá trình trích xuất đặc trưng và có độ tin cậy cao. Vì vậy, luận văn đặt ra mục tiêu nghiên cứu, ứng dụng các kỹ thuật học sâu để xây dựng ChatBot giúp chẩn đoán bệnh và chăm sóc sức khỏe, phòng ngừa bệnh về da.

Ở nước ta, việc giải đáp thắc mắc của bộ phận chăm sóc khách hàng qua tin nhắn trực tuyến đang được ưa chuộng. Tuy nhiên, việc này còn thực hiện một cách thủ công và gặp nhiều khó khăn như: tốn rất nhiều thời gian và chi phí chi trả cho nhân viên chỉ để trả lời những câu hỏi đơn giản và giống nhau. Chính vì vậy, nhu cầu cấp thiết là cần một hệ thống hỗ trợ giải đáp tự động để mang lại hiệu quả cao hơn và Chatbot là một sự lựa chọn hợp lý.

Với một khối lượng lớn câu hỏi mà chúng ta phải giải quyết mỗi ngày như: khách hàng hỏi về sản phẩm, dịch vụ, nhân viên hỏi về các quy chế công ty, con cái hỏi về những sự việc chúng đang tò mò...ngoài ra chatbot còn được áp dụng trong rất nhiều lĩnh vực. Đặc biệt trong lĩnh vực y tế: ChatBot này sẽ hỏi về các triệu chứng, các thông số cơ thể và lịch sử y tế, sau đó biên soạn một danh sách các nguyên nhân gây ra hầu hết các triệu chứng và xếp hạng chúng theo thứ tự nghiêm trọng. ChatBot có thể hướng dẫn bệnh nhân điều trị các bệnh có thể được chữa khỏi mà không cần đến bác sĩ.

2. Mục tiêu luận văn

Nhằm mục đích xây dựng hệ thống Chatbot giúp chẩn đoán bệnh da liễu nhanh hơn, dễ dàng hơn và minh bạch hơn cho cả bệnh nhân và bác sĩ – sử dụng hệ thống máy học để cung cấp các câu trả lời ngày càng chính xác cho các câu hỏi của người dùng dựa trên các hành vi mà nó “học” bằng cách tương tác với con người.

Chatbot trong lĩnh vực y tế da liễu làm giảm khối lượng công việc của các chuyên gia chăm sóc sức khỏe bằng cách giảm số lần đến bệnh viện, giảm các phương pháp điều trị và thủ tục không cần thiết, đồng thời giảm số lần nhập viện khi tuân thủ điều trị và kiến thức về các triệu chứng của họ được cải thiện. Đối với bệnh nhân, điều này mang lại rất nhiều lợi ích:

- ✓ Giảm bớt thời gian đi lại đến văn phòng bác sĩ
- ✓ Giảm bớt tài chính cho các phương pháp điều trị và xét nghiệm không cần thiết
- ✓ Dễ dàng tiếp cận bác sĩ thông qua ứng dụng

Luận văn này sẽ tập trung nghiên cứu các kỹ thuật xây dựng Chatbot, bao gồm: các kỹ thuật xử lý ngôn ngữ tự nhiên, phân loại ý định, trích xuất thông tin, quản lý hội thoại... Sử dụng nền tảng RASA để phát triển, xây dựng giao diện ứng dụng Chatbot trong ngành Y Tế da liễu, kết hợp các hệ thống NLP thông minh với công nghệ học máy để cung cấp cho người dùng trải nghiệm chính xác và nhanh nhạy.

3. Cấu trúc luận văn

Luận văn có cấu trúc như sau:

MỞ ĐẦU: Giới thiệu và đưa ra hướng nghiên cứu bài toán chatbot.

CHƯƠNG 1: Tổng quan hệ thống chatbot: Giới thiệu, Cấu tạo và Nhiệm vụ các thành phần hệ thống chatbot.

CHƯƠNG 2: Một số kỹ thuật sử dụng trong chatbot: Hiểu và nắm được một số kỹ thuật hay thuật toán cơ bản sử dụng trong chatbot để từ đó có thể điều chỉnh phù hợp với ngôn ngữ tiếng việt, giúp chatbot xử lý thông minh hơn.

CHƯƠNG 3: Xây dựng chatbot tư vấn người dùng trong Y Tế da liễu : Xây dựng ứng dụng chatbot trên nền tảng Rasa và kết quả thực nghiệm.

KẾT LUẬN: Đưa ra những kết luận, đánh giá và định hướng nghiên cứu tiếp theo

TÀI LIỆU THAM KHẢO: Tài liệu tham khảo và phụ lục

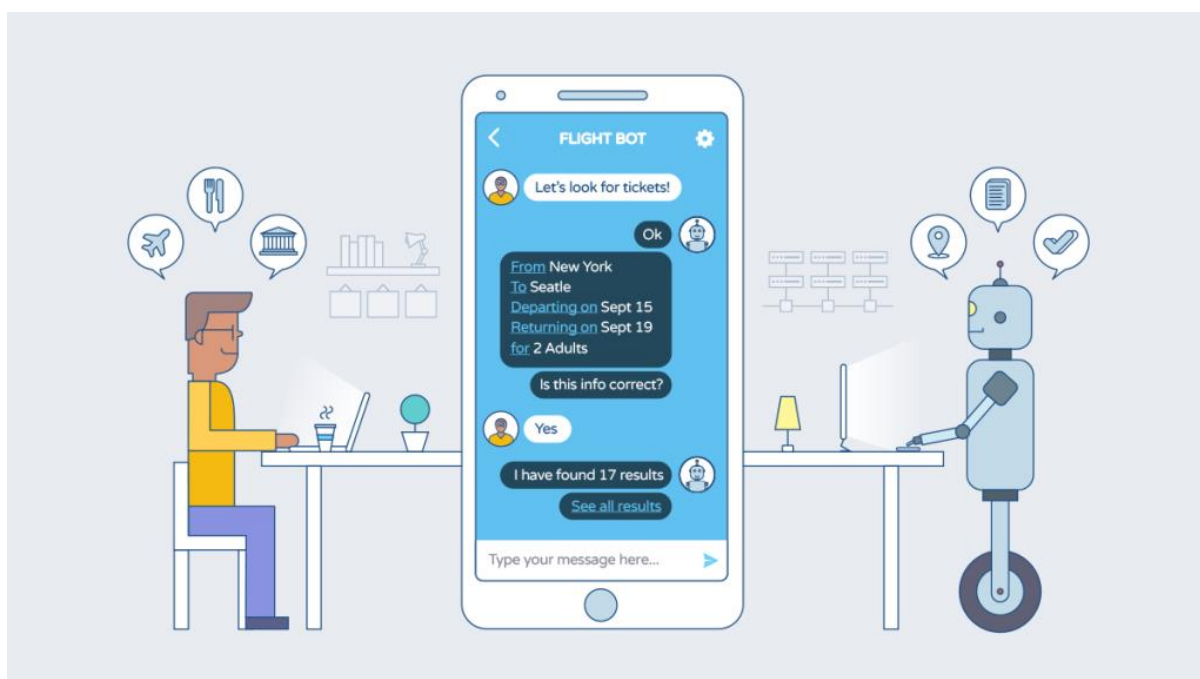
CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN HỆ THỐNG CHATBOT

Trong chương này sẽ giới thiệu tổng quan về Chatbot, kiến trúc và các thành phần cơ bản trong hệ thống chatbot.

1.1. Giới thiệu

Trước tiên, ta hiểu “Bot” là gì? Một “Bot” là một phần mềm thực hiện các nhiệm vụ, công việc con người yêu cầu một cách tự động. Hoặc “Bot” cũng có thể là một chương trình máy tính được thiết kế để “giao tiếp” với người dùng thông qua kết nối Internet.

“Chat” là từ quá quen thuộc mà ai cũng biết, nghĩa là trò chuyện, giao tiếp qua lại giữa hai người. Chatbot chính là hệ thống các Bot ở trong trạng thái trực tuyến, trên các website hoặc trên các nền tảng, giao diện chat khác của mạng xã hội để “chat tự động” với người dùng.



Hình 1.1 Ảnh minh họa Chatbot [1]

Chatbot được hiểu thông thường như một “cái máy” có thể đối thoại một cách tự nhiên với con người. Ví dụ: bất kỳ người dùng nào cũng có thể hỏi chatbot một câu hỏi hoặc một câu lệnh bất kỳ và chatbot sẽ trả lời hoặc thực hiện một hoạt động phù hợp đáp lại người dùng.

Chatbot tương tác với chúng ta như một hệ thống trả lời tin nhắn nhanh chóng, tự động. Bằng cách xây dựng, giả lập các mô hình tương tác, kích bản tương tác như của

con người sử dụng phương pháp trong học máy, hệ thống Chatbot có thể “tự học”, “tự hiểu” các câu hỏi, nhu cầu của người dùng, khách hàng và thực hiện, đưa ra các phản hồi sao cho phù hợp.

Chatbot sau khi được lập trình và huấn luyện nó sẽ tự động làm việc một cách độc lập như một con người. Chỉ những câu hỏi, tin nhắn của người dùng được cấu trúc lại thành các câu, ý định ngắn gọn với ngôn ngữ tự nhiên và thêm vào hệ thống kèm theo các kịch bản đối thoại tương ứng đã xây dựng trước đó thì Chatbot mới có khả năng đưa ra phản hồi.

Chatbot sử dụng cơ sở dữ liệu, nơi lưu trữ các câu hỏi, câu đối thoại đã được “huấn luyện” cho Chatbot – để phản hồi lại người dùng tại bất kỳ thời điểm nào. Trong trường hợp Chatbot không hiểu câu hỏi của người dùng, có thể do Chatbot chưa được “huấn luyện” kỹ thì Chatbot sẽ phản hồi sai thông tin, không phù hợp với mong muốn của người dùng.

Chatbot được “huấn luyện” và hoàn thiện trong thời gian dài sẽ tăng khả năng “tự học”, tự phát triển về phạm vi hiểu biết các ý định của người dùng và đạt được độ chính xác, độ tin cậy cao trong các phản hồi đưa ra.

Ngoài Alexa, Siri, hay Cortana còn có các ứng dụng Chatbot nổi tiếng trên thế giới khác như Chatbot Harafunnel⁵, Messnow⁶, ManyChat⁷,... với các tiện ích hỗ trợ ở nhiều lĩnh vực khác nhau từ E-commerce đến ngành du lịch,...

Chatbot được chia làm hai loại:

- ✓ Chatbot theo mô hình end-to-end: sử dụng mô hình học sâu Seq2Seq và dữ liệu là dạng câu hỏi – câu trả lời.
- ✓ Chatbot theo mô hình hướng mục tiêu: giúp người dùng đạt được mục tiêu xác định trước. Bước đầu tiên là hiểu mục tiêu của người dùng bằng cách sử dụng các kỹ thuật hiểu ngôn ngữ tự nhiên. Khi đã biết mục tiêu, bot phải quản lý một cuộc đối thoại để đạt được mục tiêu đó, cuộc đối thoại này được tiến hành theo phương pháp và thuật toán đã học.

Đề tài luận văn này sẽ sử dụng Chatbot theo mô hình hướng mục tiêu trên một miền ứng dụng đóng. Các phần tiếp theo trong luận văn sẽ mô tả cụ thể các thành phần, kỹ thuật và phương pháp để xây dựng chatbot.

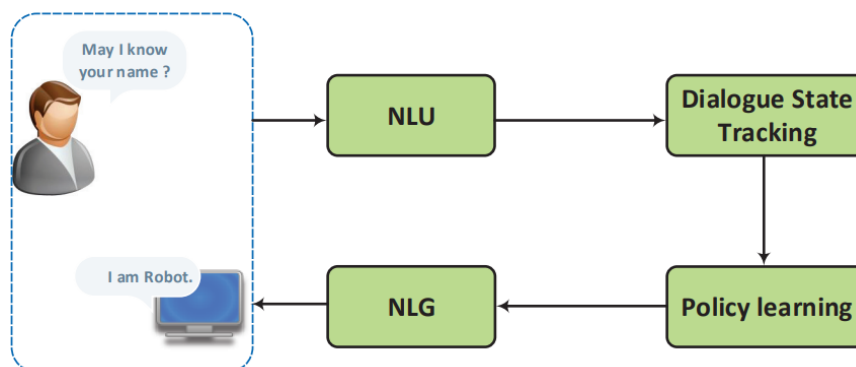
⁵ <https://en.harafunnel.com/>

⁶ <https://chatfuel.com/>

⁷ <https://manychat.com/>

1.2. Kiến trúc cơ bản hệ thống Chatbot theo hướng mục tiêu

Kiến trúc cơ bản của một hệ thống chatbot giao tiếp tự động bao gồm các thành phần như sau:



Hình 1.2 Các thành phần cơ bản của Chatbot [1]

Trên hình là cơ cấu cơ bản của một Chatbot sử dụng NLP và công nghệ học máy. Khi người dùng gửi tin nhắn đến Chatbot trên nền tảng nhắn tin như Facebook messenger thì thông tin sẽ được đưa đến hệ thống NLP để Chatbot phân tích và hiểu được ý định người dùng. Sau khi nắm được ý định người dùng, Chatbot sẽ phân loại và gửi đến cơ sở thông tin để chọn ra các câu trả lời tương ứng, chính xác và ra lệnh để phản hồi lại người dùng. Ngoài khả năng tự phân tích dựa vào NLP, Chatbot sẽ tự nhận dạng nhanh chóng các tin nhắn của người dùng và tạo khả năng tự học thông qua các thuật toán được nhà phát triển áp dụng và quá trình “huấn luyện lâu dài” trong tương lai.

Chatbot có ba thành phần chính là hiểu ngôn ngữ tự nhiên (NLU), quản lý hội thoại (DM), thành phần sinh ngôn ngữ (NLG). Mỗi thành phần trong chatbot đều có vai trò riêng:

- ✓ NLU: bao gồm việc xử lý ngôn ngữ tự nhiên có nhiệm vụ xác định được ý định câu hỏi và trích chọn thông tin.
- ✓ DM: Quản lý hội thoại có nhiệm vụ xác định được hành động tiếp theo dựa vào trạng thái hành động trước đó hay ngữ cảnh hội thoại. Các ngữ cảnh này phải được đối chiếu trong các kịch bản dựng sẵn đã huấn luyện cho bot. Thành phần này cũng đảm nhiệm việc lấy dữ liệu từ hệ thống khác qua các API gọi trong hành động.
- ✓ NLG: là thành phần sinh ngôn ngữ dựa vào chính sách và hành động được xác định trong DM thông qua các tập hội thoại. NLG có thể được sinh ra câu trả lời dựa vào tập mẫu câu trả lời đã huấn luyện cho bot.

1.2.1. Hiểu ngôn ngữ tự nhiên

Các bot xử lý ngôn ngữ tự nhiên được thiết kế để chuyển đổi văn bản hoặc lời nói của người dùng thành dữ liệu có cấu trúc. Dữ liệu được tiếp tục sử dụng để chọn một câu trả lời có liên quan

Xử lý ngôn ngữ tự nhiên bao gồm các bước dưới đây:

- ✓ Mã thông báo: bộ lọc NLP tập hợp các từ ở dạng mã thông báo.
- ✓ Phân tích cảm xúc: Bot diễn giải phản ứng của người dùng để phù hợp với cảm xúc của họ.
- ✓ Chuẩn hóa: Nó kiểm tra các lỗi đánh máy có thể làm thay đổi ý nghĩa trong yêu cầu của người dùng.
- ✓ Nhận dạng thực thể: Bot tìm kiếm các loại thông tin cần thiết khác nhau.
- ✓ Phân tích cú pháp phụ thuộc: Chatbot tìm kiếm các cụm từ phổ biến mà những gì người dùng muốn truyền tải.

1.2.1.1. Xác định ý định người dùng

Thông thường, người dùng thường truy cập hệ thống chatbot với mong muốn hệ thống sẽ đưa ra những hành động trợ giúp mình về một vấn đề nào đó. Ví dụ, người dùng của hệ thống chatbot hỗ trợ đặt lịch khám bệnh có thể đưa ra yêu cầu đặt lịch của mình khi bắt đầu cuộc hội thoại. Để đưa ra hỗ trợ được chính xác, chatbot cần xác định được ý định đó của người dùng. Việc xác định ý định của người dùng sẽ quyết định hội thoại tiếp theo giữa người và chatbot sẽ diễn ra như thế nào. Vì thế, nếu xác định sai ý định người dùng, chatbot sẽ đưa ra những phản hồi không đúng, không hợp ngữ cảnh. Khi đó, người dùng có thể thấy chán ghét và không quay lại sử dụng hệ thống. Bài toán xác định ý định người dùng vì thế đóng vai trò rất quan trọng trong hệ thống chatbot.

Đối với miền ứng dụng đóng, chúng ta có thể giới hạn rằng số lượng ý định của người dùng nằm trong một tập hữu hạn những ý định đã được định nghĩa sẵn, có liên quan đến những nghiệp vụ doanh nghiệp mà chatbot có thể hỗ trợ. Với giới hạn này, bài toán xác định ý định người dùng có thể quy về bài toán phân lớp văn bản. Với đầu vào là một câu giao tiếp của người dùng, hệ thống phân lớp sẽ xác định ý định tương ứng với câu đó trong tập các ý định đã được định nghĩa.

Để xây dựng một mô hình phân lớp ý định, chúng ta cần một tập dữ liệu huấn luyện bao gồm các cách diễn đạt khác nhau cho mỗi ý định. Ví dụ, cùng một mục đích hỏi về triệu chứng bệnh, người dùng có thể dùng những cách diễn đạt sau:

Bệnh này là gì?

Bệnh này là như thế nào?

Tôi muốn hỏi thông tin về bệnh

Tư vấn bệnh

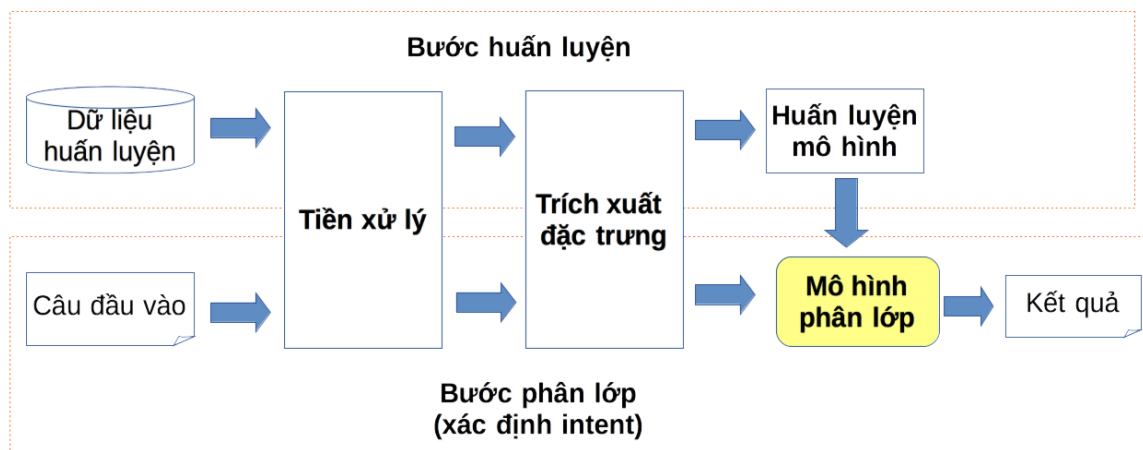
Tôi không thấy khỏe trong người, cần được tư vấn

Có thể nói, bước tạo dữ liệu huấn luyện cho bài toán phân lớp ý định là một trong những công việc quan trọng nhất khi phát triển hệ thống chatbot và ảnh hưởng lớn tới chất lượng sản phẩm của hệ thống chatbot về sau. Công việc này cũng đòi hỏi thời gian, công sức khá lớn của nhà phát triển chatbot.

Mô hình học máy cho bài toán phân lớp ý định người dùng:

Khi đã có dữ liệu huấn luyện cho bài toán phân lớp ý định, chúng ta sẽ mô hình bài toán thành bài toán phân lớp văn bản. Bài toán phân lớp văn bản là một bài toán kinh điển trong ngành NLP và khai phá văn bản. Mô hình phân lớp văn bản cho bài toán phân lớp ý định được phát biểu một cách hình thức như sau:

Chúng ta được cho trước một tập huấn luyện bao gồm các cặp (câu hội thoại, ý định), $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$, trong đó x_i là các câu hội thoại và y_i là các ý định tương ứng cho x_i . Các ý định y_i nằm trong một tập hữu hạn K các ý định được định nghĩa trước. Chúng ta cần học từ tập huấn luyện này một mô hình phân lớp có chức năng phân lớp một câu hội thoại mới vào một trong các ý định thuộc tập K. Kiến trúc của hệ thống phân lớp ý định được minh họa trong hình dưới đây.



Hình 1.3 Kiến trúc của hệ thống phân lớp ý định

Hệ thống phân lớp ý định có một số thành phần cơ bản:

- ✓ Tiền xử lý dữ liệu
- ✓ Trích xuất đặc trưng

- ✓ Huấn luyện mô hình
- ✓ Phân lớp

Trong bước tiền xử lý dữ liệu, chúng ta sẽ thực hiện các thao tác “làm sạch” dữ liệu như: loại bỏ các thông tin dư thừa, chuẩn hoá dữ liệu như chuyển các từ viết sai chính tả thành đúng chính tả, chuẩn hoá các từ viết tắt, v.v. Việc tiền xử lý dữ liệu có vai trò quan trọng trong hệ thống chatbot do đặc thù của ngôn ngữ chat, nói: viết tắt, sai chính tả, hay dùng từ lóng, v.v.

Sau khi tiền xử lý dữ liệu và thu được dữ liệu đã được làm sạch, chúng ta sẽ trích xuất những đặc trưng từ dữ liệu này. Trong học máy, bước này được gọi là trích xuất đặc trưng. Trong mô hình học máy truyền thống (trước khi mô hình học sâu được áp dụng rộng rãi), bước trích xuất đặc trưng ảnh hưởng lớn đến độ chính xác của mô hình phân lớp. Để trích xuất được những đặc trưng tốt, chúng ta cần phân tích dữ liệu khá tỉ mỉ và cần cả những tri thức chuyên gia trong từng miền ứng dụng cụ thể.

Bước huấn luyện mô hình nhận đầu vào là các đặc trưng đã được trích xuất và áp dụng các thuật toán học máy để học ra một mô hình phân lớp. Các mô hình phân lớp có thể là các luật phân lớp (nếu sử dụng cây quyết định) hoặc là các véc tơ trọng số tương ứng với các đặc trưng được trích xuất (như trong các mô hình logistic regression, SVM, hay mạng Noron).

Sau khi có một mô hình phân lớp ý định, chúng ta có thể sử dụng nó để phân lớp một câu hội thoại mới. Câu hội thoại này cũng đi qua các bước tiền xử lý và trích xuất đặc trưng, sau đó mô hình phân lớp sẽ xác định “điểm số” cho từng ý định trong tập các ý định và đưa ra ý định có điểm cao nhất.

1.2.1.2. Trích xuất thông tin

Bên cạnh việc xác định ý định trong câu hội thoại của người dùng, chúng ta cần trích xuất các thông tin cần thiết trong đó. Các thông tin cần trích xuất trong một câu hội thoại thường là các thực thể thuộc về một loại nào đó. Ví dụ, khi một khách hàng muốn đặt lịch khám chữa bệnh, hệ thống cần biết thông tin khách hàng, ngày giờ khách hàng muốn chữa trị,... Thành phần NLU của các hệ thống chatbot thường hỗ trợ các loại thực thể sau :

- ✓ Vị trí (Location)
- ✓ Thời gian (Datetime)
- ✓ Địa chỉ liên lạc (Contact)

✓ Khoảng thời gian (Duration)

Bảng 1.1 Bảng trích xuất thực thể trong một câu

Tôi	muốn	đặt	lịch	khám bệnh	tại	Hà Nội	vào	9	giờ	ngày	mai
o	o	o	o	o	o	B- Location	o	B- Time	I- Time	I- Time	I-Time

Đầu vào của một chức năng trích xuất thông tin là một câu hội thoại. Chức năng trích xuất thông tin cần xác định vị trí của các thực thể trong câu (vị trí bắt đầu và vị trí kết thúc của thực thể). Ví dụ sau minh họa một câu hội thoại và các thực thể được trích xuất từ đó.

Câu hội thoại: Tôi muốn đặt lịch khám bệnh tại Hà Nội lúc 9 giờ ngày mai.

Câu có các thực thể được xác định: Tôi muốn đặt lịch khám bệnh tại [Hà Nội] *LOCATION* lúc [9 giờ ngày mai] *TIME*.

Trong câu trên có hai thực thể (nằm trong các dấu []) với các loại thực thể tương ứng (được viết với font in nghiêng bên cạnh).

Cách tiếp cận phổ biến cho bài toán trích xuất thông tin là mô hình hoá bài toán thành bài toán gán nhãn chuỗi (sequence labeling). Đầu vào của bài toán gán nhãn chuỗi là một dãy các từ, và đầu ra là một dãy các nhãn tương ứng các từ trong đầu vào. Chúng ta sẽ sử dụng các mô hình học máy để học một mô hình gán nhãn từ một tập dữ liệu đầu vào bao gồm các cặp $(x_1 \dots x_n, y_1 \dots y_n)$, trong đó $x_1 \dots x_n$ là dãy các từ, $y_1 \dots y_n$ là dãy các nhãn. Độ dài của các dãy từ trong tập dữ liệu có thể khác nhau.

Trong bài toán trích xuất thông tin, tập nhãn cho các từ trong câu đầu vào thường được tạo ra theo mô hình BIO, với B là viết tắt của “Beginning”, I là viết tắt của “Inside”, và O là viết tắt của “Outside”. Khi biết vị trí từ bắt đầu của một thực thể và các từ nằm trong thực thể đó, chúng ta có thể xác định vị trí của thực thể trong câu. Trong ví dụ ở trên, dãy các nhãn tương ứng với dãy của các từ trong câu hội thoại đầu vào.

Thuật toán huấn luyện mô hình gán nhãn chuỗi phổ biến là mô hình Markov ẩn (HMM – Hidden Markov Models), mô hình CRF (Conditional Random Fields). Với dữ liệu văn bản, mô hình CRF thường cho kết quả tốt hơn mô hình HMM. Có khá nhiều các công cụ mã nguồn mở cài đặt mô hình CRF cho bài toán gán nhãn chuỗi như CRF++, CRF Suite, Mallet, v.v.

Gần đây, các mô hình mạng Noron hồi quy (Recurrent Neural Networks) được áp dụng khá nhiều cho bài toán gán nhãn chuỗi. Mô hình mạng Noron hồi quy tỏ ra hiệu quả với dữ liệu dạng chuỗi vì nó mô hình mối quan hệ phụ thuộc giữa các từ trong câu. Ví dụ, mạng Noron hồi quy được áp dụng cho bài toán gán nhãn từ loại (POS Tagging), bài toán xác định thực thể tên gọi.

1.2.2. Quản lý hội thoại

Trong các phiên trao đổi dài giữa người và chatbot, chatbot sẽ cần ghi nhớ những thông tin về ngữ cảnh hay quản lý các trạng thái hội thoại. Vấn đề quản lý hội thoại khi đó là quan trọng để đảm bảo việc trao đổi giữa người và máy là thông suốt.

Chức năng của thành phần quản lý hội thoại là nhận đầu vào từ thành phần NLU, quản lý các trạng thái hội thoại, ngữ cảnh hội thoại, và truyền đầu ra cho thành phần sinh ngôn ngữ (Natural Language Generation, viết tắt là NLG). Ví dụ chức năng quản lý hội thoại trong một chatbot phục vụ đặt vé máy bay cần biết khi nào người dùng đã cung cấp đủ thông tin cho việc đặt vé để tạo một vé tới hệ thống hoặc khi nào cần phải xác nhận lại thông tin do người dùng đưa vào. Hiện nay, các sản phẩm chatbot thường dùng mô hình máy trạng thái hữu hạn (Finite State Automata – FSA), mô hình Frame-based (Slot Filling), hoặc kết hợp hai mô hình này.

FSA là mô hình quản lý hội thoại đơn giản nhất. Ví dụ, trong một hệ thống chăm sóc khách hàng của một công ty viễn thông, phục vụ cho những khách hàng than phiền về vấn đề mạng chậm. Nhiệm vụ của chatbot là hỏi tên khách hàng, số điện thoại, tên gói Internet khách hàng đang dùng, tốc độ Internet thực tế của khách hàng. Các trạng thái của FSA tương ứng với các câu hỏi mà hệ thống hội thoại hỏi người dùng. Các kết nối giữa các trạng thái tương ứng với các hành động của chatbot sẽ thực hiện. Các hành động này phụ thuộc phản hồi của người dùng cho các câu hỏi. Trong mô hình FSA, chatbot là phía định hướng người sử dụng trong cuộc hội thoại.

Ưu điểm của mô hình FSA là đơn giản và chatbot sẽ định trước dạng câu trả lời mong muốn từ phía người dùng. Tuy nhiên, mô hình FSA không thực sự phù hợp cho các hệ thống chatbot phức tạp hoặc khi người dùng đưa ra nhiều thông tin khác nhau trong cùng một câu hội thoại. Trong ví dụ chatbot ở trên, khi người dùng đồng thời cung cấp cả tên và số điện thoại, nếu chatbot tiếp tục hỏi số điện thoại, người dùng có thể cảm thấy khó chịu.

Mô hình Frame-based (hoặc tên khác là Form-based) có thể giải quyết vấn đề mà mô hình FSA gặp phải. Mô hình Frame-based dựa trên các frame định sẵn để định

hướng cuộc hội thoại. Mỗi khung sẽ bao gồm các thông tin (slot) cần điền và các câu hỏi tương ứng mà quản lý hội thoại hỏi người dùng. Mô hình này cho phép người dùng điền thông tin vào nhiều slot khác nhau trong frame. Bảng dưới là một ví dụ về một frame cho chatbot hỏi thông tin triệu chứng khách hàng:

Bảng 1.2 Ví dụ khung hội thoại với người dùng

Slot	Câu hỏi
Họ tên	Bạn vui lòng cho biết tên?
Tuổi	Tuổi của bạn là gì?
Giới tính	Giới tính của bạn?
Triệu chứng	Triệu chứng bệnh của bạn là gì?

Thành phần quản lý hội thoại theo mô hình Frame-based sẽ đưa ra câu hỏi cho khách hàng, điền thông tin vào các slot dựa trên thông tin khách hàng cung cấp cho đến khi có đủ thông tin cần thiết. Khi người dùng trả lời nhiều câu hỏi cùng lúc, hệ thống sẽ phải điền vào các slot tương ứng và ghi nhớ để không hỏi lại những câu hỏi đã có câu trả lời.

Trong các miền ứng dụng phức tạp, một cuộc hội thoại có thể có nhiều frame khác nhau. Vấn đề đặt ra cho người phát triển chatbot khi đó là làm sao để biết khi nào cần chuyển đổi giữa các frame. Cách tiếp cận thường dùng để quản lý việc chuyển điều khiển giữa các frame là định nghĩa các luật (production rule). Các luật này dựa trên một số các thành tố như câu hội thoại hoặc câu hỏi gần nhất mà người dùng đưa ra.

1.2.3. Sinh ngôn ngữ tự nhiên

NLG là thành phần sinh câu trả lời của chatbot, NLG có thể được sinh ra câu trả lời dựa vào tập mẫu câu trả lời đã huấn luyện. Nó dựa vào việc ánh xạ các hành động của quản lý hội thoại vào ngôn ngữ tự nhiên để trả lời người dùng.

inform(ten_thuoc= salicylic, ten_benh=vảy_nén)



Thuốc bôi salicylic điều trị bệnh vảy nến

Trong NLG có ba phương pháp kỹ thuật được sử dụng là: Template-Base, Class-base, Phrase-Based

1.2.3.1. Template-based NLG

Phương pháp ánh xạ câu trả lời này là dùng những câu mẫu trả lời của bot đã được định nghĩa từ trước để sinh câu trả lời

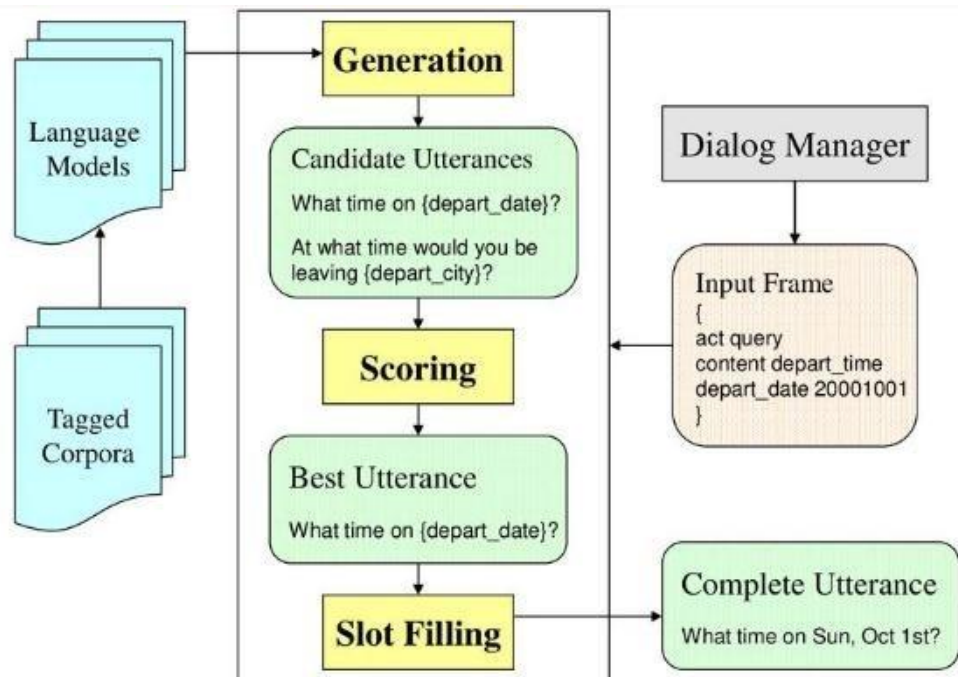
Semantic	Frame Natural Language
Confirm()	Bạn vui lòng cho biết tên bệnh bạn đang gặp phải?
Confirm (ten_benh=\$X)	Bạn muốn tìm hiểu bệnh <i>vảy nến</i> ?
Confirm (ten_thuoc=\$Y)	Bạn muốn tìm hiểu thuốc <i>salicylic</i> ?
Confirm (ten_benh=\$X, ten_thuoc=\$Y)	Bạn muốn tìm hiểu thuốc <i>salicylic</i> chữa bệnh <i>vảy nến</i> ?

Ưu điểm: Phương pháp này đơn giản, kiểm soát dễ dàng. Phù hợp cho các bài toán miền đóng.

Nhược điểm: Phương pháp này có nhược điểm là tốn thời gian định nghĩa các luật, không mang tính tự nhiên trong câu trả lời. Đối với các hệ thống lớn thì khó kiểm soát các luật dẫn đến hệ thống cũng khó phát triển và duy trì.

1.2.3.2. Class-based NLG

Phương pháp này dựa trên việc cho bot học những câu trả lời đầu vào đã được gán nhãn. Ứng với các hành động (action) và thông tin (slot) từ quản lý hội thoại thì bot sẽ đưa ra câu trả lời gần nhất dựa trên tập dữ liệu trả lời được đào tạo trước đó.



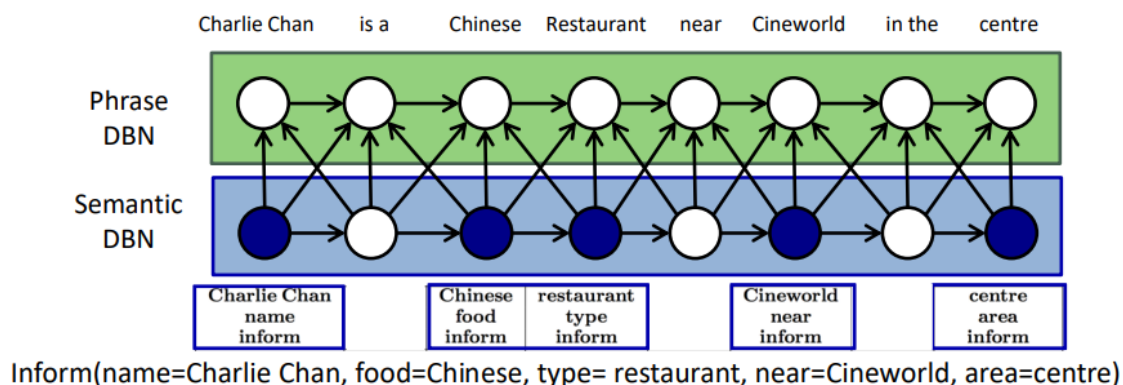
Hình 1.4 Phương pháp sinh ngôn ngữ class-based[2]

Ưu điểm: phương pháp này dễ thực hiện và dễ hiểu, các quy tắc đơn giản.

Nhược điểm: phụ thuộc vào dữ liệu trả lời đã được gán nhãn đào tạo trước đó. Bên cạnh đó việc tính toán điểm số không hiệu quả cũng dẫn đến việc sinh câu trả lời sai.

1.2.3.3. Phrase-based NLG

Phương pháp này dựa trên việc cho bot đào tạo hoàn toàn từ dữ liệu mà không cần phải làm thủ công. Tất cả các yêu cầu nhiệm vụ như là sắp xếp nội dung, tổng hợp, lựa chọn từ vựng và hiện thực hóa được học từ dữ liệu sử dụng một mô hình thống nhất, biểu diễn ngữ nghĩa ở cấp độ cụm từ, đủ biểu cảm để tạo ra những lời nói tự nhiên.



Hình 1.5 Phương pháp sinh ngôn ngữ Phrase-based[2]

Ưu điểm: phương pháp này cho hiệu quả, hiệu suất tốt.

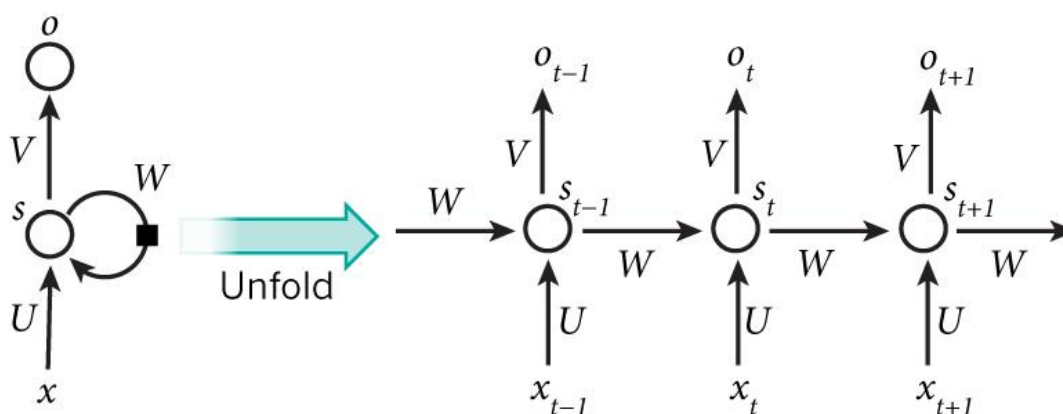
Nhược điểm: Phương pháp này yêu cầu liên kết ngữ nghĩa, thiết kế nặng nề, đòi hỏi phải rõ miền kiến thức

CHƯƠNG 2: CÁC KỸ THUẬT SỬ DỤNG TRONG CHATBOT

Trong chương này sẽ mô tả một số kỹ thuật sử dụng trong Chatbot như: kiến trúc nền tảng về mạng Noron nhân tạo, cách thức hoạt động của mạng Noron và một số các kỹ thuật được ứng dụng trong việc xử lý ngôn ngữ tự nhiên, các phương pháp trích xuất đặc trưng trên dữ liệu đã được làm sạch sử dụng các mạng Noron.

2.1. Mạng hồi quy RNN

Mạng nơ-ron hồi quy (RNN[5] - Recurrent Neural Network) là một thuật toán được chú ý rất nhiều trong thời gian gần đây bởi các kết quả tốt thu được trong lĩnh vực xử lý ngôn ngữ tự nhiên. Ý tưởng chính của RNN là sử dụng chuỗi các thông tin. Trong các mạng nơ-ron truyền thống tất cả các đầu vào và cả đầu ra là độc lập với nhau. Tức là chúng không liên kết thành chuỗi với nhau. Nhưng các mô hình này không phù hợp trong rất nhiều bài toán. Ví dụ, nếu muốn đoán từ tiếp theo có thể xuất hiện trong một câu thì ta cũng cần biết các từ trước đó xuất hiện lần lượt như thế nào. RNN được gọi là hồi quy bởi lẽ chúng thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng nhớ các thông tin được tính toán trước đó. Trên lý thuyết, RNN có thể sử dụng được thông tin của một văn bản rất dài, tuy nhiên thực tế thì nó chỉ có thể nhớ được một vài bước trước mà thôi. Về cơ bản một mạng RNN có dạng như sau:



Hình 2.1 Mạng Nơ-ron hồi quy RNN [2]

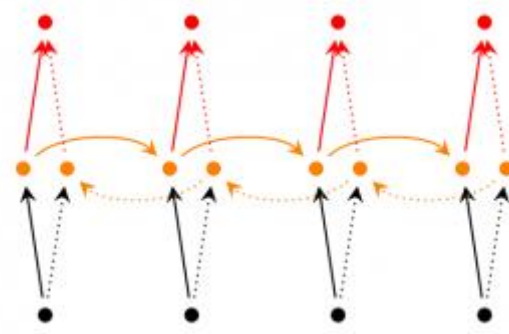
Mô hình trên mô tả phép triển khai nội dung của một RNN. Triển khai ở đây có thể hiểu đơn giản là ta vẽ ra một mạng nơ-ron chuỗi tuần tự. Ví dụ ta có một câu gồm năm chữ “*Thông tin bệnh vậy nên*”, thì mạng nơ-ron được triển khai sẽ gồm năm tầng nơ-ron tương ứng với mỗi chữ một tầng. Lúc đó việc tính toán bên trong RNN được thực hiện như sau:

- ✓ x_t : là đầu vào tại bước t . Ví dụ, x_1 là một vec-tơ one-hot tương ứng với từ thứ hai của câu.
- ✓ s_t : là trạng thái ẩn tại t . Nó chính là bộ nhớ của mạng. s_t được tính toán dựa trên cả các trạng thái ẩn phía trước và đầu vào tại bước đó: $s_t = f(Ux_t + Ws_{t-1})$. Hàm f thường là một hàm phi tuyến tính như tang hyperbolic (tanh) hay ReLu. Để làm phép toán cho phần tử ẩn đầu tiên ta cần khởi tạo thêm s_{-1} , thường giá trị khởi tạo được gán bằng 0.
- ✓ o_t : là đầu ra tại bước t . Ví dụ, muốn dự đoán từ tiếp theo có thể xuất hiện trong câu thì o_t chính là một vector xác suất các từ trong danh sách từ vựng: $o_t = \text{softmax}(Vs_t)$

Hiện nay, các nhà nghiên cứu đã phát triển nhiều kiểu RNN tinh vi để xử lý các nhược điểm của mô hình RNN truyền thống. Trong đó, có hai mô hình mở rộng sau:

2.1.1. RNN hai chiều

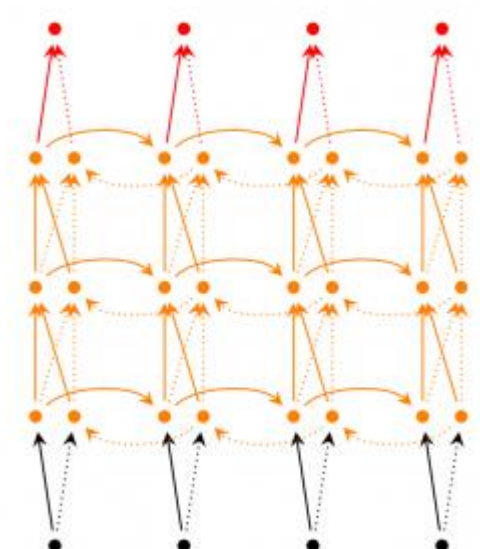
Ở mô hình RNN hai chiều (Bidirectional RNN), đầu ra tại bước t không những phụ thuộc vào các phần tử phía trước mà còn phụ thuộc cả vào các phần tử phía sau. Ví dụ, để dự đoán từ còn thiếu trong câu, thì việc xem xét cả phần trước và phần sau của câu là cần thiết. Vì vậy, có thể coi mô hình là việc chồng hai mạng RNN ngược hướng nhau lên nhau. Lúc này đầu ra được tính toán dựa vào cả hai trạng thái ẩn của hai mạng RNN ngược hướng này.



Hình 2.2 Mạng RNN hai chiều [16]

2.1.2. RNN (hai chiều) sâu

RNN sâu (Deep RNN) [16] cũng tương tự như RNN hai chiều, nhưng khác nhau ở chỗ chúng chứa nhiều tầng ẩn ở mỗi bước. Trong thực tế, chúng giúp cho việc học ở mức độ cao hơn, tuy nhiên ta cũng cần phải có nhiều dữ liệu huấn luyện hơn.



Hình 2.3 Mạng RNN (hai chiều) sâu [16]

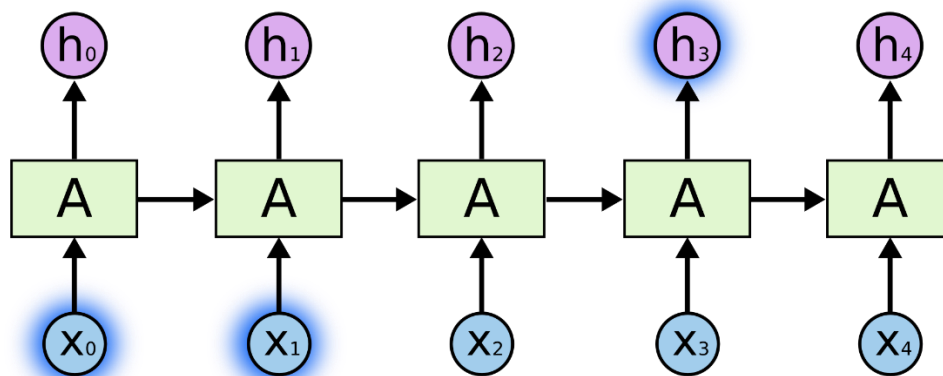
2.2. Mạng bộ nhớ dài-ngắn LSTM

Mạng LSTM[7] được sử dụng khá phổ biến, về cơ bản mô hình của LSTM không khác mô hình truyền thống của RNN, nhưng chúng sử dụng hàm tính toán khác ở các trạng thái ẩn. Bộ nhớ của LSTM được gọi là tế bào (Cell) và bạn có thể tưởng tượng rằng chúng là các hộp đen nhận đầu vào là trạng thái phía trước và đầu vào hiện tại. Bên trong hộp đen này sẽ tự quyết định cái gì cần phải nhớ và cái gì sẽ xóa đi. Sau đó, chúng sẽ kết hợp với trạng thái phía trước, nhớ hiện tại và đầu vào hiện tại. Vì vậy, ta có thể truy xuất được quan hệ của các từ phụ thuộc xa nhau rất hiệu quả.

2.2.1. Vấn đề phụ thuộc xa

Một điểm nổi bật của RNN chính là ý tưởng kết nối các thông tin phía trước để dự đoán cho hiện tại. Việc này tương tự như ta sử dụng các cảnh trước của bộ phim để hiểu được cảnh hiện thời. Nếu mà RNN có thể làm được việc đó thì chúng sẽ cực kỳ hữu dụng, tuy nhiên liệu chúng có thể làm được không còn tùy vào từng trường hợp.

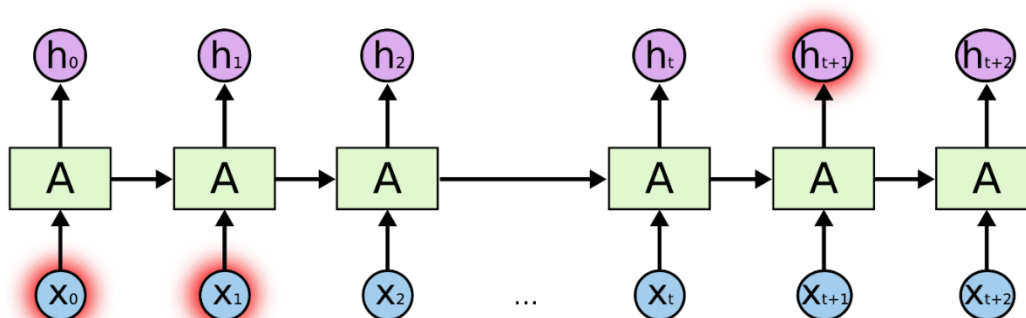
Đôi lúc ta chỉ cần xem lại thông tin vừa có thôi là đủ để biết được tình huống hiện tại. Ví dụ, với câu: “*Cách điều trị bệnh vảy nến*” thì chỉ cần đọc tới “*cách điều trị bệnh vảy*” là đủ biết được chữ tiếp theo là “*nến*” rồi. Trong tình huống này, khoảng cách tới thông tin có được cần để dự đoán là nhỏ, nên RNN hoàn toàn có thể học được.



Hình 2.4 RNN phụ thuộc short-term [5]

Nhưng trong nhiều tình huống ta buộc phải sử dụng nhiều ngữ cảnh hơn để suy luận. Ví dụ, dự đoán chữ cuối cùng trong đoạn: “*Tôi bị bệnh vậy nên... Da lỗi có vậy, màu hồng hoặc đỏ.*”. Rõ ràng là các thông tin gần (“*Da lỗi có vậy, màu hồng hoặc đỏ*”) chỉ ta biết được đằng sau nó sẽ là tên của một loại bệnh nào đó, còn không thể nào biết được đó là bệnh gì. Muốn biết là bệnh gì, thì ta cần phải có thêm ngữ cảnh “*Da lỗi có vậy, màu hồng hoặc đỏ*” nữa mới có thể suy luận được. Rõ ràng là khoảng cách thông tin lúc này có thể đã khá xa rồi.

Thật không may là với khoảng cách càng lớn dần thì RNN bắt đầu không thể nhớ và học được nữa.



Hình 2.5 RNN phụ thuộc long-term [4]

Về lý thuyết, RNN hoàn toàn có khả năng xử lý “*long-term dependencies*”, nghĩa là thông tin hiện tại có được là nhờ vào chuỗi thông tin trước đó. Nhưng trong thực tế, RNN không có khả năng này. Vấn đề này đã được Hochreiter và Bengio, và cộng sự đưa ra như một thách thức cho mô hình RNN [15].

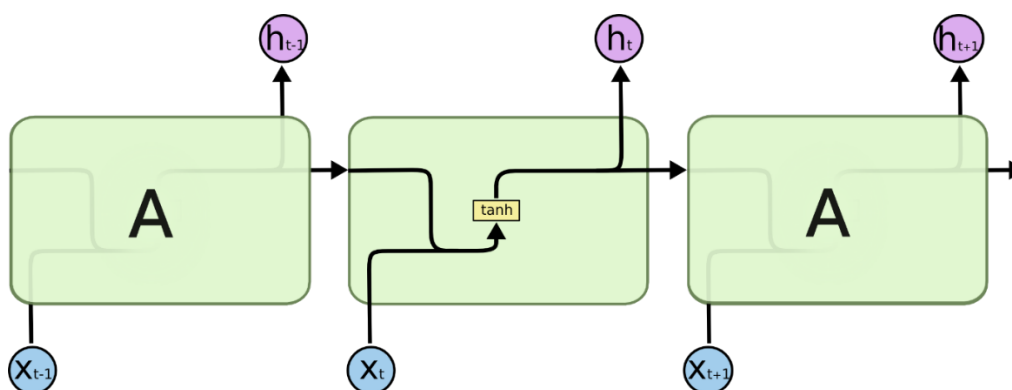
2.2.2. Kiến trúc mạng LSTM

Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks), thường được gọi là LSTM - là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa.

LSTM được giới thiệu bởi Hochreiter & Schmidhuber [15], và sau đó đã được cải tiến và phổ biến bởi rất nhiều người trong ngành. Chúng hoạt động cực kì hiệu quả trên nhiều bài toán khác nhau nên dần đã trở nên phổ biến như hiện nay.

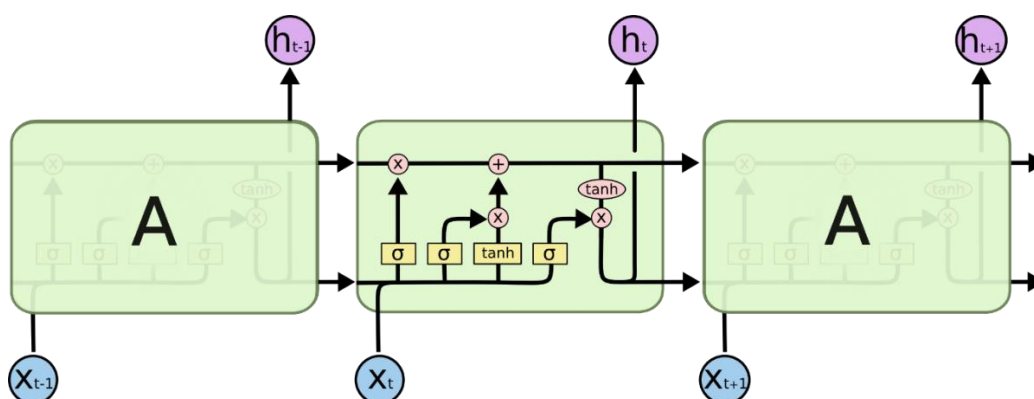
LSTM được thiết kế để tránh được vấn đề phụ thuộc xa. Việc nhớ thông tin trong suốt thời gian dài là đặc tính mặc định của chúng, chứ ta không cần phải huấn luyện nó để có thể nhớ được. Tức là ngay nội tại của nó đã có thể ghi nhớ được mà không cần bất kì can thiệp nào.

Mọi mạng hồi quy đều có dạng là một chuỗi các mô-đun lặp đi lặp lại của mạng nơ-ron. Với mạng RNN chuẩn, các mô-đun này có cấu trúc rất đơn giản, thường là một tầng *tanh*.








Hình 2.6 Các mô-đun lặp của mạng RNN chứa một lớp [4]

LSTM cũng có kiến trúc dạng chuỗi như vậy, nhưng các mô-đun trong nó có cấu trúc khác với mạng RNN chuẩn. Thay vì chỉ có một tầng mạng nơ-ron, chúng có tới bốn tầng tương tác với nhau một cách rất đặc biệt.



Hình 2.7 Các mô-đun lặp của mạng LSTM chứa bốn lớp [4]

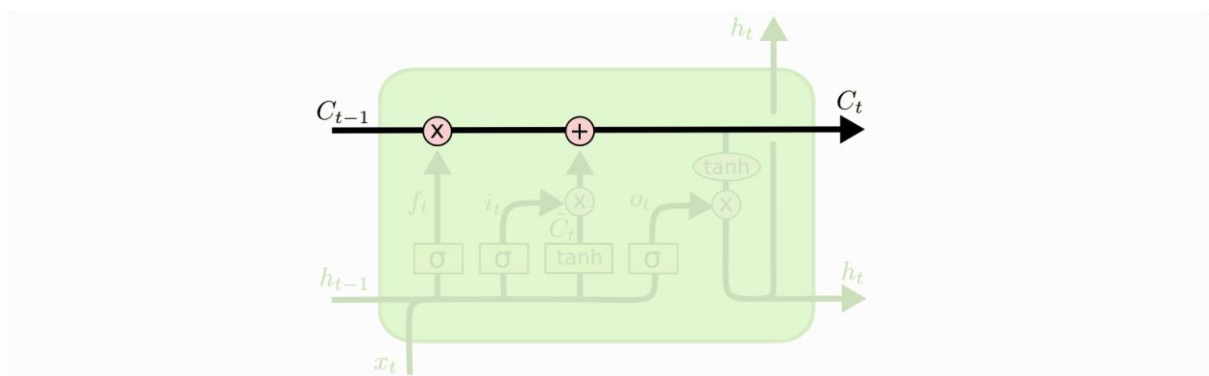
Trong đó, các ký hiệu sử dụng trong mạng LSTM có ý nghĩa như sau:

 Neural Network Layer	Là các lớp ẩn của mạng nơ-ron
 Pointwise Operation	Là toán tử Pointwise, biểu diễn các phép toán như cộng, nhân véc tơ
 Vector Transfer	Là véc tơ chỉ đầu vào và đầu ra của một nút
 Concatenate	Là biểu thị phép nối các toán hạng
 Copy	Là biểu thị cho sự sao chép từ vị trí này sang vị trí khác

2.2.3. Ý tưởng cốt lõi của LSTM

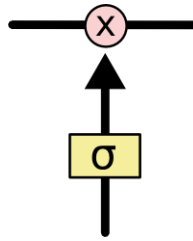
Chìa khóa của LSTM là trạng thái tế bào - chính đường chạy thông ngang phía trên của sơ đồ hình vẽ.

Trạng thái tế bào là một dạng giống như băng truyền. Nó chạy xuyên suốt tất cả các mắt xích (các nút mạng) và chỉ tương tác tuyến tính đôi chút. Vì vậy mà các thông tin có thể dễ dàng truyền đi thông suốt mà không sợ bị thay đổi.



Hình 2.8 Tế bào trạng thái LSTM giống như một băng truyền [4]

LSTM có khả năng bỏ đi hoặc thêm vào các thông tin cần thiết cho trạng thái tế bào, chúng được điều chỉnh cẩn thận bởi các nhóm được gọi là cổng. Các cổng là nơi sàng lọc thông tin đi qua nó, chúng được kết hợp bởi một tầng mạng sigma và một phép nhân.



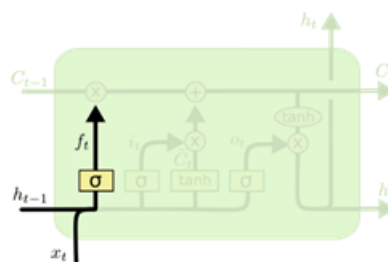
Hình 2.9 Cổng trạng thái LSTM [4]

Tầng sigma sẽ cho đầu ra là một số trong khoảng $[0, 1]$, mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là 00 thì có nghĩa là không cho thông tin nào qua cả, còn khi là 11 thì có nghĩa là cho tất cả các thông tin đi qua nó.

Một LSTM gồm có ba cổng như vậy để duy trì và điều hành trạng thái của tế bào. Và hoạt động thông qua các bước sau:

Bước đầu tiên của LSTM là quyết định xem thông tin nào cần bỏ đi từ trạng thái tế bào. Quyết định này được đưa ra bởi tầng sigma - gọi là “tầng cổng quên”. Nó sẽ lấy đầu vào là h_{t-1} và x_t rồi đưa ra kết quả là một số trong khoảng $[0,1]$ cho mỗi số trong trạng thái tế bào C_{t-1} . Đầu ra là một thể hiện rằng nó giữ toàn bộ thông tin lại, còn 0 chỉ rằng toàn bộ thông tin sẽ bị bỏ đi.

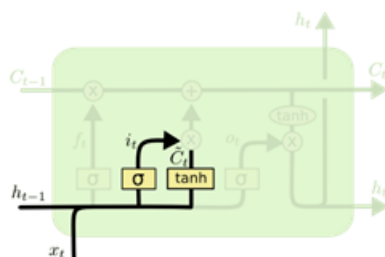
Quay trở lại với ví dụ mô hình ngôn ngữ dự đoán từ tiếp theo dựa trên tất cả các từ trước đó, với những bài toán như vậy, thì trạng thái tế bào có thể sẽ mang thông tin về giới tính của một nhân vật nào đó giúp ta sử dụng được đại từ nhân xưng chuẩn xác. Tuy nhiên, khi đề cập tới một người khác thì ta sẽ không muốn nhớ tới giới tính của nhân vật nữa, vì nó không còn tác dụng gì với chủ thể mới này.



Hình 2.10 LSTM focus f [4]

Bước tiếp theo là quyết định xem thông tin mới nào ta sẽ lưu vào trạng thái tế bào. Việc này gồm hai phần. Đầu tiên là sử dụng một tầng sigma được gọi là “tầng cổng vào” để quyết định giá trị nào ta sẽ cập nhật. Tiếp theo là một tầng \tanh tạo ra một véc-tơ cho giá trị mới C^t nhằm thêm vào cho trạng thái. Trong bước tiếp theo, ta sẽ kết hợp hai giá trị đó lại để tạo ra một cập nhật cho trạng thái.

Với ví dụ mô hình ngôn ngữ ta sẽ muốn thêm giới tính của nhân vật mới này vào trạng thái tế bào và thay thế giới tính của nhân vật trước đó.

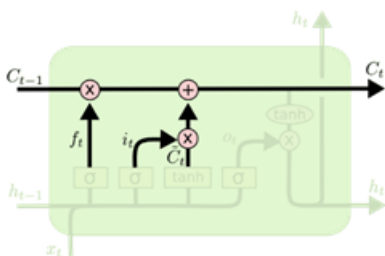


Hình 2.11 LSTM focus i [4]

Giờ là lúc cập nhật trạng thái tế bào cũ C_{t-1} thành trạng thái mới C_t . Ở các bước trước đó đã quyết định những việc cần làm, nên giờ ta chỉ cần thực hiện là xong.

Ta sẽ nhân trạng thái cũ với f_t để bỏ đi những thông tin ta quyết định quên lúc trước. Sau đó cộng thêm $i_t * C_t$. Trạng thái mới thu được này phụ thuộc vào việc quyết định cập nhật mỗi giá trị trạng thái ra sao.

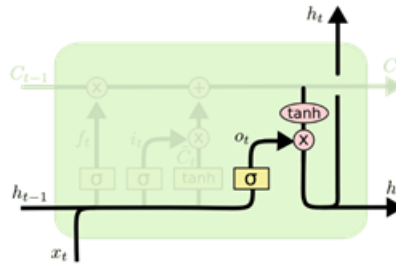
Với bài toán mô hình ngôn ngữ, chính là việc ta bỏ đi thông tin về giới tính của nhân vật cũ, và thêm thông tin về giới tính của nhân vật mới như ta đã quyết định ở các bước trước đó.



Hình 2.12 LSTM focus C [4]

Cuối cùng, cần quyết định xem ta muốn đầu ra là gì. Giá trị đầu ra sẽ dựa vào trạng thái tế bào, nhưng sẽ được tiếp tục sàng lọc. Đầu tiên, chạy một tầng sigma để quyết định phần nào của trạng thái tế bào muốn xuất ra. Sau đó, đưa nó trạng thái tế bào qua một hàm \tanh để có giá trị nó về khoảng $[-1, 1]$, và nhân nó với đầu ra của cổng sigma để được giá trị đầu ra mong muốn.

Với ví dụ về mô hình ngôn ngữ, chỉ cần xem chủ thể mà có thể đưa ra thông tin về một trạng từ đi sau đó. Ví dụ, nếu đầu ra của chủ thể là số ít hoặc số nhiều thì có thể biết được dạng của trạng từ đi theo sau nó phải như thế nào.



Hình 2.13 LSTM focus O [4]

2.3. Mô hình nhúng từ

Word Embedding là tên gọi chung của các mô hình ngôn ngữ và các phương pháp học theo đặc trưng trong Xử lý ngôn ngữ tự nhiên, ở đó các từ hoặc cụm từ được ánh xạ sang các véc tơ số (thường là số thực). Đây là một công cụ đóng vai trò quan trọng đối với hầu hết các thuật toán, kiến trúc học máy, học sâu trong việc xử lý dữ liệu đầu vào ở dạng văn bản, do chúng chỉ có thể hiểu được đầu vào ở dạng là số, từ đó mới thực hiện các công việc phân loại, hồi quy, v.v. Nhúng từ được phân chủ yếu thành hai loại:

- ✓ Phương pháp Véc tơ hóa dựa trên tần số xuất hiện (Frequency-based embedding)
- ✓ Phương pháp Véc tơ hóa dựa vào dự đoán (Prediction-based embedding)

2.3.1. Phương pháp Véc tơ hóa dựa trên tần số xuất hiện

Frequency-based Embedding [1] dựa vào tần số xuất hiện của các từ để tạo ra các véc tơ từ, trong đó có ba loại phổ biến nhất:

- ✓ Véc tơ đếm.
- ✓ Véc tơ $tf - idf$.
- ✓ Ma trận đồng xuất hiện.

Véc tơ đếm là dạng đơn giản nhất của Frequency-based Embedding, giả sử ta có D documents d_1, d_2, \dots, d_D và N là độ dài của từ điển, véc tơ biểu diễn của một từ là một véc tơ số nguyên và có độ dài là D , ở đó phần tử tại vị trí i chính là tần số của từ đó xuất hiện trong document d_i . Trong một số trường hợp, ta có thể lược bớt các từ có tần số xuất hiện thấp hoặc thay đổi mục nhập của véc tơ (thay vì tần số có thể thay bằng một giá trị nhị phân biểu thị sự xuất hiện của từ) tùy vào mục đích cụ thể.

Khác với véc tơ đếm chỉ xét đến tần số xuất hiện của từ trong một tài liệu, $tf - idf$ Véc tơ quan tâm cả tần số xuất hiện của từ trong toàn bộ tập dữ liệu, chính do đặc điểm này mà $tf - idf$ Véc tơ có tính phân loại cao hơn so với Count Véc tơ. $tf - idf$ (Term

Frequency-Inverse Document Frequency) Véc tơ là một véc tơ số thực cũng có độ dài D với D là số văn bản, nó được tính bằng tích của hai phần bao gồm tf và idf , công thức của mỗi phần tử của véc tơ được tính như sau:

$$tf_i = \frac{n_i}{N_i} \quad (2.1)$$

Trong đó:

- $i: 1 \dots D$
- n_i : tần số xuất hiện của từ trong văn bản i .
- N_i : tổng số từ trong văn bản i .

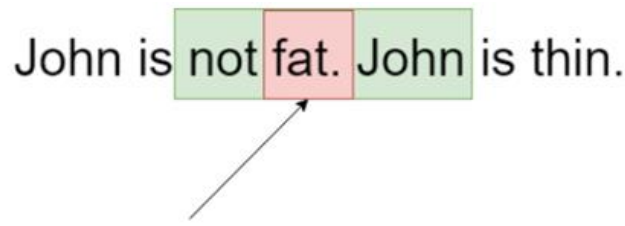
$$idf_i = \log_2 \frac{D}{d} \quad (2.2)$$

Trong đó:

- D : tổng số documents trong tập dữ liệu.
- d : số lượng documents có sự xuất hiện của từ.

$$tfidf_i = tf_i \times idf_i \quad (2.3)$$

Như đã đề cập ở trên, $tf - idf$ véc tơ có tính phân loại cao hơn so với véc tơ đếm chính là bởi nó được điều chỉnh bởi trọng số idf , dựa trên công thức của nó ta có thể hiểu rằng nếu từ xuất hiện ở càng nhiều văn bản (tính phân loại thấp) thì giá trị của nó càng nhỏ, từ đó kết quả cuối cùng sẽ bị nhỏ theo. Tuy nhiên, nhược điểm của cả hai phương pháp trên chính là việc nó chỉ chú trọng đến tần số xuất hiện của một từ, dẫn tới nó hầu như không mang ý nghĩa gì về mặt ngữ cảnh, Ma trận đồng xuất hiện phần nào giải quyết vấn đề đó. Ma trận đồng xuất hiện có ưu điểm là bảo tồn mối quan hệ ngữ nghĩa giữa các từ, được xây dựng dựa trên số lần xuất hiện của các cặp từ trong Context Window. Một Context Window được xác định bởi kích thước và hướng của nó. Hình dưới đây là một ví dụ của *Context Window*:



Context Window with size=1, direction=around

Thông thường, Ma trận đồng xuất hiện là một ma trận vuông đối xứng, mỗi hàng hoặc mỗi cột sẽ chính là véc tơ biểu thị của từ tương ứng. Tiếp tục ví dụ trên ta sẽ có ma trận đồng xuất hiện:

Bảng 2.1 Ví dụ về ma trận đồng xuất hiện

	John	is	not	fat	thin
John	0	2	0	1	0
is	2	0	1	0	1
not	0	1	0	1	0
fat	1	0	1	0	0
thin	0	1	0	0	0

Tuy nhiên, trong thực tế, do số lượng từ vựng nhiều, ta thường chọn cách bỏ đi một số từ không cần thiết (ví dụ như các stopwords) hoặc sử dụng phân tách SVD (Singular Value Decomposition) để giảm kích thước của véc tơ từ nhằm giúp cho biểu diễn của từ được rõ ràng hơn đồng thời tiết kiệm bộ nhớ dùng để lưu trữ ma trận đồng xuất hiện (do các ma trận đồng xuất hiện có kích thước rất lớn).

GloVe (Global Véc tơ) là một trong những phương pháp mới để xây dựng vec-tơ từ (được giới thiệu vào năm 2014), nó thực chất được xây dựng dựa trên ma trận đồng xuất hiện. GloVe có bản chất là xác suất, ý tưởng xây dựng phương pháp này đến từ tỉ số sau:

$$\frac{P(k|i)}{P(k|j)} \quad (2.4)$$

Trong đó:

- $P(k|i)$ là xác suất xuất hiện của từ k trong ngữ cảnh của từ i , tương tự với $P(k|j)$
- Công thức của $P(k|i)$:

$$P(k|i) = \frac{X_{ik}}{X_i} = \frac{X_{ik}}{\sum_m X_{im}} \quad (2.5)$$

Trong đó:

- X_{ik} : số lần xuất hiện của từ k trong ngữ cảnh của từ i (hoặc ngược lại).
- X_i : số lần xuất hiện từ i trong ngữ cảnh của toàn bộ các từ còn lại ngoại trừ i .

Ý tưởng chính của GloVe: độ tương tự ngữ nghĩa giữa hai từ i, j có thể được xác định thông qua độ tương tự ngữ nghĩa giữa từ k với mỗi từ i, j , những từ k có tính xác định ngữ nghĩa tốt chính là những từ làm cho (1) $\gg 1$ hoặc xấp xỉ bằng 0. Ví dụ, nếu i là “cái bàn”, j là “con mèo” và k là “cái ghế” thì (1) sẽ khá lớn do “cái ghế” có nghĩa gần hơn với “cái bàn” hơn là “con mèo”, ở trường hợp khác, nếu ta thay k là “cái kem” thì (1) sẽ xấp xỉ bằng 1 do “cái kem” hầu như chẳng liên quan gì tới “cái bàn” và “con mèo”.

Dựa trên tầm quan trọng của (1), GloVe khởi đầu bằng việc là nó sẽ tìm một hàm F sao cho nó ánh xạ từ các vec-tơ từ trong vùng không gian V sang một giá trị tỉ lệ với (1). Việc tìm F không đơn giản, tuy nhiên, sau nhiều bước đơn giản hóa cũng như tối ưu, ta có thể đưa nó về bài F toán hồi quy với việc tối ưu hóa hàm chi phí sau:

$$J = \sum_{i,j=1}^V f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \quad (2.6)$$

Trong đó:

- w_i, w_j là các vector từ.
- b_i, b_j là các bias tương ứng (được thêm vào ở các bước đơn giản hóa và tối ưu).
- X_{ij} : mục nhập tương ứng với cặp từ i, j trong ma trận đồng xuất hiện.

Hàm f được gọi là hàm trọng số, được thêm vào để giảm bớt sự ảnh hưởng của các cặp từ xuất hiện quá thường xuyên, hàm này thỏa ba tính chất:

- ✓ Có giới hạn tại 0.
- ✓ Là hàm không giảm.
- ✓ Có giá trị nhỏ khi x rất lớn.

Thực tế, có nhiều hàm số thỏa các tính chất trên, nhưng ta sẽ lựa chọn hàm số sau:

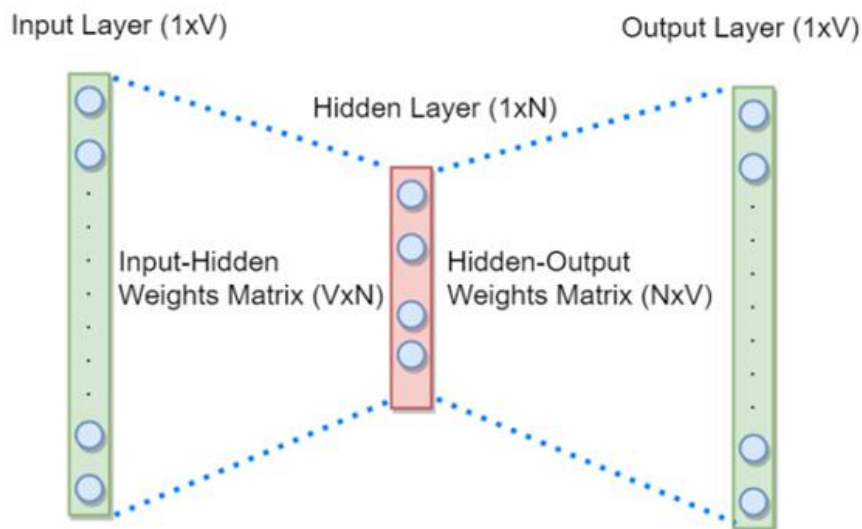
$$f(x) = \begin{cases} \left(\frac{x}{x_{\max}}\right)^\alpha & \text{nếu } x < x_{\max} \\ 1 & \text{khác} \end{cases} \quad (2.7)$$

Với $\alpha=3/4$. Việc thực hiện tối ưu hóa hàm chi phí J để tìm ra các vec-tơ từ W_i , W_j thể được thực hiện bằng nhiều cách, trong đó cách tiêu chuẩn nhất là sử dụng phương pháp Gradient Descent [1].

2.3.2. Phương pháp Véc tơ hóa dựa vào dự đoán

Prediction-based Embedding [1] xây dựng các véc tơ dựa vào các mô hình dự đoán. Tiêu biểu nhất chính là Word2vec, nó là sự kết hợp của hai mô hình: CBOW (Continuous Bag Of Words) và Skip-gram. Cả hai mô hình này đều được xây dựng dựa trên một mạng Noron gồm ba lớp: một lớp đầu vào, một lớp ẩn và một lớp đầu ra. Mục đích chính của các mạng Noron này là học các trọng số biểu diễn véc tơ từ.

CBOW hoạt động dựa trên cách thức là nó sẽ dự đoán xác suất của một từ được đưa ra theo ngữ cảnh (một ngữ cảnh có thể gồm một hoặc nhiều từ), với đầu vào là một hoặc nhiều One-hot véc tơ của các từ ngữ cảnh có chiều dài V (với V là độ lớn của từ điển), đầu ra sẽ là một véc tơ xác suất cũng với chiều dài V của từ liên quan hoặc còn thiếu, Lớp ẩn (Hidden Layer) có chiều dài N, N cũng chính là độ lớn của véc tơ từ biểu thị. Dưới đây là mô hình CBOW với ngữ cảnh là một từ đơn:



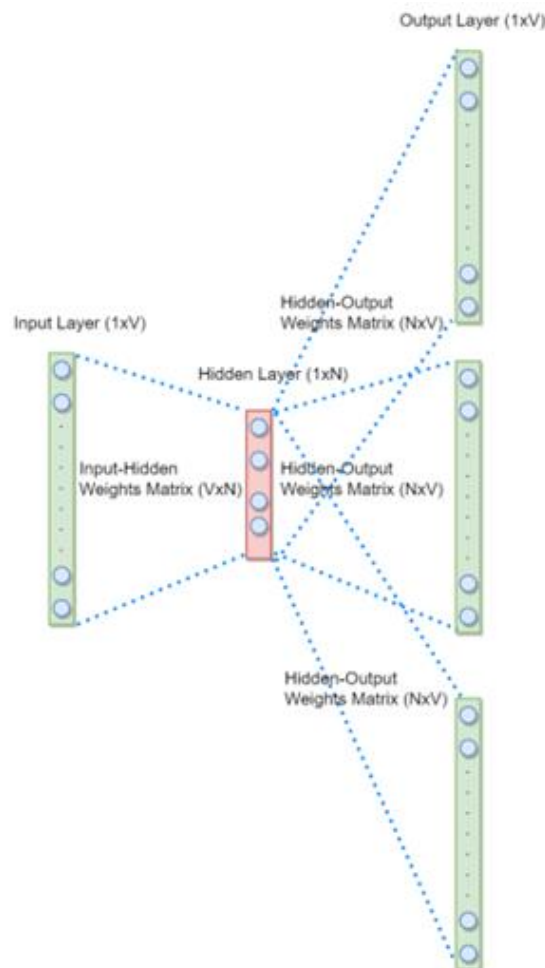
Hình 2.14 Mô hình CBOW với 1 đầu vào [3]

Về bộ dữ liệu dùng để train, đầu vào sẽ bao gồm các bộ One-hot véc tơ ngữ cảnh và các One-hot véc tơ của từ mong muốn.

Về cách thức hoạt động, ban đầu hai ma trận trọng số Ma trận trọng số ẩn đầu

vào và ma trận trọng số ẩn đầu ra được khởi tạo ngẫu nhiên, Đầu vào sẽ được nhân với ma trận đầu vào ra được một kết quả gọi là Kích hoạt ẩn, kết quả này sẽ được nhân tiếp với Ma trận trọng số đầu ra và cuối cùng được đưa vào một hàm softmax để ra được Kết quả là một véc tơ xác suất, Kết quả này sẽ được so sánh với kết quả mong muốn và tính toán độ lỗi, dựa vào độ lỗi này mà mạng Noron sẽ lan truyền ngược trở lại để cập nhật các giá trị của các ma trận trọng số. Đối với mô hình CBOW nhiều đầu vào, các thức hoạt động là tương tự, chỉ khác ở chỗ các kết quả thu được khi nhân các đầu vào với ma trận trọng số đầu vào sẽ được lấy trung bình để ra được Kích hoạt ẩn cuối cùng. Các trọng số của Ma trận trọng số đầu ra sau khi học xong sẽ được lấy làm biểu diễn của các véc tơ từ.

Mô hình Skip-gram có cấu trúc tương tự như CBOW, nhưng mục đích của nó là dự đoán ngữ cảnh của một từ đưa vào. Dưới đây là hình ảnh của mô hình Skip-gram:



Hình 2.15 Mô hình Skip-gram [3]

Về bộ dữ liệu dùng để huấn luyện cũng như cách thức hoạt động của mô hình Skip-gram hoàn toàn tương tự với mô hình CBOW 1 đầu vào, chỉ khác ở điểm thay vì chỉ có 1 độ lỗi, ta có nhiều độ lỗi do có nhiều véc tơ đầu ra, các độ lỗi này sẽ được tổng

hợp lại thành 1 độ lỗi cuối cùng để lan truyền ngược trở lại cập nhật các trọng số. Các trọng số của Ma trận trọng số đầu vào sau khi học xong sẽ được lấy làm biểu diễn của các véc tơ từ.

Ưu điểm của CBOW là nó không tốn nhiều bộ nhớ để lưu trữ các ma trận lớn, cũng như do nó có bản chất là xác suất cho nên việc hiện thực được cho là vượt trội hơn so với phương pháp khác, tuy nhiên vẫn còn tồn tại khuyết điểm các từ giống nhau nhưng nghĩa khác nhau vẫn chỉ được biểu diễn bằng một vec-tơ từ duy nhất.

So sánh giữa Word2vec và GloVe:

Về bản chất, rõ ràng Word2vec và GloVe khác nhau do thuộc hai loại nhúng khác nhau nhưng đều bắt nguồn từ Context Window, Word2vec sử dụng Context Window để tạo ra các tập huấn luyện cho mạng nơ-ron còn GloVe sử dụng nó để tạo ra ma trận đồng xuất hiện. Để ý kỹ một chút, ta thấy rằng GloVe mang tính “toàn cục” hơn là Word2vec vì GloVe tính toán xác suất từ dựa trên toàn bộ tập dữ liệu còn Word2vec học dựa trên các ngữ cảnh đơn lẻ, cũng chính vì lý do này mà GloVe có trội hơn Word2vec cũng như vài mô hình khác trong một số vấn đề về ngữ nghĩa, nhận dạng thực thể có gắn tên, v.v. Ngoài ra, GloVe có độ ổn định trung bình tốt hơn Word2vec, độ ổn định ở đây chính là độ biến thiên của kết quả giữa hai lần ta thực hiện việc học với cùng một điều kiện xác định (cùng bộ dữ liệu, cùng tham số, cùng điều kiện phần cứng, v.v.).

2.4. Mô hình BERT

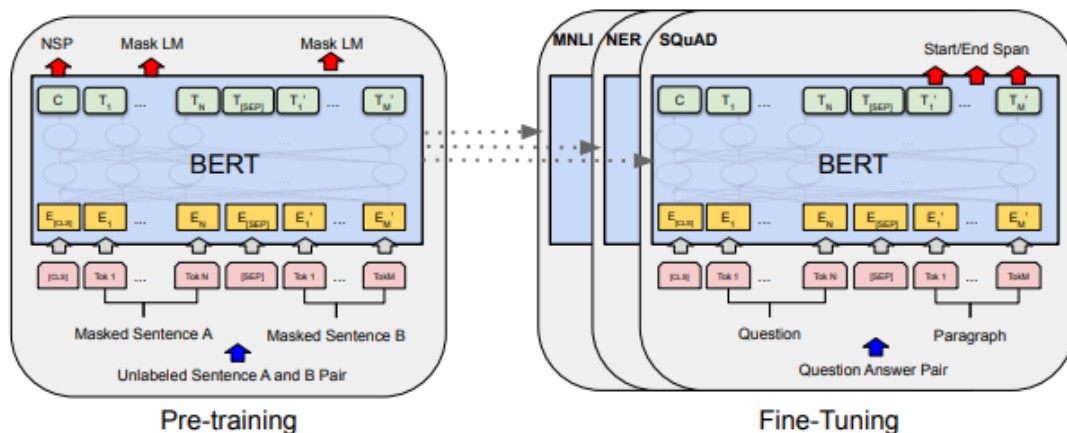
BERT[10] là viết tắt của cụm từ *Bidirectional Encoder Representation from Transformer* có nghĩa là mô hình biểu diễn từ theo hai chiều ứng dụng kỹ thuật biến đổi. BERT được thiết kế để huấn luyện trước các biểu diễn từ. Điểm đặc biệt ở BERT đó là nó có thể điều hòa cân bằng bối cảnh theo cả hai chiều trái và phải.

Cơ chế chú ý của Transformer sẽ truyền toàn bộ các từ trong câu văn đồng thời vào mô hình một lúc mà không cần quan tâm đến chiều của câu. Do đó Transformer được xem như là huấn luyện hai chiều mặc dù trên thực tế chính xác hơn thì có thể nói rằng đó là huấn luyện không chiều (non-directional). Đặc điểm này cho phép mô hình học được bối cảnh của từ dựa trên toàn bộ các từ xung quanh nó bao gồm cả từ bên trái và từ bên phải.

2.4.1. Mô hình Fine-tuning BERT

Một điểm đặc biệt ở BERT mà các mô hình trước đây chưa từng có đó là kết quả

huấn luyện có thể tinh chỉnh được. Ta sẽ thêm vào kiến trúc mô hình một lớp đầu ra để tùy biến theo tác vụ huấn luyện.



Hình 2.16 Toàn bộ tiến trình huấn luyện trước và tinh chỉnh của BERT [10]

Một kiến trúc tương tự được sử dụng cho cả mô hình huấn luyện và mô hình tinh chỉnh. Chúng ta sử dụng cùng một tham số huấn luyện để khởi tạo mô hình cho các tác vụ khác nhau. Trong suốt quá trình tinh chỉnh thì toàn bộ các tham số của các lớp học chuyển giao sẽ được chỉnh sửa. Đối với các tác vụ sử dụng đầu vào là một cặp nối tiếp (pair-sequence) ví dụ như câu hỏi và câu trả lời thì ta sẽ thêm mã khởi tạo là [CLS] ở đầu câu, mã [SEP] ở giữa để ngăn cách hai câu. Tiến trình áp dụng tinh chỉnh sẽ như sau:

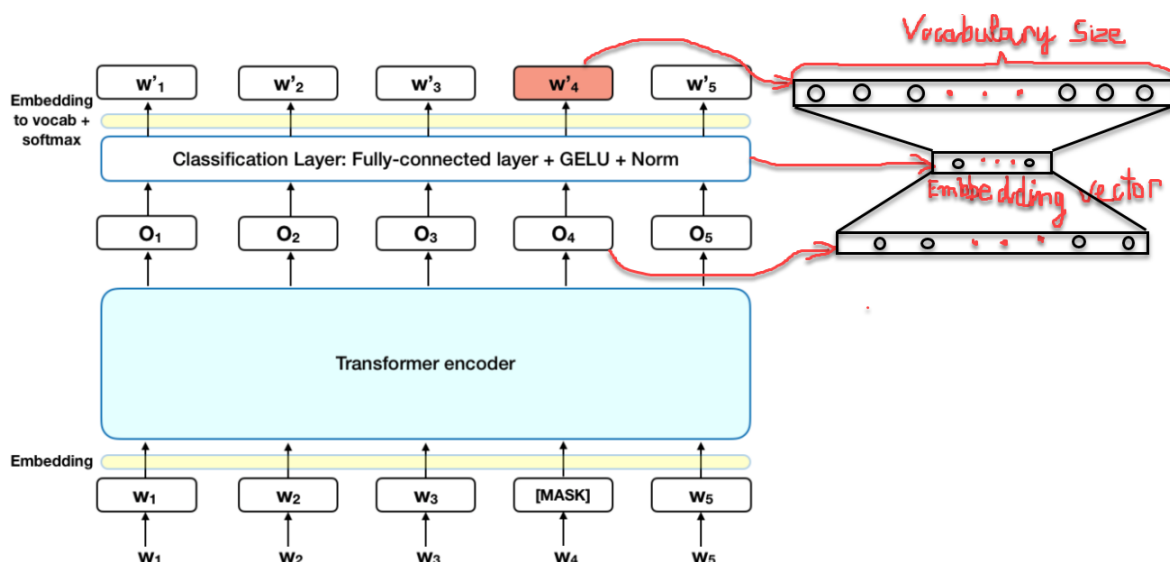
- Bước 1: Embedding toàn bộ các mã của cặp câu bằng các véc tơ nhúng từ mô hình huấn luyện trước. Các mã nhúng bao gồm cả hai mã là [CLS] và [SEP] để đánh dấu vị trí bắt đầu của câu hỏi và vị trí ngăn cách giữa 2 câu. 2 mã này sẽ được dự báo ở đầu ra để xác định các phần Bắt đầu/Kết thúc của câu đầu ra.
- Bước 2: Các véc tơ nhúng sau đó sẽ được truyền vào kiến trúc có chế chú ý multi-head với nhiều khối mã (thường là 6, 12 hoặc 24 khối tùy theo kiến trúc BERT). Ta thu được một véc tơ đầu ra ở bộ mã hóa.
- Bước 3: Để dự báo phân phối xác suất cho từng vị trí từ ở bộ giải mã, ở mỗi bước thời gian thì sẽ truyền vào véc tơ giải mã đầu ra của mã hóa và véc tơ nhúng đầu vào của bộ giải mã để tính. Sau đó tham chiếu qua lớp lót và hàm softmax để thu được phân phối xác suất cho đầu ra tương ứng ở bước thời gian.
- Bước 4: Trong kết quả trả ra ở đầu ra của transformer ta sẽ cố định kết quả của câu hỏi sao cho trùng với câu câu hỏi ở đầu vào. Các vị trí còn lại sẽ là thành phần mở rộng khoảng Bắt đầu/Kết thúc tương ứng với câu trả lời tìm được từ câu đầu vào.

Lưu ý: quá trình huấn luyện sẽ tinh chỉnh lại toàn bộ các tham số của mô hình BERT đã cắt bỏ lớp tuyến tính trên cùng và huấn luyện lại từ đầu các tham số của lớp tuyến tính mà chúng ta thêm vào kiến trúc mô hình BERT để chỉnh sửa lại phù hợp với bài toán.

2.4.2. Mô hình ngôn ngữ được che giấu (Masked ML)

Mô hình ngôn ngữ được che giấu là một tác vụ cho phép tinh chỉnh lại các biểu diễn từ trên các bộ dữ liệu văn bản không giám sát bất kỳ. Chúng ta có thể áp dụng mô hình ngôn ngữ được che giấu cho những ngôn ngữ khác nhau để tạo ra biểu diễn embedding cho chúng. Các bộ dữ liệu của tiếng anh có kích thước lên tới vài vài trăm tới vài nghìn GB được huấn luyện trên BERT đã tạo ra những kết quả khá ấn tượng.

Bên dưới là sơ đồ huấn luyện BERT theo tác vụ mô hình ngôn ngữ được che giấu



Hình 2.17 Sơ đồ kiến trúc BERT cho tác vụ Masked ML [10]

Theo đó:

- ✓ Khoảng 15% các mã của câu đầu vào được thay thế bởi mã $[MASK]$ trước khi truyền vào mô hình đại diện cho những từ bị che dấu (masked). Mô hình sẽ dựa trên các từ không được che (non-masked) dấu xung quanh $[MASK]$ và đồng thời là bối cảnh của $[MASK]$ để dự báo giá trị gốc của từ được che dấu. Số lượng từ được che dấu được lựa chọn là một số ít (15%) để tỷ lệ bối cảnh chiếm nhiều hơn (85%).
- ✓ Bản chất của kiến trúc BERT vẫn là một mô hình seq2seq gồm hai pha mã hóa giúp các từ đầu vào và bộ giải mã giúp tìm ra phân phối xác suất của các từ ở

đầu ra. Kiến trúc mã hóa Transformer được giữ lại trong tác vụ Mô hình ngôn ngữ được che giấu. Sau khi thực hiện cơ chế chú liên quan đến các vị trí khác nhau của một chuỗi ta sẽ thu được các véc tơ ở đầu ra là O_1, O_2, \dots, O_5 .

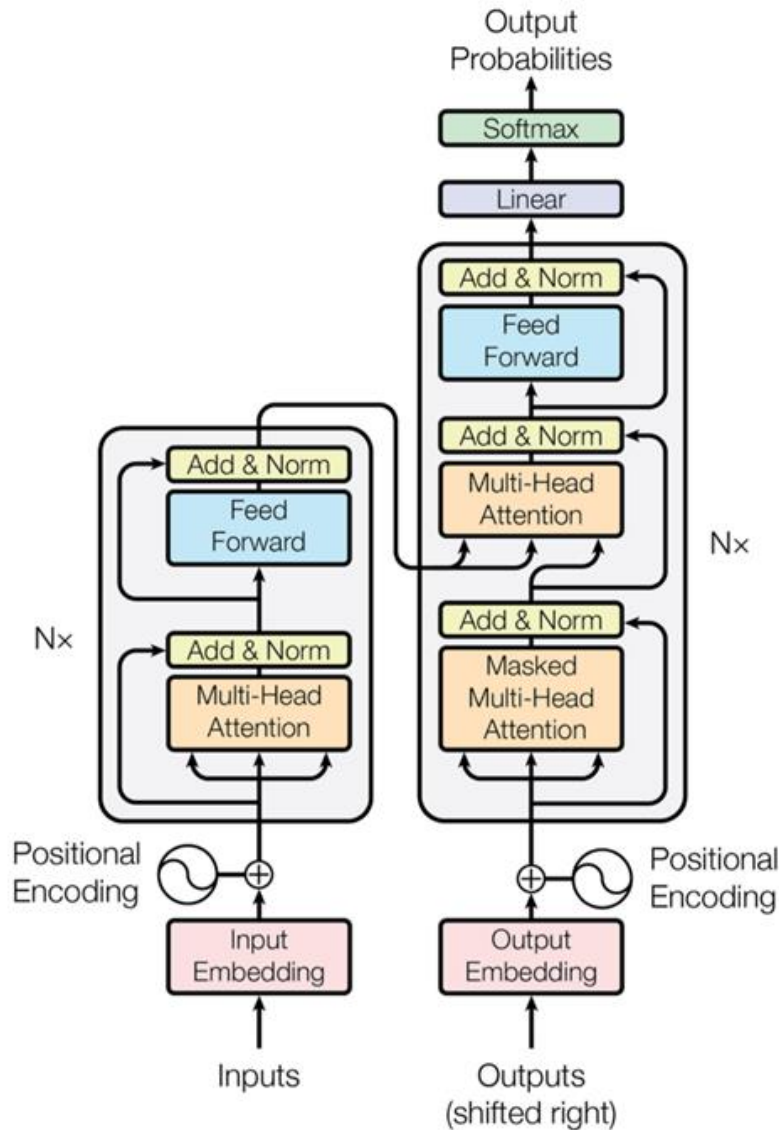
- ✓ Để tính toán phân phối xác suất cho từ đầu ra, chúng ta thêm một lớp kết nối đầy đủ ngay sau bộ mã hóa Transformer. Hàm softmax có tác dụng tính toán phân phối xác suất. Số lượng units của lớp kết nối đầy đủ phải bằng với kích thước của từ điển.
- ✓ Cuối cùng ta thu được véc tơ nhúng của mỗi một từ tại vị trí MASK sẽ là véc tơ O_i giảm chiều của véc tơ sau khi đi qua lớp kết nối đầy đủ như mô tả trên hình vẽ bên phải.

Hàm mất mát của BERT sẽ bỏ qua mất mát từ những từ không bị che dấu và chỉ đưa vào mất mát của những từ bị che dấu. Do đó mô hình sẽ hội tụ lâu hơn nhưng đây là đặc tính bù trừ cho sự gia tăng ý thức về bối cảnh. Việc lựa chọn ngẫu nhiên 15% số lượng các từ bị che dấu cũng tạo ra vô số các kịch bản đầu vào cho mô hình huấn luyện nên mô hình sẽ cần phải huấn luyện rất lâu mới học được toàn diện các khả năng.

2.4.3. Dự đoán câu tiếp theo

Đây là một bài toán phân loại học có giám sát với hai nhãn (hay còn gọi là phân loại nhị phân). Dữ liệu đầu vào của mô hình là một cặp câu sao cho 50% câu thứ hai được lựa chọn là câu tiếp theo của câu thứ nhất và 50% được lựa chọn một cách ngẫu nhiên từ bộ văn bản mà không có mối liên hệ gì với câu thứ nhất. Nhãn của mô hình sẽ tương ứng với *IsNext* khi cặp câu là liên tiếp hoặc *NotNext* nếu cặp câu không liên tiếp.

Cũng tương tự như mô hình câu hỏi và câu trả lời, chúng ta cần đánh dấu các vị trí đầu câu thứ nhất bằng mã $[CLS]$ và vị trí cuối các câu bằng mã $[SEP]$. Các mã này có tác dụng nhận biết các vị trí bắt đầu và kết thúc của từng câu thứ nhất và thứ hai.



Hình 2.18 Sơ đồ kiến trúc mô hình BERT cho tác vụ NSP [10]

Thông tin đầu vào được xử lý trước khi đưa vào mô hình huấn luyện bao gồm:

- ✓ Ngữ nghĩa của từ: Thông qua các véc tơ cho từng từ. Các véc tơ được khởi tạo từ mô hình huấn luyện.

Ngoài véc tơ biểu diễn từ của các từ trong câu, mô hình còn gắn vào thêm một số thông tin:

- ✓ Loại câu: Gồm hai véc tơ là nếu từ thuộc câu thứ nhất và nếu từ thuộc câu thứ hai.
- ✓ Vị trí của từ trong câu: là các véc tơ. Tương tự như vị trí từ trong transformer.

Véc tơ đầu vào sẽ bằng tổng của cả ba thành phần gắn theo *từ*, *câu* và *vị trí*.

2.4.4. Các kiến trúc mô hình BERT

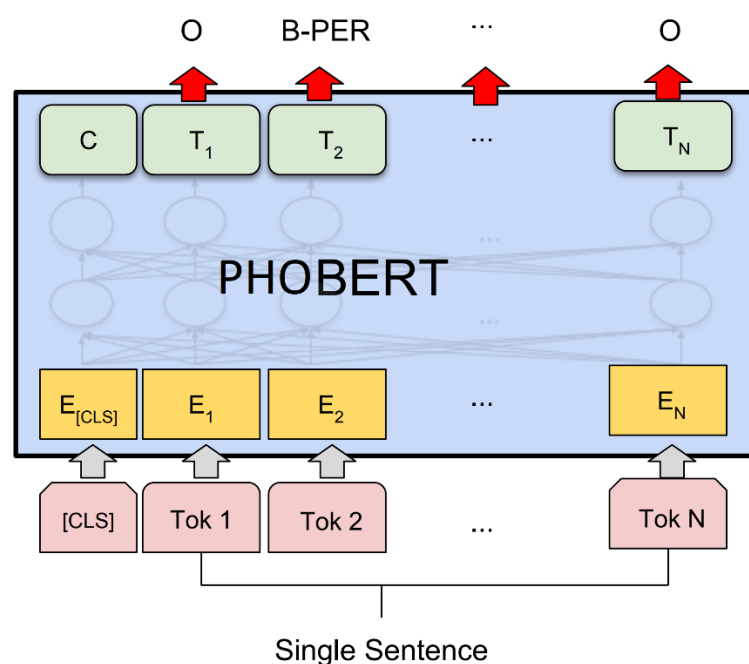
Hiện tại có nhiều phiên bản khác nhau của mô hình BERT. Các phiên bản đều dựa trên việc thay đổi kiến trúc của Transformer tập trung ở ba tham số: L : số lượng các khối lớp con trong transformer, H : kích thước của véc tơ (hidden size), A : Số lượng đầu (head) trong lớp multi-head, mỗi một đầu sẽ thực hiện một cơ chế chú ý (self-attention). Tên gọi của hai kiến trúc bao gồm:

- BERT_{BASE} ($L = 12, H = 768, A = 12$): Tổng tham số 110 triệu.
- BERT_{LARGE} ($L = 24, H = 1024, A = 16$): Tổng tham số 340 triệu.

Như vậy ở kiến trúc BERT Large chúng ta tăng gấp đôi số tầng, tăng kích thước ẩn của véc tơ gấp 1.33 lần và tăng số lượng head trong tầng multi-head gấp 1.33 lần.

2.5. Mô hình PhoBERT

Mô hình PhoBERT được đào tạo trước là mô hình ngôn ngữ hiện đại nhất dành cho Tiếng Việt (Pho , tức "Phở", là một món ăn phổ biến ở Việt Nam):



Hình 2.19 Minh họa về PhoBERT[13]

Hiện có hai phiên bản PhoBERT là "base" và "large" là mô hình ngôn ngữ đơn ngữ quy mô lớn đầu tiên được đào tạo trước cho tiếng Việt. Phương pháp tiếp cận đào tạo trước của PhoBERT dựa trên RoBERTa, tối ưu hóa quy trình đào tạo trước BERT để có hiệu suất mạnh mẽ hơn.

PhoBERT vượt trội hơn các phương pháp tiếp cận đơn ngữ và đa ngôn ngữ trước đây, đạt được những trình diễn hiện đại mới về bốn nhiệm vụ NLP của tiếng Việt là gán thẻ Phần của giọng nói, Phân tích cú pháp phụ thuộc, Nhận dạng thực thể được đặt tên và Suy luận ngôn ngữ tự nhiên. Chúng ta có thể ứng dụng PhoBERT trong một số tác vụ như:

- ✓ Tìm từ đồng nghĩa, trái nghĩa, cùng nhóm dựa trên khoảng cách của từ trong không gian biểu diễn đa chiều.
- ✓ Xây dựng các véc tơ cho các tác vụ NLP như phân tích cảm xúc, phân loại văn bản, NER, POS, huấn luyện chatbot.
- ✓ Gợi ý từ khóa tìm kiếm trong các hệ thống tìm kiếm.
- ✓ Xây dựng các ứng dụng seq2seq như robot viết báo, tóm tắt văn bản, sinh câu ngẫu nhiên với ý nghĩa tương đồng.
- ✓ Đây là một mô hình huấn luyện trước được huấn luyện ngôn ngữ đơn ngữ, tức là chỉ huấn luyện dành riêng cho tiếng Việt. Việc huấn luyện dựa trên kiến trúc và cách tiếp cận giống RoBERTa của Facebook được Facebook giới thiệu giữa năm 2019. Đây là một cải tiến so với BERT trước đây.
- ✓ Tương tự như BERT, PhoBERT cũng có hai phiên bản là **BERT_{BASE}** với 12 khối transformers và **BERT_{LARGE}** với 24 khối transformers.
- ✓ PhoBERT được huấn luyện trên khoảng 20GB dữ liệu bao gồm khoảng 1GB Vietnamese Wikipedia corpus và 19GB còn lại lấy từ Vietnamese news corpus. Đây là một lượng dữ liệu khá ổn để huấn luyện một mô hình như BERT.
- ✓ PhoBERT sử dụng RDRSegmenter của VnCoreNLP ⁸ để tách từ cho dữ liệu đầu vào trước khi qua bộ mã hóa BPE.
- ✓ Như đã nói ở trên, do tiếp cận theo tư tưởng của RoBERTa, PhoBERT chỉ sử dụng nhiệm vụ MLM để huấn luyện, bỏ đi nhiệm vụ NSP.

2.6 Tổng quan về Rasa

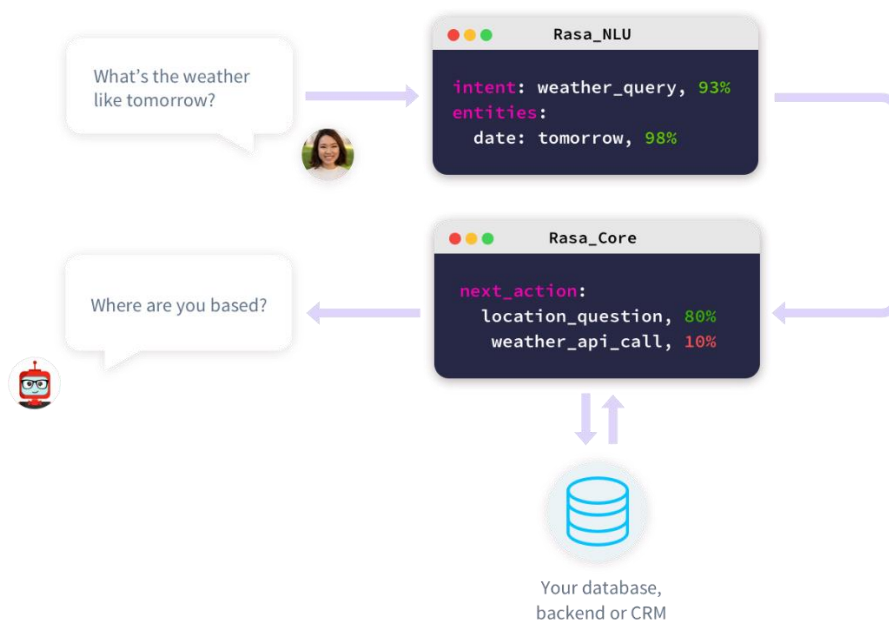
Rasa⁹ là một thư viện mã nguồn mở để hiểu ngôn ngữ tự nhiên được sử dụng để phân loại ý định, tạo và truy xuất phản hồi, trích xuất thực thể trong thiết kế các cuộc hội thoại chatbot. Thành phần NLU của Rasa từng tách biệt nhưng được hợp nhất với Rasa Core thành một khuôn khổ duy nhất.

⁸ <https://github.com/vncorenlp/VnCoreNLP>

⁹ <https://rasa.com/docs/rasa/nlu-training-data/>

Trước tiên, để bắt đầu đi vào quy trình xây dựng một chatbot bằng Rasa, chúng ta cần biết Rasa là gì và Rasa có những ưu điểm vượt trội gì để được lựa chọn cho việc xây dựng chatbot.

2.6.1 Các thành phần trong Rasa



Hình 2.20 Các thành phần chính trong Rasa [12]

1. **Rasa NLU** - một thư viện để hiểu ngôn ngữ tự nhiên thực hiện phân loại ý định và trích xuất thực thể từ đầu vào của người dùng và giúp bot hiểu những gì người dùng đang nói.

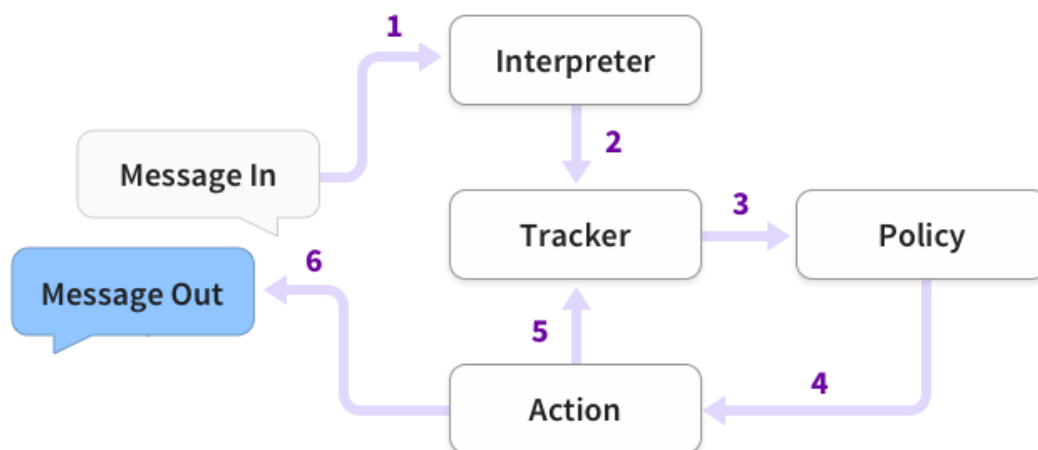
2. **Rasa Core** - một khung chatbot với quản lý hội thoại, lấy đầu vào có cấu trúc từ NLU. NLU và Core là độc lập và người ta có thể sử dụng NLU mà không cần Core, và ngược lại.

Hiện nay khung Rasa được phát triển tốt và có một số điểm mạnh sau:

- ✓ Nền tảng Rasa là mã nguồn mở, do đó Rasa giúp bạn biết chính xác được bạn đang làm gì với chatbot của mình, thậm chí có thể chỉnh sửa chatbot theo ý thích của bản thân, ví dụ như: Action, Tokenizer, hay cả việc quyết định nền tảng tin nhắn của chatbot.
- ✓ Rasa hoạt động khá tốt và mạnh mẽ, đặc biệt trong vấn đề xác định ý định người dùng và đối tượng được nhắc đến trong câu dù dữ liệu bạn thu thập và cung cấp cho rasa là vô cùng ít.
- ✓ Rasa hoạt động trên cơ sở học tập tương tác. Ngay cả khi không có đủ dữ liệu

để huấn luyện chatbot, vẫn có thể dễ dàng tạo dữ liệu bằng cách nói chuyện với chatbot trong giai đoạn đầu của quá trình phát triển.

2.6.2 Kiến trúc của Rasa



Hình 2.21 Kiến trúc của Rasa [12]

Trong kiến trúc trên, Rasa framework có bốn thành phần quan trọng là: Trình thông dịch (Interpreter), Trình theo dõi (Tracer), Chính sách (Policy), Hành động (Action).

Bất cứ khi nào một người nhập và gửi tin nhắn tới chatbot RASA, tin nhắn đó sẽ được nhận và chuyển cho Interpreter. Chính trình thông dịch này xác định mục đích của thông điệp và trích xuất các thực thể ra khỏi nó.

Trình theo dõi (Tracer) có thể coi là bộ nhớ của chatbot, sẽ theo dõi trạng thái của cuộc trò chuyện giữa người dùng và bot. Bao gồm các thông tin như: các thông tin sự kiện, hành động, slot, v.v.

Chính sách trong kiến trúc theo dõi trạng thái hội thoại hiện tại và quyết định đâu là hành động thích hợp của bot. Hành động đã chọn sau đó cũng được theo dõi bởi Tracer và sau đó được gửi đến người dùng dưới dạng phản hồi.

CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG CHATBOT TƯ VẤN NGƯỜI DÙNG TRONG Y TẾ DA LIỄU

Để có một ứng dụng chatbot hiệu quả và trải nghiệm người dùng thú vị, chatbot phải được thiết kế để tạo các tương tác tự nhiên nhất có thể; và điều này yêu cầu các mô hình học máy có thể cho phép bot hiểu được mục đích và ngữ cảnh của các cuộc trò chuyện. Đây là lúc các công cụ xử lý và hiểu ngôn ngữ tự nhiên ra đời.

Hiện nay có rất nhiều nền tảng hỗ trợ và phát triển chatbot tốt như: Chatfuel, Manychat, Harafunnel, Messnow... nhưng qua quá trình nghiên cứu thì thấy Rasa là nền tảng tốt và phù hợp nhất với bài toán này. Vì thế chương này sẽ tập trung mô tả chi tiết các bước xây dựng Chatbot dựa trên nền tảng Rasa.

3.1. Xây dựng dữ liệu

Nguồn dữ liệu sử dụng thử nghiệm trong luận văn được thu thập từ tài liệu chuyên khoa ngành y tế da liễu [14] và tham khảo qua một số chuyên gia bác sỹ trong ngành. Quá trình xây dựng dữ liệu chatbot chính là việc xây dựng các ý định người dùng, trích xuất các entity từ ý định người dùng, tạo các khung kịch bản và xây dựng các câu trả lời cho bot.

3.1.1. Xây dựng các Ý Định

Như đã nói ở trên, NLP được xây dựng với mục đích là xác định mục đích/ý định của người dùng trong tin nhắn và trích xuất các thực thể có trong tin nhắn đó. Hiện tại bot tạo ra 20 intent mà người dùng thường hay tương hỏi. Sau đây là ví dụ cho tình huống hỏi thông tin bệnh và thuốc:

```
- intent: hoi_thong_tin_benh
examples: |
- bệnh này là gì?
- bệnh này là như thế nào?
- bệnh gì
- tôi muốn hỏi thông tin về bệnh da liễu
- hỏi bệnh da liễu
- tư vấn bệnh da liễu

- intent: hoi_thong_tin_thuoc
examples: |
- thuốc chữa bệnh
- thuốc da liễu
- thuốc điều trị
- tác dụng của thuốc là gì?
- tác dụng thuốc
- cách sử dụng thuốc như thế nào
- tác dụng phụ của thuốc là gì?
- chống chỉ định thuốc
- tác dụng phụ khi sử dụng thuốc
```

3.1.2. Xây dựng các loại Entity và Slot

Thực thể là các thông tin được trích xuất từ tin nhắn của người dùng.

Slot như là bộ nhớ của bot, chúng hoạt động như một kho lưu trữ dạng key-value, có thể được sử dụng để lưu trữ thông tin mà người dùng cung cấp (ví dụ: tên bệnh hay triệu chứng của khách) cũng như thông tin thu thập được bên ngoài (ví dụ: kết quả của một truy vấn từ cơ sở dữ liệu). Hiện tại, bot có 12 Entity và 10 Slot.

```
entities:
- thongtinbenh_value
- hinh_anh_benh
- thongtinthuoc_value
- ten_benh
- ten_thuoc
- trieu_chung
- vi_tri
- tuoi
- tinh_trang
- dieu_tri
- bien_chung
- tac_dung

slots:
  fullname:
    type: text
    influence_conversation: true
    mappings:
      - type: from_text
        conditions:
          - active_loop: name_form
            requested_slot: fullname
  age:
    type: text
    influence_conversation: true
    mappings:
      - type: from_text
        conditions:
          - active_loop: age_form
```

3.1.3. Xây dựng Câu trả lời cho Bot

Sau khi chatbot đã hiểu người dùng muốn cái gì rồi thì nó sẽ cần trả lời lại người dùng. Và phần câu trả lời sẽ đảm nhận nhiệm vụ đấy. Dưới đây là một số mẫu trả lời người dùng

```
responses:
  utter_ask_ten_benh:
    - text: Bạn vui lòng cho biết bệnh đang gặp là gì?
  utter_ask_trieu_chung:
    - text: Bạn vui lòng cung cấp thêm triệu chứng đang gặp?
  utter_ask_ten_thuoc:
    - text: Bạn vui lòng cung cấp tên thuốc là gì?
  utter_chao_hoi:
    - text: "Xin chào! Leskin bot có thể hỗ trợ gì cho anh chị?"
    - text: Xin chào bạn nha!
    - text: Xin chào bạn
```



```

- text: Xin chào bạn, cảm ơn bạn đã liên hệ
utter_did_that_help:
- text: Tôi có thể giúp gì cho bạn?
- text: Mình là chatbot mình có thể giúp gì cho bạn?
utter_cam_on:
- text: Không có gì bạn nhé. Cảm ơn bạn đã liên hệ
utter_happy:
- text: Tuyệt vời!
utter_dong_y:
- text: Vâng bạn nhé!
- text: Chúc bạn thành công

```

Ở đây, ta có thể cho bot trả lời một hoặc nhiều câu. Nếu có nhiều sự lựa chọn cho bot thì nó sẽ dùng phương pháp ngẫu nhiên để chọn ra câu trả lời, như thế sẽ tạo tính tự nhiên trong hội thoại chat.

3.1.4. Xây dựng Khung kịch bản và Quy tắc

Stories là bản trình bày cuộc trò chuyện giữa người dùng và bot, được chuyển đổi thành một định dạng cụ thể trong đó thông tin người dùng nhập vào được thể hiện dưới dạng ý định (và thực thể khi cần thiết), trong khi phản hồi và hành động của bot được thể hiện dưới dạng tên hành động. Dưới đây là ví dụ của Story hỏi thông tin bệnh của người dùng:

```

stories:
- story: thong_tin_benh
  steps:
  - intent: chao_hoi
  - action: utter_chao_hoi
  - intent: hoi_thong_tin_benh
  - action: thong_tin_benh_form
  - active_loop: thong_tin_benh_form
  - slot_was_set:
    - requested_slot: ten_benh
  - slot_was_set:
    - ten_benh: ten_benh_test
  - slot_was_set:
    - requested_slot: trieu_chung
  - slot_was_set:
    - trieu_chung: trieu_chung_test
  - slot_was_set:
    - requested_slot: null
  - active_loop: null
  - action: action_thong_tin_benh_form
  - action: utter_xac_nhan_benh
  - action: utter_hoi_them_thong_tin_benh

```

Rules là các quy tắc, các cuộc hội thoại của người dùng sẽ luôn đi theo một đường mòn nhất định. Các quy tắc rất tuyệt vời để xử lý các mẫu hội thoại nhỏ cụ thể, nhưng không giống như Stories, các quy tắc không có khả năng tổng quát hóa các đường dẫn hội thoại không nhìn thấy được. Kết hợp các quy tắc và câu chuyện để làm cho bot mạnh mẽ và có thể xử lý hành vi của người dùng thực.

```

- rule: Activate thong_tin_benh form

```

```

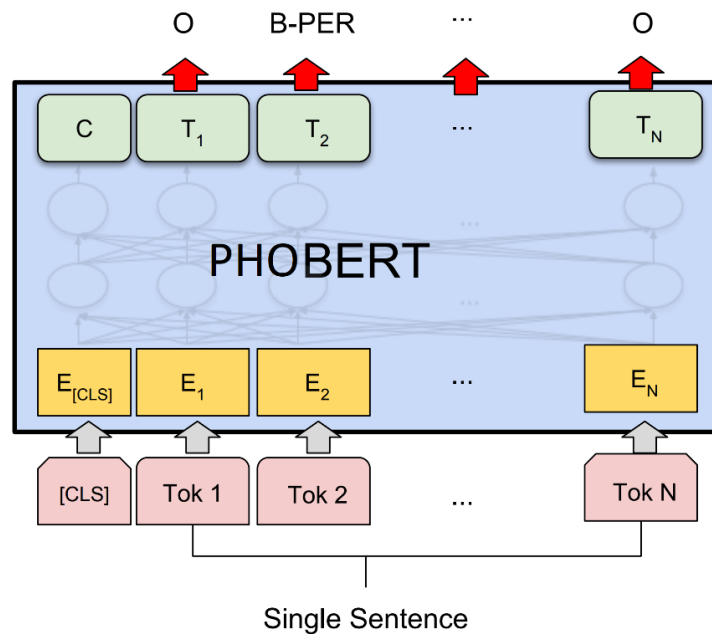
steps:
- intent: hoi_thong_tin_benh
- action: thông_tin_benh_form
- active_loop: thông_tin_benh_form

- rule: Submit thông_tin_benh_form
condition:
- active_loop: thông_tin_benh_form
steps:
- action: thông_tin_benh_form
- active_loop: null
- slot_was_set:
  - requested_slot: null
- action: action_thông_tin_benh_form
- action: utter_xác_nhan_benh
- action: utter_hoi_them_thông_tin_benh

```

3.2. Cải tiến nhận dạng ý định và tên riêng sử dụng PhoBERT

Bert thường ứng dụng để cải tiến hiệu quả các bài toán NLP đặc biệt như bài toán xác định ý định và nhận dạng tên riêng.

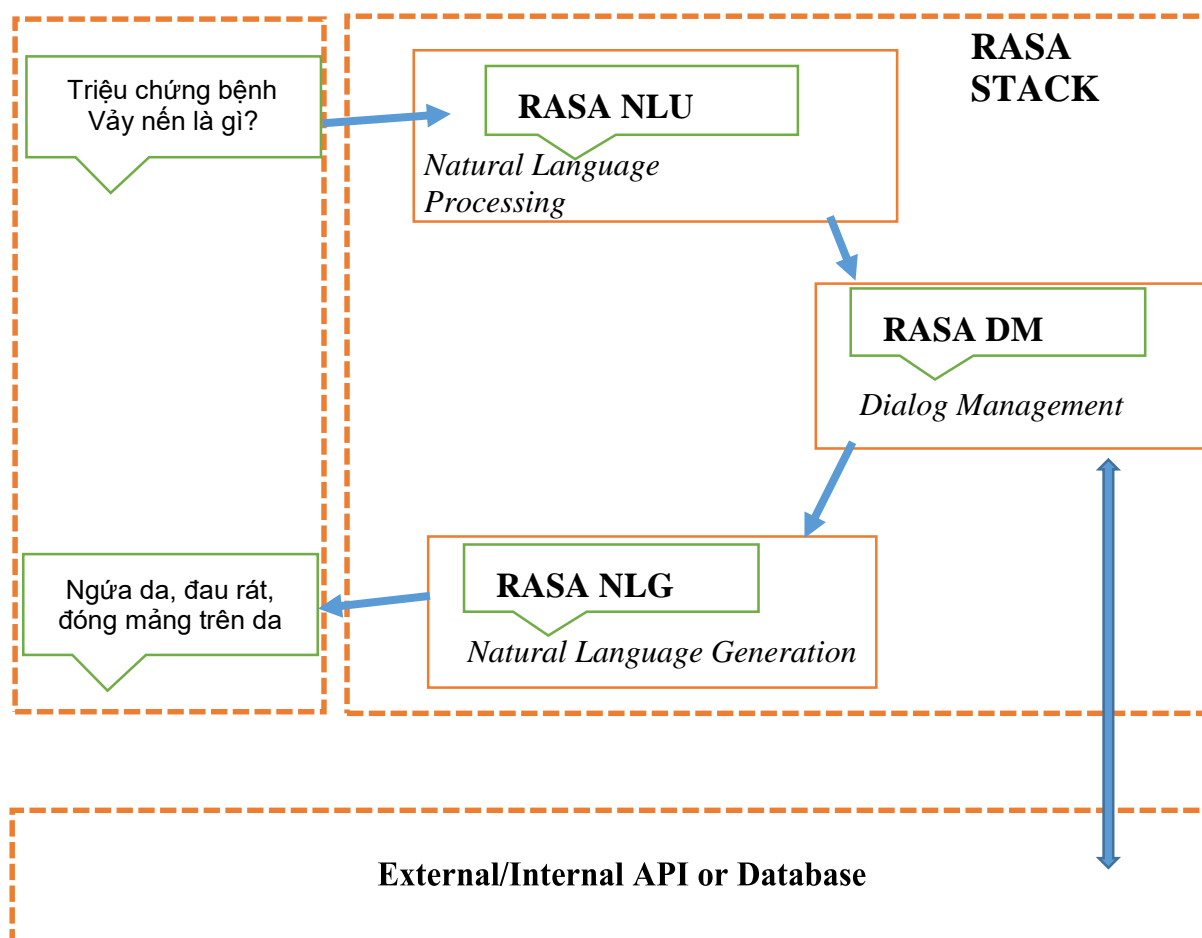


Hình 3.1 Kiến trúc mô hình PhoBERT

Vì hệ thống chatbot này sử dụng với ngôn ngữ tiếng Việt, nên ứng dụng sử dụng PhoBERT cho bài toán xác định ý định và bài toán nhận dạng tên riêng. Mô hình PhoBERT áp dụng cho các bài toán trên được mô tả trong hình trên.

3.3. Áp dụng Rasa xây dựng chatbot trong ngành Y Tế đa liễu

Cấu trúc hệ thống chatbot trong lĩnh vực Y Tế đa liễu dựa trên nền tảng Rasa được thiết kế như sau:



Hình 3.2 Cấu trúc hệ thống chatbot trong ngành Y Tế da liễu

Trong cấu trúc chatbot trên, từ câu hỏi người dùng thông qua Rasa Stack sẽ cho đầu ra là câu trả lời của Bot. Các thành phần trong Framework có vai trò như sau:

Rasa NLU (Hiểu ngôn ngữ tự nhiên): là phần đầu tiên và cũng là phần mà Rasa được sử dụng nhiều nhất, có nhiệm vụ véc tơ hóa ngôn ngữ, phân loại ý định người dùng và trích xuất ra các thông tin người dùng. Ví dụ câu đầu vào người dùng hỏi “triệu chứng bệnh vảy nến?” thì hệ thống sẽ véc tơ hóa nó rồi đối chiếu với các tập dữ liệu training đã được gán nhãn để đưa ra ý định là “trieu_chung”. Tiếp đến hệ thống trích xuất được thông tin với bệnh là “vảy nến”.

Rasa DM (Quản lý hội thoại): Dựa vào trạng thái và ngữ cảnh hội thoại để xác định ra action xử lý cho câu đầu vào trên. Thành phần này cũng có nhiệm vụ lấy dữ liệu từ hệ thống Database hoặc API để phục vụ việc sinh dữ liệu trả lời của bot cho thành phần NLG. Ở ví dụ trên thì hệ thống xác nhận bot đang trong ngữ cảnh hỏi về thông tin triệu chứng bệnh nhưng chưa rõ hỏi về triệu chứng nào nên bot sẽ đưa ra quyết định hỏi lại người dùng. Trong trường hợp mà bot có đầy đủ thông tin hỏi về triệu chứng thì sẽ lấy dữ liệu từ Cơ sở dữ liệu để trả về cho người dùng.

Rasa NLG (Sinh ngôn ngữ tự nhiên): Bot sinh câu trả lời dựa vào dữ liệu từ thành phần DM theo các mẫu câu template đã được xây dựng trước.

3.4. Cài đặt Rasa

Để bắt đầu cài đặt Rasa yêu cầu có Python 3.6, 3.7 hoặc 3.8, sau đó sử dụng lệnh sau:

```
pip install rasa==1.7.0
```

Sau đó di chuyển đến thư mục muốn khởi tạo Rasa và sử dụng:

```
Rasa init
```

Một folder sẽ được tạo ra bao gồm các file sau:

- ✓ data/nlu.md
- ✓ config.yml
- ✓ data/stories.md
- ✓ domain.yml
- ✓ action.py
- ✓ endpoints.yml
- ✓ credentials.yml

Đầu tiên, ta cần cấu hình file *config.yml*, đây là phần cấu hình cho NLU, nơi chúng ta lựa chọn ngôn ngữ, mô hình cần thiết. Chúng ta sẽ lựa chọn ngôn ngữ ở đây là vi (vietnam).

```
# Configuration for Rasa NLU.  
language: vi  
pipeline: supervised_embeddings
```

Tiếp theo là đến cấu hình đường ống:

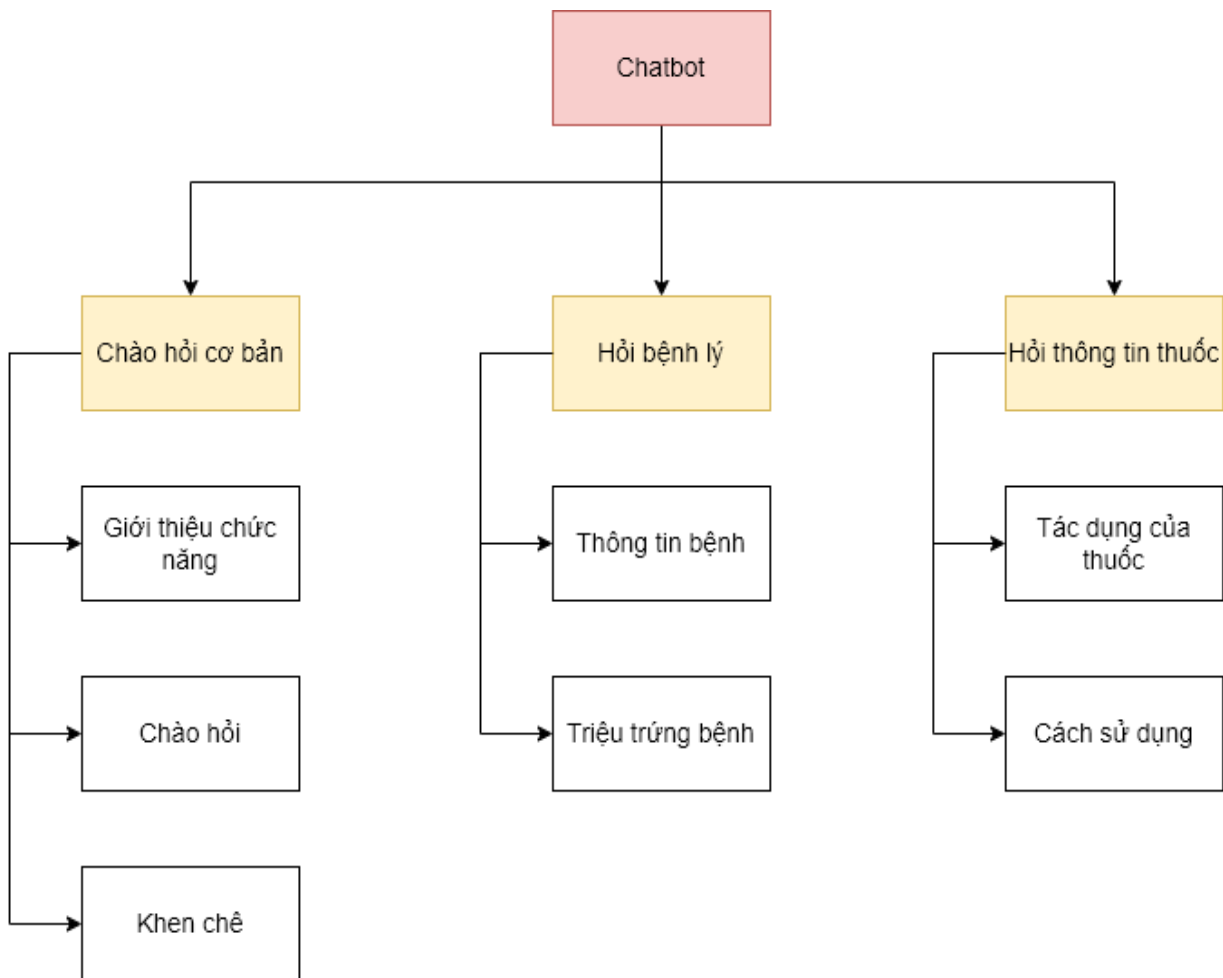
```
pipeline: supervised_embeddings  
=  
pipeline:  
  - name: "WhitespaceTokenizer"  
  - name: "RegexFeaturizer"  
  - name: "CRFEntityExtractor"  
  - name: "EntitySynonymMapper"  
  - name: "CountVectorFeaturizer"  
  - name: "CountVectorFeaturizer"  
    analyzer: "char_wb"  
    min_ngram: 1
```

```
max_ngram: 4
- name: "EmbeddingIntentClassifier"
```

Đây là một quy trình hoàn chỉnh từ lựa chọn mã thông báo, chuẩn hóa, nhận dạng thực thể đến phân loại. Tất nhiên các ta hoàn toàn có thể lựa chọn thay thế bất cứ một công đoạn nào trong pipeline này nếu ta thấy nó sẽ đạt hiệu quả tốt hơn. Việc lựa chọn phương án thay thế có thể là trong chính Rasa hoặc một package nào đó bên ngoài mà thấy phù hợp.

Bên cạnh supervised_embeddings thì Rasa cũng cung cấp sẵn thêm một vài mẫu pipeline khác là pretrained_embeddings_spacy, pretrained_embeddings_convert, bert v.v. Trong trường hợp này của luận văn, với ngôn ngữ tiếng việt thì PhoBERT là lựa chọn tốt nhất vì nó thực hiện huấn luyện lại từ đầu, dựa vào file nlu.md mà chúng ta cung cấp để xây dựng các ý định và tên định danh của người dùng.

3.5. Phân tích thiết kế hệ thống

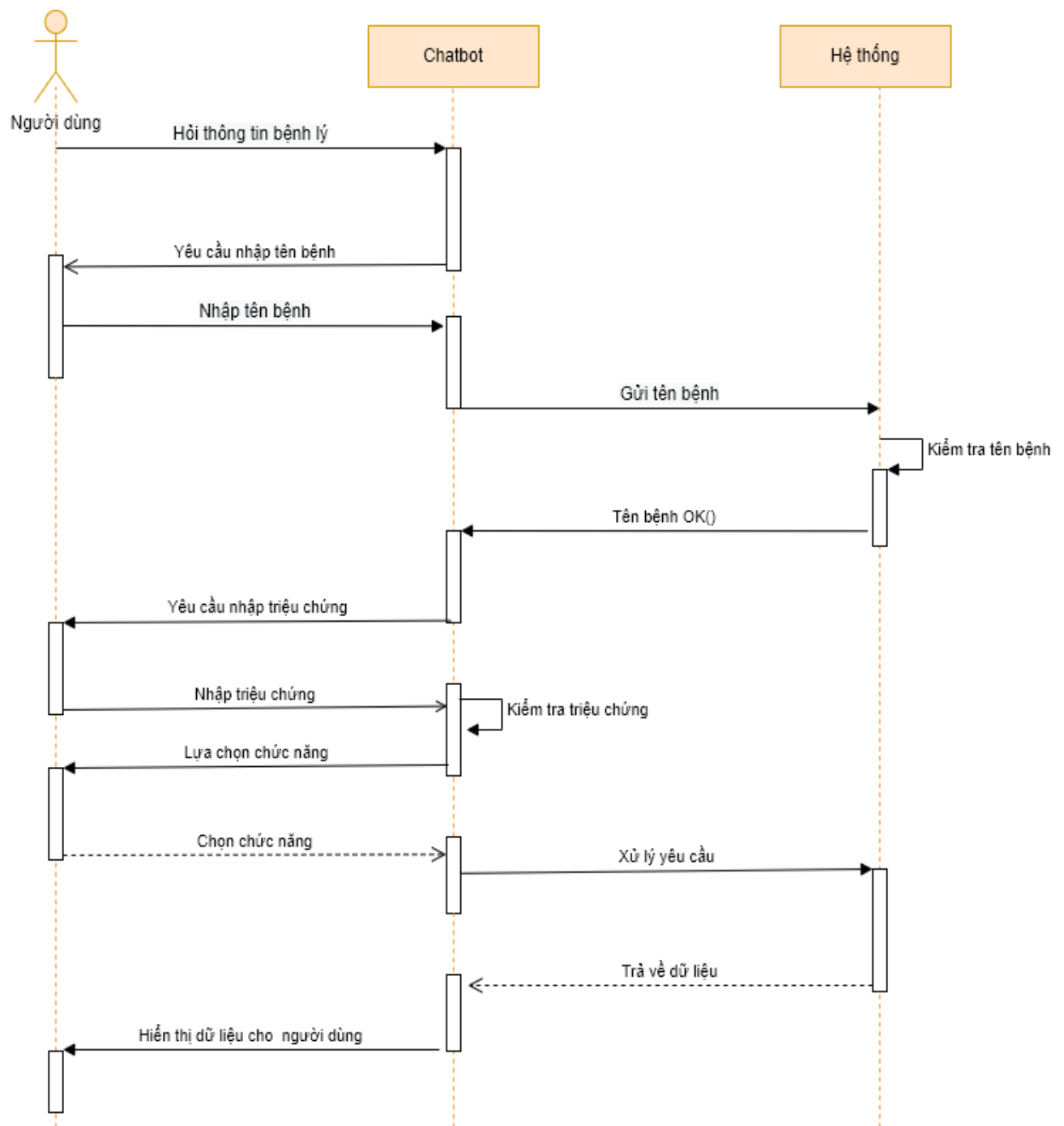


Hình 3.3 Biểu đồ cung cấp thông tin của chatbot

Hệ thống bao gồm ba phạm vi kiến thức cung cấp chính như sau:

- ✓ Chức năng giao tiếp cơ bản với người dùng như chào hỏi, giới thiệu, đưa ra nhận xét với chatbot
- ✓ Chức năng hỏi bệnh lý: giúp người dùng có thể hỏi các thông tin liên quan đến bệnh như : triệu chứng, nguyên nhân, chẩn đoán, cách điều trị và các biến chứng...
- ✓ Chức năng hỏi thông tin thuốc: cung cấp cho người dùng các thông tin bổ ích về thuốc như: giới thiệu, tác dụng, cách dùng, tác dụng phụ, chống chỉ định... của thuốc

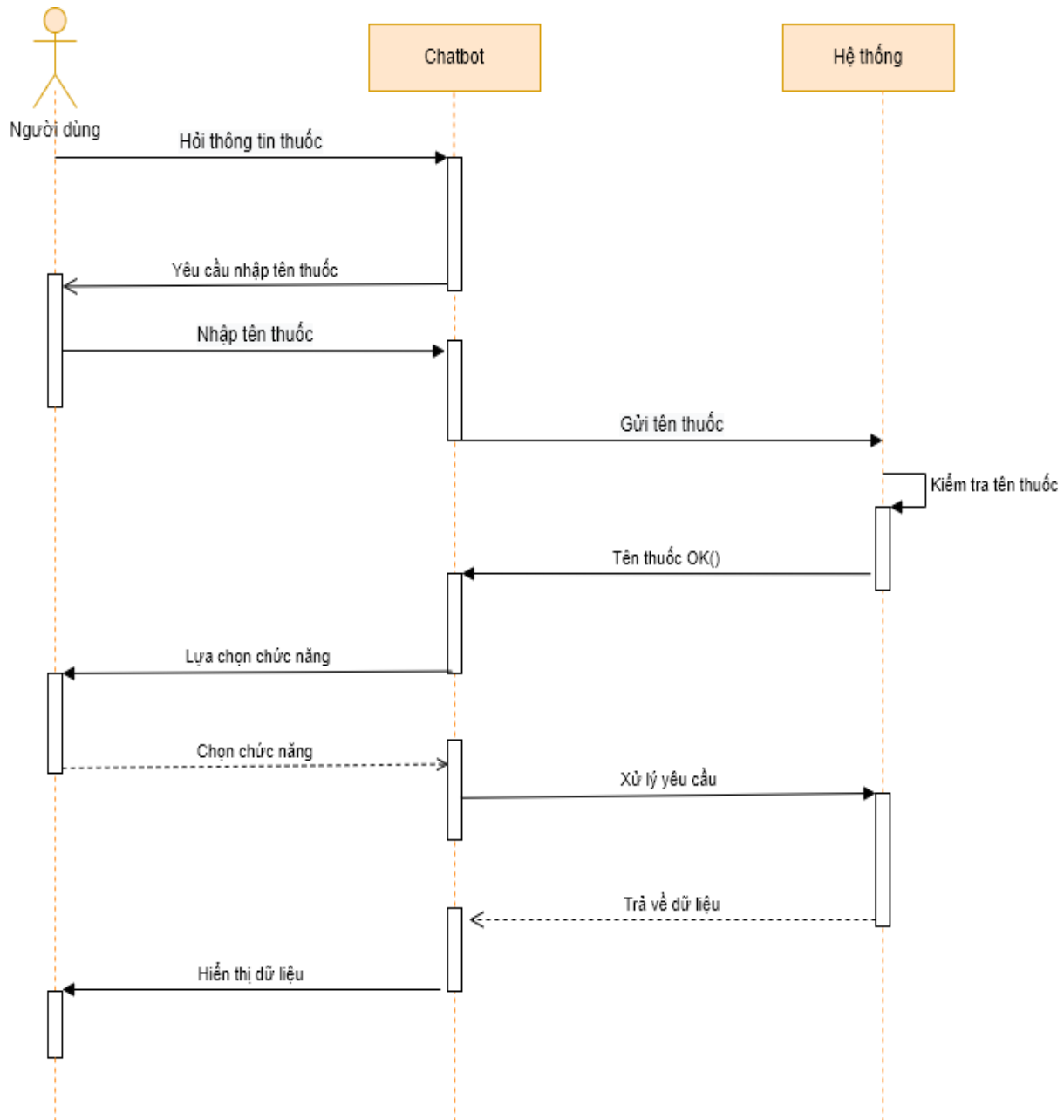
Sequence Diagram cho hành động hỏi thông tin bệnh:



Hình 3.4 Sequence Diagram cho hành động hỏi thông tin bệnh

Quá trình tương tác giữa người dùng và chatbot sẽ yêu cầu kiểm tra dữ liệu thông tin có chính xác hay không trước khi xử lý và trả lại thông tin cho người dùng.

Sequence Diagram cho hành động hỏi thông tin thuốc:



Hình 3.5 Sequence Diagram cho hành động hỏi thông tin thuốc

Trong chức năng hỏi thông tin về thuốc, tên thuốc sẽ được kiểm tra trong danh mục thuốc ngành da liễu, nếu nhập sai hệ thống sẽ yêu cầu nhập lại, sau đó sẽ trả kết quả cho người dùng.

3.6. Kết quả thực nghiệm

Để tiến hành thử nghiệm ứng dụng ta cần xây dựng môi trường thực nghiệm, phương pháp đánh giá mô hình và các bước tiến hành thực nghiệm khác nhau.

3.6.1. Môi trường thực nghiệm

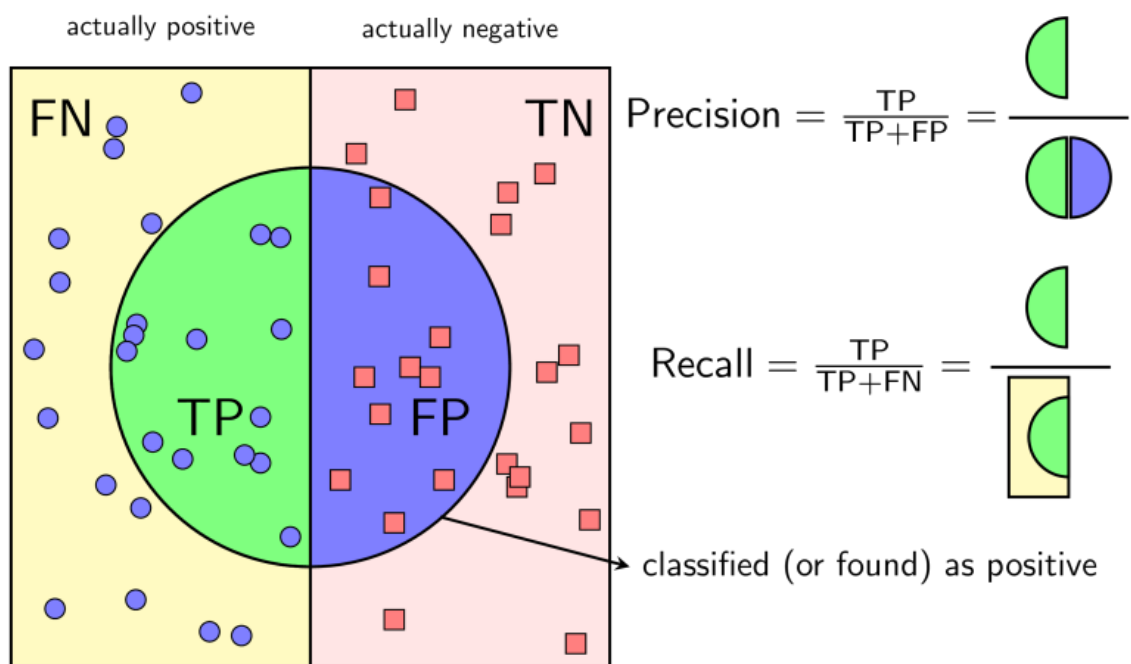
Môi trường tiến hành thử nghiệm như sau:

- Hệ điều hành: Window Server 2012, 64 bit, Ram 16Gb
- Database: SQL Server 2012
- Mô hình huấn luyện: vinai/phobert-base
- Python: 3.7.0 và PyTorch 1.1.0+ (hoặc TensorFlow 2.0+)
- Rasa: 3.5

3.6.2. Phương pháp đánh giá

Để đánh giá mô hình trong bài toán xác định ý định và tên riêng trong NLU dựa trên ba số liệu:

$$Precision = \frac{TP}{TP + FP} \quad Recall = \frac{TP}{TP + FN} \quad F1 = 2 \frac{Precision \cdot Recall}{Precision + Recall}$$



Hình 3.6 Mô tả cách các chỉ số đánh giá mô hình huấn luyện

Precision – là tỷ lệ bao nhiêu cái đúng được lấy ra, Cân nhắc trên tập dữ liệu kiểm soát xem có bao nhiêu dữ liệu được mô hình phán đoán đúng.

Recall – là tỷ lệ bao nhiêu cái được lấy ra là đúng, Chỉ số này còn được gọi là độ bao phủ, tức là xét xem mô hình tìm được có khả năng tổng quát hóa như nào. Từ hai yếu tố độ chuẩn xác và độ bao phủ người ta đặt ra một chỉ số khác gọi là F1-Score

F1-Score: Đây được gọi là một bình quân điều hòa(harmonic mean) của các tiêu chuẩn Precision và Recall. Nó có khuynh hướng lấy giá trị gần với giá trị nào bé hơn

giữa hai giá trị Precision và Recall và song song nó có giá trị lớn nếu cả hai giá trị Precision và Recall đều lớn. Chính vì như vậy F1-Score thể hiện được một cách khách quan hơn của một mô hình học máy.

3.6.3. Thử nghiệm

Thử nghiệm 1: So sánh các mô hình sử dụng trong bài toán xác định ý định và thực thể.

Trong thử nghiệm của luận văn, với mỗi bài toán xác định ý định ý định và tên riêng thì hệ thống sử dụng ba mô hình chính sau:

WhitespaceTokenizer: mô hình tách các token dựa vào các khoảng trắng hoặc bất kể các ký tự nào không được hiển thị.

ConveRT: mô hình mã hóa câu để xác định ý định người dùng

CustomPhoBERT: mô hình sử dụng PhoBert để cải tiến chất lượng nhận dạng ý định và tên riêng.

Kết quả thể hiện ở các bảng dưới đây.

Bảng 3.1 Bảng kết quả xác định ý định

Thuật toán	Precision	Recall	F1-Score
WhitespaceTokenizer	0.78	0.78	0.78
ConveRT	0.83	0.81	0.82
CustomPhoBERT	0.86	0.87	0.86

Bảng 3.2 Bảng kết quả nhận dạng thực thể

Thuật toán	Precision	Recall	F1-Score
WhitespaceTokenizer	0,84	0,76	0,79
ConveRT	0,89	0,78	0,82
CustomPhoBERT	0,91	0,83	0,86

Từ kết quả trên ta thấy mô hình CustomPhoBERT thực hiện tốt nhất trong cả việc xác định ý định lẫn thực thể của người dùng.

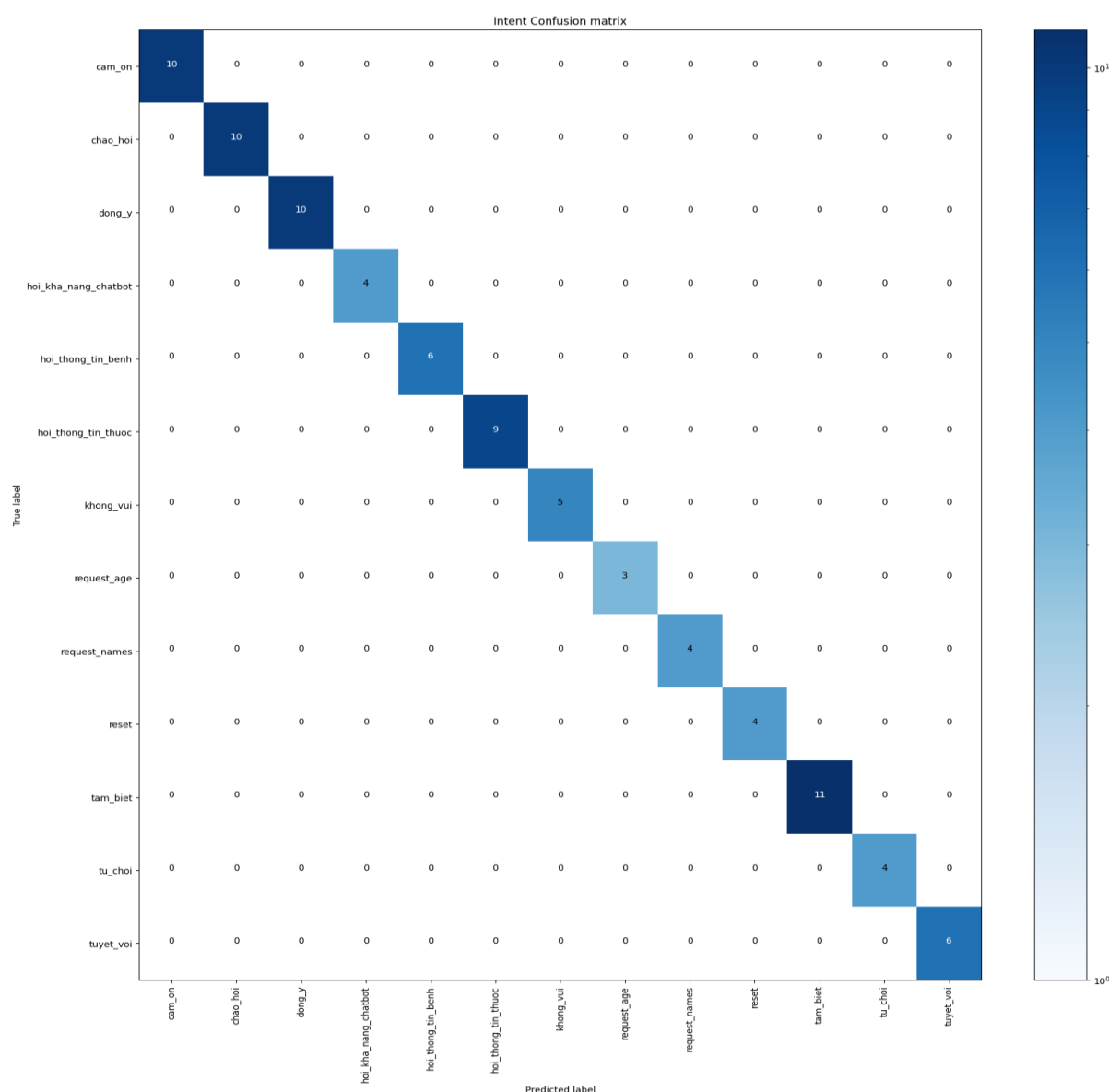
Thử nghiệm 2: Tiến hành thử nghiệm trên các câu ngẫu nhiên để đánh giá kết quả cuối cùng của người dùng

Sau khi lựa chọn được mô hình đào tạo CustomPhoBERT tối ưu nhất thì kết quả thực hiện thử nghiệm ngẫu nhiên 100 câu với chatbot ta được kết quả sau các lần thử như sau:

Bảng 3.3 Bảng mô tả số lần thử nghiệm với người dùng

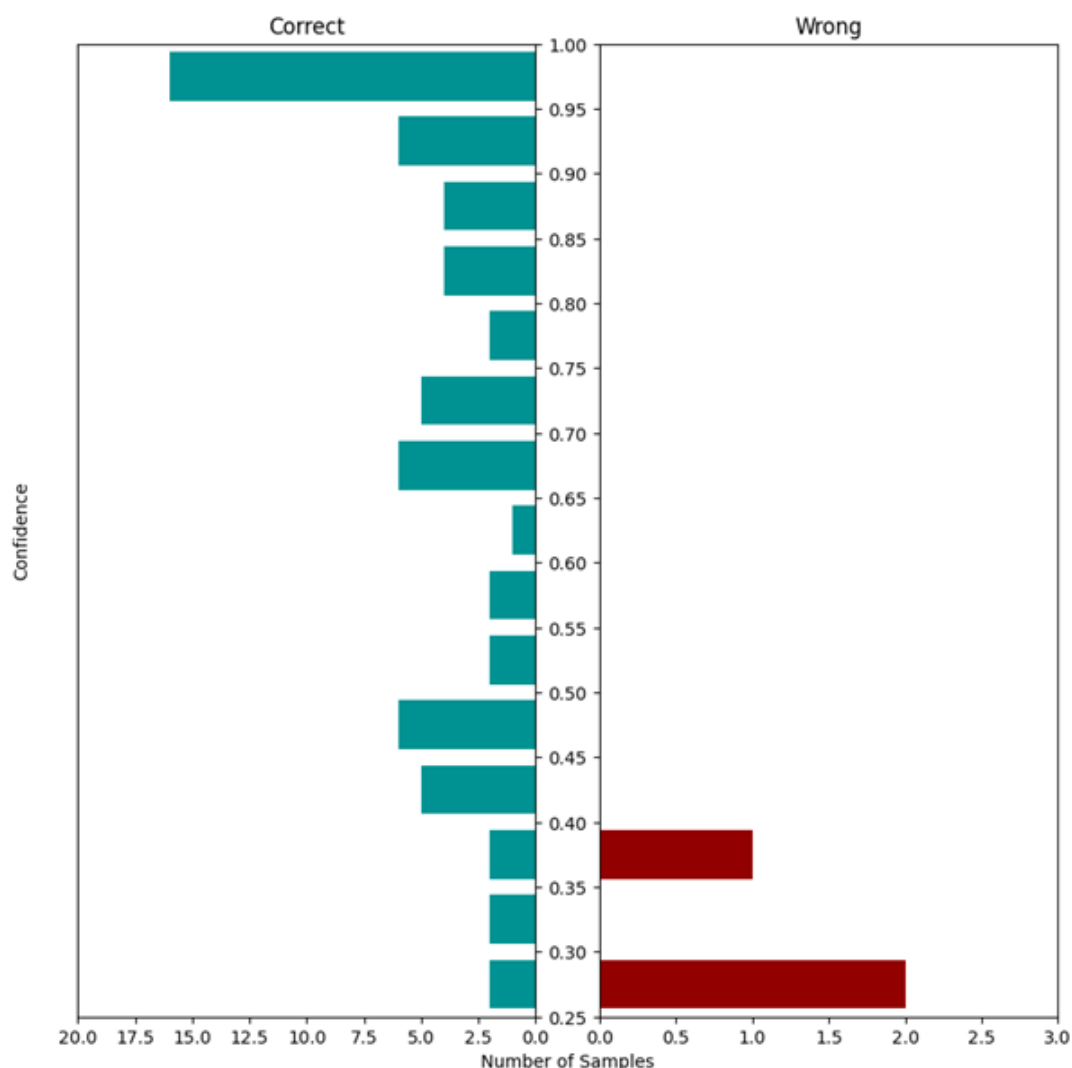
Lần thử nghiệm	Số câu đúng	Độ chính xác
1	54/100	54%
2	65/100	65%
3	75/100	75%
4	86/100	86%
5	89/100	89%

Sau khi thử nghiệm lần cuối cùng với 100 câu thì độ chính xác của chatbot đạt khoảng 89%. Đối với những câu hỏi mà bot chưa được đào tạo thì hệ thống sẽ đưa ra một số câu lựa chọn để điều hướng người dùng sang kịch bản khác mà bot có thể đáp ứng.



Hình 3-7 Ma trận ước lượng nhầm lẫn xây dựng dữ liệu intent

Ma trận nhầm lẫn trên cho thấy ý định nào bị nhầm lẫn với ý định khác. Bất kỳ mẫu nào đã được dự đoán không chính xác sẽ được ghi lại và lưu lại để gỡ lỗi dễ dàng hơn.



Hình 3-8 Biểu đồ độ tin cậy cho các dự đoán

Biểu đồ giúp ta hình dung độ tin cậy cho tất cả các dự đoán, với các dự đoán đúng và sai được hiển thị bằng các thanh màu xanh lam và màu đỏ tương ứng. Việc cải thiện chất lượng dữ liệu đào tạo của bạn sẽ di chuyển các thanh biểu đồ màu xanh lam lên trên biểu đồ và các thanh biểu đồ màu đỏ xuống dưới biểu đồ. Nó cũng sẽ giúp giảm số lượng thanh biểu đồ màu đỏ.

Dưới đây là đoạn hội thoại với chatbot sau lần thử nghiệm cuối cùng:

Bảng 3.4 Bảng hội thoại với chatbot sau lần thử nghiệm cuối

STT	Người dùng	Bot
1	Xin chào	<i>Chào mừng bạn đến với Leskin Chatbot</i>

2	Bạn có thể hỗ trợ gì?	<i>Tôi là Leskin Bot. Chuyên gia hỗ trợ các bệnh về da liễu. Bạn hãy cung cấp thông tin bệnh và triệu chứng để hệ thống đưa ra lời khuyên cho bạn!</i>
3	Tôi muốn tư vấn bệnh?	<i>Trước tiên, Bạn vui lòng cung cấp họ tên?</i>
4	Bùi Ngọc Quang	<i>Bạn năm nay bao nhiêu tuổi?</i>
5	Tôi 40 tuổi	<i>Hệ thống ghi nhận họ tên bạn là Bùi Ngọc Quang / 40 tuổi. Bạn vui lòng cho biết bệnh đang gặp là gì?</i>
6	Bệnh ze na	<i>Tên bệnh này không có danh mục bệnh liên quan da liễu. Vui lòng nhập lại?</i>
7	Bệnh zona	<i>Bạn vui lòng cung cấp thêm triệu chứng đang gặp?</i>
8	ban đỏ, mụn nước, bọng nước tập trung thành đám	<i>Hệ thống ghi nhận bệnh của bạn là "bệnh zona" với triệu chứng " ban đỏ, mụn nước, bọng nước tập trung thành đám "</i> <i>Bạn muốn biết thêm thông tin khác về bệnh như sau?</i> <ul style="list-style-type: none"> ▪ Tổng quan bệnh ▪ Cách điều trị ▪ Các biến chứng ▪ Cách phòng bệnh
9	Tổng quan bệnh	<i>Bệnh zona hay herpes zoster là bệnh nhiễm trùng da với biểu hiện là các ban đỏ, mụn nước, bọng nước tập trung thành đám.</i>
10	Cách điều trị	<i>Uống acyclovir: thuốc kháng virus có tác dụng nhanh lành vết thương, giảm số tổn thương mới và giảm đau sau zona. Thuốc nên được dùng sớm, tốt nhất trong vòng 72 giờ đầu.</i>
11	Tôi cần tư vấn thuốc?	<i>Bạn vui lòng cung cấp tên thuốc là gì?</i>
12	naomycin	<i>Tên thuốc này không tồn tại trong danh mục thuốc. Vui lòng nhập lại?</i>

13	neomycin	<i>Neomycin thường phối hợp với một số kháng sinh khác như polymixin B, bacitracin, colistin, gramicidin, hoặc các corticoid như dexametason trong các thuốc dùng ngoài.</i> <i>Bạn muốn biết thêm thông tin khác về thuốc như sau?</i> <ul style="list-style-type: none">▪ Tác dụng▪ Cách dùng▪ Tác dụng phụ▪ Chống chỉ định
14	Tác dụng	<i>Neomycin có tác dụng với phần lớn các vi khuẩn Gram âm và Gram dương gây nên các nhiễm khuẩn ngoài da.</i>
15	Cách dùng	<i>Liều uống: 1 g/lần mỗi 1 giờ, liên tục trong 4 giờ; sau đó 1g/lần mỗi 4 giờ, trong 2 hoặc 3 ngày trước khi phẫu thuật...</i>
16	Tác dụng phụ	<i>Dùng uống với liều cao: Buồn nôn, nôn, tiêu chảy, độc thận, độc với thính giác ngay cả khi dùng liều điều trị. Uống kéo dài có thể gây hội chứng kém hấp thu và bội nhiễm...</i>
17	Chống chỉ định	<i>Neomycin chống chỉ định với các trường hợp quá mẫn với neomycin hoặc với các aminoglycoside...</i>
18	Cám ơn	<i>Không có gì bạn nhé. Cảm ơn bạn đã liên hệ</i>
19	Tạm biệt	<i>OK, tạm biệt bạn</i>

3.8. Đánh giá

Từ kết quả thực nghiệm rút ra một số đánh giá như sau:

- ✓ Xác định được đúng ý định có ý nghĩa quan trọng nhất đối với chatbot. Đối với bài toán miền đóng cần xác định rõ ràng các ý định, xây dựng tập dữ liệu đủ lớn, gán nhãn và tiến hành huấn luyện.
- ✓ Xây dựng dữ liệu, huấn luyện cho bot với các kịch bản là rất cần thiết để cho độ chính xác cao của chatbot.
- ✓ Chatbot ứng dụng AI có khả năng đáp ứng tốt với các kịch bản dựng sẵn, và được huấn luyện. Đối với các kịch bản nằm ngoài kịch bản dựng sẵn, có thể điều hướng người dùng về các câu mặc định hoặc các giao diện menu để người dùng lựa chọn.
- ✓ Việc xác định và phản hồi đa ý định có thể thực hiện bằng việc kết hợp các ý định với nhau.

- ✓ Qua bài toán thử nghiệm trên có thể thấy rằng áp dụng chatbot cho việc hỗ trợ người dùng trong ngành y tế da liễu là khả thi, có tính thực tiễn cao, có thể áp dụng được ngay trong thực tiễn.

Tuy nhiên, trong quá trình thực hiện cũng gặp phải những thách thức bổ sung bao gồm:

- Vấn đề thứ nhất là vấn đề đồng tham chiếu (coreference). Trong nói và viết, thông thường chúng ta thường dùng những cách gọi rút gọn cho những đối tượng mà chúng ta đã đề cập trước đó. Ví dụ trong tiếng Việt, người nói và viết có thể dùng các đại từ xưng hô khác nhau hoặc các từ lóng theo từng vùng miền,... Nếu không có thông tin ngữ cảnh và bộ phân tích đồng tham chiếu, sẽ là rất khó để chatbot biết các từ này tham chiếu đến đối tượng nào. Việc không xác định được đúng đối tượng mà từ thay thế tham chiếu đến có thể khiến chatbot hiểu không chính xác câu hỏi thoại của người dùng. Thách thức này là khá rõ ràng trong các đoạn hội thoại dài.

- Vấn đề thứ hai là làm sao để giảm bớt công sức làm dữ liệu khi phát triển chatbot. Theo những cách tiếp cận ở trên, khi phát triển một ứng dụng chatbot, ta cần gán nhãn dữ liệu huấn luyện cho bộ phân lớp ý định và nhận dạng thực thể tên gọi. Trong những miền ứng dụng phức tạp như ngành y tế, công sức để tạo ra những bộ dữ liệu đó là khá đắt đỏ. Vì thế, việc phát triển các phương pháp để tận dụng những nguồn dữ liệu sẵn có trong doanh nghiệp để giảm lượng dữ liệu cần gán nhãn, trong khi vẫn đảm bảo được độ chính xác của các mô hình xử lý ngôn ngữ tự nhiên là cần thiết.

- Sự đa dạng trong cách mọi người nhập thông điệp của người dùng có thể khiến ta khó hiểu được ý định của họ. Chatbot phải có khả năng xử lý cả câu dài và ngắn, cũng như bong bóng trò chuyện có nội dung dài so với nhiều lần gửi rất ngắn.

- Con người là ngẫu nhiên và hành vi của người dùng được điều khiển bởi cảm xúc và tâm trạng; người dùng có thể nhanh chóng thay đổi ý định của họ. Sau khi yêu cầu một gợi ý ban đầu, họ có thể chuyển sang muốn đưa ra một lệnh. Công nghệ chatbot phải có khả năng thích ứng và hiểu được yếu tố ngẫu nhiên và tự phát này.

KẾT LUẬN

Sau khi tìm hiểu và thực hiện luận văn “Xây dựng chatbot tư vấn người dùng trong ngành y tế da liễu” tôi đã hoàn thành đề tài và đạt được các kết quả như sau:

- ✓ Tìm hiểu tổng quan về bài toán và quy trình xây dựng chatbot để giải quyết theo hướng tiếp cận miền đóng, cụ thể là trong nghiệp vụ y tế da liễu.
- ✓ Nắm được các kiến thức bao gồm kiến trúc và nhiệm vụ các thành phần, thuật toán và các kỹ thuật sử dụng trong chatbot như:
 - Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks).
 - Mô hình Word embeddings.
 - Mô hình BERT và PhoBERT áp dụng cho Tiếng Việt
- ✓ Xây dựng bộ dữ liệu thử nghiệm chuyên ngành da liễu trong lĩnh vực y tế cho bài toán chatbot.
- ✓ Nghiên cứu và ứng dụng Framework Rasa, tối ưu hóa các thuật toán khi áp dụng framework này. Tạo mô hình huấn luyện riêng để xác định ý định người dùng, cũng như chỉnh sửa và thêm các hành động khi người dùng tương tác với bot. Đồng thời sử dụng PhoBERT để cải tiến chất lượng của bài toán xác định ý định và nhận dạng thực thể tên.
- ✓ Xây dựng ứng dụng chatbot tích hợp trên website để tương tác với người dùng một cách trực quan và dễ dàng sử dụng.

Định hướng nghiên cứu tiếp theo:

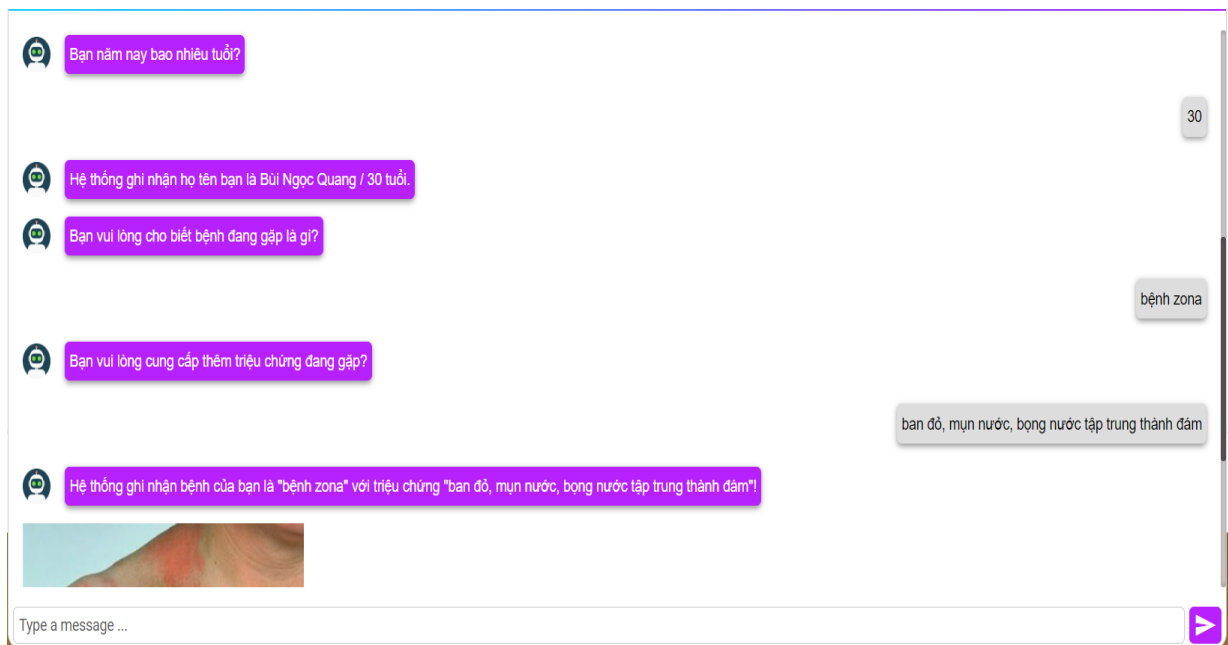
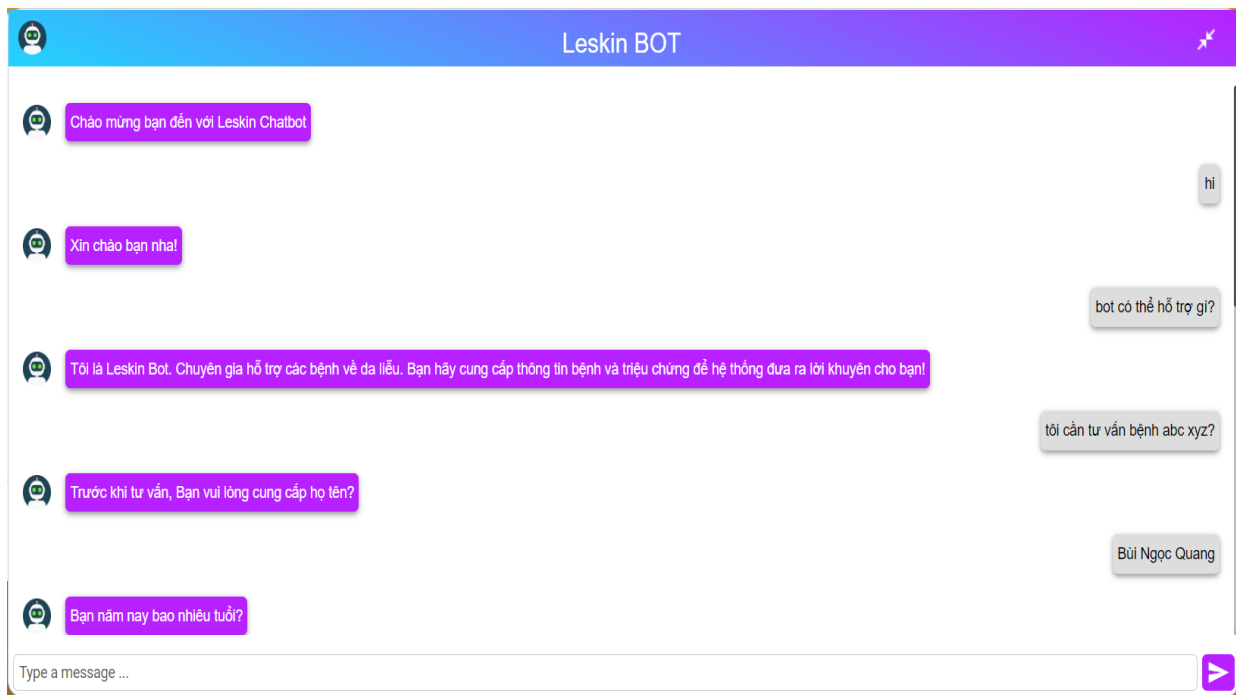
- ✓ Chuẩn đoán bệnh thông qua hình ảnh kết hợp với các triệu chứng sẽ cho ra thông tin bệnh chính xác hơn.
- ✓ Hỗ trợ người dùng qua giọng nói, tích hợp giọng nói sang văn bản và văn bản sang giọng nói cho bot, người dùng sử dụng sẽ tiện hơn.
- ✓ Xây dựng nhiều kịch bản và tính năng hơn cho chatbot như đặt lịch khám chữa bệnh, lựa chọn bác sỹ khám tại nhà,...
- ✓ Huấn luyện chatbot trả lời linh hoạt và thông minh hơn thông qua việc mở rộng xây dựng và cải tiến cơ sở dữ liệu của hệ thống trở nên tối ưu hơn, đa dạng hơn.

TÀI LIỆU THAM KHẢO

1. R. J. Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*”, 8(3-4):229–256, 2012.
2. X. Zhou, D. Dong, H. Wu, S. Zhao, D. Yu, H. Tian, X. Liu, and R. Yan. Multi-view response selection for human-computer conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 372–381, 2016.
3. L. Bahl, P. Brown, P. De Souza, and R. Mercer. “Maximum mutual information estimation of hidden markov model parameters for speech recognition”. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’86.*, volume 11, pages 49–52. IEEE, 2006.
4. A. Graves. Long short-term memory. *Neural Computation*, 9(8):1735, 1997.
5. Naoaki Okazaki, “CRFsuite: A fast implementation of Conditional Random Fields (CRFs)”, 2011.
6. T.-H. Wen, M. Gasic, P.-H. Su, D. Vandyke, and S. Young. In “*Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*”, pages 1711–1721, Lisbon, Portugal, September 2015.
7. Zhiheng Huang, Wei Xu, Kai Yu, “*Bidirectional LSTM-CRF Models for Sequence Tagging*”, On arxiv, 2015.
8. Jurafsky, D., & Martin, “*Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*”, Chapter 24. “*Dialogue and Conversational Agents*”.
9. Ming-Wei Chang “*BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*” Version 2, 2019
10. A. Jiao, “An intelligent chatbot system based on entity extraction using RASA NLU and neural network,” *J. Phys. Conf. Ser.*, vol. 1487, p. 012014, 2020.
11. Rasa Community, “Memoization Policy with max history”, 2021.
12. PhoBERT, “Pre-trained language models for Vietnamese”, 2020
13. Hongshen Chen, Xiaorui Liu, Dawei Yin, Jiliang Tang “*A Survey on Dialogue Systems: Recent Advances and New Frontiers*” , On arxiv, Version 3, 2018

14. PGS.TS Nguyễn Thị Xuyên, “Hướng dẫn chẩn đoán và điều trị các bệnh da liễu”, Số 75/QĐ-BYT, 2015
15. Sepp Hochreiter, Jürgen Schmidhuber “Long Short-term Memory”, 1997
16. Denny Britz, “Recurrent Neural Networks Tutorial” , 2015

PHỤ LỤC





Bạn muốn biết thêm thông tin khác về bệnh như sau?

Tổng quan bệnh

Cách điều trị

Các biến chứng

Cách phòng bệnh



Bệnh zona hay herpes zoster là bệnh nhiễm trùng da với biểu hiện là các ban đỏ, mụn nước, bóng nước tập trung thành đám, thành chùm dọc theo đường phân bố của thần kinh ngoại biên. Bệnh do sự tái hoạt của virút Varicella zoster (VZV) tiềm ẩn ở rễ thần kinh cảm giác cạnh cột sống. Bệnh hay gặp ở những người già, những người suy giảm miễn dịch, đặc biệt ở người nhiễm HIV/AIDS.

a) Mục tiêu điều trị: làm liền tổn thương; giảm đau; ngăn ngừa biến chứng.

Type a message ...



Uống acyclovir: thuốc kháng virus có tác dụng nhanh lành vết thương, giảm số tổn thương mới và giảm đau sau zona. Thuốc nên được dùng sớm, tốt nhất trong vòng 72 giờ đầu. Liều 800mg x 5 lần/ngày trong 7-10 ngày

Hoặc, famciclovir 500mg mỗi 8 giờ (3 lần mỗi ngày) x 7 ngày

Hoặc, valacyclovir 1000mg mỗi 8 giờ (3 lần mỗi ngày) x 7 ngày

Ngoài ra: kháng sinh chống bội nhiễm; giảm đau, kháng viêm, an thần, sinh tố nhóm B liều cao. Nếu đau dai dẳng: bôi kem chứa lidocain và prilocalin, kem capsaicin, lidocain gel, uống thuốc chống trầm cảm ba vòng, phong bế thần kinh và vật lý trị liệu kết hợp. Corticoid: có tác dụng giảm đau trong thời kỳ cấp tính và nhiều tác giả cho rằng thuốc có tác dụng giảm đau sau zona

tôi cần tư vấn thuốc?



Bạn vui lòng cung cấp tên thuốc là gì?

Neomycin



Neomycin thường phối hợp với một số kháng sinh khác như polymixin B, bacitracin, colistin, gramicidin, hoặc các corticoid như dexametason trong các thuốc dùng ngoài.



Hệ thống ghi nhận thông tin thuốc cần hỏi là "Neomycin"!

Type a message ...



Neomycin thường phối hợp với một số kháng sinh khác như polymixin B, bacitracin, colistin, gramicidin, hoặc các corticoid như dexametason trong các thuốc dùng ngoài.

Hệ thống ghi nhận thông tin thuốc cần hỏi là "Neomycin"!

Bạn muốn biết thêm thông tin khác về thuốc như sau?

Tác dụng

Cách dùng

Tác dụng phụ

Chống chỉ định

Neomycin có tác dụng với phần lớn các vi khuẩn Gram âm và Gram dương gây nên các nhiễm khuẩn ngoài da.

Những vi khuẩn nhạy cảm với neomycin như: *Staphylococcus aureus*, *Escherichia coli*, *Haemophilus influenzae*, *Klebsiella*, *Enterobacter* các loại, *Neisseria* các loại.

Neomycin không có tác dụng với *Pseudomonas aeruginosa*, *Serratia marcescens*, *Streptococci* bao gồm cả *Streptococcus pneumoniae* hoặc *Streptococcus tan máu*.

Neomycin không được dùng đường tiêm hoặc toàn thân vì độc tính của thuốc.

Type a message ...



Thuốc uống

Dùng sát khuẩn ruột trước phẫu thuật bụng:

- Trẻ em > 1 tuổi: 90 mg/kg/ngày, chia làm 6 lần, cách 4 giờ uống một lần, trong 2 ngày, trước khi phẫu thuật để hỗ trợ làm sạch ruột và kết hợp với erythromycin base.
- Trẻ em từ 6 đến 12 tuổi: 250 – 500 mg/lần mỗi 4 giờ trong 2 đến 3 ngày trước khi phẫu thuật.
- Trẻ em trên 12 tuổi: uống 1g/lần mỗi 4 giờ trong 2 hoặc 3 ngày trước khi phẫu thuật.

Trong hôn mê gan:

- Liều uống: 50 – 100 mg/kg/ngày, chia làm 3 – 4 lần uống liên tục trong 5 – 6 ngày, dùng không quá 12 g/ngày. Uống kéo dài sẽ gây hội chứng kém hấp thu.

cảm ơn bot đã hỗ trợ

Không có gì bạn nhé. Cảm ơn bạn đã liên hệ

tạm biệt

OK, tạm biệt bạn

Type a message ...

