

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**VIỆN HÀN LÂM KHOA HỌC
VÀ CÔNG NGHỆ VIỆT NAM**

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ VIỆT NAM



Đỗ Việt Mạnh

XÂY DỰNG CHATBOT BÁN HÀNG DỰA TRÊN MÔ HÌNH SINH

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

Hà Nội – 2020

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**VIỆN HÀN LÂM KHOA HỌC
VÀ CÔNG NGHỆ VIỆT NAM**

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ VIỆT NAM



Đỗ Việt Mạnh

XÂY DỰNG CHATBOT BÁN HÀNG DỰA TRÊN MÔ HÌNH SINH

Chuyên ngành: Hệ thống thông tin

Mã số: 8480104

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

CÁN BỘ HƯỚNG DẪN KHOA HỌC

TS. Nguyễn Việt Anh

Hà Nội – 2020

LỜI CAM ĐOAN

Tôi là Đỗ Viết Mạnh, học viên khóa 2018A, ngành Công nghệ thông tin, chuyên ngành Hệ Thống Thông Tin. Tôi xin cam đoan luận văn “*Xây dựng Chatbot bán hàng dựa trên mô hình sinh*” là do tôi nghiên cứu, tìm hiểu và phát triển dưới sự hướng dẫn của TS. Nguyễn Việt Anh, không phải sự sao chép từ các tài liệu, công trình nghiên cứu của người khác mà không ghi rõ trong tài liệu tham khảo. Tôi xin chịu trách nhiệm về lời cam đoan này.

Hà Nội, ngày tháng năm 2020

Tác giả

Đỗ Viết Mạnh

LỜI CẢM ƠN

Lời cảm ơn trân trọng đầu tiên em muốn dành tới các thầy cô Học viện khoa học và công nghệ Việt Nam, Viện công nghệ thông tin, Viện Hàn lâm khoa học và công nghệ Việt Nam nói chung và các thầy cô trong bộ môn Hệ thống thông tin cũng như khoa Công nghệ thông tin nói riêng đã tận tình giảng dạy và truyền đạt những kiến thức quý báu trong suốt khoá cao học vừa qua, giúp em có những kiến thức chuyên môn nền tảng để làm cơ sở lý luận khoa học cho luận văn này.

Đặc biệt em xin chân thành cảm ơn thầy TS. Nguyễn Việt Anh đã dùi dắt và hướng dẫn em trong suốt quá trình làm luận văn, sự chỉ bảo và định hướng của thầy giúp em tự tin nghiên cứu những vấn đề mới và giải quyết bài toán một cách khoa học.

Em xin trân trọng cảm ơn Ban giám hiệu Học viện khoa học công nghệ Việt Nam - Viện Hàn lâm khoa học và công nghệ Việt Nam đã tạo các điều kiện cho em được học tập và làm luận văn một cách thuận lợi.

Mặc dù đã cố gắng rất nhiều, nhưng chắc chắn trong quá trình học tập cũng như luận văn không khỏi những thiết sót. Em rất mong được sự thông cảm và chỉ bảo tận tình của các thầy cô và các bạn.

Hà Nội, ngày tháng năm 2020

Tác giả

Đỗ Viết Mạnh

MỤC LỤC

DANH MỤC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT	8
DANH MỤC HÌNH VẼ VÀ ĐỒ THỊ	10
MỞ ĐẦU	1
1. Động lực nghiên cứu	2
2. Mục tiêu luận văn	3
3. Cấu trúc luận văn	4
CHƯƠNG 1 : TỔNG QUAN CÁC HỆ THỐNG CHATBOT	5
1.1 Giới thiệu	5
1.2 Các mô hình chatbot bán hàng tiêu biểu hỗ trợ Tiếng Việt hiện nay	6
1.2.1 Chatbot theo kịch bản (menu/button)	6
1.2.2 Chatbot nhận dạng từ khoá	7
1.2.3 Mô hình Chatbot bán hàng mà luận văn nghiên cứu	8
1.3 Cấu trúc các thành phần hệ thống Chatbot	9
1.4 Hiểu ngôn ngữ tự nhiên (NLU)	10
1.4.1 Xác định ý định người dùng	13
1.5 Quản lý hội thoại (DM)	15
1.5.1 Mô hình máy trạng thái hữu hạn FSA	16
1.5.2 Mô hình Frame-based	17
1.6 Mô hình sinh ngôn ngữ (NLG)	18
1.6.1 Template-based NLG	18
1.6.2 Plan-based NLG	19
1.6.3 Class-based NLG	19
1.7 Kết luận chương	20
CHƯƠNG 2: CÁC KỸ THUẬT SỬ DỤNG TRONG CHATBOT	21
2.1. Kiến trúc mạng nơ-ron nhân tạo	21
2.2. Mạng nơ-ron hồi quy RNN	23

2.2.1 Vấn đề phụ thuộc quá dài	26
2.2.2 Kiến trúc mạng LSTM.....	27
2.2.3 Phân tích mô hình LSTM.....	29
2.3. Word embeddings.....	32
2.3.1 Word2vec	32
2.3.2 Glove.....	34
2.4. Ứng dụng RNN vào quản lý hội thoại.....	35
2.4.1 Mô hình word-based DST.....	35
2.4.2 Mô hình Global-Locally Self-Attentive DST (GLAD)	37
2.5 Mô hình CRF	38
2.5.1 Định nghĩa CRF	38
2.5.2 Huấn luyện CRF	40
2.5.3 Suy diễn CRF.....	42
2.6 Giải thuật phân loại văn bản Starspace.....	43
2.7 Kết luận chương.....	44
CHƯƠNG 3: XÂY DỰNG CHATBOT BÁN HÀNG	45
3.1. Bài toán.....	45
3.2. Xây dựng Chatbot hỗ trợ nghiệp vụ bán hàng.....	45
3.3. Ứng dụng RASA xây dựng Chatbot.....	47
3.4. Xây dựng dữ liệu Chatbot.....	49
3.4.1 Xây dựng ý định.....	50
3.4.2 Xây dựng entity.....	51
3.4.3 Xây dựng câu trả lời cho bot.....	52
3.4.4 Xây dựng khung kịch bản (history)	53
3.5. Thử nghiệm.....	54
3.5.1 Dữ liệu thử nghiệm	54
3.5.2 Môi trường và công cụ sử dụng thực nghiệm.....	55
3.5.3 Thiết kế chương trình thử nghiệm	55
3.5.4 Thử nghiệm.....	56

3.6. Đánh giá.....	62
CHƯƠNG 4: KẾT LUẬN.....	63
TÀI LIỆU THAM KHẢO	1
PHỤ LỤC	3

DANH MỤC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

Từ viết tắt	Từ chuẩn	Điễn giải
AI	Artificial Intelligence	Trí tuệ nhân tạo
ANN	Artificial Nerual Network	Mạng nơ-ron nhân tạo
CBOW	Continuous Bag of Words	
CNN	Convolution Neural Network	Mạng nơ-ron tích chập
CRF	Conditional Random Fields	Mô hình xác xuất trường điều kiện ngẫu nhiên
DM	Dialogue Management	Quản lý hội thoại
DNN	Deep Neural Networks	Mô hình học sâu
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	
HMM	Hiden Markov Models	Mô hình Markov ẩn
LSTM	Long short-term memory	Mạng cải tiến để giải quyết vấn đề phụ thuộc quá dài
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
POS	Part Of Speech	Gán nhãn từ loại
RNN	Recurrent Neural Network	Mạng nơ-ron hồi quy
SVM	Vector Support Machine	Máy vector hỗ trợ

DANH MỤC HÌNH VẼ VÀ ĐỒ THỊ

Hình 1: Ví dụ về dạng Chatbot (menu/button).....	6
Hình 2: Ví dụ về Chatbot nhận dạng từ khoá.....	7
Hình 3: Cấu trúc các thành phần cơ bản hệ thống Chatbot [12]	9
Hình 4: Mô hình các thành phần xử lý trong Chatbot [1]	10
Hình 5: Các bước xử lý chính trong pipeline của NLU [1]	11
Hình 6: Các bước xử lý trong NLU [2].....	11
Hình 7: Mô hình các bước xác định ý định.....	13
Hình 8: Mô hình quản lý trạng thái và quyết định action trong hội thoại [2]	15
Hình 9: Quản lý hội thoại theo mô hình máy trạng thái hữu hạn FSA	16
Hình 10: Frame cho Chatbot hỏi thông tin khách hàng	17
Hình 11: Phương pháp sinh ngôn ngữ dựa trên tập mẫu câu trả lời [1].....	18
Hình 12: Phương pháp sinh ngôn ngữ Plan-based [1]	19
Hình 13: Phương pháp sinh ngôn ngữ class-based [1].....	19
Hình 14: Kiến trúc mạng nơ-ron nhân tạo [15].....	21
Hình 15: Quá trình xử lý thông tin của một mạng nơ-ron nhân tạo [15].....	22
Hình 16: Mạng RNN [15]	24
Hình 17: Mạng RNN 2 chiều [15].....	25
Hình 18: Mạng RNN nhiều tầng [15].....	26
Hình 19: RNN phụ thuộc short-term [17]	27
Hình 20: RNN phụ thuộc long-term [17]	27
Hình 21: Các mô-đun lặp của mạng RNN chứa một layer [17]	28
Hình 22: Các mô-đun lặp của mạng LSTM chứa bốn layer [17]	28
Hình 23: Tế bào trạng thái LSTM giống như một băng truyền [17]	29
Hình 24: Cỗng trạng thái LSTM [17].....	30
Hình 25: LSTM focus f [17].....	30
Hình 26: LSTM focus I [17].....	31
Hình 27: LSTM focus c [17]	31
Hình 28: LSTM focus o [17]	32
Hình 29: Mô hình từ nhúng [16]	33
Hình 30: Mô hình CBOW và Skip-Ngram [16]	33
Hình 31: Xác xuất từ k trên ngữ cảnh của từ i và j [16].....	34

Hình 32: Công thức tính xác xuất từ k trên ngữ cảnh của từ i [16]	34
Hình 33: Công thức tính hàm chi phí tối thiểu [16]	35
Hình 34: Hàm trọng số (weighting function) [16]	35
Hình 35: Mô hình word-based DST với mạng RNN [20]	36
Hình 36: Mô hình Global-Locally Self-Attentive DST (GLAD) [21].....	37
Hình 37: Global-locally self-attentive encoder modul [21]	38
Hình 38: Đồ thị vô hướng mô tả CRF	39
Hình 39: Cấu trúc hệ thống Chatbot.....	46
Hình 40: Cấu hình pipeline xử lý ngôn ngữ tự nhiên.....	48
Hình 41: Các bước xây dựng Chatbot.....	50
Hình 42: Xây dựng ý định người dùng	51
Hình 43: Danh sách các entities	52
Hình 44: Mẫu câu trả lời của Chatbot cho ý định hỏi giá sản phẩm.....	52
Hình 45: Mẫu câu trả lời mặc định của bot khi không nhận ra ý định người dùng	53
Hình 46: Custom action xử lý slot maSP	53
Hình 47: Huấn luyện cho Chatbot.....	54
Hình 48: Kiến trúc của chương trình thử nghiệm	55
Hình 49: Hình ma trận ước lượng nhầm lẫn xây dựng dữ liệu intent	57
Hình 50: Ước lượng độ chính xác tập dữ liệu trainning intent	58
Hình 51: Ước lượng độ chính xác trích chọn thông tin	58
Hình 52: Bảng mô tả đoạn hội thoại test với Chatbot.....	61

MỞ ĐẦU

Mạng xã hội đang ngày càng phát triển, đặc biệt thương mại điện tử đang trở thành xu thế, không chỉ các doanh nghiệp mà tất cả cá nhân đều có thể bán hàng trực tiếp thông qua internet. Dưới góc độ người mua hàng, họ rất cần nắm rõ các thông tin của sản phẩm, chính vì vậy người bán hàng cần đưa ra những cuộc trao đổi để cung cấp thêm nhiều thông tin về sản phẩm, nhằm thuyết phục người mua đưa ra quyết định mua hàng. Để giải quyết bài toán trên, người bán hàng cần xây dựng một hệ thống Chatbot bán hàng tự động giúp giảm thiểu được chi phí về nhân sự, tăng hiệu quả bán hàng, chăm sóc khách hàng và tăng khả năng tương tác. Vậy Chatbot bán hàng tự động là gì ? Tại sao lại cần mô hình như vậy ? Những lợi ích và thuận tiện khi sử dụng mô hình này là gì ?

Để giải đáp cho những câu hỏi ở trên, đặt dưới góc độ của người bán hàng ta thấy rằng khi gặp phải các trường hợp như yêu cầu tư vấn về sản phẩm vào lúc giờ nghỉ buổi trưa, buổi tối hay khi có quá nhiều khách hàng muốn tư vấn về sản phẩm vào cùng một thời điểm hoặc khách hàng thường xuyên đưa ra các câu hỏi mang tính chất trùng lặp ...vv. Ở các trường hợp trên nếu như không có Chatbot tự động phản hồi các yêu cầu nhanh nhất thì chắc chắn rằng hiệu quả bán hàng sẽ giảm đáng kể, cũng như uy tín và sự chuyên nghiệp của người bán hàng sẽ được khách hàng đánh giá thấp. Hiện nay rất nhiều người bán hàng sử dụng các công cụ quảng cáo từ Facebook, google...vv, chi phí cho việc quảng cáo cũng khá cao, nếu sử dụng Chatbot bán hàng tự động cũng sẽ có thể tạo được rất nhiều chiến dịch quảng cáo, giảm được chi phí đi rất nhiều mà lại mang lại tính hiệu quả cao. Những vấn đề nêu trên, chứng minh không phải lúc nào chúng ta cũng đủ thời gian và nguồn nhân lực để sẵn sàng kết nối với khách hàng. Do đó, mô hình trả lời bán hàng tự động là rất thiết thực trong bối cảnh hiện nay.

Các hệ thống bán hàng tự động hiện nay chỉ dừng lại ở mức độ trả lời những câu hỏi đơn giản có sẵn, việc hỗ trợ Tiếng Việt không đầy đủ, khó khăn trong việc cải tiến. Những bất cập này làm cho việc vận hành và sử dụng hệ thống không mang lại nhiều lợi ích thiết thực. Dựa vào mô hình sinh, tôi xây dựng Chatbot trả lời tự động cho Tiếng Việt nhằm phục vụ riêng cho nghiệp vụ bán hàng.

Thời đại của Chatbot chỉ mới và đang bắt đầu phát triển nhưng lợi ích từ những ứng dụng của chúng mang lại giúp chúng ta hưởng rất nhiều lợi ích. Với những sự phát triển và tiến bộ của trí tuệ nhân tạo trong những năm gần đây,

chúng ta hoàn toàn có thể mong đợi một tương lai nơi Chatbot không chỉ thay con người đưa ra các quyết định mà còn giúp giải quyết các vấn đề trong cuộc sống.

1. Động lực nghiên cứu

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 điều khiển thông minh, tự động để mang lại hiệu quả cao hơn và Chatbot là một sự lựa chọn hoàn hảo.

Hiện nay, các ứng dụng trò chuyện trực tuyến được mọi người sử dụng đang bắt đầu trở thành một phương tiện ưa thích để giao tiếp với các doanh nghiệp và giải quyết thắc mắc của khách hàng. Ứng dụng nhắn tin nhanh đã trở thành điểm đến hàng đầu cho mọi thương hiệu nhằm tiếp cận người tiêu dùng, bởi vậy không có gì đáng ngạc nhiên khi Chatbot ngày càng trở nên phổ biến.

Với một khối lượng lớn câu hỏi, yêu cầu mà chúng ta đang phải giải quyết mỗi ngày như: khách hàng hỏi về thông tin sản phẩm, tư vấn dịch vụ, nhân viên hỏi về các nội quy, quy định của công ty, con cái hỏi về những sự việc chúng đang muốn tìm hiểu trong lứa tuổi...ngoài ra Chatbot còn được áp dụng trong rất nhiều lĩnh vực:

Lĩnh vực giải trí: Các Chatbot giải trí trực tuyến tốt nhất dựa trên AI hiện đang được ứng dụng là Mitsuku, Rose, Insomno Bot...người dùng có thể nói chuyện và tương tác với chúng hàng giờ, nó trả lời câu hỏi của bạn theo cách nhân văn nhất và hiểu được tâm trạng của bạn với ngôn ngữ bạn đang sử dụng.

Lĩnh vực thời tiết: Poncho là Chatbot điển hình được thiết kế để trở thành một chuyên gia thời tiết, ngoài dự báo thời tiết chúng còn gửi cảnh báo khi thời tiết xấu với sự chấp thuận của người dùng.

Lĩnh vực hoạt động xã hội: Để nâng cao nhận thức của con người về cuộc khủng hoảng nước ở Ethiopia (dưới 50% dân số được sử dụng nước sạch), tổ chức từ thiện về nước hợp tác với Lokai để tạo ra Yeshi. Yeshi là một Chatbot đại diện các cô gái trẻ ở Ethiopia, người phải đi bộ 2,5 giờ mỗi ngày để tìm nước sạch. Khi ai đó bắt đầu trò chuyện với bot, Yeshi sẽ gửi hình ảnh, video, clip âm thanh và

bản đồ để tạo ra trải nghiệm cảm xúc sâu sắc giúp người dùng khám phá ra thực tế khắc nghiệt của người Ethiopia như Yeshi.

Lĩnh vực nhà hàng và các ngành bán lẻ: Khách hàng được Chatbot chào đón và được cung cấp các tiện ích menu như: chọn vị trí chỗ ngồi khi đến nhà hàng, hỗ trợ thanh toán và được thông báo khi nào họ có thể bắt đầu lấy thức ăn của họ.

Lĩnh vực du lịch và khách sạn: Chatbot có thể trợ giúp các khách sạn trong một số nghiệp vụ, bao gồm quản lý quỹ thời gian, dịch vụ chăm sóc khách hàng và giảm chi phí nhân lực. Chúng có thể được xây dựng để trò chuyện với khách bằng nhiều loại ngôn ngữ khác nhau, giúp cho các khách hàng khi nói chuyện bằng ngôn ngữ địa phương của mình dễ dàng hơn.

Lĩnh vực y tế: Chatbot lĩnh vực y tế sẽ hỏi về các triệu chứng, các thông số cơ thể và quá trình khám bệnh, 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 tự điều trị một số bệnh mà không cần sự trợ giúp của bác sĩ.

Lĩnh vực hàng không: Khách hàng sử dụng dịch vụ của ngành hàng không có thể nhận tài liệu chuyến bay của mình qua Messenger, bao gồm xác nhận đặt vé, thông báo đăng ký, thẻ lên máy bay và cập nhật trạng thái chuyến bay.

Lĩnh vực Ngân hàng: Chatbot lĩnh vực Ngân hàng hỗ trợ tư vấn cho khách hàng về các sản phẩm dịch vụ của Ngân hàng như thông tin về lãi suất tiền gửi, lãi suất tiền vay, các gói vay ưu đãi, ..vv giúp khách hàng có thể có được thông tin mà không cần gặp trực tiếp nhân viên Ngân hàng.

2. Mục tiêu luận văn

Tìm hiểu và trình bày các kỹ thuật xử lý ngôn ngữ tự nhiên trong NLU, NLP như phân loại ý định các câu (intent classification hay intent detection), yêu cầu của người dùng, biểu diễn ngôn ngữ, trích chọn thông tin (information extraction) và quản lý cuộc hội thoại, ... trong ứng dụng cụ thể là việc xây dựng Chatbot bán hàng.

Luận văn tập trung đưa giải pháp và xây dựng mô hình Chatbot **ứng dụng trong miền đóng (closed domain)** và có khả năng sinh ra các câu trả lời phù hợp

với những câu hỏi, yêu cầu từ phía người dùng. Phần lớn các hệ thống Chatbot hiện nay triển khai trong thực tế thì phần lớn là được xây dựng trên mô hình truy xuất thông tin và được áp dụng trong những miền ứng dụng cụ thể.

Với bài toán này thì luận văn sẽ tập trung xây dựng mô hình Chatbot hỗ trợ người dùng trong nghiệp vụ bán hàng dựa vào framework **Rasa** và áp dụng những kiến thức nền tảng để có thể làm chủ và tùy chỉnh trên mã nguồn mở này. Đối tượng nghiên cứu ở đây cụ thể là đơn vị bán hàng hoặc cá nhân bán hàng online trên mạng xã hội.

3. Cấu trúc luận văn

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

CHƯƠNG 1: Tổng quan các hệ thống Chatbot: Chương này sẽ giới thiệu những kiến thức tổng quan về một hệ thống Chatbot, tìm hiểu chi tiết cấu trúc các thành phần và những vấn đề gặp phải khi xây dựng hệ thống Chatbot.

CHƯƠNG 2: Các kỹ thuật sử dụng trong Chatbot: Chương này giới thiệu một số kiến thức nền tảng về mạng nơ-ron nhân tạo, cách thức hoạt động của mạng nơ-ron và một số các kỹ thuật được ứng dụng trong việc xử lý ngôn ngữ tự nhiên nói riêng hay xây dựng Chatbot nói chung.

CHƯƠNG 3: Xây dựng Chatbot bán hàng: Chương này sẽ mô tả từng bước xây dựng bài toán trên nền tảng mã nguồn mở Rasa. Phần thực nghiệm và đánh giá sẽ cho ta biết khả năng phục vụ của Chatbot cũng như chỉ ra những điểm hạn chế của Chatbot nhằm tìm cách cải tiến và tìm hướng đi mới cho việc xây dựng Chatbot nhằm phục vụ nghiệp vụ bán hàng.

CHƯƠNG 4: Kết luận: Đưa ra những kết luận, đánh giá và định hướng nghiên cứu tiếp theo.

TÀI LIỆU THAM KHẢO: Liệt kê các tài liệu mà luận văn tham khảo trên nhiều nguồn khác nhau.

PHỤ LỤC: Danh sách các đoạn hội thoại với bot được đính kèm ở phần thử nghiệm.

CHƯƠNG 1 : TỔNG QUAN CÁC HỆ THỐNG CHATBOT

Chương này sẽ giới thiệu những kiến thức tổng quan về một hệ thống Chatbot, các mô hình Chatbot bán hàng hiện nay, tìm hiểu chi tiết cấu trúc các thành phần và những vấn đề gặp phải khi xây dựng hệ thống Chatbot.

1.1 Giới thiệu

Hệ thống trả lời tự động hay còn gọi là Chatbot là một chương trình máy tính có khả năng giao tiếp với con người bằng cách tự động trả lời những câu hỏi hoặc xử lý tình huống. Trí thông minh của Chatbot được xác định bằng thuật toán của người tạo nên chúng. Chatbot được ứng dụng trong rất nhiều lĩnh vực như thương mại điện tử, dịch vụ khách hàng, tài chính ngân hàng, giải trí, y tế, giáo dục,...vv.

Chatbot có thể được chia thành hai loại:

- Loại thứ nhất: Hệ thống không có định hướng mục tiêu (Miền mở)

Miền mở (Open Domain): Mô hình trả lời tự động trên miền mở cho phép người dùng có thể tham gia trò chuyện với một chủ đề bất kỳ, không nhất thiết phải có một mục tiêu rõ ràng hay một ý định cụ thể nào. Các cuộc trò chuyện trên mạng xã hội như Facebook, Twitter thường là miền mở, chúng có thể đi vào tất cả các chủ đề. Số lượng các chủ đề thảo luận được đề cập đến là không giới hạn, do đó, tri thức yêu cầu được tạo ra để trả lời các câu đố thoại thuộc miền mở trở nên khó hơn. Tuy nhiên, việc thu thập trích rút dữ liệu từ miền này khá phong phú và đơn giản.

- Loại thứ hai: Hệ thống hướng mục tiêu trên một miền ứng dụng (Miền đóng)

Miền đóng (Close Domain): Mô hình trả lời tự động thuộc miền đóng thường tập trung vào trả lời các câu hỏi đối thoại liên quan đến một miền cụ thể, ví dụ như: Y tế, giáo dục, thời tiết, du lịch, mua sắm, ... Trong một miền đóng cụ thể, không gian các mẫu hỏi input và output là có giới hạn, bởi vì các hệ thống này đang cố gắng để đạt được một mục tiêu rất cụ thể. Hệ thống hỗ trợ kỹ thuật (Technical Customer Support) hay tư vấn và hỗ trợ mua hàng (Shopping Assistants) là các ứng dụng thuộc miền đóng. Các hệ thống này không thể đối thoại về cách lĩnh vực khác, chúng chỉ cần thực hiện các nhiệm vụ cụ thể một

cách hiệu quả nhất có thể. Chắc chắn, người dùng vẫn có thể hỏi đáp bất cứ gì, nhưng hệ thống không yêu cầu phải xử lý những trường hợp ngoại lệ này.

Mỗi cách tiếp cận bài toán đều có hướng giải quyết khác nhau dẫn tới các kỹ thuật sử dụng khác nhau. **Trong luận văn này, tôi sẽ tập trung vào xây dựng Chatbot thuộc loại thứ hai, cụ thể là bài toán hướng mục tiêu tư vấn hỗ trợ mua hàng.**

1.2 Các mô hình chatbot bán hàng tiêu biểu hỗ trợ Tiếng Việt hiện nay

1.2.1 Chatbot theo kịch bản (menu/button)

Chatbot theo kịch bản (menu/button) là các hệ thống phân cấp cây quyết định được trình bày cho người dùng dưới dạng các nút (button). Chatbot xây dựng sẵn một tập các menu với các lựa chọn như một chiếc điều khiển, người dùng phải giao tiếp với Chatbot thông qua các thao tác click vào nút đúng theo yêu cầu mình mong muốn, để nhận được câu trả lời của Chatbot. Sau đây là một vài ví dụ về loại Chatbot này:



Hình 1: Ví dụ về dạng Chatbot (menu/button)

Ở ví dụ trên Chatbot đã xây dựng sẵn ba nút chọn tương ứng với ba yêu cầu của người dùng, chỉ cần chọn click vào đúng nút theo nhu cầu của mình thì Chatbot sẽ trả lời theo đúng yêu cầu đó.

Ưu điểm của Chatbot này là xây dựng rất dễ dàng, độ chính xác cao vì người dùng đưa ra yêu cầu dựa trên những nút đã được xây dựng trước, tuy nhiên người dùng sẽ bị động trước những mong muốn của mình, mà phải phụ thuộc vào sự cung cấp các menu lựa chọn của Chatbot.

1.2.2 Chatbot nhận dạng từ khóa

Khác với các Chatbot dạng menu/button, các Chatbot dựa trên nhận dạng từ khóa có thể lắng nghe những câu nói của người dùng và trả lời một cách thích hợp. Những Chatbot sử dụng các từ khóa tùy biến và AI để xác định làm thế nào để đưa ra câu trả lời phù hợp nhất cho người dùng. Sau đây là ví dụ về loại Chatbot này:



Hình 2: Ví dụ về Chatbot nhận dạng từ khoá

Trong ví dụ trên khi câu nói của người dùng xuất hiện từ khoá “xin giá” thì chatbot sẽ đưa ra câu trả lời về thông tin giá của sản phẩm và các thông tin liên quan đến sản phẩm.

Ưu điểm của mô hình Chatbot này là giúp người dùng chủ động hơn trong việc đưa ra yêu cầu, như có thể đưa ra yêu cầu của mình thông qua câu nói mà không cần lựa chọn các nút nội dung yêu cầu làm cho cuộc trò chuyện tự nhiên hơn.

Tuy mô hình này có những ưu điểm hơn so với Chatbot dựa trên menu/button nhưng nó vẫn còn khá nhiều những nhược điểm như khi người dùng sử dụng các từ đồng nghĩa với các từ khoá thì Chatbot không thể phát hiện được để trả lời phù hợp, và không thể nắm bắt được ngữ cảnh cuộc trò chuyện.

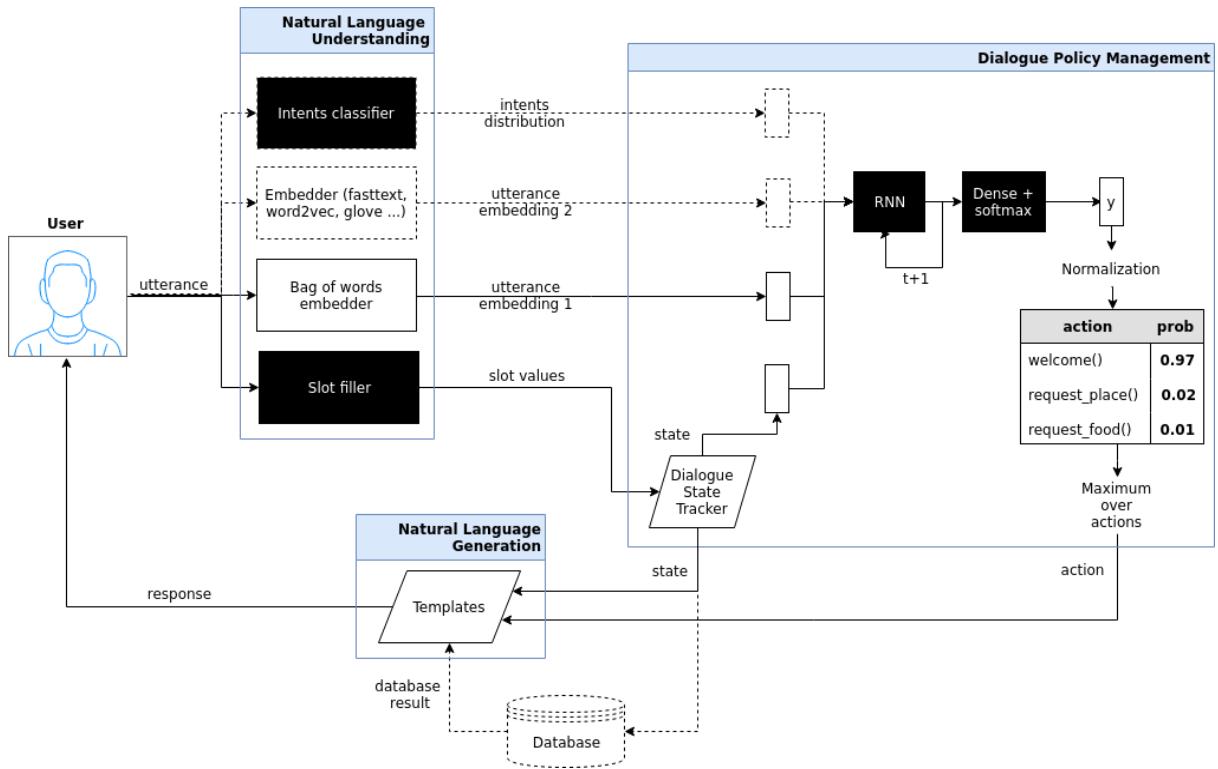
Hiện nay loại chatbot nhận dạng từ khoá kết hợp menu/button đang rất phổ biến trong lĩnh vực dịch vụ. Những dịch vụ hay sử dụng loại chatbot này là dịch vụ bán hàng, ship đồ ăn,...vv.

1.2.3 Mô hình Chatbot bán hàng mà luận văn nghiên cứu

Từ những phân tích về ưu nhược điểm của 2 mô hình Chatbot dựa trên menu/button và nhận dạng từ khoá, tôi lựa chọn xây dựng Chatbot dựa trên các phương pháp học máy và trí tuệ nhân tạo để có thể lắng nghe và hiểu được những yêu cầu của người dùng một cách tự nhiên nhất. Ví dụ như khi người dùng đưa ra một yêu cầu “Bộ sản phẩm mã sp90 này có giá bao nhiêu ?” thì Chatbot sẽ hiểu được ý định của người dùng đang muốn hỏi về thông tin giá sản phẩm và cụ thể là sản phẩm có mã sp90.

Với mô hình này thì mô hình này thì Chatbot thông minh hơn, có thể hiểu được các ý định và có thể trích chọn được các thông tin từ yêu cầu của người dùng, lưu được ngữ cảnh và sinh ra được câu trả lời phù hợp nhất, giúp cho trải nghiệm của người dùng được tự nhiên hơn. Để xây dựng được mô hình trên thì Chatbot phải có cấu trúc và các thành phần hệ thống như mục 1.3 của chương.

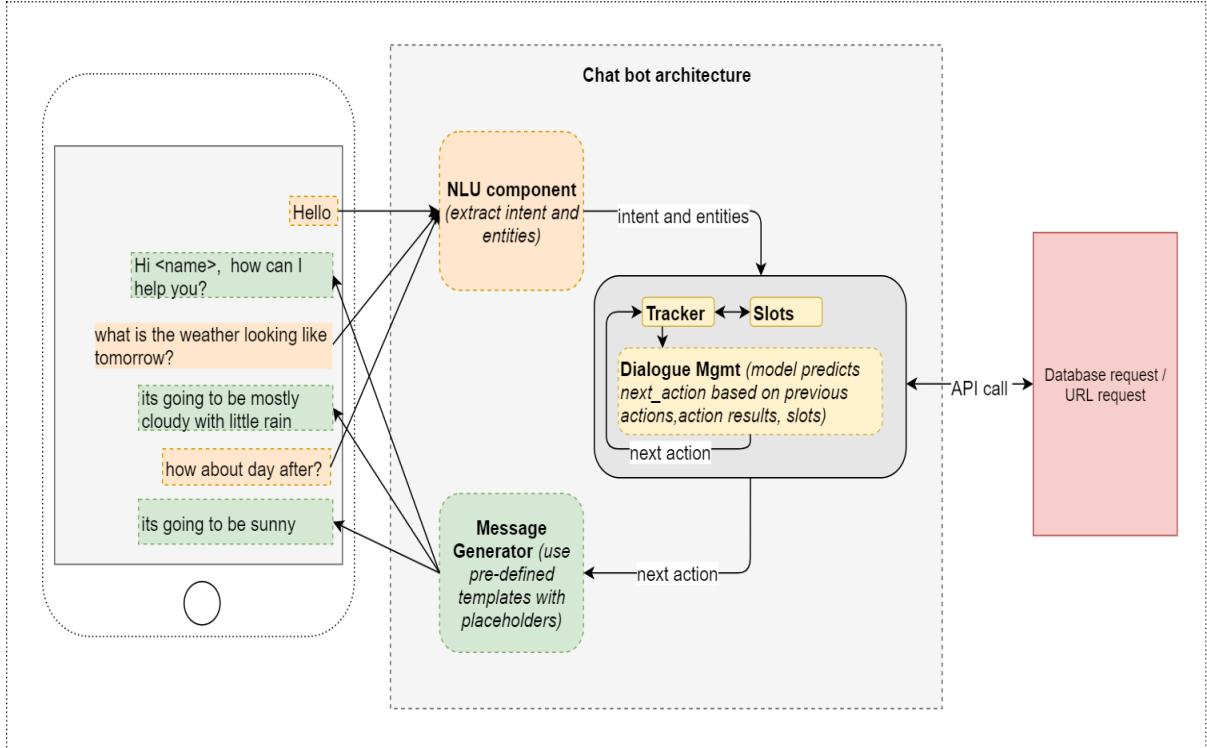
1.3 Cấu trúc các thành phần hệ thống Chatbot



Hình 3: Cấu trúc các thành phần cơ bản hệ thống Chatbot [12]

Chatbot có ba thành phần chính là: thành phần hiểu ngôn ngữ tự nhiên (**NLU**), thành phần 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 (NLP) có nhiệm vụ xác định được ý định câu hỏi của người dùng (intent classification) và trích chọn thông tin (slots filter).
- **DM:** Quản lý hội thoại có nhiệm vụ xác định được hành động (action) tiếp theo dựa vào trạng thái hành động trước đó hoặc ngữ cảnh của cuộc hội thoại. Các ngữ cảnh này phải được tham chiếu trong các kịch bản dụng sẵn (history) được đào tạo cho Chatbot. Thành phần này cũng chịu trách nhiệm việc truy xuất dữ liệu từ hệ thống khác qua các lệnh gọi API trong action.
- **NLG:** là thành phần sinh ngôn ngữ dựa vào chính sách (policy) và hành động được xác định trong DM thông qua các bộ hội thoại. NLG có thể sinh ra câu trả lời dựa vào tập mẫu câu trả lời (pre-defined template) đã đào tạo cho bot. Để hiểu rõ chi tiết hơn về luồng xử lý tin nhắn từ các thành phần của Chatbot ta xem Hình 4:



Hình 4: Mô hình các thành phần xử lý trong Chatbot [1]

1.4 Hiệu ứng ngôn ngữ tự nhiên (NLU)

Đây có thể nói là thành phần quan trọng nhất của Chatbot. Chatbot có thông minh hay không thì đây là thành phần quyết định. Mục tiêu của thành phần này là trích xuất ra hai thành phần thông tin từ câu nói của người dùng:

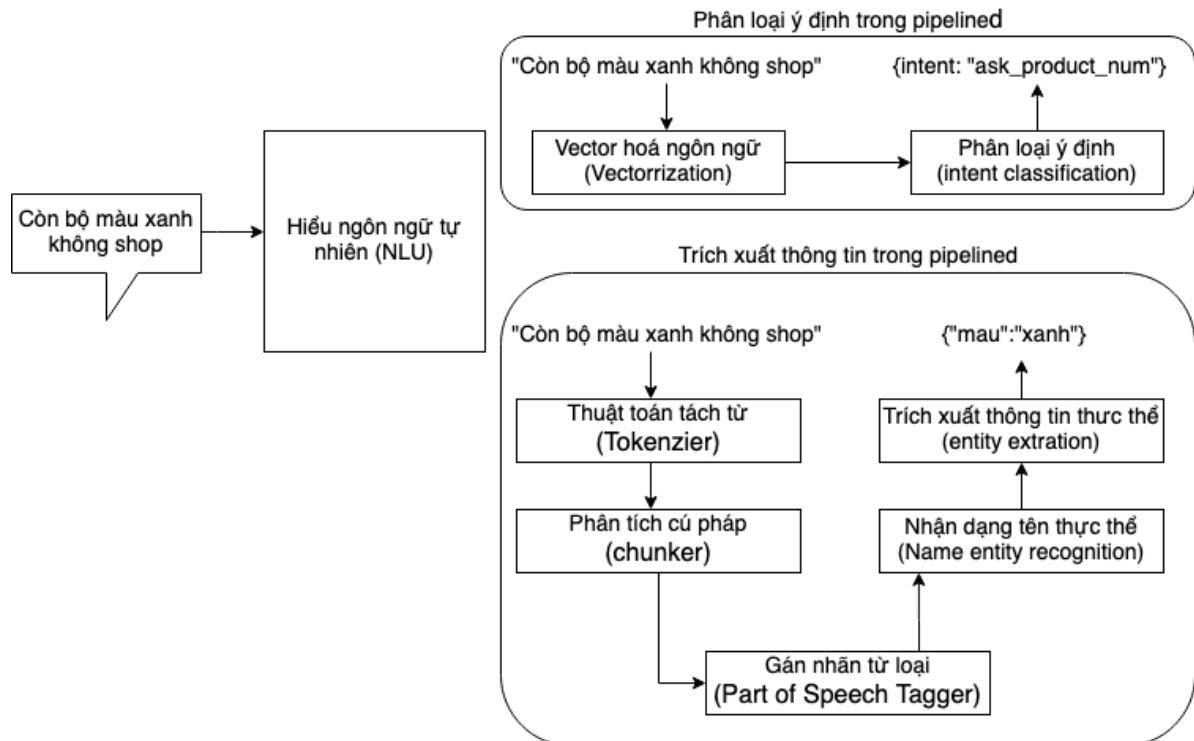
- Đầu tiên là phân loại ý định (intent classification), ví dụ như xác định được ý định của khách hàng như hỏi về giá sản phẩm, size mặc phù hợp, hỏi về dịch vụ ship hàng...
- Cuối cùng là bước trích xuất thông tin (slot fillter hay entity extraction) trong câu hỏi người dùng. Ví dụ ta phải trích chọn được thông tin loại sản phẩm trong câu hỏi người dùng: “Chiếc áo này bao nhiêu tiền”. Từ việc trích xuất được thông tin sản phẩm là “áo” thì Chatbot mới có cơ sở trả lời cho người dùng.
- NLU xử lý tin nhắn người dùng bằng một đường ống (pipeline) nơi mà cấu hình các bước xử lý liên tiếp theo tuần tự:



Hình 5: Các bước xử lý chính trong pipeline của NLU [1]

Trong đường ống này thì bạn có thể tùy chỉnh các thành phần từ bước tiền xử lý dữ liệu, mô hình hóa ngôn ngữ, các thuật toán dùng để tách từ và trích xuất thông tin thực thể...

Để chi tiết các bước xử lý xem trong Hình 6:



Hình 6: Các bước xử lý trong NLU [2]

Để phân loại ý định của một câu nói của người dùng, chúng ta cần mô hình hóa ngôn ngữ, nghĩa là biểu diễn ngôn ngữ dưới dạng một vectơ số học để máy tính hiểu (vector hóa). Phương pháp phổ biến nhất hiện nay là nhúng từ (word embedding). Word embedding là tên chung cho một tập hợp các mô hình và phương pháp ngôn ngữ dành riêng cho xử lý ngôn ngữ tự nhiên (NLP), trong đó các từ hoặc cụm từ vựng được ánh xạ tới các vectơ số thực. Về mặt khái niệm, nó liên quan đến việc nhúng toán học từ một không gian có một chiều cho mỗi từ vào không gian vectơ liên tục với các kích thước thấp hơn nhiều. Một số phương thức đại diện phổ biến như Word2Vec, GloVe hoặc FastText mới hơn sẽ được giới thiệu trong phần sau.

Sau khi mô hình hóa ngôn ngữ bao gồm dữ liệu đầu vào training cho bot thì việc xác định ý định người dùng từ câu hỏi người dùng dựa trên tập đã training

là bước phân loại ý định (intent classification) hay phân loại văn bản. Ở bước này ta có thể dùng một số kỹ thuật như:

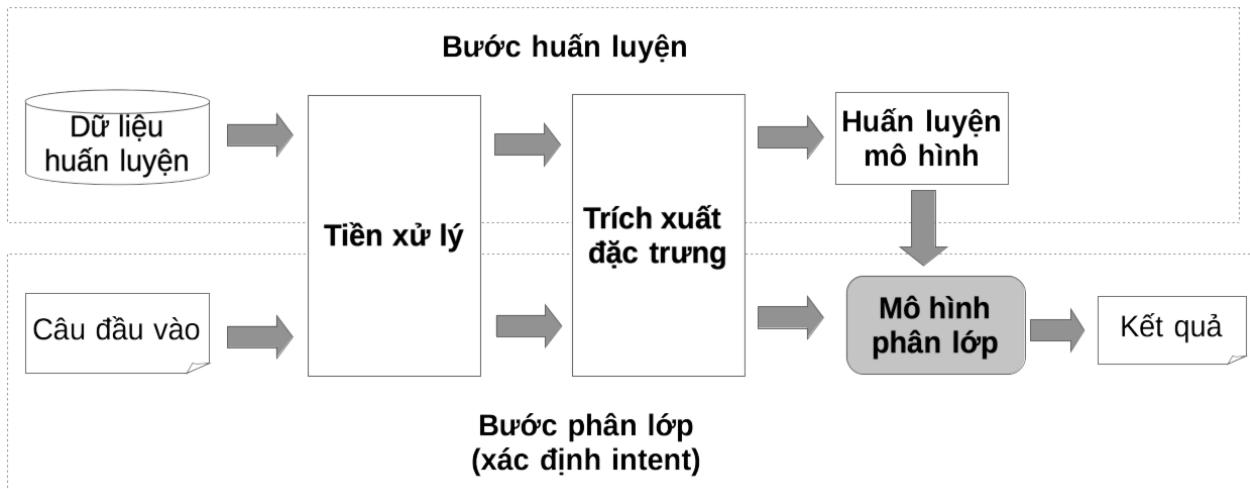
- Naive Bayes.
- Decision Tree (Random Forest).
- Vector Support Machine (SVM).
- Convolution Neural Network (CNN).
- Recurrent Neural Network (RNN).
- Long Short-Term Memory (LSTM, Bi-LSTM).

Hầu hết các Chatbot hiện tại đều ứng dụng mô hình deep learning như RNN và LSTM để phân loại ý định người dùng. Bài toán thách thức lớn nhất cho các Chatbot ở bước này là xác định nhiều ý định (multiple intents) trong một câu nói người dùng. Ví dụ nếu bạn nói “Xin chào, giá áo này bao nhiêu vậy shop” thì bot phải xác định được hai ý định là “chào hỏi” và “giá áo” trong câu nói người dùng. Nếu bot có thể hiểu và trả lời được câu hỏi loại này sẽ giúp việc tương tác với bot trở nên tự nhiên hơn.

Tiếp đến là việc trích xuất thông tin trong câu hỏi thoại người dùng. Các thông tin cần trích xuất thường dưới dạng số, chuỗi hoặc thời gian và chúng phải được khai báo và huấn luyện trước.

Phân tách các từ (Tokenization hay word segmentation): Tách từ là một quá trình xử lý nhằm mục đích xác định ranh giới của các từ trong câu văn, cũng có thể hiểu đơn giản rằng tách từ là quá trình xác định các từ đơn, từ ghép... có trong câu. Đối với xử lý ngôn ngữ, để có thể xác định cấu trúc ngữ pháp của câu, xác định từ loại của một từ trong câu, yêu cầu nhất thiết đặt ra là phải xác định được đâu là từ trong câu. Vấn đề này tưởng chừng đơn giản với con người nhưng đối với máy tính, đây là bài toán rất khó giải quyết. Thông thường thì các ngôn ngữ phân tách các từ bởi khoảng trắng nhưng đối với ngôn ngữ Tiếng Việt thì có rất nhiều từ ghép và cụm từ. Ví dụ trong câu “ship cho mình về Mỹ Đình” thì từ ghép “Mỹ Đình” được tạo bởi hai từ đơn “Mỹ” và “Đình”. Có một số thuật toán hỗ trợ giải quyết bài toán này như mô hình so khớp từ dài nhất (longest matching), so khớp cực đại (Maximum Matching), Markov ẩn (Hidden Markov Models- HMM) hay mô hình CRF (conditional random field), ở luận văn này tôi sử dụng thư viện Underthesea [26] của tác giả Vũ Anh cho phép hỗ trợ tách từ Tiếng Việt với tỷ lệ chính xác cao.

1.4.1 Xác định ý định người dùng



Hình 7: Mô hình các bước xác định ý định

Mô hình phân loại ý định của người dùng bao gồm một số bước cơ bản:

- ✓ **Bước 1:** Tiền xử lý dữ liệu
- ✓ **Bước 2:** Trích xuất đặc trưng
- ✓ **Bước 3:** Huấn luyện mô hình
- ✓ **Bước 4:** Phân lớp

Trong bước thứ nhất tiền xử lý dữ liệu chính là thao tác “làm sạch” dữ liệu như: xử lý loại bỏ các thông tin dư thừa, chuẩn hoá dữ liệu và chuẩn hoá các từ viết sai chính tả thành đúng chính tả theo chuẩn Tiếng Việt, chuẩn hoá các từ viết tắt, tách các từ trong câu... Bước tiền xử lý dữ liệu đóng một vai trò vô cùng quan trọng trong hệ thống Chatbot. Nếu ở bước tiền xử lý này dữ liệu đầu vào được làm sạch và chuẩn hoá tốt thì sẽ làm tăng khả năng độ chính xác cũng như sự thông minh cho Chatbot.

Tiếp đến bước thứ hai là trích xuất đặc trưng (feature extraction hay feature engineering) từ những dữ liệu đã được chuẩn hoá và làm sạch. Trong mô hình học máy truyền thống, bước trích xuất đặc trưng này có sự ảnh hưởng lớn đến độ chính xác của mô hình phân lớp. Để trích xuất được những đặc trưng tốt nhất, chúng ta cần phân tích dữ liệu tỉ mỉ, chuyên sâu và cần cả những chuyên gia trong lĩnh vực bán hàng để giúp phân tích được dữ liệu một cách chính xác nhất.

Ở bước thứ ba này là bước huấn luyện mô hình có input đầu vào là các đặc trưng quan trọng đã được trích xuất ở bước thứ hai và áp dụng các thuật toán học máy để tạo ra một mô hình phân lớp. Các mô hình phân lớp ở đây có thể là các

quy tắc phân lớp (nếu sử dụng cây quyết định) hoặc là các vector trọng số tương ứng với các đặc trưng được trích xuất (như trong các mô hình hồi quy logistic, mô hình mạng SVM hoặc mạng Nơ-ron).

Sau khi đã xây dựng được một mô hình phân lớp intent, chúng ta có thể sử dụng nó để phân lớp một câu hỏi hoặc yêu cầu mòi. Câu hỏi, yêu cầu này cũng phải tuân theo các bước tiền xử lý dữ liệu và trích xuất đặc trưng, sau đó mô hình phân lớp sẽ chấm “điểm số” cho từng ý định trong tập các ý định và đưa ra ý định nào có điểm cao nhất. Để đưa câu trả lời chính xác nhất với câu hỏi và yêu cầu của người dùng, Chatbot cần xác định được chính xác ý định của người dùng. Việc xác định ý định của người dùng sẽ quyết định hội thoại tiếp được diễn ra như thế nào. Vì thế, nếu xác định sai ý định người dùng, Chatbot sẽ đưa ra những câu trả lời sai, không hợp ngữ cảnh. Khi đó, người dùng có thể cảm thấy không hài lòng dẫn đến việc không sử dụng hệ thống và sẽ không mua hàng. Từ lý do trên cho thấy, bài toán xác định chính xác ý định người dùng đóng một vai trò rất quan trọng trong hệ thống Chatbot bán hàng.

Đối với miền ứng cụ thể, còn gọi là miền đóng, chúng ta giới hạn 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 tính năng và nghiệp vụ mà Chatbot có thể hỗ trợ. Như vậy với việc giới hạn như vậy thì bài toán xác định ý định người dùng chúng ta có thể quy về bài toán phân lớp văn bản. Với input đầu vào là một câu nói của người dùng, mô hình phân lớp sẽ xác định được ý định tương ứng với câu đó trong tập các intent đã được gán nhãn và định nghĩa trước đó.

Để xây dựng một mô hình phân lớp ý định chính xác, chúng ta cần một tập dữ liệu huấn luyện bao gồm các cách diễn đạt câu hỏi hoặc yêu cầu khác nhau cho mỗi ý định. Ví dụ, khi cùng một mục đích hỏi về giá sản phẩm thì người dùng có thể sử dụng những cách diễn đạt sau:

- *Giá áo này bao nhiêu?*
- *Áo này giá bao nhiêu?*
- *Quần này shop bán giá thế nào?*
- *Quần này giá bao nhiêu vậy shop?*

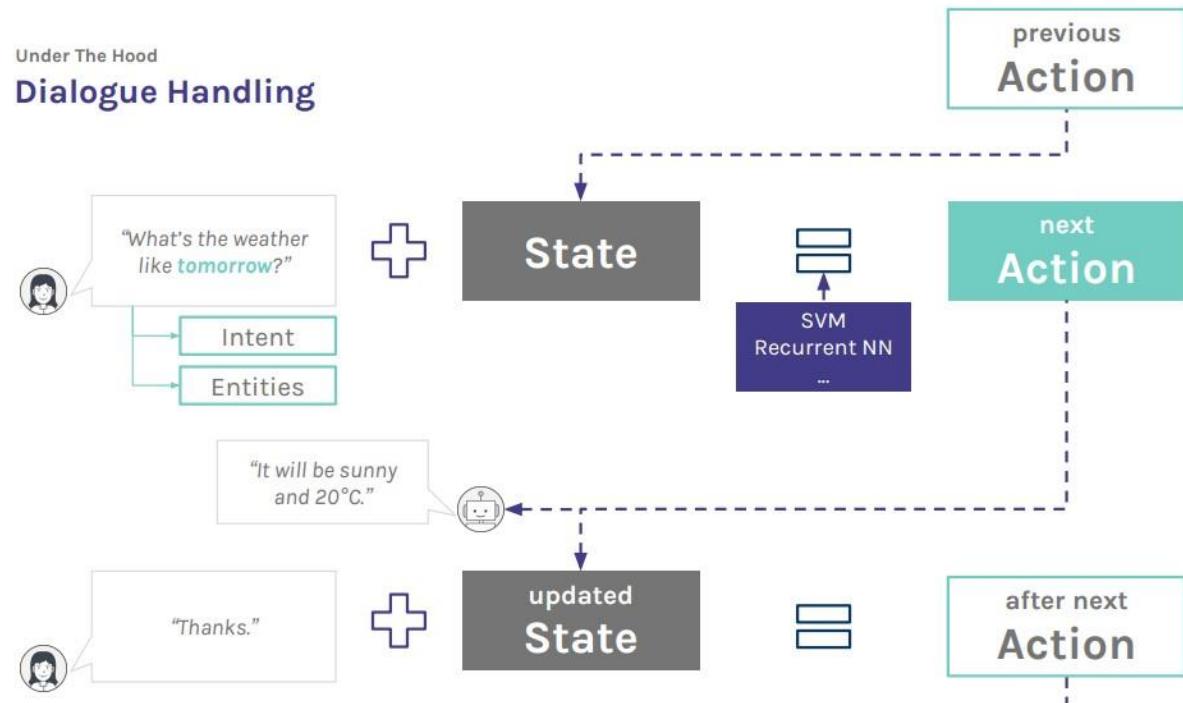
Như vậy có thể nói, trong việc tạo và xây dựng dữ liệu huấn luyện cho bài toán phân lớp ý định người dùng là một trong những nhiệm vụ quan trọng nhất khi phát triển hệ thống Chatbot và ảnh hưởng lớn rất lớn tới độ thông minh của

hệ thống Chatbot khi triển khai thực tế. Nhiệm này cũng đòi hỏi rất nhiều thời gian, cũng như công sức khá lớn của nhà phát triển Chatbot.

1.5 Quản lý hội thoại (DM)

Trong các cuộc trò chuyện dài (long conversation) Chatbot và người dùng, Chatbot sẽ cần phải ghi nhớ được những thông tin về ngữ cảnh (context) hoặc quản lý các trạng thái hội thoại (dialog state). Nhiệm vụ quản lý hội thoại (dialogue management) khi đó đóng vai trò quan trọng để đảm bảo việc trao đổi giữa dùng người và Chatbot là thông suốt.

Nhiệm vụ của thành phần quản lý hội thoại là nhận input đầu vào từ thành phần hiểu ngôn ngữ tự nhiên NLU, quản lý các trạng thái cuộc hội thoại (dialogue state), ngữ cảnh hội thoại (dialogue context), và là đầu ra quan trọng cho thành phần sinh ngôn ngữ (Natural Language Generation, viết tắt là NLG).



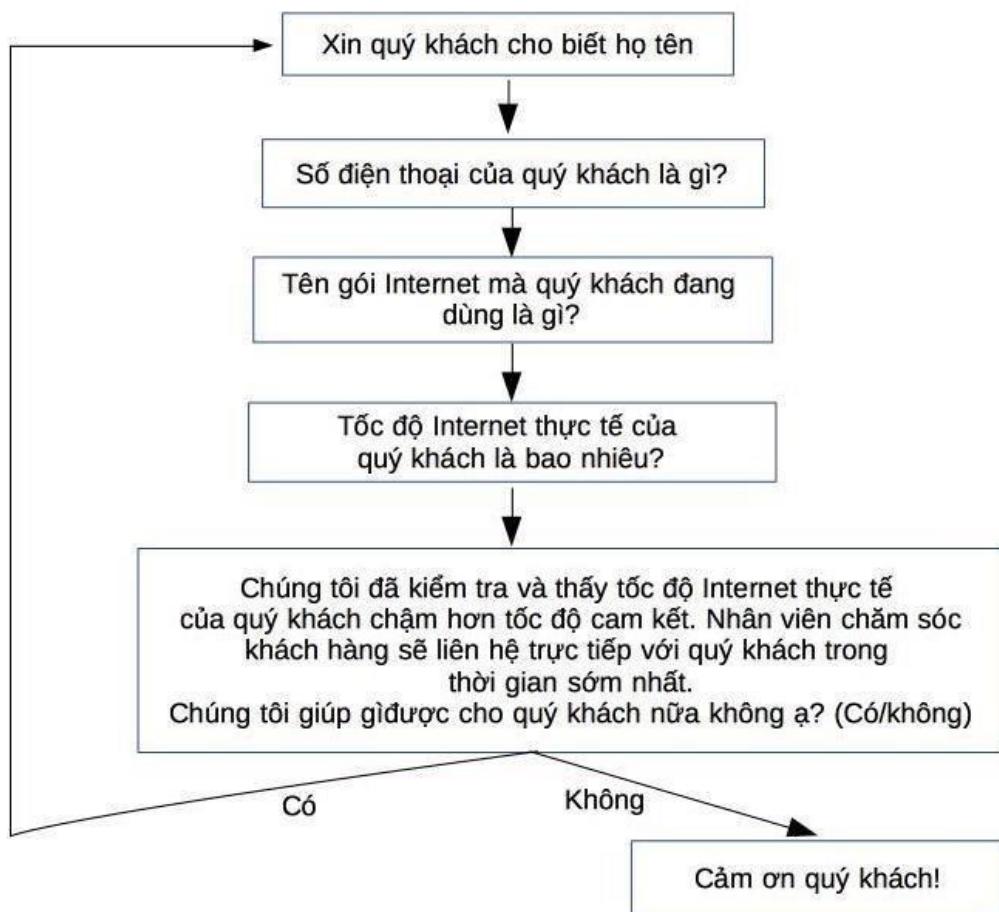
Hình 8: Mô hình quản lý trạng thái và quyết định action trong hội thoại [2]

Trạng thái hội thoại (dialog state) được lưu lại và dựa vào tập luật hội thoại (dialog policy) để quyết định hành động tiếp theo cho câu trả lời của Chatbot trong một kịch bản hội thoại, hay hành động (action) chỉ phụ thuộc vào trạng thái (dialog state) trước của nó.

Ví dụ module quản lý hội thoại trong một mô hình Chatbot phục vụ việc đặt mua vé máy bay cần biết khi nào người dùng đã cung cấp đầy đủ các thông tin cho việc đặt mua vé để tạo một ticket tới hệ thống hoặc khi nào cần phải xác

nhận lại những thông tin do người dùng cung cấp. Hiện nay, các mô hình Chatbot thường dùng là mô hình máy trạng thái hữu hạn (Finite State Automata – FSA), mô hình Frame-based (Slot Filling), hoặc mô hình lai kết hợp giữa hai mô hình này. Một số hướng nghiên cứu mới có áp dụng mô hình nơ ron ANN vào việc quản lý hội thoại giúp Chatbot thông minh hơn.

1.5.1 Mô hình máy trạng thái hữu hạn FSA



Hình 9: Quản lý hội thoại theo mô hình máy trạng thái hữu hạn FSA

Mô hình FSA là mô hình quản lý hội thoại đơn giản cơ bản nhất. Ví dụ hệ thống tư vấn và chăm sóc khách hàng của một công ty viễn thông, xây dựng để phục vụ cho những khách hàng hay than phiền về vấn đề mạng chậm. Nhiệm vụ của mô hình Chatbot là hỏi thông tin khách hàng như tên khách hàng, số điện thoại, tên gói Internet đang sử dụng, tốc độ Internet thực tế tại nơi của khách hàng. Hình 9 minh họa một mô hình quản lý hội thoại cho Chatbot của công ty viễn thông. Các trạng thái của FSA tương ứng với các câu hỏi, yêu cầu mà Chatbot hỏi người dùng. Các liên kết giữa các trạng thái tương ứng với các hành động của

Chatbot sẽ được thực hiện. Các hành động này phụ thuộc vào phản hồi của người dùng ứng với các câu hỏi. Trong mô hình FSA, Chatbot đóng vai trò định hướng người sử dụng trong cuộc hội thoại.

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

1.5.2 Mô hình Frame-based

Để giải quyết vấn đề bên trên mà mô hình FSA gặp phải, mô hình Frame-based (hoặc tên khác là Form-based) có đáp ứng được nhiệm vụ này. Mô hình Frame-based dựa trên các khung được định sẵn để định hướng cuộc hội thoại. Mỗi frame sẽ bao gồm các thông tin (slot) cần phải điền và các câu hỏi tương ứng mà Chatbot hỏi người dùng. Mô hình này cho phép người dùng điền thông tin vào nhiều các vị trí khác nhau trong khung. Hình 10 là một ví dụ trực quan về một khung cho Chatbot ở trên.

Slot	Câu hỏi
Họ tên	Xin quý khách vui lòng cho biết họ tên
Số điện thoại	Số điện thoại của quý khách là gì ạ?
Tên gói Internet	Tên gói Internet mà quý khách đang dùng là gì?
Tốc độ Internet	Tốc độ Internet thực tế của quý khách đang là bao nhiêu?

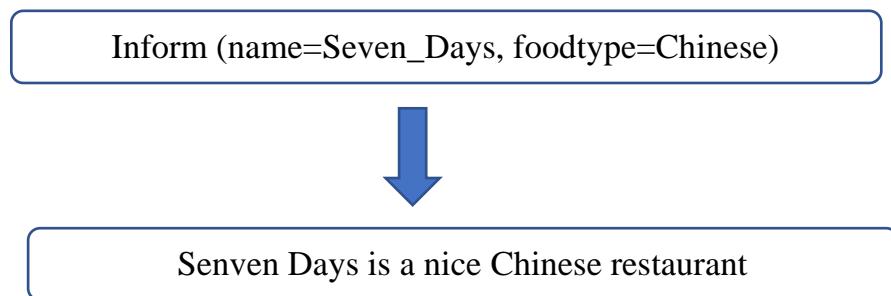
Hình 10: Frame cho Chatbot hỏi thông tin khách hàng

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

Khi xây dựng Chatbot trong các miền ứng dụng phức tạp, trong một cuộc hội thoại có thể có rất nhiều các khung khác nhau. Vấn đề đặt ra ở đây cho người phát triển Chatbot là khi đó làm sao để biết khi nào cần chuyển đổi giữa các khung. Cách tiếp cận thường được sử dụng để quản lý việc chuyển đổi điều khiển giữa các khung là định nghĩa các luật (production rule). Các luật này dựa trên một số các câu hỏi hoặc câu hỏi, yêu cầu gần nhất mà người dùng đưa ra.

1.6 Mô hình sinh ngôn ngữ (NLG)

NLG là mô hình sinh câu trả lời của Chatbot. Nó dựa vào việc ánh xạ các hành động của quản lý hội thoại vào ngôn ngữ tự nhiên để trả lời người dùng.



Có 4 phương pháp ánh xạ hay dùng là: Template-Based, Plan-based, Class-base, RNN-base

1.6.1 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.

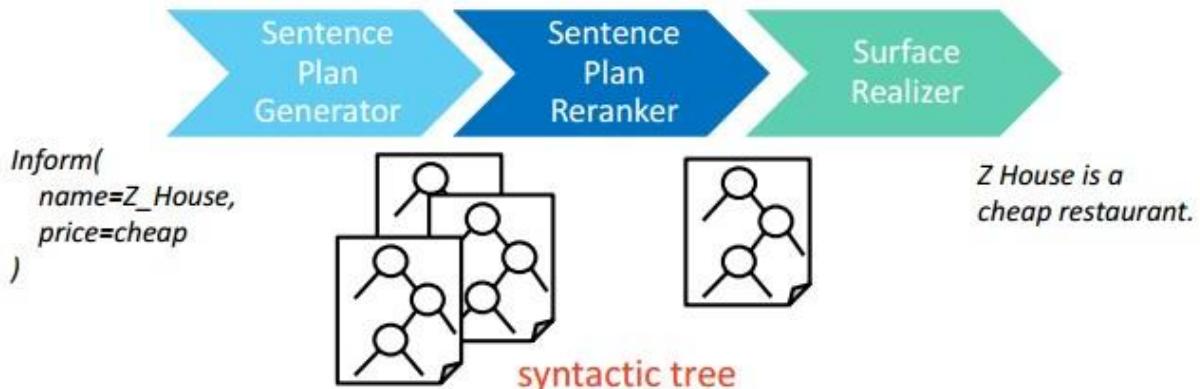
Semantic Frame	Natural Language
confirm()	"Please tell me more about the product you are looking for."
confirm(area=\$V)	"Do you want somewhere in the \$V?"
confirm(food=\$V)	"Do you want a \$V restaurant?"
confirm(food=\$V,area=\$W)	"Do you want a \$V restaurant in the \$W?"

Hình 11: Phương pháp sinh ngôn ngữ dựa trên tập mẫu câu trả lời [1]

- **Ưu điểm:** Đơn giản, kiểm soát dễ dàng. Phù hợp cho các bài toán miền đóng.

- **Nhược điểm:** 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.

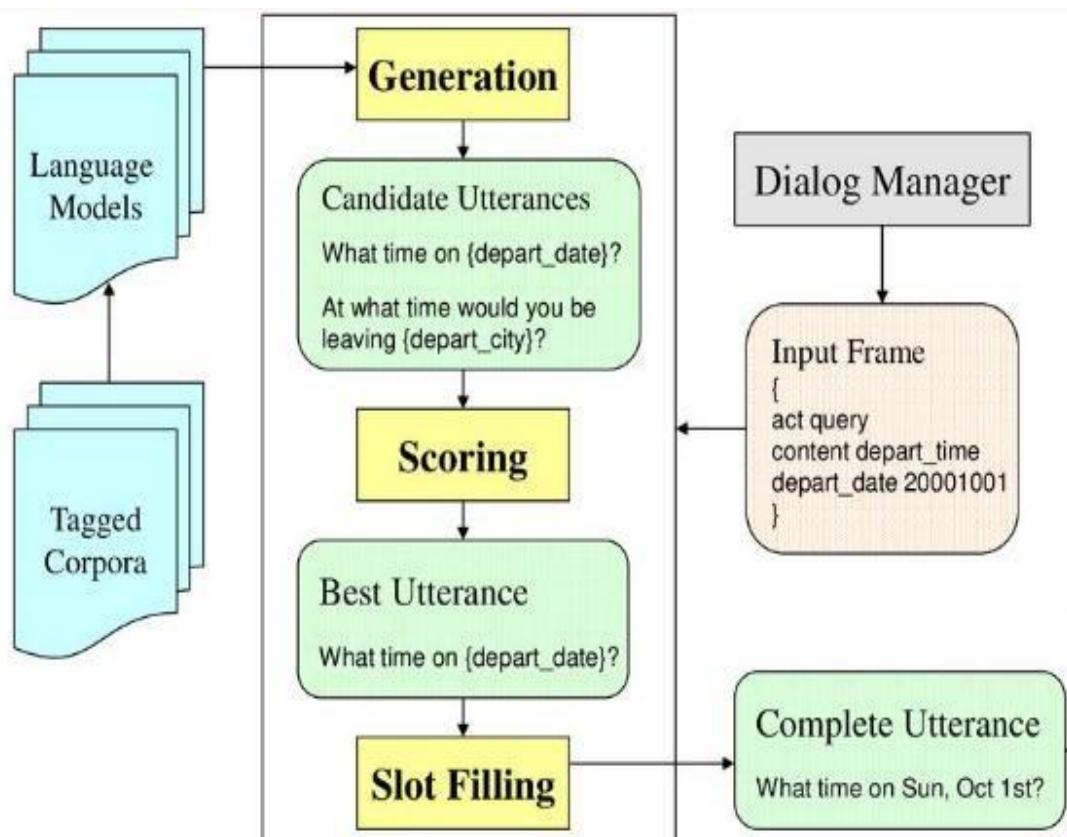
1.6.2 Plan-based NLG



Hình 12: Phương pháp sinh ngôn ngữ Plan-based [1]

- **Ưu điểm:** Có thể mô hình hóa câu trúc ngôn ngữ phức tạp
- **Nhược điểm:** Thiết kế nặng nề, đòi hỏi phải rõ miền kiến thức

1.6.3 Class-based NLG



Hình 13: Phương pháp sinh ngôn ngữ class-based [1]

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.

- **Ưu điểm:** Dễ dàng thực thi.
- **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.

1.7 Kết luận chương

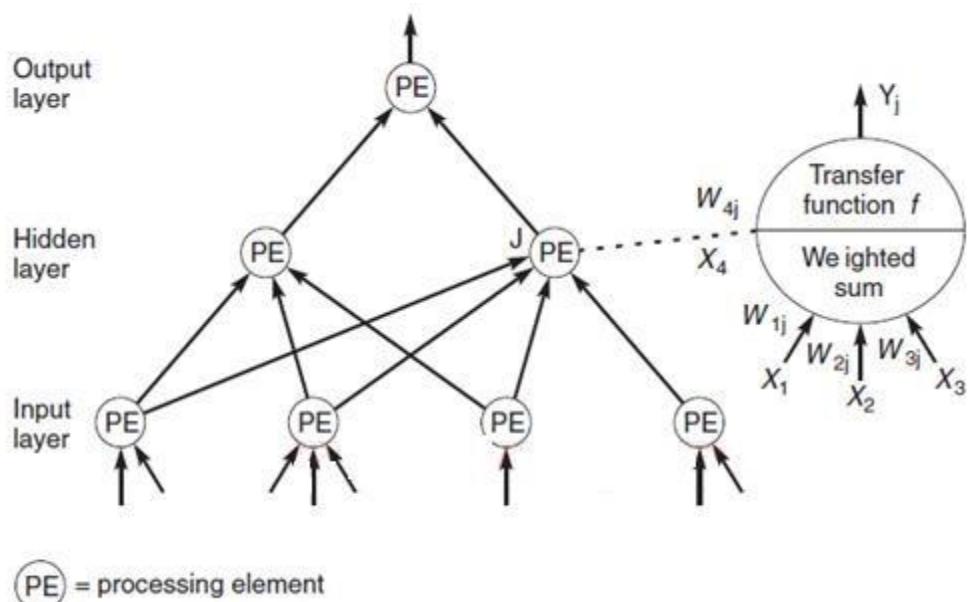
Chương này đã trình bày và giới thiệu những kiến thức tổng quan nhất về một hệ thống Chatbot, phân tích các ưu nhược điểm của mô hình Chatbot bán hàng hiện nay từ đó định hướng mô hình Chatbot mà luận văn nghiên cứu và xây dựng, đưa ra giới thiệu về chi tiết các thành phần cấu trúc và những vấn đề khi gặp phải khi xây dựng hệ thống Chatbot.

CHƯƠNG 2: CÁC KỸ THUẬT SỬ DỤNG TRONG CHATBOT

Chương này giới thiệu một số kiến thức nền tảng về mạng nơ-ron nhân tạo, cách thức hoạt động của mạng nơ-ron và một số các kỹ thuật được ứng dụng trong việc xử lý ngôn ngữ tự nhiên nói riêng hay xây dựng Chatbot nói chung.

2.1. Kiến trúc mạng nơ-ron nhân tạo

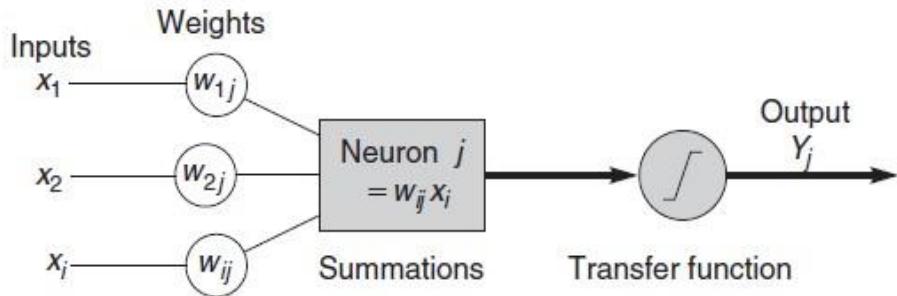
Mạng nơ-ron nhân tạo (Artificial Neural Network – ANN) là một mô phỏng xử lý thông tin hay thường được gọi ngắn gọn là mạng nơ-ron, được nghiên cứu dựa trên hệ thống thần kinh của sinh vật, giống như một bộ não con người để xử lý các thông tin. ANN bao gồm một số lượng lớn các mối gắn kết để xử lý các yếu tố làm việc trong mối quan hệ giải quyết vấn đề rõ ràng. ANN được giới thiệu năm 1943 bởi nhà thần kinh học Warren McCulloch và nhà logic học Walter Pits, hoạt động của nó giống như hoạt động bộ não của con người, được học hỏi bởi kinh nghiệm thông qua việc đào tạo, huấn luyện, có khả năng lưu giữ được các tri thức và sử dụng những tri thức đó trong việc phán đoán các dữ liệu mới, chưa biết (unseen data). Processing Elements (PE) của ANN gọi là nơ-ron, nhận các dữ liệu vào (inputs) xử lý chúng và cho ra một kết quả (output) duy nhất. Kết quả xử lý của một nơ-ron có thể làm input cho các nơ-ron khác.



Hình 14: Kiến trúc mạng nơ-ron nhân tạo [15]

Kiến trúc chung của ANN gồm 3 thành phần đó là input layer, hidden layer và output layer. Trong đó, lớp ẩn (hidden layer) gồm các nơ-ron, nhận dữ liệu

input từ các nơ-ron ở lớp trước đó và chuyển đổi các input này cho các lớp xử lý tiếp theo. Quá trình xử lý thông tin của một ANN như sau:



Hình 15: Quá trình xử lý thông tin của một mạng nơ-ron nhân tạo [15]

Trong đó, mỗi input tương ứng với 1 thuộc tính của dữ liệu. Ví dụ như trong ứng dụng của ngân hàng xem xét có chấp nhận cho khách hàng vay tiền hay không thì mỗi input là một thuộc tính của khách hàng như thu nhập, nghề nghiệp, tuổi, số con... Output là một giải pháp cho một vấn đề, ví dụ như với bài toán xem xét chấp nhận cho khách hàng vay tiền hay không thì output là yes - cho vay hoặc no - không cho vay. Trọng số liên kết (Connection Weights) là thành phần rất quan trọng của một ANN, nó thể hiện mức độ quan trọng hay có thể hiểu là độ mạnh của dữ liệu đầu vào đối với quá trình xử lý thông tin, chuyển đổi dữ liệu từ layer này sang layer khác. Quá trình học (Learning Processing) của ANN thực ra là quá trình điều chỉnh các trọng số (Weight) của các input data để có được kết quả mong muốn. Hàm tổng (Summation Function) cho phép tính tổng trọng số của tất cả các input được đưa vào mỗi nơ-ron. Hàm tổng của một nơ-ron đối với n input được tính theo công thức sau:

$$Y = \sum_{i=1}^n X_i W_i \quad (2.1)$$

Kết quả trên cho biết khả năng kích hoạt của nơ-ron đó. Các nơ-ron này có thể sinh ra một output hoặc không trong ANN, hay nói cách khác rằng có thể output của 1 nơ-ron có thể được chuyển đến layer tiếp trong mạng nơ-ron hoặc không là do ảnh hưởng bởi hàm chuyển đổi (Transfer Function). Việc lựa chọn Transfer Function có tác động lớn đến kết quả của ANN. Vì kết quả xử lý tại các nơ-ron là hàm tính tổng nên đối khi rất lớn, nên transfer function được sử dụng để

xử lý output này trước khi chuyển đến layer tiếp theo. Hàm chuyển đổi phi tuyến được sử dụng phổ biến trong ANN là sigmoid (logical activation) function.

$$Y_T = \frac{1}{1 + e^{-Y}} \quad (2.2)$$

Kết quả của Sigmoid Function thuộc khoảng $[0, 1]$ nên còn gọi là hàm chuẩn hóa (Normalized Function). Đôi khi thay vì sử dụng hàm chuyển đổi, ta sử dụng giá trị ngưỡng (Threshold value) để kiểm soát các output của các nơ-ron tại một layer nào đó trước khi chuyển các output này đến các layer tiếp theo. Nếu output của một nơ-ron nào đó nhỏ hơn Threshold thì nó sẽ không được chuyển đến Layer tiếp theo. Ứng dụng thực tế của mạng nơ-ron thường được sử dụng trong các bài toán nhận dạng mẫu như nhận dạng chữ cái quang học (Optical character recognition), nhận dạng chữ viết tay, nhận dạng tiếng nói, nhận dạng khuôn mặt.

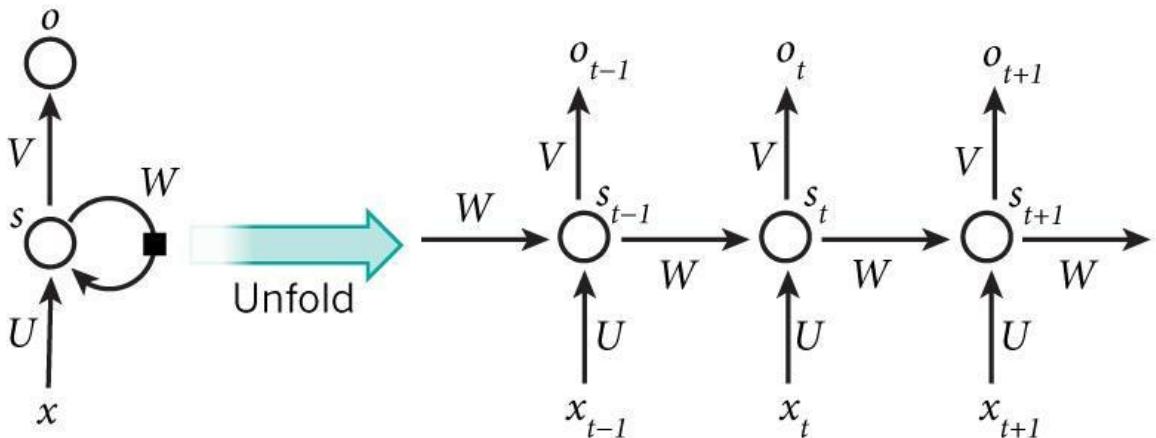
2.2. Mạng nơ-ron hồi quy RNN

Mạng nơ-ron hồi quy (RNN) là một trong những mô hình Deep Learning được đánh giá là có nhiều lợi thế trong các tác vụ xử lý ngôn ngữ tự nhiên (NLP). Ý tưởng của RNN là thiết kế một mạng lưới thần kinh có khả năng xử lý thông tin giống như chuỗi như một câu hỏi. Lặp lại có nghĩa là thực hiện cùng một nhiệm vụ lặp đi lặp lại cho từng yếu tố trong chuỗi. Cụ thể, đầu ra ở thời điểm hiện tại phụ thuộc vào kết quả tính toán của các thành phần ở thời điểm trước. Nói cách khác, RNN là một mô hình có bộ nhớ, có thể nhớ thông tin được tính toán trước đó, không giống như các mạng thần kinh truyền thống là thông tin đầu vào hoàn toàn độc lập với thông tin đầu ra. Về mặt lý thuyết, RNN có thể nhớ thông tin chuỗi có độ dài bất kỳ, nhưng trong thực tế mô hình này chỉ có thể nhớ thông tin từ một vài bước trước đó.

Các ứng dụng của RNN khá đa dạng trong các lĩnh vực như mô hình ngôn ngữ và tạo văn bản (Generating text). Mô hình ngôn ngữ cho chúng ta biết xác suất của câu trong ngôn ngữ là gì. Đây cũng là vấn đề dự đoán xác suất của từ tiếp theo của một câu nhất định. Từ vấn đề này, chúng ta có thể mở rộng sang vấn đề tạo văn bản (generative model/generating text). Mô hình này cho phép chúng tôi tạo văn bản mới dựa trên bộ dữ liệu đào tạo. Ví dụ, khi đào tạo mô hình này với dữ liệu từ văn bản hàng, có thể tạo câu trả lời cho các câu hỏi liên quan đến thương mại điện tử. Tùy thuộc vào loại dữ liệu đào tạo, chúng tôi sẽ có nhiều loại ứng dụng khác nhau. Trong mô hình ngôn ngữ, đầu vào là một chuỗi các từ (được mã

hóa thành one-hot vector [13]), đầu ra là một chuỗi các từ được dự đoán từ mô hình này. Một lĩnh vực khác của RNN là Dịch máy. Vấn đề dịch máy tương tự như mô hình ngôn ngữ. Cụ thể, đầu vào là chuỗi các từ của ngôn ngữ nguồn (ví dụ: tiếng Việt), đầu ra là chuỗi các từ của ngôn ngữ đích (ví dụ: tiếng Anh). Sự khác biệt ở đây là đầu ra chỉ có thể dự đoán được khi đầu vào đã được phân tích hoàn toàn. Điều này là do từ được dịch phải chứa tất cả thông tin từ từ trước đó. Hoặc RNN có thể áp dụng cho các vấn đề toán học được mô tả cho hình ảnh Generating Image Descriptions). RNN kết hợp với Convolution Neural Networks có thể tạo văn bản mô tả cho hình ảnh. Mô hình này hoạt động bằng cách tạo các câu mô tả từ các tính năng được trích xuất trong hình ảnh.

Đào tạo RNN tương tự như đào tạo ANN truyền thống. Giá trị ở mỗi đầu ra không chỉ phụ thuộc vào kết quả tính toán của bước hiện tại mà còn phụ thuộc vào kết quả tính toán của các bước trước đó.



Hình 16: Mạng RNN [15]

RNN có khả năng biểu diễn mối quan hệ phụ thuộc giữa các thành phần trong chuỗi. Ví dụ, nếu chuỗi đầu vào là một câu có 5 từ thì RNN này sẽ unfold (dàn ra) thành RNN có 5 layer, mỗi layer tương ứng với mỗi từ, chỉ số của các từ được đánh từ 0 tới 4. Trong hình vẽ ở trên, x_t là input (one-hot vector) tại thời điểm thứ t. s_t là hidden state (memory) tại thời điểm thứ t, được tính dựa trên các hidden state trước đó kết hợp với input của thời điểm hiện tại với công thức:

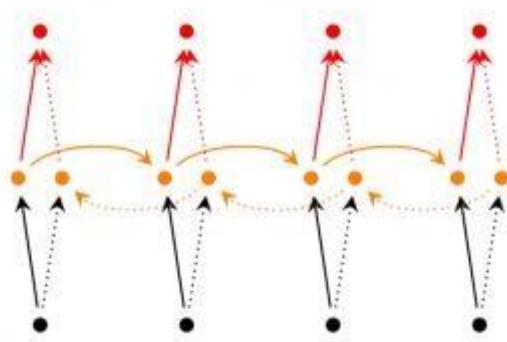
$$s_t = \tanh(U_{x_t} + W_{s_{t-1}}) \quad (2.3)$$

s_{t-1} là hidden state được khởi tạo là một vector không. o_t là output tại thời điểm thứ t, là một vector chứa xác suất của toàn bộ các từ trong từ điển.

$$O_t = \text{softmax}(V_{S_t}) \quad (2.4)$$

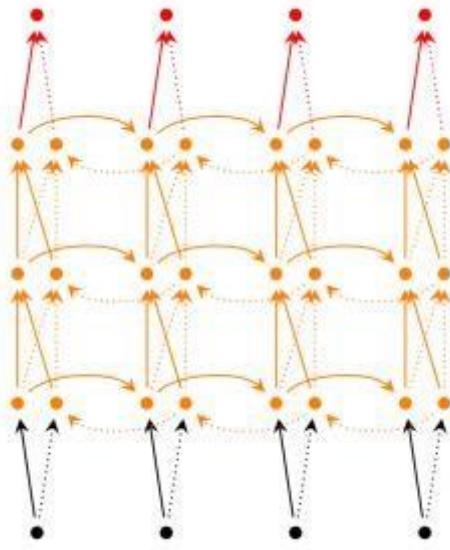
Không giống như ANN truyền thống, ở mỗi lớp cần sử dụng một tham số khác, RNN chỉ sử dụng một bộ tham số (U, V, W) cho tất cả các bước. Về mặt lý thuyết, RNN có thể xử lý và lưu trữ thông tin của một chuỗi dữ liệu có độ dài bất kỳ. Tuy nhiên, trong thực tế, RNN chỉ hiệu quả đối với chuỗi dữ liệu có độ dài không quá dài (short-term memory hay còn gọi là long-term dependency problem). Nguyên nhân của vấn đề này là do vấn đề độ dốc biến mất (gradient được sử dụng để cập nhật giá trị của weight matrix trong RNN và nó có giá trị nhỏ dần theo từng layer khi thực hiện back propagation). Khi độ dốc trở nên rất nhỏ (với giá trị gần bằng 0), giá trị của ma trận trọng số sẽ không được cập nhật nữa và do đó, mạng nơ-ron sẽ ngừng học ở lớp này. Đây cũng là lý do tại sao RNN không thể lưu trữ thông tin của các dấu thời gian đầu tiên trong một chuỗi dữ liệu dài. Trong vài năm qua, các nhà khoa học đã nghiên cứu và phát triển nhiều RNN ngày càng tinh vi để giải quyết các hạn chế của RNN.

Bidirectional RNN (2 chiều): dựa trên ý tưởng rằng đầu ra tại thời điểm t không chỉ phụ thuộc vào các thành phần trước mà còn phụ thuộc vào các thành phần trong tương lai. Ví dụ, để dự đoán một từ còn thiếu trong chuỗi, chúng ta cần xem xét các từ trái và phải xung quanh từ đó. Mô hình này chỉ bao gồm hai RNN chồng chéo. Cụ thể, trạng thái ẩn được tính toán dựa trên cả hai thành phần bên trái và bên phải của mạng.



Hình 17: Mạng RNN 2 chiều [15]

Deep RNN: tương tự như Bidirectional RNN, sự khác biệt là mô hình này bao gồm nhiều lớp RNN hai chiều tại một thời điểm. Mô hình này sẽ cho chúng ta khả năng thực hiện các tính toán nâng cao nhưng yêu cầu đào tạo phải đú lớn.

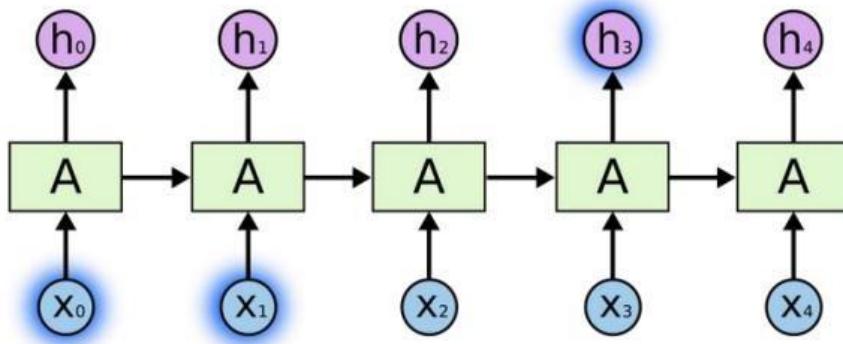


Hình 18: Mạng RNN nhiều tầng [15]

Long short-term memory network (LSTM): mô hình này có cấu trúc giống như mạng RNN nhưng có cách tính khác cho các trạng thái ẩn. Bộ nhớ trong LSTM được gọi là các tế bào. Chúng ta có thể thấy đây là một hộp đen nhận thông tin đầu vào bao gồm cả trạng thái và giá trị ẩn. Trong các hạt nhân này, họ xác định thông tin nào sẽ lưu trữ và thông tin nào cần xóa, để mô hình có thể lưu trữ thông tin dài hạn.

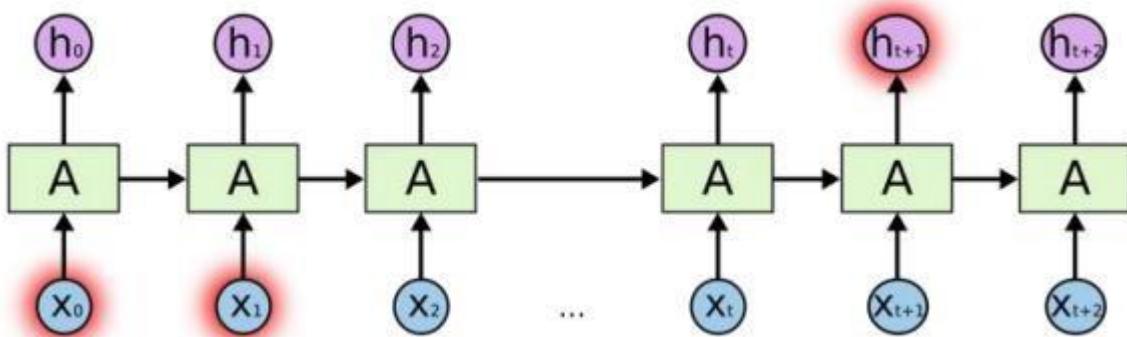
2.2.1 Vấn đề phụ thuộc quá dài

Ý tưởng ban đầu của RNN là liên kết thông tin trước đó để hỗ trợ các quy trình hiện tại. Nhưng đôi khi, chỉ cần dựa vào một số thông tin gần nhất để thực hiện nhiệm vụ hiện tại. Ví dụ, trong mô hình hóa ngôn ngữ, chúng tôi có gắng dự đoán từ tiếp theo dựa trên các từ trước đó. Nếu chúng ta dự đoán từ cuối cùng trong câu "mây bay trên bầu trời", thì chúng ta không cần phải tìm kiếm quá nhiều từ trước đó, chúng ta có thể đoán từ tiếp theo sẽ là "bầu trời". Trong trường hợp này, khoảng cách đến thông tin liên quan được rút ngắn, nặng RNN có thể tìm hiểu và sử dụng thông tin trong quá khứ.



Hình 19: RNN phụ thuộc short-term [17]

Nhưng cũng có những trường hợp chúng ta cần thêm thông tin, đó là phụ thuộc vào ngữ cảnh. Chẳng hạn, khi dự đoán từ cuối cùng trong đoạn "Tôi sinh ra và lớn lên ở Việt Nam ... tôi có thể nói tiếng Việt trôi chảy". Từ thông tin mới nhất cho thấy từ tiếp theo là tên của một ngôn ngữ, nhưng khi chúng ta muốn biết ngôn ngữ cụ thể, chúng ta cần quay lại quá khứ xa hơn, để tìm bối cảnh của Việt Nam. Và do đó, RNN có thể phải tìm thông tin liên quan và số điểm trở nên rất lớn. Thật bất ngờ, RNN không thể học cách kết nối thông tin lại với nhau:



Hình 20: RNN phụ thuộc long-term [17]

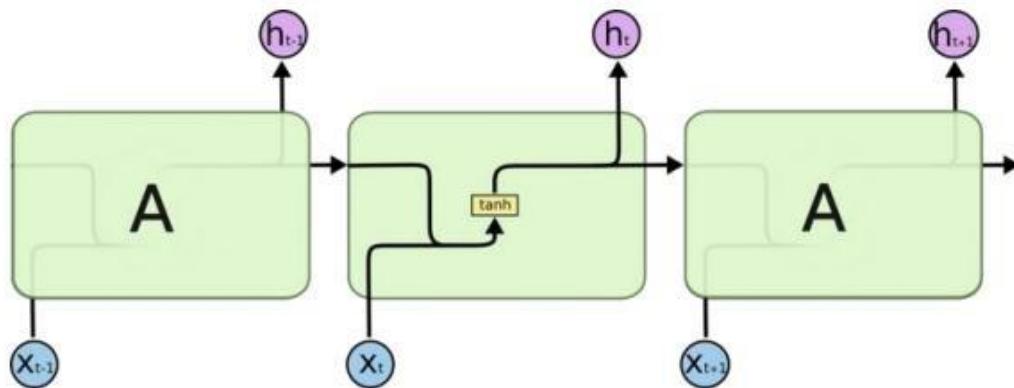
Về mặt lý thuyết, RNN hoàn toàn có khả năng xử lý "các phụ thuộc dài hạn", nghĩa là thông tin hiện tại thu được là do chuỗi thông tin trước đó. Đáng buồn thay, trong thực tế, RNN dường như không có khả năng này. Vấn đề này đã được đặt ra bởi Hochreiter (1991) [German] và và công sự đặt ra một thách thức cho mô hình RNN.

2.2.2 Kiến trúc mạng LSTM

Long Short Term Memory network (LSTM) là trường hợp đặc biệt của RNN, có khả năng học các phụ thuộc dài hạn. Mô hình này được giới thiệu bởi Hochreiter & Schmidhuber (1997), và được cải tiến một lần nữa. Sau đó, mô hình

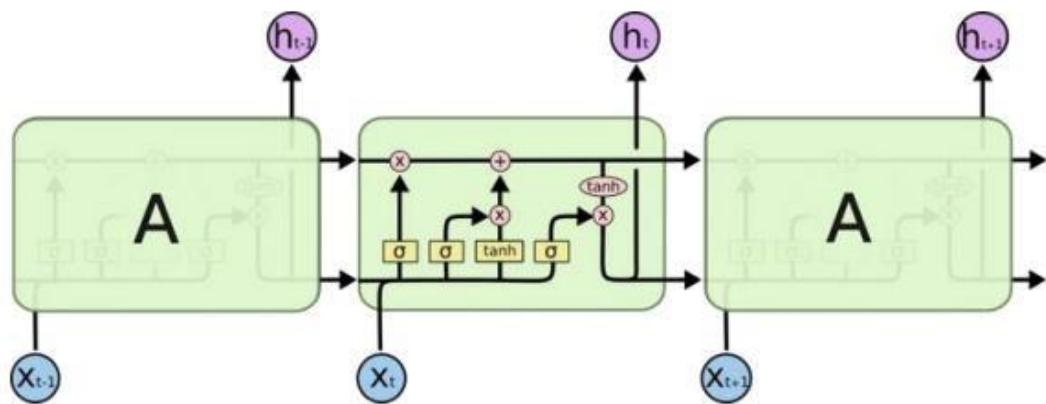
này dần trở nên phổ biến nhờ các công trình nghiên cứu gần đây. Mô hình này tương thích với nhiều vấn đề, vì vậy nó được sử dụng rộng rãi trong các ngành liên quan.

LSTM được thiết kế để loại bỏ các vấn đề phụ thuộc quá dài. Nhìn vào mô hình RNN bên dưới, các lớp được kết nối với nhau thành các mô đun mạng thần kinh. Trong RNN tiêu chuẩn, mô-đun lặp lại này có cấu trúc rất đơn giản bao gồm một lớp đơn giản tanh layer.



Hình 21: Các mô-đun lặp của mạng RNN chứa một layer [17]

LSTM có cùng cấu trúc liên kết, nhưng các mô-đun lặp lại có cấu trúc khác. Thay vì chỉ có một lớp mạng thần kinh, LSTM có tới bốn lớp, tương tác với nhau theo một cấu trúc cụ thể.



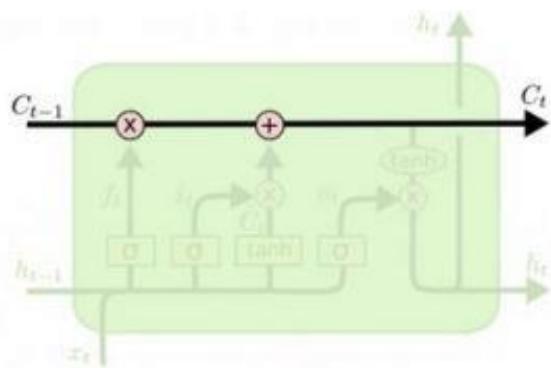
Hình 22: Các mô-đun lặp của mạng LSTM chứa bốn layer [17]

Trong đó, các ký hiệu sử dụng trong mạng LSTM được giải nghĩa như sau:

	Là các lớp ẩn của mạng nơ-ron
	Toán tử Pointwise, biểu diễn các phép toán như cộng, nhân vector
	Vector chỉ đầu vào và đầu ra của một nút
	Biểu thị phép nối các toán hạng
	Biểu thị cho sự sao chép từ vị trí này sang vị trí khác

2.2.3 Phân tích mô hình LSTM

Máu chốt của LSTM là trạng thái ô, đường ngang chạy dọc theo đỉnh của sơ đồ. Trạng thái té bào giống như một băng chuyền. Nó chạy thẳng qua toàn bộ chuỗi, chỉ một vài tương tác tuyến tính nhỏ được thực hiện. Điều này làm cho thông tin ít có khả năng thay đổi trong suốt quá trình lan truyền.

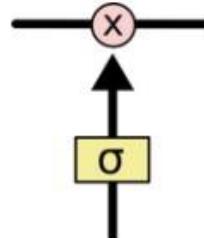


Hình 23: Tế bào trạng thái LSTM giống như một băng truyền [17]

LSTM có khả năng thêm hoặc bớt thông tin vào cell state, được quy định một cách cẩn thận bởi các cấu trúc gọi là cổng (gate). Các cổng này là một cách

(tùy chọn) để định nghĩa thông tin băng qua. Chúng được tạo bởi hàm sigmoid và một toán tử nhân pointwise.

LSTM có khả năng thêm hoặc xóa thông tin về trạng thái tế bào, được điều chỉnh cẩn thận bởi các cấu trúc gọi là cổng (gate). Các cổng này là một cách (tùy chọn) để xác định thông tin truyền qua. Chúng được tạo bởi hàm sigmoid và toán tử pointwise.

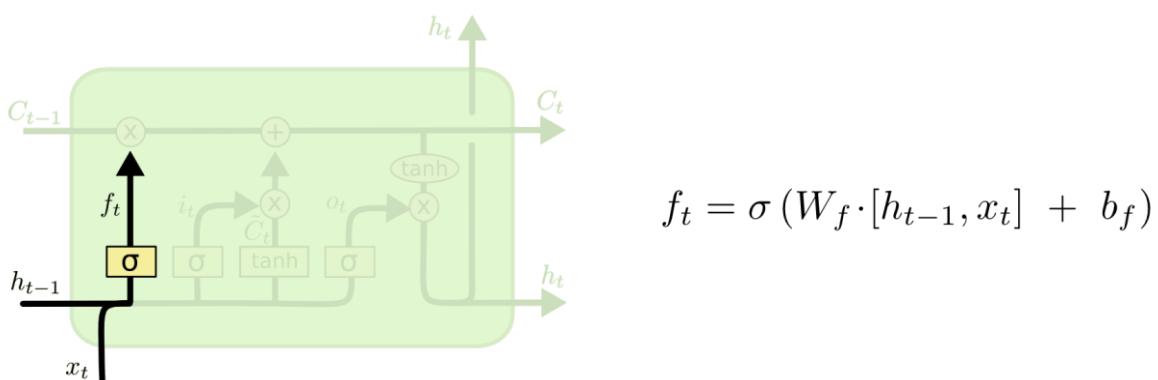


Hình 24: Cổng trạng thái LSTM [17]

Hàm kích hoạt Sigmoid có giá trị từ $[0 - 1]$, mô tả độ lớn thông tin được phép truyền qua tại mỗi lớp mạng. Nếu ta thu được 0 điều này có nghĩa là “không cho bất kỳ cái gì đi qua”, ngược lại nếu thu được giá trị là 1 thì có nghĩa là “cho phép mọi thứ đi qua”. Một LSTM có ba cổng như vậy để bảo vệ và điều khiển cell state.

Quá trình hoạt động của LSTM được thông qua các bước cơ bản sau:

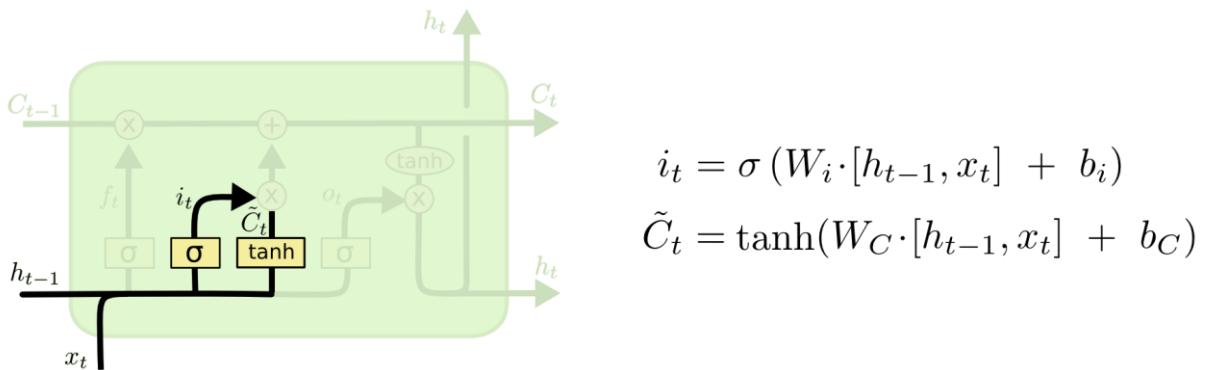
Bước đầu tiên trong mô hình LSTM là quyết định thông tin nào chúng ta cần xóa khỏi trạng thái tế bào. Quá trình này được thực hiện thông qua một lớp sigmoid gọi là "lớp cổng chẵn" - cổng chẵn. Đầu vào là h_{t-1} và x_t , đầu ra là một giá trị trong phạm vi $[0, 1]$ cho trạng thái ô C_{t-1} . 1 tương đương với "lưu giữ thông tin", 0 tương đương với "xóa thông tin"



Hình 25: LSTM focus f [17]

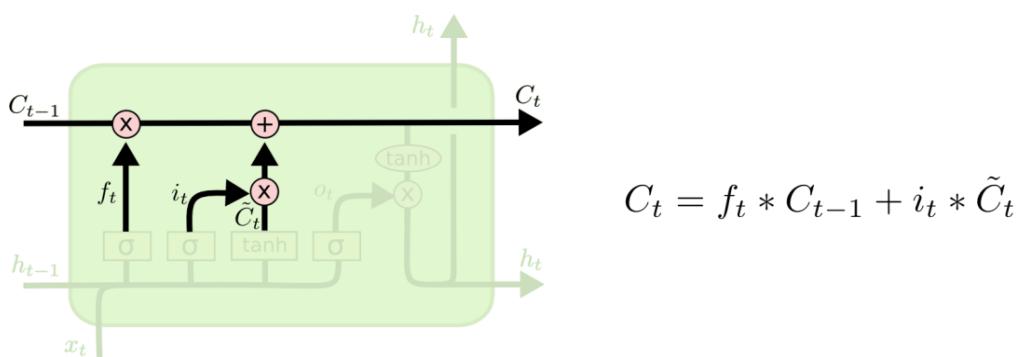
Bước tiếp theo, ta cần quyết định thông tin nào cần được lưu lại tại cell state. Ta có hai phần. Một, single sigmoid layer được gọi là “input gate layer” quyết định các giá trị chúng ta sẽ cập nhật. Tiếp theo, một *tanh* layer tạo ra một vector ứng viên mới, C_t được thêm vào trong ô trạng thái.

Trong bước tiếp theo, chúng ta cần quyết định thông tin nào cần được lưu trữ trong trạng thái tế bào. Chúng tôi có hai phần. Một, lớp sigmoid duy nhất được gọi là "lớp cổng đầu vào" xác định các giá trị chúng tôi sẽ cập nhật. Tiếp theo, một lớp *creates* tạo ra một vector ứng cử viên mới, C_t được thêm vào trong hộp trạng thái.



Hình 26: LSTM focus I [17]

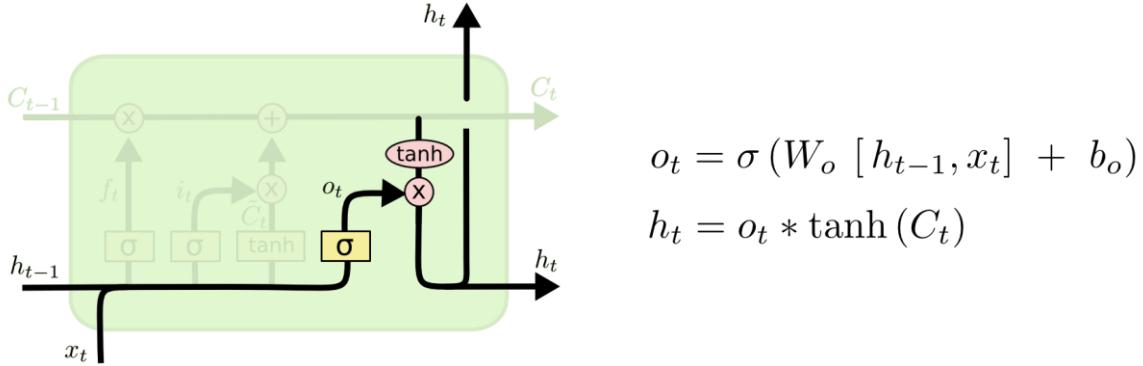
Ở bước tiếp theo, ta sẽ kết hợp hai thành phần này lại để cập nhật vào cell state. Lúc cập nhật vào cell state cũ, C_{t-1} , vào cell state mới C_t . Ta sẽ đưa state cũ hàm ff , để quên đi những gì trước đó. Sau đó, ta sẽ thêm $i_t * \tilde{C}_t$. Đây là giá trị ứng viên mới, co giãn (scale) số lượng giá trị mà ta muốn cập nhật cho mỗi state.



Hình 27: LSTM focus c [17]

Cuối cùng, ta cần quyết định xem thông tin output là gì. Output này cần dựa trên cell state của chúng ta, nhưng sẽ được lọc bớt thông tin. Đầu tiên, ta sẽ áp dụng single sigmoid layer để quyết định xem phần nào của cell state chúng ta

dự định sẽ output. Sau đó, ta sẽ đẩy cell state qua tanh giá trị khoảng [-1 và 1] và nhân với một output sigmoid gate, để giữ lại những phần ta muốn output ra ngoài.



Hình 28: LSTM focus o [17]

Ví dụ về một mô hình ngôn ngữ, chỉ cần nhìn vào chủ đề có thể cung cấp thông tin về một trạng từ sau. Ví dụ: nếu đầu ra của chủ ngữ là số ít hoặc số nhiều thì chúng ta có thể biết dạng trạng từ sau nó là gì.

2.3. Word embeddings

Biểu diễn ngôn ngữ hoặc vector hóa các từ là một thành phần quan trọng để giúp máy tính có thể hiểu được ngôn ngữ từ văn bản sang dạng số. Đó là đưa văn bản dạng text vào một không gian mới được gọi là embedding space (không gian từ nhúng).

Word embeddings (tập nhúng từ) là một phương pháp ánh xạ mỗi từ vào một không gian thực đa chiều nhưng nhỏ hơn nhiều so với kích thước từ điển. Word embedding có 2 model nổi tiếng là **word2vec** và **Glove** [16].

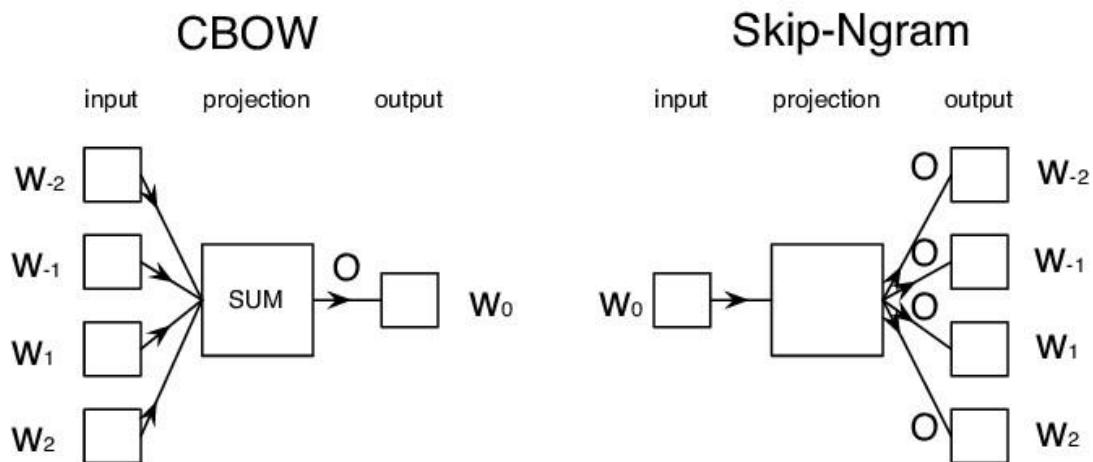
2.3.1 Word2vec

Word2vec được tạo ra vào năm 2013 bởi một kỹ sư tại google có tên là Tomas Mikolov. Về mặt toán học, Word2Vec về cơ bản là ánh xạ từ từ một tập hợp các từ vựng vào không gian vector, mỗi từ được biểu thị bằng n số thực. Mỗi từ tương ứng với một vector cố định. Sau khi đào tạo mô hình bằng thuật toán backpropagation, trọng lượng của mỗi vector từ được cập nhật liên tục. Từ đó, chúng ta có thể thực hiện một phép tính khoảng cách giữa các từ và các từ "gần" nhau thường là các từ xuất hiện cùng nhau trong ngữ cảnh, từ đồng nghĩa.

giận_hờn
 câu_giận
 giận_dữ
 tức_giận
 giận

Hình 29: Mô hình từ nhúng [16]

Word2vec có hai word vector là skip-gram và Continuous Bag-of-Words (Cbow).



Hình 30: Mô hình CBOW và Skip-Ngram [16]

CBoW: Dự đoán từ hiện tại dựa trên ngữ cảnh của các từ trước đó

- ✓ Cho các từ ngữ cảnh
- ✓ Đoán xác suất của một từ đích

Skip-gram: Dự đoán các từ xung quanh khi cho bởi từ hiện tại.

- ✓ Cho từ đích
- ✓ Đoán xác suất của các từ ngữ cảnh

2.3.2 Glove

GloVe [16] là một trong những phương pháp mới để xây dựng vectơ từ (giới thiệu vào năm 2014), nó được xây dựng dựa trên ma trận đồng xảy ra (Co-occurrence Matrix). GloVe dựa trên ý tưởng tính tỉ lệ xác xuất:

$$\frac{P(k|i)}{P(k|j)} \quad (2.5)$$

Hình 31: Xác xuất từ k trên ngữ cảnh của từ i và j [16]

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

$$P(k|i) = \frac{X_{ik}}{X_i} = \frac{X_{ik}}{X_{im}} \quad (2.6)$$

Hình 32: Công thức tính xác xuất từ k trên ngữ cảnh của từ i [16]

X_{ik} : là số lần xuất hiện của từ k trong ngữ cảnh của từ i (và ngược lại).

X_i : là số lần xuất hiện của từ i trong ngữ cảnh của toàn bộ các từ còn lại ngoại trừ i.

Ý tưởng chính của GloVe được tính toán dựa trên sự giống nhau về ngữ nghĩa giữa hai từ i, j được xác định thông qua sự tương đồng về ngữ nghĩa giữa các từ k và mỗi từ i, j, các từ k có định nghĩa ngữ nghĩa tốt. là những từ khiến giá trị được tính từ công thức (2.5) nằm trong phạm vi $[0, 1]$

Ví dụ: nếu từ i là "table", từ j là "cat" và từ k là "chair" thì công thức (2.5) sẽ cho giá trị tiệm cận là 1 vì "chair" có nghĩa gần với "table" hơn. thay vì "cat", trong các trường hợp khác, nếu chúng ta thay thế thuật ngữ k bằng "ice cream" thì giá trị của công thức (1) sẽ xấp xỉ bằng 0 vì "ice cream" gần như không liên quan gì đến "table" và "cat".

Dựa trên tầm quan trọng của công thức (2.5), GloVe bắt đầu bằng việc nó sẽ tìm thấy hàm F để nó ánh xạ từ các vectơ trong không gian V sang giá trị tỷ lệ được tính theo công thức (2.5). Việc tìm F không đơn giản, tuy nhiên, sau nhiều bước đơn giản hóa và tối ưu hóa, chúng ta có thể đưa nó trở lại vấn đề hồi quy với việc tính toán hàm chi phí tối thiểu (hàm chi phí tối thiểu) như sau:

$$J = \sum_{i,j=1}^V f(X_{ij})(W_i^T \tilde{W}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \quad (2.7)$$

Hình 33: Công thức tính hàm chi phí tối thiểu [16]

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 xảy ra

Hàm $f(x)$ được gọi là hàm trọng số (weighting function), được thêm vào để giảm bớt sự ảnh hưởng của các cặp từ xuất hiện quá thường xuyên, hàm này thỏa mãn 3 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:

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{if } x < x_{max} \\ 1 & \text{otherwise} \end{cases} \quad (2.8)$$

Với $\alpha = 3/4$

Hình 34: Hàm trọng số (weighting function) [16]

Việc thực hiện tính hàm chi phí tối thiểu J để tìm ra các vec-tor từ W_i, W_j thể được thực hiện bằng nhiều cách, trong đó cách tiêu chuẩn nhất là sử dụng tìm cực tiểu hàm số theo thuật toán Gradient Descent.

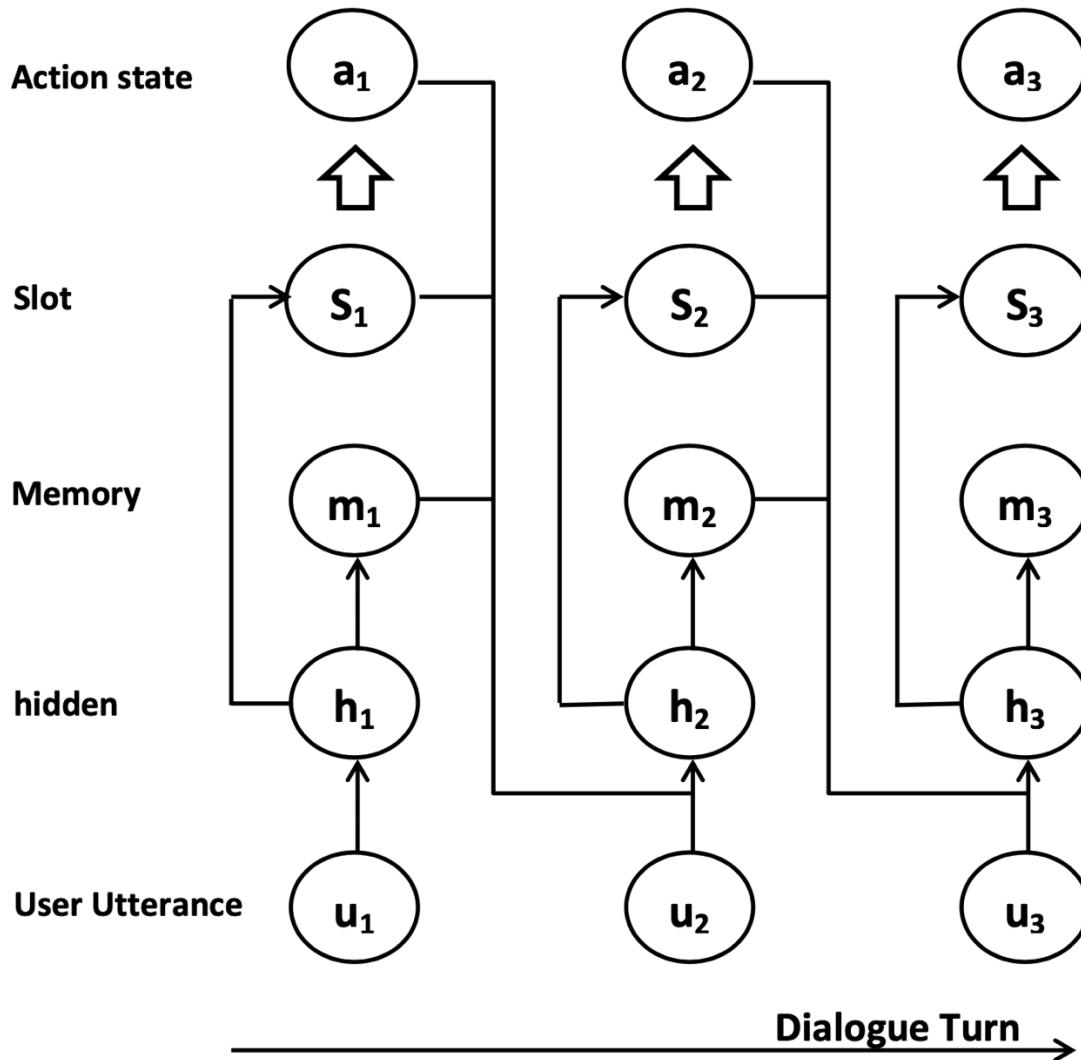
2.4. Ứng dụng RNN vào quản lý hội thoại

Phần này sẽ giới thiệu thuật toán RNN áp dụng cho chức năng theo dõi trạng thái hay ngữ cảnh hội thoại **Dialogue State Tracking (DTS)** trong thành phần quản lý hội thoại đã được mô tả chi tiết trong Hình 7 như đã đề cập.

2.4.1 Mô hình word-based DST

Như đã biết, thành phần DM có nhiệm vụ quan trọng nhất là quản lý các trạng thái hoặc ngữ cảnh của cuộc hội thoại để xác định hành động tiếp theo. Vì vậy, quyết định của hành động tiếp theo là dựa trên đâu. Đó là đầu vào của người dùng, dữ liệu vị trí được lưu trữ trong bộ nhớ và trạng thái của các hành động trước đó. Với khả năng lưu trữ thông tin trong quá trình xử lý các vấn đề về chuỗi,

RNN được sử dụng để xác định bối cảnh và xác định hành động tiếp theo nhờ thông tin được lưu trữ trong bộ nhớ RNN.



Hình 35: Mô hình word-based DST với mạng RNN [20]

User Utterance: là câu dữ liệu đầu vào của người dùng.

Hidden: là lớp ẩn được sử dụng trong thành phần NLU để vector hóa ngôn ngữ, phân loại ý định và trích xuất được các thông tin người dùng.

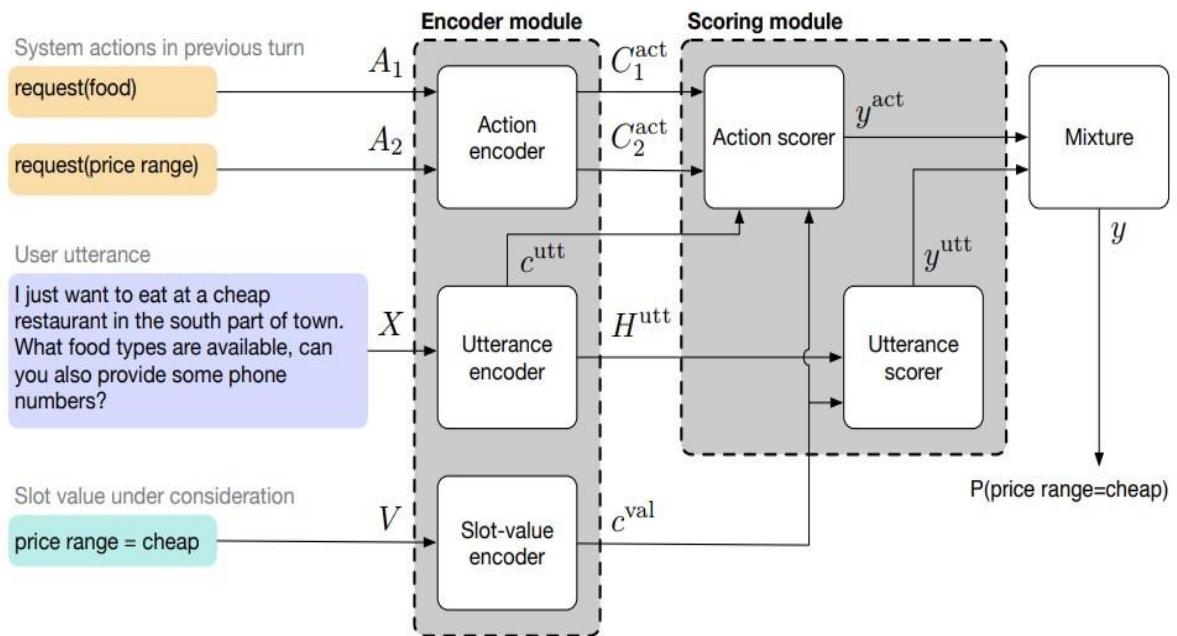
Memory: là bộ nhớ lưu các giá trị vector hóa ngôn ngữ và ngữ cảnh hội thoại bao gồm cả slot.

Slot: thông tin được trích xuất được lưu lại trong các câu nói người dùng

Action State: trạng thái action trước. Nó mang tính ngữ cảnh ở trong đoạn hội thoại.

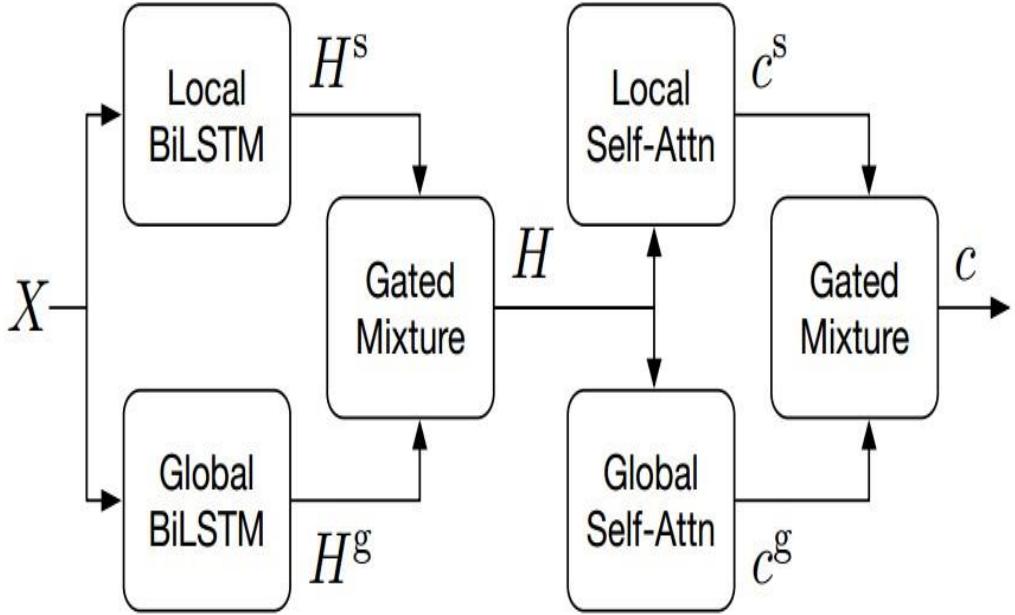
2.4.2 Mô hình Global-Locally Self-Attentive DST (GLAD)

Với các mô hình theo dõi trạng thái bắt thường của việc xác định mục đích của người dùng trực tiếp trong danh sách các cuộc hội thoại đã được đào tạo cho bot, sẽ có những vấn đề mơ hồ xác định ý định trong các cuộc hội thoại sử dụng tương tự 1 ý định. Đó là, với cùng một tuyên bố rằng người dùng xác định một mục đích, ý định đó có thể được sử dụng trong nhiều cuộc hội thoại. Với ý tưởng xác định bối cảnh cho mỗi cuộc trò chuyện, việc xác định ý định nằm trong cuộc trò chuyện sẽ tăng độ chính xác.



Hình 36: Mô hình Global-Locally Self-Attentive DST (GLAD) [21]

Các tham số đầu vào cho phần local và global là các giá trị intent người dùng (X), các slot và action state (C)



Hình 37: Global-locally self-attentive encoder modul [21]

Việc dùng các slot trong mô hình cần lưu ý là sau kết thúc mỗi hội thoại thì nên xóa các slot không dùng nữa để tránh nhập nhằng xác định ví trí chính xác intent của người dùng trong danh sách các đoạn hội thoại mà bot được đào tạo.

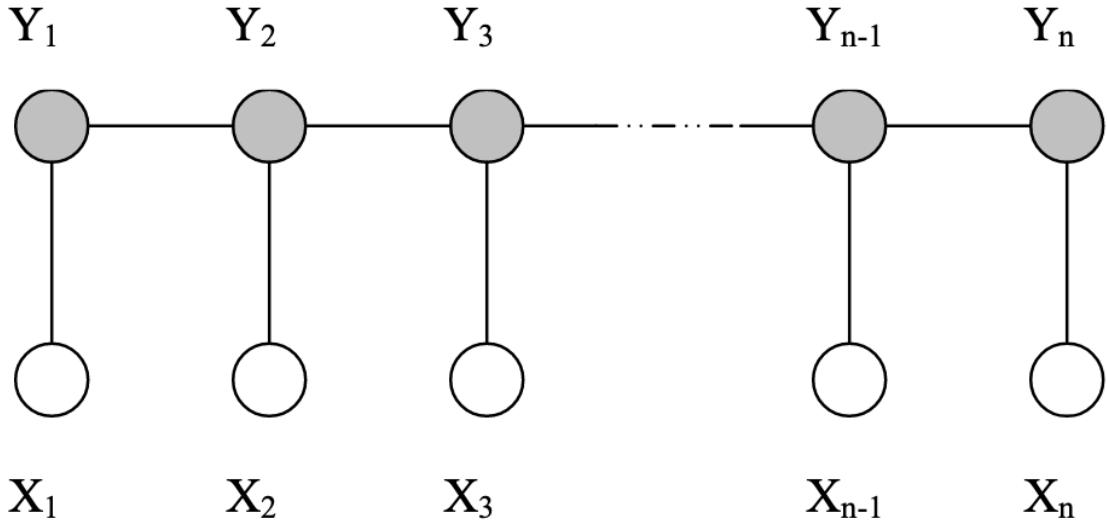
2.5 Mô hình CRF

2.5.1 Định nghĩa CRF

Kí hiệu X là biến ngẫu nhiên có tương ứng với chuỗi dữ liệu cần gán nhãn và Y là biến ngẫu nhiên tương ứng với chuỗi nhãn. Mỗi thành phần Y_i của Y là một biến ngẫu nhiên nhận giá trị trong tập hữu hạn các trạng thái S . Ví dụ trong bài toán phân đoạn từ, X nhận giá trị là các câu trong ngôn ngữ tự nhiên, còn Y là chuỗi nhãn tương ứng với các câu này. Mỗi thành phần Y_i của Y là một nhãn xác định phạm vi của một từ trong câu (bắt đầu một từ, ở trong một từ và kết thúc một từ).

Cho một đồ thị vô hướng không có chu trình $G = (V, E)$, trong đó E là tập các cạnh vô hướng của đồ thị, V là tập các đỉnh của đồ thị sao cho $Y = \{Y_v | v \in V\}$. Nói cách khác là tồn tại ánh xạ một – một giữa một đỉnh đồ thị và một thành phần Y_v của Y . Nếu mỗi biến ngẫu nhiên Y_v tuân theo tính chất Markov đối với đồ thị G – tức là xác suất của biến ngẫu nhiên Y_v cho bởi X và tất cả các biến ngẫu nhiên khác $Y_u | u \neq v, \{u, v\} \in V\}$: $p(Y_v | X, Y_u, u \neq v, \{u, v\} \in V)$ bằng xác suất của biến ngẫu nhiên Y_v cho bởi X và các biến ngẫu nhiên khác tương ứng với các

đỉnh kè với đỉnh v trong đồ thị: $p(Y_v|X, Y_u, (u, v) \in E)$, thì ta gọi (X, Y) là một trường ngẫu nhiên điều kiện (Conditional Random Field). Như vậy, một CRF là một trường ngẫu nhiên phụ thuộc toàn cục vào chuỗi quan sát X. Trong bài toán phân đoạn từ nói riêng và các bài toán xử lý dữ liệu dạng chuỗi nói chung, thì đồ thị G đơn giản chỉ là dạng chuỗi $V = \{1, 2, \dots, m\}$, $E = \{(i, i + 1)\}$ Kí hiệu $X = (X_1, X_2, \dots, X_n)$ và $Y = (Y_1, Y_2, \dots, Y_n)$ thì mô hình đồ thị G có dạng sau



Hình 38: Đồ thị vô hướng mô tả CRF

Gọi C là tập các đồ thị con đầy đủ của G. Vì G có dạng chuỗi nên đồ thị con đầy đủ thực ra chỉ là một đỉnh hoặc một cạnh của đồ thị G. Áp dụng kết quả của Hammerley - Clifford[27] cho các trường ngẫu nhiên Markov thì phân phối của chuỗi nhãn Y với chuỗi quan sát X cho trước có dạng:

$$P(y|x) = \prod_{A \in C} \psi_A (A | x)$$

Trong đó Ψ_A gọi là hàm tiềm năng, nhận giá trị thực dương. Lafferty xác định hàm tiềm năng này dựa trên nguyên lý cực đại entropy. Việc xác định một phân phối theo nguyên lý cực đại entropy có thể hiểu là ta phải xác định một phân phối sao cho “phân phối đó tuân theo mọi giải thích suy ra từ thực nghiệm, ngoài ra không đưa thêm bất kì giả thiết nào khác” và gần nhất với phân phối đều.

Entropy là độ đo thể hiện tính không chắc chắn, hay độ không đồng đều của phân phối xác suất. Độ đo entropy điều kiện $H(Y|X)$ được cho bởi công thức

$$H(Y|X) = - \sum_{x,y} \tilde{p}(x,y) \log q(y|x) \quad (2.9)$$

Với $\tilde{p}(x, y)$ là phân phối thực nghiệm của dữ liệu.

Theo cách trên, Lafferty đã chỉ ra hàm tiềm năng của mô hình CRFs có dạng

$$\psi_A(A|x) = \exp \sum_k \lambda_k f_k(A|x) \quad (2.10)$$

Trong đó λ_k là thừa số lagrangian ứng với thuộc tính f_k . Ta cũng có thể xem như λ_k là trọng số xác định độ quan trọng của thuộc tính f_k trong chuỗi dữ liệu. Có hai loại thuộc tính là thuộc tính chuyển (kí hiệu là f) và thuộc tính trạng thái (kí hiệu là g) tùy thuộc vào A là một đỉnh hay một cạnh của đồ thị. Thay công thức hàm tiềm năng vào công thức (3.1) và thêm thừa số chuẩn hóa để đảm bảo thỏa mãn điều kiện xác suất ta được

$$p(y|x) = \frac{1}{Z(x)} \exp \left(\sum_i \sum_k \lambda_k f_k(y_{i-1}, y_i, x) + \sum_i \sum_k \mu_k g_k(y_i, x) \right) \quad (2.11)$$

Ở đây, x là chuỗi dữ liệu, y là chuỗi trạng thái tương ứng. $f_k(y_{i-1}, y_i, x)$ là thuộc tính của chuỗi quan sát ứng và các trạng thái ứng với vị trí thứ i và $i-1$ trong chuỗi trạng thái $g_k(y_i, x)$ là thuộc tính của chuỗi quan sát và trạng thái ứng với vị trí i trong chuỗi trạng thái.

Các thuộc tính này được rút ra từ tập dữ liệu và có giá trị cố định. Ví dụ:

$$f_i = \begin{cases} 1 & \text{nếu } x_{i-1} = \text{"Học"}, x_i = \text{"sinh"} \text{ và } y_{i-1} = \text{"B_W"}, y_i = \text{"I_W"} \\ 0 & \text{nếu ngược lại} \end{cases} \quad (2.12)$$

$$g_i = \begin{cases} 1 & \text{nếu } x_{i-1} = \text{"Học"} \text{ và } y_i = \text{"B_W"} \\ 0 & \text{nếu ngược lại} \end{cases} \quad (2.13)$$

Vấn đề của ta bây giờ là phải ước lượng được các tham số $(\lambda_1, \lambda_2, K; \mu_1, \mu_2, K)$ từ tập dữ liệu huấn luyện.

2.5.2 Huấn luyện CRF

Việc huấn luyện mô hình CRF thực chất là đi tìm tập tham số của mô hình. Kỹ thuật được sử dụng là làm cực đại độ đo likelihood giữa phân phối mô hình và phân phối thực nghiệm. Vì thế việc huấn luyện mô hình CRFs trở thành bài toán tìm cực đại của hàm logarit của hàm likelihood.

Giả sử dữ liệu huấn luyện gồm một tập N cặp, mỗi cặp gồm một chuỗi quan sát và một chuỗi trạng thái tương ứng, $D = \{(x^{(i)}, y^{(i)})\} \forall i = 1, \dots, N$. Hàm log-likelihood có dạng sau:

$$l(\theta) = \sum_{x,y} \tilde{p}(x,y) \log(p(y|x, \theta)) \quad (2.14)$$

Ở đây $\Theta(\lambda_1, \lambda_2, \dots, \mu_1, \mu_2, \dots)$ là các tham số của mô hình và $\tilde{p}(x, y)$ là phân phối thực nghiệm đồng thời của x, y trong tập huấn luyện.

Thay $p(y|x)$ của CRFs trong công thức trên ta được:

$$l(\theta) = \sum_{x,y} \tilde{p}(x,y) \left[\sum_{i=1}^{n+1} \lambda f_i + \sum_{i=1}^n \mu g_i \right] - \sum_x \tilde{p}(x) \log Z \quad (2.15)$$

Ở đây, $\lambda(\lambda_1, \lambda_2, \dots, \lambda_n)$ và $\mu(\mu_1, \mu_2, \dots, \mu_m)$ là các vector tham số của mô hình, \mathbf{f} là vector các thuộc tính chuyên, \mathbf{g} là vector các thuộc tính trạng thái.

Người ta đã chứng minh được hàm log-likelihood là một hàm lõm và liên tục trong toàn bộ không gian của tham số. Vì vậy ta có thể tìm cực đại hàm log-likelihood bằng phương pháp vector gradient. Mỗi thành phần trong vector gradient sẽ được gán bằng 0:

$$\frac{\partial l(\theta)}{\partial \lambda_k} = E_{\tilde{p}(x,y)}[f_k] - E_{p(y|x,\theta)}[f_k] \quad (2.16)$$

Việc thiết lập phương trình trên bằng 0 tương đương với việc đưa ra ràng buộc với mô hình là : giá trị kì vọng của thuộc tính f_k đối với phân phối mô hình phải bằng giá trị kì vọng của thuộc tính f_k đối với phân phối thực nghiệm.

Hiện nay có khá nhiều phương pháp để giải quyết bài toán cực đại hàm loglikelihood, ví dụ như các phương pháp lặp (IIS và GIS), các phương pháp tối ưu số (Conjugate Gradient, phương pháp Newton...). Theo đánh giá của Malouf (2002) thì phương pháp được coi là hiệu quả nhất hiện nay đó là phương pháp tối ưu số bậc hai L-BFGS (limited memory BFGS)

Dưới đây em xin trình bày tư tưởng chính của phương pháp L-BFGS dùng để ước lượng tham số cho mô hình CRFs

L-BFGS là phương pháp tối ưu số bậc hai, ngoài tính toán giá trị của vector gradient, L-BFGS còn xem xét đếm yếu tố về đường cong hàm log-likelihood.

Theo công thức khai triển Taylor tới bậc hai của $l(\theta + \Delta)$ ta có:

$$l(\theta + \Delta) \approx l(\theta) + \Delta^T G(\theta) + \frac{1}{2} \Delta^T H(\theta) \quad (2.17)$$

Trong đó $G(\theta)$ là vector gradient còn $H(\theta)$ là đạo hàm bậc hai của hàm loglikelihood, gọi là ma trận Hessian. Thiết lập đạo hàm của xấp xỉ trong công thức trên bằng 0 ta tìm được giá số để cập nhật tham số mô hình như sau:

$$\Delta(k) = H^{-1}(\theta(k))G(\theta(k)) \quad (2.18)$$

Ở đây, k là chỉ số bước lặp. Ma trận Hessian thường có kích thước rất lớn, đặc biệt với bài toán ước lượng tham số của mô hình CRFs, vì vậy việc tính trực tiếp nghịch đảo của nó là không thực tế. Phương pháp L-BFGS thay vì tính toán trực tiếp với ma trận Hessian nó chỉ tính toán sự thay đổi độ cong của vector gradient so với bước trước đó và cập nhật lại. Công thức trên có thể viết lại là:

$$\Delta(k) = B^{-1}(\theta(k))G(\theta(k)) \quad (2.19)$$

Trong đó ma trận $B^{-1}(\theta(k))$ phản ánh sự thay đổi độ cong qua từng bước lặp của thuật toán. Yếu tố “giới hạn bộ nhớ” (limited memory) của thuật toán thể hiện ở mỗi bước lặp, các tham số dùng để tính toán $B^{-1}(\theta(k))$ sẽ được lưu riêng biệt nhau và khi bộ nhớ bị sử dụng hết thì những tham số cũ sẽ được xóa đi để thay vào đó là các tham số mới.

Việc xấp xỉ ma trận Hessian theo $B(\theta)$ cho phép phương pháp L-BFGS hội tụ nhanh dù lượng dữ liệu rất lớn. Những thực nghiệm gần đây đã chứng minh rằng phương pháp L-BFGS đạt kết quả vượt trội so với các phương pháp khác.

2.5.3 Suy diễn CRF

Sau khi tìm được mô hình CRFs từ tập dữ liệu huấn luyện, nhiệm vụ của ta lúc này là làm sao dựa vào mô hình đó để gán nhãn cho chuỗi dữ liệu quan sát, điều này tương đương với việc làm cực đại phân phối xác suất giữa chuỗi trạng thái y và dữ liệu quan sát x . Chuỗi trạng thái y^* mô tả tốt nhất chuỗi dữ liệu quan sát x sẽ là nghiệm của phương trình:

$$y^* = \operatorname{argmax}\{ p(y|x) \} \quad (2.20)$$

Chuỗi y^* có thể xác định được bằng thuật toán Viterbi. Gọi S là tập tất cả trạng thái có thể, ta có $S = m$. Xét một tập hợp các ma trận cỡ $m \times m$ kí hiệu $\{ M_i(x) \mid i=0,1,\dots,n-1 \}$ được định nghĩa trên từng cặp trạng thái $y, y' \in S$ như sau:

$$M_i(y', y|x) = \exp\left(\sum_k \lambda_k f_k(y', y_i, x) + \sum_k \mu_k g_k(y, x)\right) \quad (2.21)$$

Bằng việc đưa thêm hai trạng thái y_{i-1} và y_n vào trước và sau chuỗi trạng thái. Coi như chúng ứng với trạng thái “start” và “end”, phân phối xác suất có thể viết là:

$$p(y|x, \lambda) = \frac{1}{Z(x)} \prod_{i=0}^n M_i(y', y|x) \quad (2.22)$$

Ở đây $Z(x)$ là thừa số chuẩn hóa được đưa thêm vào và có thể tính được dựa vào các M_i , nhưng vấn đề ta quan tâm là cực đại hóa $p(y|x)$ nên không cần thiết phải tính $Z(x)$. Như vậy ta chỉ cần cực đại hóa tích $n+1$ phần tử trên. Tư tưởng chính của thuật toán Viterbi là tăng dần chuỗi trạng thái tối ưu bằng việc quét các ma trận từ vị trí 0 cho đến vị trí n. Tại mỗi bước i ghi lại tất cả các chuỗi tối ưu kết thúc bởi trạng thái y với $\forall y \in S$ (ta kí hiệu là $y_i^*(y)$) và tích tương ứng $P_i(y)$:

Bước 1: $P_0(y) = M_0(\text{start}, y|x)$ và $y_0^*(y) = y$

Bước lặp: Cho i chạy từ 1 đến n tính: $P_i(y) = \max_{y' \in S} P_{i-1}(y') \times M_i(y', y|x)$ $y_i^*(y) = y_{i-1}^*(y').(y)$, trong đó $y^* = \operatorname{argmax}_{y' \in S} P_{i-1}(y') \times M_i(y', y|x)$ và “.” là toán tử

Bước lặp: Cho i chạy từ 1 đến n tính:

$P_i(y) = \max_{y' \in S} P_{i-1}(y') \times M_i(y', y|x) y_i^*(y) = y_{i-1}^*(\hat{y}).(y)$ cộng chuỗi. Chuỗi $y_{i-1}^*(y)$ chính là chuỗi có xác suất $p(y^*|x)$ lớn nhất, đó cũng chính là chuỗi nhẫn phù hợp nhất với chuỗi dữ liệu quan sát x cho trước.

2.6 Giải thuật phân loại văn bản Starspace

Mô hình StarSpace bao gồm việc học các thực thể. Mỗi thực thể được mô tả bằng một tập hợp các tính năng riêng biệt. Mục tiêu là học ma trận có kích thước $D \times d$, trong đó D là số lượng các đặc trưng và d là chiều dài của vectơ embedding. Một thực thể a được biểu diễn dưới dạng $\sum_{i \in \alpha} F_i$, trong đó F_i là hàng thứ i (có kích thước d) trong ma trận embedding.

Hàm loss sau sẽ được cực tiểu hóa trong quá trình huấn luyện:

$$\sum_{\substack{(a,b) \in E^+ \\ b^- \in E^-}} L^{batch}(\text{sim}(a, b), \text{sim}(a, b), \dots, \text{sim}(a, b_k^-)) \quad (2.23)$$

Trong đó, việc tạo ra các cặp thực thể dương (a, b) thuộc E^+ và thực thể âm b^- thuộc E^- (phương pháp lấy mẫu k-âm (tương tự như trong word2vec) được sử dụng để lấy mẫu cho b_i^-) phụ thuộc vào ứng dụng cụ thể của mô hình

Hàm $sim(\cdot, \cdot)$ là hàm tương tự, trong mô hình được đề xuất, triển khai cả hai phương pháp tính tương tự là cosine (cosine similarity) và tích trong (inner product), sau đó, để mô hình tự lựa chọn phương pháp phù hợp trong quá trình huấn luyện. Thông thường, các phương pháp này đều hoạt động tốt đối với số lượng nhãn nhỏ, tuy nhiên đối với tập nhãn kích thước lớn, hàm cosine cho kết quả tốt hơn.

Hàm loss L^{batch} sẽ so sánh cặp thực thể dương (a, b) với các cặp thực thể âm (a, b_i^-) với $i=1, \dots, k$. Quá trình huấn luyện được tối ưu hóa dựa vào giải thuật Stochastic gradient descent (SGD). Sau khi huấn luyện xong, hàm $sim(\cdot, \cdot)$ sẽ được sử dụng. Ví dụ trong các bài toán phân loại, nhãn b cho thực thể a sẽ được tính bằng $\max_{\hat{b}} sim(a, \hat{b})$ đối với mọi nhãn \hat{b} . Hiểu một cách đơn giản là nhãn nào có tính tương đồng với thực thể a nhất sẽ được lựa chọn. Tùy vào ứng dụng cụ thể, mô hình này có thể được lựa chọn cấu hình khác nhau.

Đối với bài toán phân loại văn bản, cặp thực thể dương (a, b) được lấy trực tiếp từ tập huấn luyện, trong đó, a là nhóm từ đầu vào và b là nhãn tương ứng trong tập huấn luyện. Các thực thể âm b^- là các nhãn còn lại trong tập huấn luyện. Mô hình sẽ học cách cực đại hóa $sim(a, b)$ và cực tiểu hóa $sim(a, b_i^-)$.

2.7 Kết luận chương

Trong chương này luận văn đã giới thiệu một số kiến thức nền tảng về mạng nơ-ron nhân tạo, cách thức hoạt động của mạng nơ-ron và một số các kỹ thuật được ứng dụng trong việc xử lý ngôn ngữ tự nhiên đặc biệt là 2 kỹ thuật **word2vec** và **Glove**, ứng dụng RNN trong quản lý cuộc hội thoại

CHƯƠNG 3: XÂY DỰNG CHATBOT BÁN HÀNG

Chương này sẽ mô tả từng bước xây dựng bài toán trên nền tảng mã nguồn mở Rasa. Phần thực nghiệm và đánh giá sẽ cho ta biết khả năng phục vụ của Chatbot cũng như chỉ ra những điểm hạn chế của Chatbot nhằm tìm cách cải tiến và tìm hướng đi mới cho việc xây dựng Chatbot nhằm phục vụ nghiệp vụ bán hàng.

3.1. Bài toán

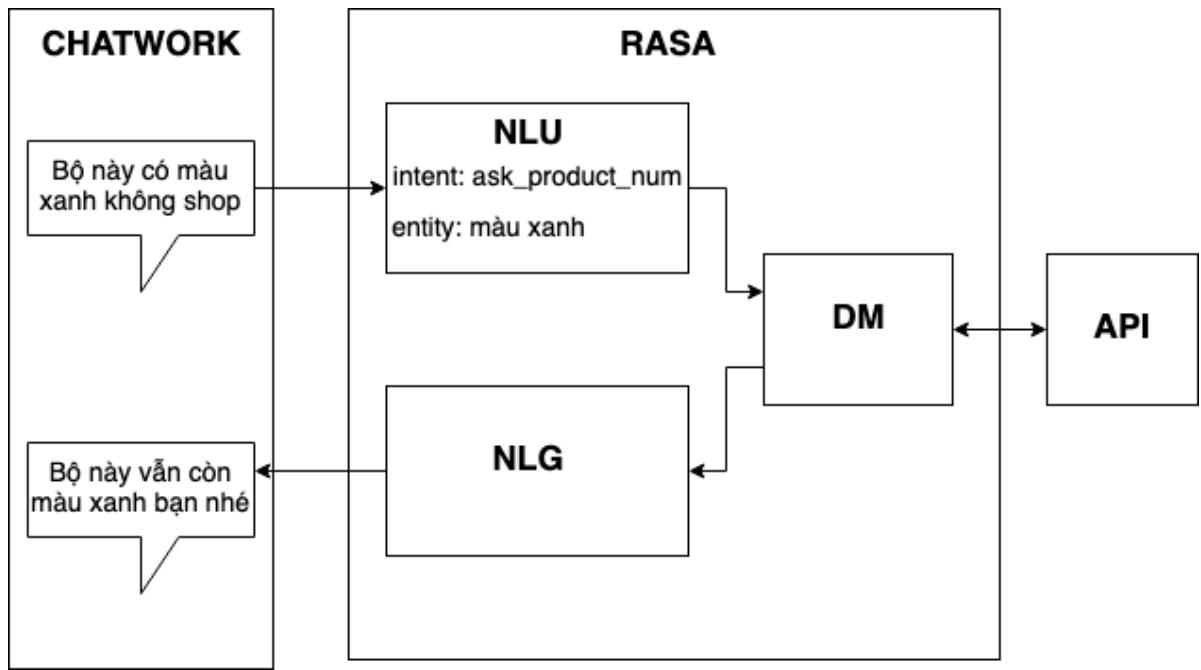
Có rất nhiều bài toán ứng dụng cho Chatbot nhưng tôi chọn bài toán cho nghiệp vụ bán hàng để giải quyết những khó khăn trong việc tư vấn và hỗ trợ bán hàng trên các trang thương mại điện tử, nhằm giảm chi phí và tăng hiệu quả hoạt động bán hàng.

Bài toán mà tôi xây dựng sẽ tập trung vào các chức năng tư vấn và hỗ trợ bán hàng trên các Fanpage Facebook. Các chức năng chính bao gồm:

- Thông tin sản phẩm
- Truy vấn sản phẩm
- Hỏi giá sản phẩm
- Tư vấn size
- Tư vấn ship hàng
- Tư vấn thanh toán
- Tạo đơn hàng
- Thông tin bảo hành

3.2. Xây dựng Chatbot hỗ trợ nghiệp vụ bán hàng

Cấu trúc hệ thống Chatbot cho nghiệp vụ bán hàng sẽ được thiết kế như sau:



Hình 39: Cấu trúc hệ thống Chatbot

Từ mô hình trên thì đầu vào của hệ thống là một câu hỏi người dùng. Đầu ra là câu trả lời của Chatbot. Các thành phần bên trong hệ thống Chatbot sẽ đảm nhiệm như sau:

NLU: Có nhiệm vụ vector hóa ngôn ngữ, phân loại ý định người dùng và trích xuất ra các thông tin. Ví dụ câu đầu vào người dùng hỏi “Áo màu xanh này giá bao nhiêu?” thì hệ thống sẽ vector hóa nó rồi đối chiếu với các tập dữ liệu training đã được gán nhãn để đưa ra ý định là “product_price” (ý định này là hỏi về giá của sản phẩm). Tiếp đến hệ thống trích xuất được thông tin màu sắc của sản phẩm là “màu xanh”.

DM: 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 kết nối với cơ sở dữ liệu để lấy dữ liệu từ hệ thống phục vụ việc sinh câu trả lời của Chatbot cho thành phần NLG. Ở ví dụ trên thì hệ thống xác nhận Chatbot đang trong ngữ cảnh hỏi về thông tin giá áo màu xanh nhưng chưa rõ hỏi về giá áo màu xanh nào nên Chatbot sẽ đưa ra quyết định hỏi lại người dùng. Trong trường hợp mà Chatbot có đầy đủ thông tin thì sẽ lấy dữ liệu từ backend để trả lời lại cho người dùng về thông tin giá theo yêu cầu.

NLG: Mô hình sinh câu trả lời dựa vào dữ liệu từ thành phần **DM** theo các tập mẫu câu template đã được xây dựng trước.

Hiện tại có rất nhiều phương pháp làm Chatbot do bên thứ ba cung cấp như Ahachat, Chatfuel, Messnow... dễ dàng xây dựng và tích hợp thông qua API nhưng lại không đảm bảo tính bảo mật về mặt dữ liệu người dùng nên trong bài toán này tôi quyết định sử dụng mã nguồn mở Rasa để xây dựng hệ thống Chatbot riêng biệt nhằm mục đích làm chủ dữ liệu và bảo mật hệ thống lẫn thông tin người dùng. Bên cạnh đó với Rasa hỗ trợ một số cơ chế học máy cho tương thích với Tiếng Việt hay có thể tùy ý kết nối tới nhiều hệ thống khác để trả lại dữ liệu cho Chatbot. Hiện tại Rasa đang có cộng đồng phát triển mạnh với hơn 3.500 thành viên, số lượng download là hơn 500.000. Các tính năng và bug mới liên tục được cập nhật và sửa đổi. Trên thực tế thì Rasa cũng đã áp dụng thành công nhiều bài toán cho các lĩnh vực như: y tế, bảo hiểm, xã hội, du lịch, ngân hàng và viễn thông. Đó cũng chính là những động lực giúp tôi lựa chọn Rasa là framework để xây dựng Chatbot giải quyết bài toán này. Chi tiết về việc tìm hiểu và làm chủ công cụ Rasa này sẽ đề cập trong Mục 3 của chương.

Về phần xây dựng giao diện hiển thị và tương tác người dùng với bot tôi sử dụng chatwork để dựng. Ngoài ra Rasa cũng hỗ trợ kết nối và tích hợp với nhiều nền tảng khác nhau như Facebook messenger,...vv

Ván đề xây dựng tập dữ liệu huấn luyện cho mô hình cũng đóng một vai trò quan trọng trong việc xây dựng hệ thống Chatbot và vấn đề này mang tính quyết định hệ thống có đáp ứng được các yêu cầu từ phía người dùng hay không. Việc xây dựng dữ liệu này sẽ được đề cập một cách chi tiết trong Mục 3.3 của chương này.

3.3. Ứng dụng RASA xây dựng Chatbot

Rasa cung cấp cho ta 2 phương pháp chính xây dựng dữ liệu training cho Chatbot:

- Pretrained Embeddings (Intent_classifier_sklearn) : Việc phân loại ý định người dùng sẽ dựa trên các tập dữ liệu được lọc trước, sau đó được sử dụng để thể hiện từng từ trong thông điệp người dùng dưới dạng từ nhúng (word embedding) hay biểu diễn ngôn ngữ dưới dạng vector(word2vec). Các tập dữ liệu này có thể được cung cấp từ Spacy hoặc MITIE ...

- Supervised Embeddings (EmbeddingIntentClassifier): Nhưng được giám sát. Với phương pháp này thì người dùng sẽ phải tự xây dựng dữ liệu từ đầu do ko có dữ liệu đào tạo sẵn có. Nhưng với các bài toán trong một miền lĩnh vực

đóng thì nó sẽ đảm bảo tính chính xác hơn nhiều và tránh dư thừa dữ liệu so với phương pháp ở trên.

Với bài toán tập trung vào miền đóng như Chatbot bán hàng, cộng thêm việc khó tìm kiếm tập dữ liệu pretrained (do hầu hết dữ liệu pretrained đều là tiếng anh) thì ta không cần sử dụng tập dữ liệu đào tạo từ trước (pretrained word embeddings) thay vào đó thì ta sẽ tự tạo tập dữ liệu training riêng của mình. Điều này cũng đảm bảo Chatbot có thời gian training ngắn mà độ chính xác cao hơn.

Một số cấu hình trong Rasa mà tôi lựa chọn để training cho Chatbot bao gồm từ việc phân tích câu, phân loại ý định(intent) đến trích chọn thông tin người dùng được cấu hình trong file config.yml như sau:

```
1 language: vi
2 ##pipeline: tensorflow_embedding_pipeline:
3 - name: tokenizer_whitespace
4 - name: ner_crfs
5 - name: ner_synonyms
6 - name: intent_featurizer_count_vectors - name: intent_classifier_tensorflow_embedding intent_tokenization_flag: true
    intent_split_symbol: "+"
```

Hình 40: Cấu hình pipeline xử lý ngôn ngữ tự nhiên

Thuật toán tách từ tôi sử dụng tokenizer_whitespace tức các từ có thể phân tách bởi dấu cách. Điều này có thể sai trong một số trường hợp đối với từ ghép.

Thuật toán tách từ Tiếng Việt tôi sử dụng **VietnameseTokenizer** dựa trên thư viện **Underthesea** [26] của tác giả Vũ Anh. Thư viện này giúp tôi tách các từ Tiếng Việt một cách chuẩn xác bao gồm cả việc tách các từ ghép. Được xây dựng như sau:

```
1 import re
2 from typing import Any, Dict, List, Text
3 from rasa.nlu.tokenizers.tokenizer import Token, Tokenizer
4 from rasa.nlu.training_data import Message
5
6 from rasa.nlu.constants import TOKENS_NAMES, MESSAGE_ATTRIBUTES
7 from underthesea import word_tokenize
8 class VietnameseTokenizer(Tokenizer):
9
10     provides = [TOKENS_NAMES[attribute] for attribute in MESSAGE_ATTRIBUTES]
11
12     def __init__(self, component_config: Dict[Text, Any] = None) -> None:
13         super().__init__(component_config)
14
15     def tokenize(self, message: Message, attribute: Text) -> List[Token]:
16         text = message.get(attribute)
17         words = word_tokenize(text)
18
19         return self._convert_words_to_tokens(words, text)
```

Để trích chọn thông tin (slot filter) thì tôi cấu hình dùng mô hình CRF. Bên cạnh đó cũng hỗ trợ cấu hình việc nhận dạng các từ đồng nghĩa (ner_synonyms). Rasa cũng tích hợp với framework hỗ trợ trích xuất thông tin nổi tiếng như **duckling** có hỗ trợ tiếng việt và rất thông minh trong việc trích xuất được thời gian và dữ liệu số. Với bài toán demo thì CRF là một lựa chọn hợp lý hơn khi đã tích hợp sẵn trong Rasa mà không phải cấu hình thêm như duckling.

Thành phần chính là trình phân loại ý định EmbeddingIntentClassifier dựa trên mô hình giải thuật **starspace** [9]. Giải thuật starspace do Facebook phát triển và công bố năm 2017.

3.4. Xây dựng dữ liệu Chatbot

Nguồn dữ liệu xây dựng để huấn luyện Chatbot bán hàng được thu thập và tham khảo qua một số Fanpage Facebook.

Một đoạn hội thoại giữa Chatbot và khách hàng là để giải quyết một yêu cầu nào đó của khách hàng. Ví dụ đoạn hội thoại của khách hàng hỏi Chatbot về mua sản phẩm thời trang:

Người dùng	Bot
Giá bộ sản phẩm này bao nhiêu	Giá bộ sản phẩm này là 299k bạn nhé
Có bán riêng áo không shop	Giá áo là 200k bạn nhé
Chất vải gì shop	Chất vải thun co dãn 4 chiều, chống tia uv, kháng khuẩn

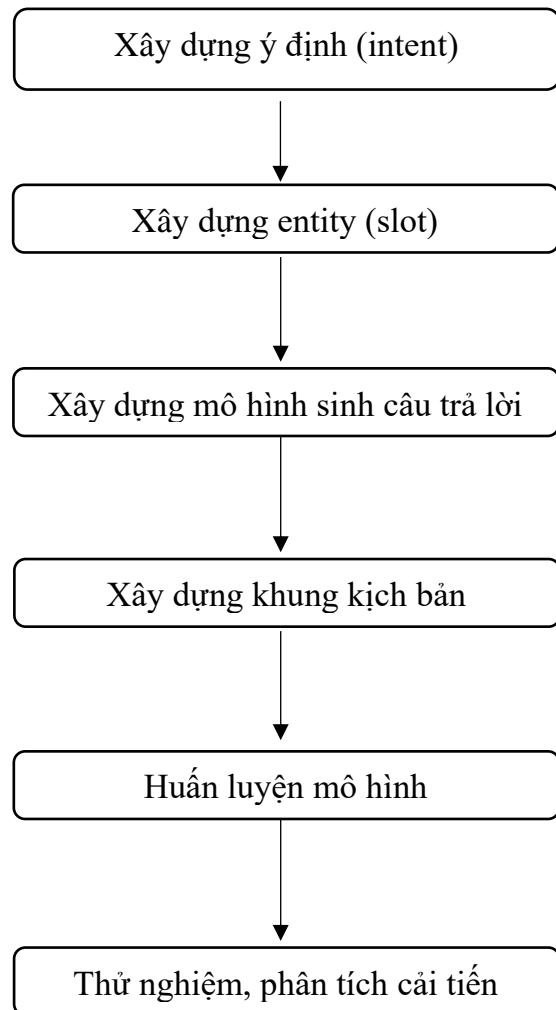
Như vậy với bài bán xây dựng dữ liệu huấn luyện cho Chatbot bán hàng nói riêng hay theo miền đóng chung là ta phải đi xây dựng danh sách các khung kịch bản như ví dụ bên trên.

Ý định của người dùng có thể diễn đạt nhiều cách nói khác nhau nhưng chung một nhãn thông điệp của họ, ứng với mỗi một câu hỏi, yêu cầu trong đoạn trò chuyện là một ý định của người dùng. Ví dụ người dùng có thể diễn đạt mục đích chào hỏi của mình bằng nhiều câu nói như “xin chào”, “chào bạn”, “chào shop”, “hi shop”,..vv.

Khi đã nắm được ý định của người dùng trong câu nói của họ qua cuộc trò chuyện thì ta cần phải trích xuất ra được những thông tin đặc trưng của người dùng. Đó chính là các thực thể (entity), các thực thể này sẽ được lưu lại vào các

slot để lưu được ngữ cảnh tránh việc phải hỏi lại nhiều lần trong cuộc trò chuyện, việc này sẽ phục vụ cho các hành động sau của Chatbot.

Quy trình xây dựng hệ thống Chatbot sẽ thực hiện qua các bước chính như sau:



Hình 41: Các bước xây dựng Chatbot

3.4.1 Xây dựng ý định

Nhiệm vụ xây dựng các tập ý định sẽ theo nguyên tắc là những mẫu câu hỏi mà người dùng khi có ý định đó hay sử dụng nhất có thể. Cần định nghĩa các ý định khớp với ngôn ngữ tự nhiên nhất, trong hệ thống này tôi định nghĩa các ý định như sau:

- Ask_hello : ý định liên quan đến các chào hỏi.
- Product_price: ý định liên quan đến hỏi về giá sản phẩm.

- Product_num: ý định hỏi về số lượng sản phẩm.
- Product_size: ý định hỏi về tư vấn size phù hợp.
- Product_quality: ý định hỏi về chất liệu sản phẩm.
- Product_ship: ý định hỏi về dịch vụ vận chuyển sản phẩm.
- Product_payment: ý định hỏi về phương thức thanh toán.
- Product_feedback: ý định về đánh giá sản phẩm.

Ví dụ cho một ý định muốn hỏi về thông tin giá sản phẩm thì người dùng có thể hỏi nhiều câu như sau:

```

11  ## intent:ask_product_price
12  - giá bộ mã [sp1](product_name) này bao nhiêu
13  - giá bộ [sp2](product_name) này bao nhiêu
14  - bộ này giá bao nhiêu
15  - giá bộ này nhiêu
16  - bộ này giá nhiêu
17  - xin giá bộ này shop ơi
18  - xin giá
19  - giá bao nhiêu vậy shop
20  - áo này giá nhiêu

```

Hình 42: Xây dựng ý định người dùng

3.4.2 Xây dựng entity

Entity là các thực thể thông tin đặc trưng quan trọng được trích xuất theo các ý định người dùng. Các thông tin được trích chọn trong câu nói của người dùng được hệ thống lưu lại trong bộ nhớ hệ thống để dùng trong các hành động hoặc để đưa ra các câu trả lời phù hợp theo ngữ cảnh, tránh việc phải hỏi lại những thông tin mà người dùng đã cung cấp từ trước.

Nhiệm vụ trích xuất thông tin được cấu hình theo mô hình CRF như đã đề cập ở trên.

```
29 |     entities:  
30 |         - product_name  
31 |         - location  
32 |         - kilogam  
33 |         - height  
34 |         - size
```

Hình 43: Danh sách các entities

3.4.3 Xây dựng câu trả lời cho bot

Khi người dùng đưa ra các câu hỏi, yêu cầu thì Chatbot phải có nhiệm vụ đưa ra được câu trả lời đáp ứng được các câu hỏi và yêu cầu đó

```
1 |     utter_product_price:  
2 |         - text: "Giá bộ này là {&price} bạn nhé"  
3 |         - text: "Bộ này có giá là {&price} bạn nhé"  
4 |         - text: "Bộ này có giá chỉ {&price} bạn nhé"
```

Hình 44: Mẫu câu trả lời của Chatbot cho ý định hỏi giá sản phẩm

Để tạo tính tự nhiên và phù hợp với độ tuổi trong cuộc hội thoại thì ta có thể xây dựng nhiều tập mẫu câu trả lời để Chatbot lựa phù hợp với lứa tuổi, giới tính khi có được thông tin về khác hàng hoặc nếu không thì sẽ chọn ngẫu nhiên các mẫu câu trả lời để tạo cảm giác không bị nhảm chán.

Có thể xây dựng các phản hồi cho Chatbot thông qua action. Khi có yêu cầu đầu vào của người dùng thì Chatbot có thể lựa chọn các hành động phù hợp để đáp ứng nhu cầu đó. Hành động này có thể cung cấp thông tin mong muốn cho người dùng dựa vào các ý định, slot và dữ liệu lấy từ hệ thống cơ sở dữ liệu thông qua các API kết nối. Bên cạnh đó action của rasa còn hỗ trợ tùy biến qua ngôn ngữ python nên ta có thể điều hướng các action tiếp theo dựa vào dialog state tracker, policy và dispatcher của rasa. Có ba loại hành động trong Rasa Core:

- **Default actions** : các hành động default như lắng nghe người dùng, restart lại hội thoại hoặc trả lời mặc định khi không phân loại được ý định người dùng.

```

1  utter_default:
2  - text: "Xin lỗi! Tôi không hiểu yêu cầu của bạn. Bạn vui lòng để lại số điện thoại để nhân viên bên mình hỗ trợ bạn
nhé"
3  - text: "Xin lỗi! câu hỏi của bạn ngoài khả năng của tôi. \nBạn có thể hỏi tôi về thông tin sản phẩm, các dịch vụ hỗ
troy" buttons:
4  - text: "Bạn có cần hỗ trợ trực tiếp qua tổng đài không"

```

Hình 45: Mẫu câu trả lời mặc định của bot khi không nhận ra ý định người dùng

- **Custom actions:** Khi các tập câu trả lời mẫu không áp dụng được với các câu trả lời cần có kết quả lấy từ một nguồn dữ liệu khác thì action tùy biến được sử dụng, nó sẽ trỏ đến một hàm trong lớp action (python). Trong đây mình sẽ tùy biến câu trả lời như lấy dữ liệu qua API rồi điền vào tham số trong câu trả lời.

```

1  class CheckSP(Action):
2      """ CHECK XEM HÀNG CÒN KHÔNG"""
3      def name(self):
4          return "action_check_sp"
5      def run(self, dispatcher, tracker, domain):
6          # maSP = tracker.get_slot('maSP')
7          text = tracker.latest_message.text
8          print(text)
9          maSP = text
10         if maSP:
11             # check maSP
12             print(maSP)
13         else:
14             print('Sản phẩm này đã hết hàng')
15
16         return [SlotSet('maSP', maSP)]
17

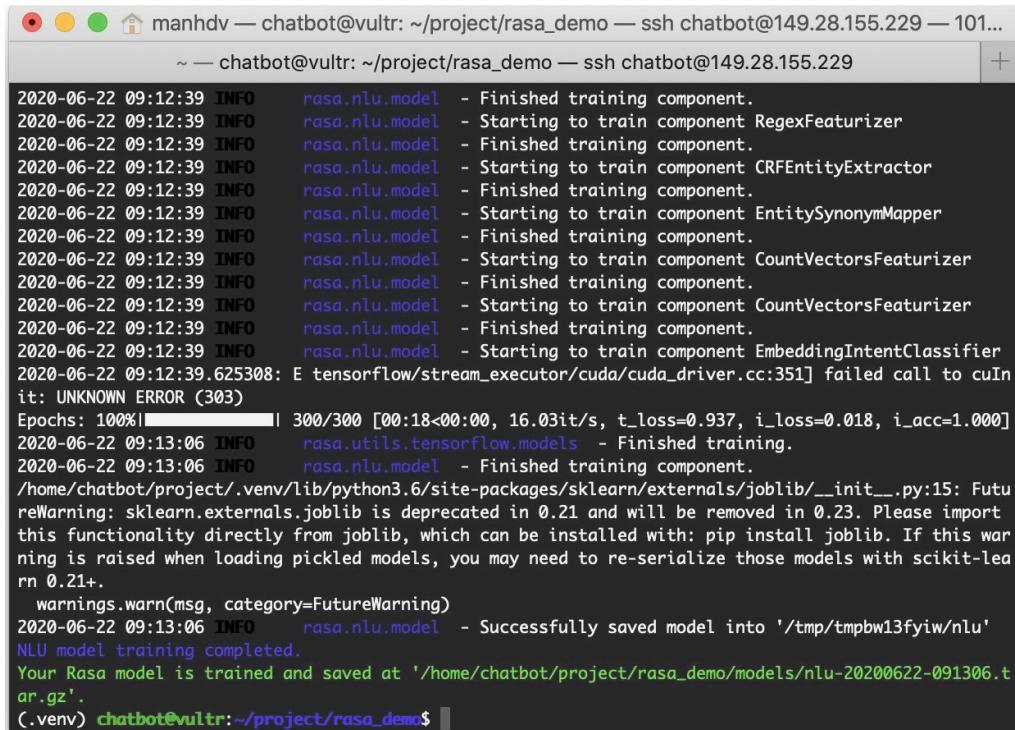
```

Hình 46: Custom action xử lý slot maSP

3.4.4 Xây dựng khung kịch bản (history)

Với mỗi một ý định của người dùng thì tương ứng với một tập các mẫu câu trả lời đã được xây dựng sẵn, ta xây dựng các khung kịch bản cho Chatbot dựa trên việc sắp xếp thành đoạn đối thoại

Việc xây dựng đoạn hội thoại này có thể viết bằng tay hoặc thông qua việc **học tương tác (Interactive Learning)** với Chatbot: Đây là một cách khác để xây dựng khung câu truyện là việc học tương tác với bot. Chế độ này cho phép người dùng tự động tạo ra các hội thoại sau khi chat trực tiếp với bot. Nếu bot nhận định các intent hay slot sai thì người dùng có huấn luyện lại cho bot đúng.



The screenshot shows a terminal window with the following log output:

```
2020-06-22 09:12:39 INFO rasa.nlu.model - Finished training component.
2020-06-22 09:12:39 INFO rasa.nlu.model - Starting to train component RegexFeaturizer
2020-06-22 09:12:39 INFO rasa.nlu.model - Finished training component.
2020-06-22 09:12:39 INFO rasa.nlu.model - Starting to train component CRFEntityExtractor
2020-06-22 09:12:39 INFO rasa.nlu.model - Finished training component.
2020-06-22 09:12:39 INFO rasa.nlu.model - Starting to train component EntitySynonymMapper
2020-06-22 09:12:39 INFO rasa.nlu.model - Finished training component.
2020-06-22 09:12:39 INFO rasa.nlu.model - Starting to train component CountVectorsFeaturizer
2020-06-22 09:12:39 INFO rasa.nlu.model - Finished training component.
2020-06-22 09:12:39 INFO rasa.nlu.model - Starting to train component CountVectorsFeaturizer
2020-06-22 09:12:39 INFO rasa.nlu.model - Finished training component.
2020-06-22 09:12:39 INFO rasa.nlu.model - Starting to train component EmbeddingIntentClassifier
2020-06-22 09:12:39.625308: E tensorflow/stream_executor/cuda/cuda_driver.cc:351] failed call to cuInit: UNKNOWN ERROR (303)
Epochs: 100%[██████████] 300/300 [00:18<00:00, 16.03it/s, t_loss=0.937, i_loss=0.018, i_acc=1.000]
2020-06-22 09:13:06 INFO rasa.utils.tensorflow.models - Finished training.
2020-06-22 09:13:06 INFO rasa.nlu.model - Finished training component.
/home/chatbot/project/.venv/lib/python3.6/site-packages/scikit-learn/experimental/joblib/__init__.py:15: FutureWarning: sklearn.externals.joblib is deprecated in 0.21 and will be removed in 0.23. Please import this functionality directly from joblib, which can be installed with: pip install joblib. If this warning is raised when loading pickled models, you may need to re-serialize those models with scikit-learn 0.21+.
    warnings.warn(msg, category=FutureWarning)
2020-06-22 09:13:06 INFO rasa.nlu.model - Successfully saved model into '/tmp/tmpbw13fyiw/nlu' NLU model training completed.
Your Rasa model is trained and saved at '/home/chatbot/project/rasa_demo/models/nlu-20200622-091306.tar.gz'.
(.venv) chatbot@vultr:~/project/rasa_demo$
```

Hình 47: Huấn luyện cho Chatbot

3.5. Thủ nghiệm

3.5.1 Dữ liệu thử nghiệm

Hệ thống được thử nghiệm trên bộ dữ liệu gồm: tập các câu hỏi thoại từ phía người dùng khi mua hàng, kết quả của thử nghiệm được lưu tại thư mục result.

```

manhdv — chatbot@vultr: ~/project/rasa_demo/results — ssh ch...
...chatbot@vultr: ~/project/rasa_demo/results — ssh chatbot@149.28.155.229 + 
https://www.techrepublic.com/article/how-to-install-microk8s-on-macos/

* Canonical Livepatch is available for installation.
- Reduce system reboots and improve kernel security. Activate at:
https://ubuntu.com/livepatch

2 packages can be updated.
0 updates are security updates.

Last login: Tue Jun 30 16:45:48 2020 from 1.55.108.126
[chatbot@vultr:~$ cd project/
[chatbot@vultr:~/project$ source .venv/bin/ac
-bash: .venv/bin/ac: No such file or directory
[chatbot@vultr:~/project$ source .venv/bin/activate
(.venv) chatbot@vultr:~/project$ cd rasa_demo
(.venv) chatbot@vultr:~/project/rasa_demo$ ls
__init__.py credentials.yml endpoints.yml models test_data.md
actions.py data get-pip.py rasa_x_playbook.yml
config.yml domain.yml install.sh results
(.venv) chatbot@vultr:~/project/rasa_demo$ cd results/
(.venv) chatbot@vultr:~/project/rasa_demo/results$ ls
CRFEntityExtractor_report.json config.png hist.png intent_report.json
(.venv) chatbot@vultr:~/project/rasa_demo/results$ ]

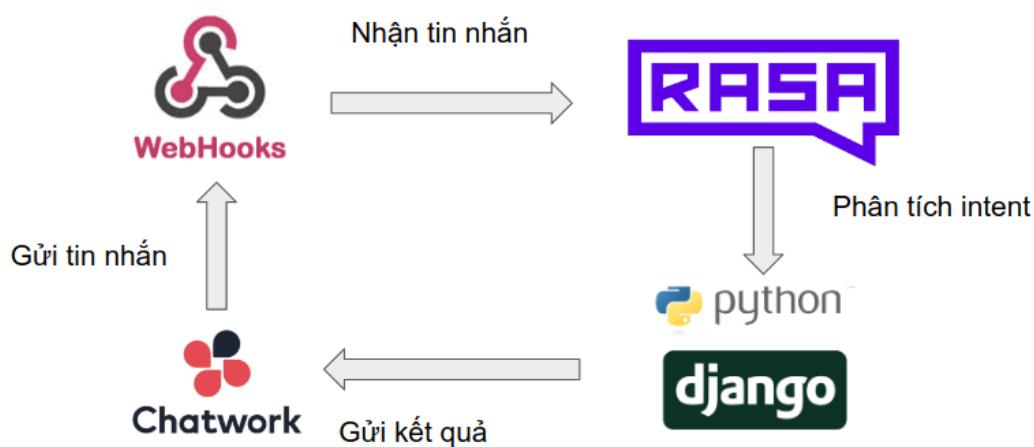
```

3.5.2 Môi trường và công cụ sử dụng thực nghiệm

Chương trình thử nghiệm được thiết kế, xây dựng và thực hiện trên môi trường hệ điều hành Ubuntu với nền tảng framework Rasa và Django, dựa trên ngôn ngữ lập trình python. Giao diện người dùng sử dụng nền tảng ChatWork được tích hợp trên app mobile để tiện sử dụng.

3.5.3 Thiết kế chương trình thử nghiệm

Kiến trúc của chương trình thử nghiệm trong luận văn được thiết kế như sau:



Hình 48: Kiến trúc của chương trình thử nghiệm

Trong hệ thống trên thì luồng dữ liệu được định nghĩa như sau:

- **Bước 1: Chatwork gửi event qua webhook** Mỗi khi có một sự kiện diễn ra trên chatwork cụ thể ở đây là sự kiện khi có một người dùng gửi tin nhắn cho Chatbot thì tài khoản chatwork này sẽ gửi một POST request đến webhook được sử dụng để lắng nghe sự kiện. Webhook này sẽ phân tích tin nhắn và chuyển tiếp đến NLU của RASA.

- **Bước 2: RASA nhận diện intent và entity:** Sau khi đã thu được message của người dùng thì sử dụng RASA để hiểu được intent và entity của câu nói người dùng. Từ các thông tin đó sẽ quyết định xử lý sao cho phù hợp

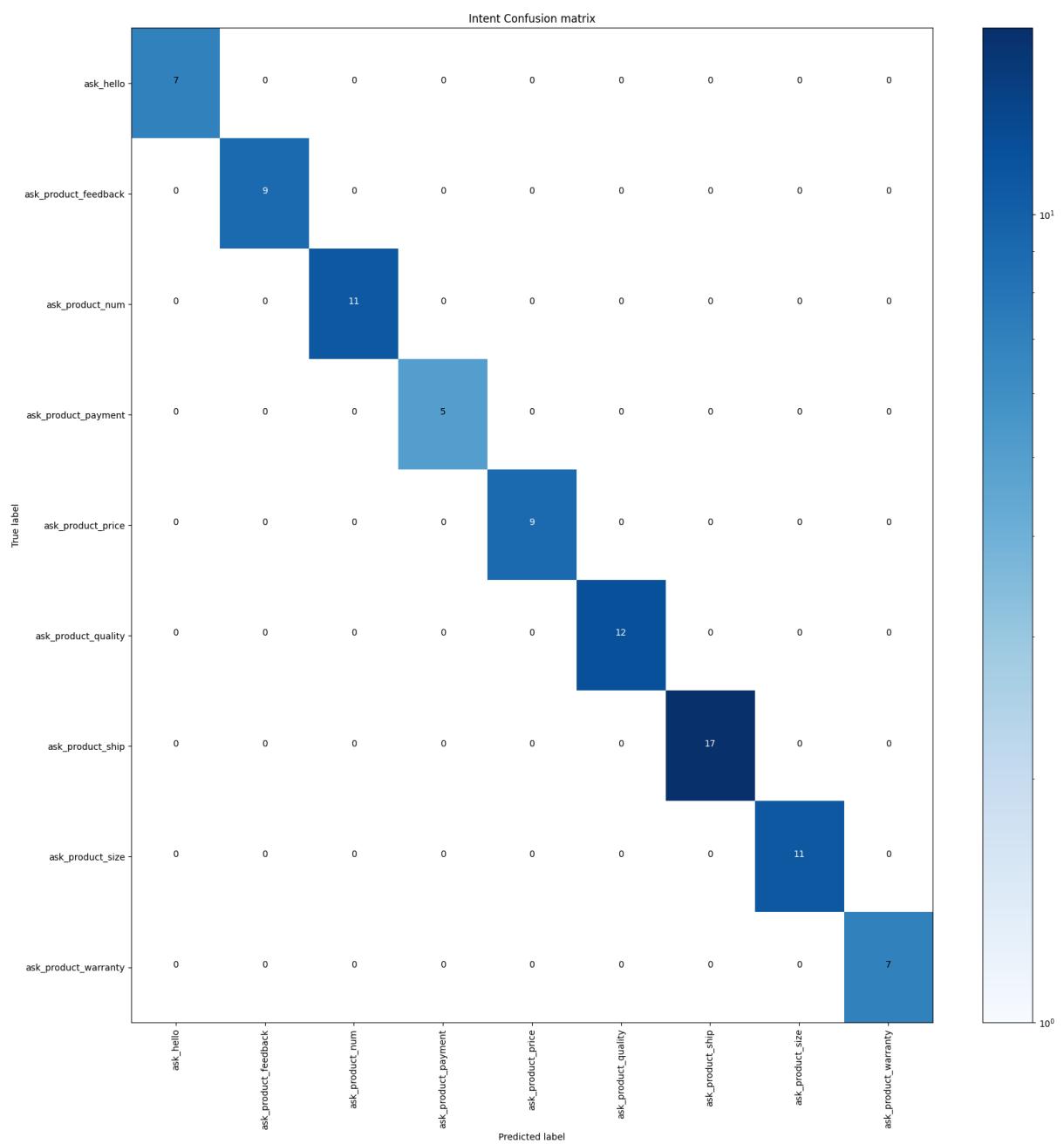
- **Bước 3: Sinh câu trả lời:** Bước này sau khi phân tích được ý định và các entity thì hệ thống sẽ sinh ra các câu trả lời phù hợp.

- **Bước 4: Gửi tin nhắn qua Chatwork** bước này sử dụng Chatwork SDK để chuyển tiếp tin nhắn sang tài khoản chatwork và hoàn thành nhiệm vụ của ứng dụng.

3.5.4 Thủ nghiệm

Tiến hành thử nghiệm tương tác với Chatbot với một số câu thì thấy độ chính xác của Chatbot đạt khoảng 80% và chỉ tính các câu hỏi xung quanh kịch bản đã đào tạo cho Chatbot. Còn đối với các câu hỏi mà chưa training(out of scope) thì bot sẽ đưa ra một số mẫu câu trả lời điều hướng người dùng tới một số câu hỏi sẵn có mà Chatbot có thể đáp ứng được.

Một số biểu đồ đánh giá ước lượng tập dữ liệu training và khung kịch bản (history) cho bot:



Hình 49: Hình ma trận ước lượng nhầm lẫn xây dựng dữ liệu intent

Với biểu đồ ước lượng trên ta thấy dữ liệu các câu đầu vào cho intent người dùng ít bị xung đột hay nhầm lẫn với nhau. Điều này đảm bảo tính đúng đắn của tập dữ liệu cho việc đào tạo bot.

Intent	Độ đo			
	precision	recall	f1-score	support
ask_hello	1.0	1.0	1.0	7
ask_product_feedback	1.0	1.0	1.0	9
ask_product_num	1.0	1.0	1.0	11
ask_product_payment	1.0	1.0	1.0	5
ask_product_price	1.0	1.0	1.0	9
ask_product_quality	1.0	1.0	1.0	12
ask_product_ship	1.0	1.0	1.0	17
ask_product_size	1.0	1.0	1.0	11
ask_product_warranty	1.0	1.0	1.0	7

Hình 50: Ước lượng độ chính xác tập dữ liệu trainning intent

Sử dụng mô hình EmbeddingIntentClassifier dựa trên nền tảng mô hình starspace [26] của facebook để đào tạo cho Chatbot phục vụ việc phân loại ý định người dùng cho thấy độ chính xác tương đối cao đối với tập dữ liệu cho bài toán ngân hàng.

Entities	Độ đo			
	precision	recall	f1-score	support
product_name	1.0	1.0	1.0	24
location	1.0	1.0	1.0	21
kilogam	1.0	1.0	1.0	4
height	1.0	1.0	1.0	3
size	1.0	1.0	1.0	2

Hình 51: Ước lượng độ chính xác trích chọn thông tin

Việc áp dụng mô hình CRF cho trích chọn thông tin đạt độ chính xác rất cao 0.99%. Qua quá trình đào tạo cho bot thì thấy việc càng training nhiều dữ liệu

thì độ chính xác trích xuất càng cao với CRF. Để đảm bảo việc trích chọn thông tin chính xác nhất thì mỗi entity phải training cho bot từ 10 đến 20 dữ liệu.

Dưới là danh sách câu hỏi và trả lời khi người dùng tương tác với Chatbot trong thử nghiệm cuối.

No	Người dùng	Chatbot	Ý định	Slot
1	Xin chào shop	Xin chào! Chúng tôi có thể giúp gì cho bạn?	ask_hello	
2	Hi shop	Xin chào! Chúng tôi rất hân hạnh khi được phục vụ bạn!	ask_hello	
3	Bộ sp1 này có còn không shop ơi	Bộ sp1 này vẫn còn bạn nhé	product_num	product_name: sp1
4	Mẫu sp2 này có còn không nhỉ	Mẫu sp2 này bên mình vẫn đang còn bạn nhé	product_num	product_name: sp2
5	Có còn bộ sp33 không shop	Bộ này bên mình vẫn còn hàng nhé	product_num	product_name: sp3
6	Bộ này còn size nào vậy	Hiện tại shop vẫn còn full size cho các sản phẩm bạn nhé	product_size	
7	Mình nặng 80kg, cao	Bạn nên mặc size XL để cho thoải mái nhé	product_size	kilogame: 80kg, height: 1m7

	1m7 thì mặc size nào vậy			
8	Giá bộ sp33 là bao nhiêu vậy shop	Giá sản phẩm sp33 là 299k nhé	product_price	
9	Giá bộ sp4 là bao nhiêu vậy shop	Giá bộ sp4 này bên mình là 299k nhé b	product_price	
10	Xin giá mẫu sp5 với shop ơi	Giá mẫu sp5 này bên mình là 299k ạ	product_price	
11	Bộ này mặc có mát không shop	Sản phẩm bên mình sử dụng chất vải thun co dãn 4 chiều, chống tia uv, kháng khuẩn, giặt không phai màu, shop xin đảm bảo chất lượng	product_quality	
12	Bộ này khi giặt có bị phai màu không nhỉ	Bên mình đảm bảo chất lượng hàng bên mình rất ok nhé, chất vải co dãn 4 chiều, chống tia uv, kháng khuẩn, giặt không phai màu,	product_quality	
13	Được đổi trả hàng trong bao lâu vậy shop	Với chính sách trả hàng thì bạn có thể trả hàng trong vòng 7 ngày, bạn vui lòng để nguyên tem mác trên sản phẩm giúp mình nhé	product_warranty	

14	Tức là trong 7 ngày là được trả lại hàng đúng không nhỉ			
15	Có ship về Thái Nguyên không shop	Bên mình đang khuyến mại free ship toàn quốc nhé	product_ship	
16	Ship hàng mất thời gian bao lâu thì nhận được nhỉ			
17	Nhận được hàng mình với thanh toán đúng không			
18	Mình thanh toán cod được không	Tất cả sản phẩm đều ship COD nhận tiền khi giao hang	product_payment	
19	Cảm ơn	Cảm ơn bạn đã phản hồi!	product_feedback	
20	Mình đã nhận được hàng nhé shop	Cảm ơn bạn đã phản hồi! Chúc bạn một ngày tốt lành	product_feedback	

Hình 52: Bảng mô tả đoạn hội thoại test với Chatbot.

3.6. Đánh giá

Từ quá trình tìm hiểu, xây dựng và các kết quả khách quan từ thực nghiệm thì tôi có những đánh giá như sau:

- Xây dựng dữ liệu đào tạo, huấn luyện cho Chatbot và các kịch bản xây dựng cuộc hội thoại là yếu tố vô cùng quan trọng. Đây là hai yếu tố có sự ảnh hưởng rất lớn đến tính linh hoạt và độ thông minh của Chatbot.

- Việc định nghĩa và xây dựng các ý định cũng rất quan trọng phải phân tích tỉ mỉ với các chủ đề khác nhau để có thể đưa ra các tập ý định tốt nhất. Đối với các ý định không rõ ràng hay gần nhau về mặt ngữ nghĩa sẽ khiến cho độ chính xác của Chatbot giảm. Nên việc thiết kế các ý định và slot là vô cùng quan trọng.

Với các đoạn hội thoại nằm trong kịch bản dựng sẵn thì Chatbot đáp ứng rất tốt khi trả lời đúng cho người dùng. Tuy nhiên việc xây dựng kịch bản cho Chatbot rất khó khăn trong các đoạn hội thoại vì nó có thể xảy ra rất nhiều trường hợp. Đối với các cuộc hội thoại dài thì rất phức tạp trong việc lưu trữ các slot để lưu được ngữ cảnh cuộc hội thoại.

Chatbot có khả năng trả lời ngẫu nhiên các mẫu câu trong template khiếu cho đoạn hội thoại trở nên tự nhiên hơn. Bên cạnh đó bot có khả năng điều hướng người dùng đến các mẫu câu trả lời sẵn, khi người dùng hỏi những câu ngoài phạm vi huấn luyện. Tuy nhiên việc điều hướng cũng dựa trên khả năng trả lời ngẫu nhiên của bot dẫn đến việc bot cũng chưa thông minh trong việc xử lý tình huống như này.

Qua bài toán trên tôi đánh giá việc áp dụng bài toán Chatbot cho nghiệp vụ bán hàng là rất khả thi, có tính thực tiễn cao do đáp ứng được một số vấn đề trong nghiệp vụ bán hàng.

CHƯƠNG 4: KẾT LUẬN

Trong luận văn này tôi đã tìm hiểu một số kiến thức tổng quan về hệ thống Chatbot, các thành phần cấu trúc và nhiệm vụ các thành phần trong Chatbot, tìm hiểu một số thuật toán cơ bản áp dụng vào việc xây dựng Chatbot để giải quyết bài toán Chatbot bán hàng. Trong quá trình tìm hiểu và xây dựng ứng dụng Chatbot hỗ trợ người dùng cho nghiệp vụ bán hàng tôi đã đạt được một số kết quả nhất định như sau:

Các vấn đề mà luận văn đã làm được:

1. Trình bày kiến thức tổng quan về một hệ thống Chatbot, các mô hình Chatbot bán hàng hiện nay, tìm hiểu chi tiết cấu trúc các thành phần và những vấn đề gặp phải khi xây dựng hệ thống Chatbot.

2. Nắm được các luồng hoạt động hay các bước xử lý của các thành phần trong mô hình Chatbot. Bên cạnh đó tôi cũng nắm được một số thuật toán và phương pháp để xử lý dữ liệu trong Chatbot.

3. Trong quá trình xây dựng tập dữ liệu đào tạo, huấn luyện cho Chatbot đã giúp tôi có được những kinh nghiệm quý báu trong việc xử lý và gán nhãn dữ liệu với ngữ nghĩa nhập nhằng. Từ đó có thể xây dựng bộ dữ liệu huấn luyện tốt hơn đem lại độ chính xác cao hơn mô hình.

4. Xây dựng thành công chương trình thử nghiệm hệ thống Chatbot phục vụ nghiệp vụ bán hàng theo đúng các mô hình và thuật toán đã trình bày với CSDL thử nghiệm tham khảo qua các kênh Fanpage Facebook đang bán hàng thật và đánh giá kết quả thử nghiệm. Tuy nhiên, việc đánh giá kết quả vẫn còn thực hiện thủ công

Định hướng nghiên cứu tiếp theo:

- ✓ Tích hợp speech to text và text to speech cho bot. Khi đó có thể tích hợp vào Chatbot để hỗ trợ người dùng qua giọng nói song song cùng với giao diện hiện tại.
- ✓ Xây dựng bot hỗ trợ multi intent. Hay người dùng có thể hỏi nhiều câu hỏi kép
- ✓ Xây dựng bot mang tính cảm xúc hơn hay nhân cách hóa Chatbot giúp Chatbot trở nên giống người hơn

TÀI LIỆU THAM KHẢO

1. Yun-Nung (Vivian) Chen, Asli Celikyilmaz and Dilek Hakkani-Tur, 2018: "Deep Learning for Dialogue Systems"
2. Tom Bocklisch, 2018: "Conversational AI with Rasa NLU & Rasa Core"
3. Hongshen Chen, Xiaorui Liu, Dawei Yin and Jiliang Tang, 11 Jan 2018: "A Survey on Dialogue Systems: Recent Advances and New Frontiers"
4. Daniel Jurafsky & James H. Martin, 23 September 2018: "Dialog Systems and Chatbots"
5. Daniel Jurafsky & James H. Martin, 23 September 2018: "Advanced Dialog Systems"
6. Jason D. Williams, Kavosh Asadi and Geoffrey Zweig, 24 Apr 2017: "Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning"
7. Andrew Maas, Spring 2017: "Dialogue System Introduction and Frame-Based Dialogue"
8. Peng Jin, Yue Zhang, Xingyuan Chen and Yunqing Xia: "Bag-of-Embeddings for Text Classification"
9. Ledell Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes and Jason Weston, 21 Nov 2017: "StarSpace: Embed All The Things!"
10. Bing Liu and Ian Lane, 2 JUN 2018: "End-to-End Learning of Task-Oriented Dialogs"
11. John A. Bullinaria, 2005: "IAI: Semantic Networks and Frames"
12. Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, Alexey Litinsky, Varvara Logacheva, Alexey Lymar, Valentin Malykh, Maxim Petrov, Vadim Polulyakh, Leonid Pugachev, Alexey Sorokin, Maria Vikhreva and Marat Zaynudinov, 15 JUN 2018: "DeepPavlov: Open-Source Library for Dialogue Systems"
13. Hao Fang, 17 Apr 2018: "Dialog Management and System Evaluation"
14. Jason Weston, "Tasks and Architectures for Language Understanding and Dialogue with memory"

15. DENNY BRITZ, 2015, “Recurrent Neural Networks Tutorial”
16. SONVX, 2018, “Học biểu diễn ngôn ngữ cho máy tính”
17. Colah’s blog, August 27, 2015, “Understanding LSTM Networks”
18. Van Deemter, Krahmer, Emiel; Theune, 1999 “Plan-based vs. template-based NLG”
19. Ondrej Plátek, Petr Belohlávek, Vojtech Hudecek, and Filip Jurcícek, 2016 “Recurrent Neural Networks for Dialogue State Tracking”
20. Matthew Henderson, Blaise Thomson and Steve Young, 2014 “Word-Based Dialog State Tracking with Recurrent Neural Networks”
21. Victor Zhong, Caiming Xiong, Richard Socher, 2018 “Global-Locally SelfAttentive Dialogue State Tracker”
22. Pei-Hao Su, Nikola Mrksic, Inigo Casanueva, Ivan Vulic, 2018 “Deep Learning for Conversational AI”
23. Jagan Jami, 2017, “INFOGRAPHIC: THE FUTURE OF CHATBOTS STATISTICS & TRENDS”
24. Larry Kim, 2018 “The Top 7 Messenger Apps in the World”
25. Liron Hakim Bobrov, January 2019, “Mobile Messaging App Map of the World”
26. Vu Anh, Jul 2019, “*Underthesea* - Vietnamese NLP Toolkit”
27. Hammersley, J., & Clifford. P, Unpublished manuscript ,1971, “Markov fields on finite graphs and lattices”

PHỤ LỤC

12:37 Wi-Fi Battery

< Mạnh Dv ⚡ 🔍 ⚡

 hidemo 11:56
Xin chào shop

 Mạnh Dv 11:56
↳ RE 
Xin chào! Chúng tôi có thể giúp gì cho bạn?

 hidemo 11:58
Hi shop

 Mạnh Dv 11:58
↳ RE 
Xin chào! Chúng tôi rất hân hạnh khi được phục vụ bạn!

 hidemo 12:00
Bộ sp1 này có còn không shop ơi

 Mạnh Dv 12:00
↳ RE 
Chúng tôi không hiểu yêu cầu của bạn! Bạn vui lòng nhập lại yêu cầu

 hidemo 12:01
Bộ sp1 này có còn không shop ơi

Nhập nội dung tin nhắn

Aa  TO   Gửi

12:37



Manh Dv



ooo



hidemo

12:01

Bộ sp1 này có còn không shop ơi



Manh Dv

12:01



Bộ sp1 này vẫn còn bạn nhé



hidemo

12:02

Mẫu sp2 này có còn không nhỉ



Manh Dv

12:02



Mẫu sp2 này bên mình vẫn đang còn bạn nhé



hidemo

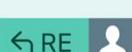
12:04

Có còn bộ sp33 không shop



Manh Dv

12:04



Bộ này bên mình vẫn còn hàng nhé



hidemo

12:06

Bộ này còn size nào vậy



Manh Dv

12:06

Nhập nội dung tin nhắn

Aa



TO



Gửi

12:37



Manh Dv



ooo



hidemo

12:06

Bộ này còn size nào vậy



Manh Dv

12:06



Hiện tại shop vẫn còn full size cho các sản
phẩm bạn nhé



hidemo

12:09

Mình nặng 80kg, cao 1m7 thì mặc size nào
vậy



Manh Dv

12:10



Bạn nên mặc size XL để cho thoải mái nhá



hidemo

12:11

Giá bộ sp33 là bao nhiêu vậy shop



Manh Dv

12:11



Giá sản phẩm sp33 là 299k nhé



hidemo

12:13

Giá bộ sp4 là bao nhiêu vậy shop

Nhập nội dung tin nhắn

Aa



TO



Gửi

12:37

Manh Dv

Manh Dv 12:11
↳ RE 
Giá sản phẩm sp33 là 299k nhé

hidemo 12:13
 Giá bộ sp4 là bao nhiêu vậy shop

Manh Dv 12:14
 ↳ RE 
Giá sản phẩm sp4 là 299k nhé

hidemo 12:15
 Xin giá mẫu sp5 với shop ơi

Manh Dv 12:15
 ↳ RE 
Giá sản phẩm sp5 là 299k nhé

hidemo 12:25
 Bộ này mặc mát không shop

Manh Dv 12:25
 ↳ RE 
Sản phẩm bên mình sử dụng chất vải thun co dãn 4 chiều, chống tia uv, kháng khuẩn, giặt

Nhập nội dung tin nhắn

Aa  TO   Gửi

12:37



Manh Dv



ooo



hidemo

12:25

Bộ này mặc mát không shop



Manh Dv

12:25



Sản phẩm bên mình sử dụng chất vải thun co dãn 4 chiều, chống tia uv, kháng khuẩn, giặt không phai màu, shop xin đảm bảo chất lượng nhé



hidemo

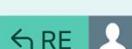
12:26

Bộ này khi giặt có bị phai màu không nhỉ



Manh Dv

12:26



Sản phẩm bên mình sử dụng chất vải thun co dãn 4 chiều, chống tia uv, kháng khuẩn, giặt không phai màu, shop xin đảm bảo chất lượng nhé



hidemo

12:26

Được đổi trả hàng trong bao lâu vậy shop



Manh Dv

12:26



Chúna tôi khôn̄a hiểu vêu cầu của ban! Ban

Nhập nội dung tin nhắn

Aa



TO



Gửi

12:38



Manh Dv



ooo



hidemo

12:33

Được đổi trả hàng trong bao lâu vậy shop



Manh Dv

12:33



Với chính sách trả hàng thì bạn có thể trả hàng trong vòng 7 ngày, bạn vui lòng để nguyên tem mác trên sản phẩm giúp mình nhé



hidemo

12:34

Tức là trong 7 ngày là được trả lại hàng đúng không



Manh Dv

12:34



Với chính sách trả hàng thì bạn có thể trả hàng trong vòng 7 ngày, bạn vui lòng để nguyên tem mác trên sản phẩm giúp mình nhé



hidemo

12:35

Có ship về Thái Nguyên không shop



Manh Dv

12:35



Nhập nội dung tin nhắn

Aa



TO



Gửi

12:38



Manh Dv



ooo

nhưng trong vòng 7 ngày, bạn vẫn có
nguyên tem mác trên sản phẩm giúp mình
nhé



hidemo

12:35

Có ship về Thái Nguyên không shop



Manh Dv

12:35



Bên mình đang khuyến mại free ship toàn
quốc nhé



hidemo

12:35

Mình thanh toán cod được không



Manh Dv

12:35



Tất cả sản phẩm đều ship COD nhận tiền khi
giao hàng.



hidemo

12:36

Cảm ơn



Manh Dv

12:36



Cảm ơn bạn đã phản hồi

Nhập nội dung tin nhắn

Aa



TO



Gửi