

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH



LUẬN VĂN TỐT NGHIỆP

**Xây dựng Chatbot cho dịch vụ đặt tiệc nhà
hàng sử dụng framework Rasa
Building a Chatbot for restaurant booking
service using Rasa framework**

HỘI ĐỒNG: Khoa học máy tính
GVHD: ThS. Võ Thanh Hùng
GVPB: PGS.TS Quản Thành Thơ
—o0o—
SVTH: Trần Đình Hậu
MSSV: 1711261

TP. HỒ CHÍ MINH, 5/2023

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA: KH & KT Máy tính _____

NHIỆM VỤ LUẬN VĂN/ ĐỒ ÁN TỐT NGHIỆPBỘ MÔN: KHMT _____
Chú ý: Sinh viên phải dán tờ này vào trang nhất của bản thuyết trình

HỌ VÀ TÊN: Trần Đình Hậu _____

MSSV: 1711261 _____

NGÀNH: Khoa học máy tính

LỚP: MT17KH01

1. Đầu đề luận văn/ đồ án tốt nghiệp:

Tiếng Việt: Xây dựng Chatbot cho dịch vụ đặt tiệc nhà hàng sử dụng framework Rasa

Tiếng Anh: Building a chatbot for restaurant booking service using Rasa framework

2. Nhiệm vụ (yêu cầu về nội dung và số liệu ban đầu):

Hiện nay nhu cầu sử dụng dịch vụ nhà hàng đang ngày càng phát triển, nhằm mang đến cho khách hàng sự tiện ích tốt nhất khi có nhu cầu sử dụng nhà hàng tại một thời điểm nhất định. Chatbot sẽ mang lại tiện ích tốt nhất cho khách hàng khi khách hàng có thể đặt nhà hàng tại bất kỳ thời điểm nào trong ngày. Chatbot cũng sẽ mang đến cho nhà hàng lợi ích khi giảm tải được khối lượng công việc cho nhân viên. Giúp đem lại hiệu quả cao cho nhà hàng.

Giai đoạn đề cương:

- Tìm hiểu về framework Rasa.
- Tìm hiểu về thuật toán NLU sử dụng training model.
- Xây dựng kịch bản đặt tiệc nhà hàng cho khách hàng, xây dựng dữ liệu.
- Đưa dữ liệu vào trong Rasa và training model.
- Đánh giá hiệu quả của model, cải thiện chất lượng mô hình

Giai đoạn luận văn:

- Cải thiện chất lượng mô hình
- Hoàn thiện hệ thống
- Viết demo sản phẩm
- Viết báo cáo

3. Ngày giao nhiệm vụ: 30/01/2023**4. Ngày hoàn thành nhiệm vụ: 09/06/2023****5. Họ tên giảng viên hướng dẫn:****Phần hướng dẫn:**

1) ThS. Võ Thanh Hùng _____

Nội dung và yêu cầu LVTN/ ĐATN đã được thông qua Bộ môn.

Ngày 15 tháng 5 năm 2023
CHỦ NHIỆM BỘ MÔN
(Ký và ghi rõ họ tên)

GIẢNG VIÊN HƯỚNG DẪN CHÍNH
(Ký và ghi rõ họ tên)

PHẦN DÀNH CHO KHOA, BỘ MÔN:

Người duyệt (châm sơ bộ): _____

Đơn vị: _____

Ngày bảo vệ: _____

Điểm tổng kết: _____

Nơi lưu trữ LVTN/ĐATN: _____

Lời cam đoan

Tôi xin cam đoan đề tài này là sản phẩm của riêng cá nhân tôi, do tôi tự nghiên cứu, đọc, dịch tài liệu, tổng hợp và thực hiện dưới sự hướng dẫn của Ths. Võ Thanh Hùng. Nội dung lý thuyết trong đề tài này tôi có sử dụng một số tài liệu tham khảo như đã trình bày trong phần tài liệu tham khảo. Kết quả được công bố trong đề tài này là trung thực. Các tài liệu sử dụng trong luận văn có nguồn gốc, xuất xứ rõ ràng.

Lời cảm ơn / Lời ngỏ

Quá trình thực hiện đề cương luận văn và luận văn tốt nghiệp là giai đoạn quan trọng nhất trong quãng đời mỗi sinh viên. Luận văn tốt nghiệp là tiền đề nhằm trang bị cho chúng em những kỹ năng nghiên cứu, những kiến thức quý báu trước khi lập nghiệp.

Trước hết, chúng em xin chân thành cảm ơn quý Thầy cô khoa Khoa học và kỹ thuật máy tính. Đặc biệt là các Thầy cô trong bộ môn Khoa học máy tính đã tận tình chỉ dạy và trang bị cho em những kiến thức cần thiết trong suốt quá trình ngồi trên ghế giảng đường, làm nền tảng cho em có thể hoàn thành được bài luận văn này. Đặc biệt, em xin gửi lời cảm ơn sâu sắc đến thầy ThS. Võ Thanh Hùng đã tận tình hướng dẫn, chỉ dẫn hướng đi cho em trong suốt quá trình thực hiện đề cương luận văn cũng như luận văn tốt nghiệp. Xin cảm ơn các thành viên lớp MT17KH01 đã cùng đồng hành và giúp đỡ mình trong suốt 2 năm học tập và rèn luyện vừa qua. Cuối cùng và quan trọng nhất, em xin gửi lời cảm ơn đến gia đình đã luôn tin tưởng, động viên và là chỗ dựa vững chắc để em có thể tiếp tục học tập.

Với kiến thức và hiểu biết có hạn, chắc chắn luận văn tốt nghiệp khó tránh khỏi thiếu sót, em rất mong nhận được nhiều ý kiến đóng góp, phê bình của quý thầy cô, bạn bè và các kỹ sư, chuyên gia trong và ngoài ngành Khoa học máy tính để luận văn sau này này được hoàn thiện hơn. Em xin chân thành cảm ơn và kính chúc tất cả mọi người mọi điều tốt đẹp nhất

Tóm tắt nội dung

Trong ngành kinh doanh dịch vụ hiện nay, việc tăng cường trò chuyện và tương tác khách hàng là một trong những nhiệm vụ quan trọng nhằm góp phần tạo sự hài lòng cho khách hàng. Việc nhân viên trực tiếp trao đổi với khách hàng gặp nhiều khó khăn khi thời điểm tương tác với khách hàng ở bất kỳ thời điểm nào trong ngày. Nên để đáp ứng được điều đó thì doanh nghiệp phải tốt thêm chi phí để thuê nhân viên để đáp ứng nhân viên để tương tác với khách hàng. Một vấn đề tiếp theo là nếu tại một thời điểm trong ngày có thể tiếp nhận nhiều khách hàng trong cùng một lúc, với lý do này doanh nghiệp cũng phải tốn thêm nhiều chi phí để có một lượng nhân viên đủ để đáp ứng tức thời tương tác khách hàng. Nếu số lượng nhân viên hạn chế để tiếp nhận khách hàng thì sẽ gây ra sự phản hồi chậm trễ đến khách hàng, điều này sẽ góp phần tạo nên sự không hài lòng đối với khách hàng.

Hiểu về vấn đề đó, đề tài này sẽ xây dựng một bot có chức năng chat trực tuyến với khách hàng. Chatbot có thể hoạt động 24/7 nên có thể tiếp nhận trao đổi với khách hàng tại mọi thời điểm trong ngày. Chatbot có thể tiếp nhận và phản hồi nhanh chóng đến khách hàng, chatbot có thể làm cho khách hàng có những sự trải nghiệm mới khi tương tác giống như đang trò chuyện với nhân viên. Chatbot có thể tương tác nhiều khách hàng cùng một lúc, điều này tránh sự không hài lòng cho khách hàng. Chatbot Rasa được sử dụng trong đề tài này là một nguồn mở, có thể sử dụng cho nhiều chủ đề khác nhau tùy vào dữ liệu được sử dụng.

Chatbot trong đề tài này được sử dụng trong nhà hàng nhằm để tương tác với khách hàng trong vấn đề giải quyết nhu cầu của khách hàng trong việc sử dụng dịch vụ của nhà hàng. Chatbot được sử dụng bằng framework Rasa. Framework này hỗ trợ nhiều kênh kết nối, vì vậy có thể được sử dụng để kết nối đến nhiều ứng dụng nhắn tin để tăng lượng khách hàng tương tác. Trong đề tài này, bot Rasa sẽ được tạo và sử dụng để đặt tiệc cho khách hàng. Bot này sẽ được kết nối với fanpage facebook để sử dụng.

Mục lục

1 Mở đầu	1
1.1 Giới thiệu đề tài	1
1.2 Mục tiêu của đề tài	3
1.2.1 Về kiến thức	3
1.2.2 Về sản phẩm	3
1.3 Bố cục của luận văn	4
2 Tổng quan về Chatbot	5
2.1 Giới thiệu	5
2.2 Một số framework Chatbot	7
2.2.1 Microsoft Bot Framework	7
2.2.2 Google Dialogflow	7
2.2.3 Wit.ai	8
2.2.4 IBM Watson	8
2.2.5 Amazon Lex Framework	8
2.2.6 Rasa Framework	9
2.3 Tổng quan về Rasa Framework	9
2.3.1 Hiểu ngôn ngữ tự nhiên	11
2.3.2 Xử lý đối thoại	17
2.4 Thuật toán sử dụng trong NLU	19
2.4.1 Mạng hồi quy RNN	19
2.4.2 Long Short-Term Memory - LSTM	20
3 Xây dựng luồng hội thoại đặt tiệc tại nhà hàng	22
3.1 Tổng quát câu chuyện tại tương tác khách hàng	22
3.2 Các thành phần câu chuyện	24
3.2.1 Kịch bản chào và hỏi dịch vụ	24
3.2.2 Kịch bản đặt đồ ăn của khách hàng	25
3.2.3 Kịch bản thay đổi thông tin món ăn	27
3.2.4 Kịch bản lấy thông tin về người tham gia và thời gian sử dụng	27
3.2.5 Kịch bản lấy thông tin khách hàng	29
3.2.6 Kịch bản xác nhận thông tin khách hàng	30
3.2.7 Kịch bản chỉnh sửa thông tin khách hàng	30
4 Xây dựng Chatbot	33
4.1 Xây dựng câu chuyện trong Rasa	33
4.2 Xây dựng quy tắc trong Rasa	37
4.3 Truy xuất và lưu thông tin từ người dùng	39

4.4	Dữ liệu đào tạo NLU	42
4.5	Tạo mẫu câu phản hồi người dùng	49
4.6	Cấu hình hành động Rasa	50
4.7	Cấu hình pipeline xử lý tin nhắn từ người dùng	54
4.8	Cấu hình policies quyết định hành động trả về	55
4.9	Xây dựng cơ sở dữ liệu	55
4.10	Kết nối chatbot Rasa với fanpage facebook	58
5	Thử nghiệm và đánh giá hệ thống	63
5.1	Thử nghiệm	63
5.2	Đánh giá hệ thống	69
5.2.1	Phương pháp đánh giá	69
5.2.2	Đánh giá dự đoán thực thể	71
5.2.3	Đánh giá dự đoán ý định	74
5.2.4	Đánh giá mô hình hiểu ngôn ngữ	76
5.2.5	Đánh giá mô hình đối thoại	80
6	Tổng kết	83
6.1	Kết luận	83
6.2	Các hướng phát triển	84

Danh sách từ viết tắt

STT	Ký hiệu từ viết tắt	Chữ viết đầy đủ
1	NLP	Natural Language Processing
2	NLU	Natural Language Understanding
3	IoT	Internet of Things
4	AI	Artificial Intelligence
5	AWS	Amazon Web Services
6	API	Application Programming Interface
7	CDD	Conversation-Driven Development
8	TED	Transformer Embedding Dialogue
9	DAG	Directed Acyclic Graph
10	DIET	Dual Intent và Entity Transformer
11	CRF	Conditional Random Field
12	MITIE	MitieEntityExtractor
13	URL	Uniform Resource Locator
14	RNNL	Recurrent Neural Network
15	LSTM	Long short term memory

Danh sách bảng

5.1	Bảng ước lượng độ chính xác dự đoán từng thực thể	73
5.2	Bảng ước lượng độ chính xác dự đoán từng ý định	76
5.3	Bảng ước lượng độ chính xác dự đoán từng thực thể dựa trên dữ liệu từ khách hàng	79
5.4	Bảng ước lượng độ chính xác dự đoán từng ý định dựa trên dữ liệu của khách hàng	79
5.5	Bảng ước lượng độ chính xác dự đoán các hành động	81

Danh sách hình vẽ

2.1	Minh họa chatbot tương tác với con người [3]	5
2.2	Kiến trúc cơ bản của rasa [10]	10
2.3	Kiến trúc Rasa [10]	10
2.4	Các ứng dụng của Rasa [12]	11
2.5	Hệ thống độc lập linear pipeline giữa Rasa NLU và Rasa Core [13]	12
2.6	Đồ thị dạng cây DAG biểu diễn pipeline trong Rasa [13]	12
2.7	Rasa NLU và Rasa Core hoạt động [15]	18
2.8	Recurrent neural network và tính toán chuyển tiếp của mạng RNN [16]	20
2.9	Kiến trúc mạng LSTM [17]	21
3.1	Luồng câu chuyện cơ bản đặt tiệc tại nhà hàng	22
3.2	Luồng câu chuyện mở rộng chỉnh sửa món ăn của khách hàng	24
3.3	Luồng câu chuyện mở rộng chỉnh sửa thông tin của khách hàng	24
3.4	Luồng kịch bản chào và hỏi dịch vụ	25
3.5	Luồng kịch bản đặt món ăn	26
3.6	Luồng kịch bản chỉnh sửa thông tin món ăn	28
3.7	Luồng kịch bản lấy thông tin về người tham gia và thời gian sử dụng	29
3.8	Luồng kịch bản lấy thông tin về khách hàng	29
3.9	Luồng kịch bản xác nhận thông tin	30
3.10	Luồng kịch bản chỉnh sửa thông tin khách hàng	31
4.1	Mô hình dữ liệu thực thể - liên kết giữa thông tin khách hàng và món ăn	56
4.2	Ràng buộc lược đồ cơ sở dữ liệu quan hệ của thông tin khách hàng	56
4.3	Sơ đồ lớp dữ liệu thông tin khách hàng và các món ăn	57
4.4	Ví dụ về thông tin khách hàng được lưu trên cơ sở dữ liệu	57
4.5	Ví dụ về các món ăn của khách hàng	58
4.6	Kiến trúc luồng dữ liệu đi qua các cổng kết nối	58
4.7	Tạo fanpage facebook	59
4.8	Lấy mã fanpage facebook	60
4.9	Lấy khóa bí mật của fanpage facebook	60
4.10	Khởi động đường hầm trong ngrok	61
4.11	Sử dụng URL gọi lại cho fanpage	61
4.12	Cấu hình trường trong fanpage	61
4.13	Trạng thái kết nối fanpage thành công	62
5.1	Kiểm tra lời chào và hỏi dịch vụ	63
5.2	Kiểm tra đặt món ăn theo chỉ số	64
5.3	Kiểm tra đặt món ăn theo tên	64
5.4	Kiểm tra đặt món ăn theo tên không đầy đủ	65
5.5	Kiểm tra đặt món ăn và xác nhận thay đổi danh sách món ăn	65

5.6	Kiểm tra thay đổi các thành phần trong danh sách món ăn	66
5.7	Kiểm tra lấy thông tin của khách hàng	67
5.8	Kiểm tra thay đổi thông tin về số lượng người sử dụng	67
5.9	Kiểm tra thay đổi thông tin về tên và số điện thoại	68
5.10	Kiểm tra xác nhận thông tin chính xác	68
5.11	Kiểm tra bản dữ liệu cập nhật	68
5.12	Kiểm tra các thông tin khách hàng trong bản dữ liệu	69
5.13	Các chỉ số để đánh giá mô hình [18]	70
5.14	Ma trận nhầm lẫn dự đoán thực thể	71
5.15	Biểu đồ độ tin cậy các dự đoán thực thể	72
5.16	Ma trận nhầm lẫn dự đoán ý định	74
5.17	Biểu đồ độ tin cậy các dự đoán ý định	75
5.18	Ma trận nhầm lẫn trích xuất thực thể từ người dùng	77
5.19	Ma trận nhầm lẫn dự đoán ý định từ người dùng	78
5.20	Ma trận nhầm lẫn các dự đoán hành động	82

Chương 1

Mở đầu

1.1 Giới thiệu đề tài

Theo sự phát triển khoa học và công nghệ ngày càng được phổ biến, ứng dụng của chúng mang lại ngày càng rộng rãi trong mọi lĩnh vực trong cuộc sống. Hiện nay, với sự phát triển mạnh mẽ của thương mại điện tử trong các lĩnh vực về dịch vụ. Bằng việc phân tích các khó khăn của doanh nghiệp cùng với nhu cầu thông tin về sản phẩm dưới góc độ của khách hàng. Chính vì vậy, sự trao đổi của doanh nghiệp và khách hàng là cần thiết để giúp doanh nghiệp tăng lợi nhuận, giúp khách hàng có thêm thông tin về sản phẩm và dịch vụ. Nhu cầu của khách hàng có thể được thực hiện vào bất kỳ thời điểm nào nên để đáp ứng nhu cầu của khách hàng thì doanh nghiệp tốn thêm chi phí nhân sự. Chính vì vậy, để giải quyết bài toán này thì Chatbot là một giải pháp mà doanh nghiệp về dịch vụ đang hướng đến. Chatbot về dịch vụ là giải pháp thay thế nhân sự tiếp nhận một số thông tin cơ bản của khách hàng và giải quyết để đáp ứng nhu cầu đó. Chatbot dịch vụ tự động góp phần giảm chi phí nhân sự, tăng hiệu quả bán hàng, tăng khả năng trao đổi với khách hàng và tăng khả năng tương tác với khách hàng.

Chatbot là một hệ thống nhẫn tin tự động sử dụng các thuật toán để trích xuất thông tin dữ liệu từ người dùng để quyết định phản hồi lại cho người dùng một thông tin cần thiết tại bất kỳ trong thời gian thực. Chatbot cho phép người dùng tương tác với máy tính giống như tương tác với con người. Chatbot được ứng dụng rộng rãi trong nhiều lĩnh vực hiện nay như chăm sóc khách hàng, trợ lý ảo, quy trình trao đổi trong các công ty, v.v. Việc sử dụng chatbot trong doanh nghiệp giúp thúc đẩy và tiết kiệm được thời gian và chi phí cho doanh nghiệp. Đồng thời cho phép doanh nghiệp dễ dàng giải quyết nhiều loại truy vấn và các vấn đề của khách hàng. Với chatbot, doanh nghiệp có thể mở rộng quy mô giải quyết vấn đề và chủ động trong cùng một lúc. Chatbot có thể sử dụng 24/7 và tốc độ phản hồi nhanh nên chatbot có thể làm tăng doanh số bán hàng, hạn chế việc phản hồi chậm trễ gây ra sự không hài lòng cho khách hàng. Chatbot có

thể liên kết được với phần lớn các nền tảng mạng xã hội như zalo, facebook, telegram, webexTeam,... Việc chatbot có thể hoạt động liên tục trên các mạng xã hội này sẽ làm tăng lượng khách hàng mới.

Hiểu được nhu cầu trên, các công ty lớn đã và đang phát triển các khung cho phép các nhà phát triển truy cập và xây dựng các tính năng phù hợp với dịch vụ của doanh nghiệp. Các tài nguyên có sẵn trong framework giúp nhà phát triển tiết kiệm thời gian hơn so với phát triển một chatbot từ đầu [1]. Các khung cũng đóng vai trò là phần mềm trung gian cho phép nhà phát triển kết nối với các nền tảng mạng xã hội như facebook, zalo, slack, telegram, .v.v. Việc sử dụng framework Chatbot sẽ mang lại hiệu quả cao hơn và giá thành rẻ hơn so với khi viết lại một Chatbot từ đầu [2].

Đề tài "Xây dựng Chatbot cho dịch vụ đặt tiệc nhà hàng sử dụng framework Rasa" là một đề tài mang tính ứng dụng cao và theo xu hướng phát triển của công nghệ. Đây là một trong những giải pháp hiệu quả trong việc trao đổi và chăm sóc khách hàng không chỉ trong lĩnh vực nhà hàng mà còn trong các lĩnh vực khác như giáo dục, y tế, tư vấn tâm lý, .v.v.

Đề tài này tập trung thiết kế và xây dựng một hệ thống Chatbot để sử dụng trong nhà hàng với mục tiêu giúp khách hàng đặt tiệc nhà hàng bao gồm các món ăn, số lượng chỗ ngồi và thời gian sử dụng. Sản phẩm được xây dựng trong đề tài này sẽ giúp cho nhà hàng có thể giảm được chi phí nhân sự để tư vấn và trao đổi với khách hàng. Sản phẩm này có thể giúp việc trao đổi với nhiều khách hàng trong cùng một thời gian nên có thể giúp nhà hàng tăng doanh số. Sản phẩm này giúp nhà hàng có thể tương tác với khách hàng ở mọi khung giờ trong ngày khi khách hàng có nhu cầu đặt tiệc. Chính vì vậy đề tài này có tính ứng dụng thực tiễn cao và có tiềm năng để phát triển trong mọi lĩnh vực trong cuộc sống.

Đặt tiệc nhà hàng là một chủ đề đóng chỉ xảy ra trong ngữ cảnh tại nhà hàng nên đề tài này sẽ xây dựng Chatbot dựa trên ngữ cảnh đặt tiệc tại nhà hàng. Luồng câu chuyện sẽ được xây dựng nhằm đáp ứng các yêu cầu cơ bản giúp khách hàng có thể đặt được tiệc được tại nhà hàng cũng như cung cấp các thông tin cơ bản để nhân viên có thể định danh được khách hàng nào đặt tiệc, đặt bàn bao nhiêu người, thời gian sử dụng và các món ăn để chuẩn bị trước. Đề tài này sẽ sử dụng framework Rasa làm công cụ để phát triển Chatbot. Với bài toán và cùng với các tính năng phù hợp với câu chuyện này thì framwork này sẽ là lựa chọn ưu tiên cho Chatbot với ngữ cảnh đóng được xác định trước. Cùng với những tính năng có sẵn từ framework này như kết nối đa kênh thì sẽ giúp cho nhà hàng có thể tiếp cận mạnh mẽ đến khách hàng. Với độ linh hoạt về cấu hình luồng câu chuyện mà framework này sẽ giúp cho nhà phát triển có thể tùy chỉnh mở rộng các

tính năng một cách dễ dàng.

1.2 Mục tiêu của đề tài

1.2.1 Về kiến thức

Có kiến thức cơ bản về các loại Chatbot, tùy thuộc vào tình huống và nhu cầu đặt ra thì loại Chatbot sử dụng để phát triển phù hợp.

Có kiến thức cơ bản về xử lý ngôn ngữ tự nhiên NLP, cách Chatbot phân tích và hiểu ý định của người dùng. Cách thức bot trích xuất được thực thể quan trọng từ câu nói của khách hàng.

Có kiến thức về cách xây dựng kịch bản giao tiếp của người chăm sóc khách hàng đối với khách hàng thông qua tin nhắn trực tiếp. Xác định các trường hợp sẽ xảy ra trong quá trình đặt tiệc của khách hàng. Các thông tin cần thiết của khách hàng khi sử dụng dịch vụ tại nhà hàng.

Có kiến thức về cách kiểm soát về luồng dữ liệu của thông tin từ người dùng đi qua Rasa NLU và sử dụng Rasa action để phản hồi lại người dùng.

Có kiến thức về việc lưu trữ dữ liệu được trích xuất ra từ câu nói của khách hàng để tạo mẫu câu trả về cho người dùng và lưu vào cơ sở dữ liệu nội bộ của nhà hàng.

1.2.2 Về sản phẩm

Tùy vào lĩnh vực kinh doanh và yêu cầu người dùng mà có thể tạo ra các chatbot có thể phản hồi lại đúng về chủ đề dịch vụ mà khách hàng đang tìm kiếm, có nhu cầu sử dụng. Dịch vụ kinh doanh nhà hàng là chủ đề mà chatbot có thể được sử dụng để chăm sóc và hỗ trợ khách hàng một cách nhanh chóng và hiệu quả. Ngoài ra, chatbot giúp khách hàng có được trải nghiệm đặt dịch vụ tốt hơn khi tạo cho khách hàng có được cảm giác đang chat trực tiếp với nhân viên của khách hàng. Việc sử dụng chatbot trong nhà hàng được thiết kế để phục vụ khách hàng khi khách hàng có nhu cầu sử dụng dịch vụ nhà hàng như đặt chỗ, quản lý đặt chỗ và phản hồi lại cho khách hàng. Chatbot trong lĩnh vực nhà hàng giúp làm giảm chi phí nhân sự của nhà hàng giúp tiết kiệm chi phí cho nhà hàng. Chatbot giúp tăng khả năng tương tác với nhiều khách hàng tại cùng một thời điểm giúp nhà hàng góp phần tăng lợi nhuận đến cho nhà hàng. Giúp người quản lý có được thông tin của các khách hàng đang có nhu cầu sử dụng dịch vụ tại nhà hàng.

Đối với bài toán đặt tiệc tại nhà hàng thì luận văn này sẽ tập trung xây dựng mô hình hệ thống Chatbot để tương tác với khách hàng dựa vào framwork Rasa. Kết hợp với các

kiến thức nền tảng về NLP để có thể tạo dữ liệu NLU để bot có thể học được dữ liệu về ý định khi được huấn luyện.

1.3 Bố cục của luận văn

Luận văn được chia thành 6 chương:

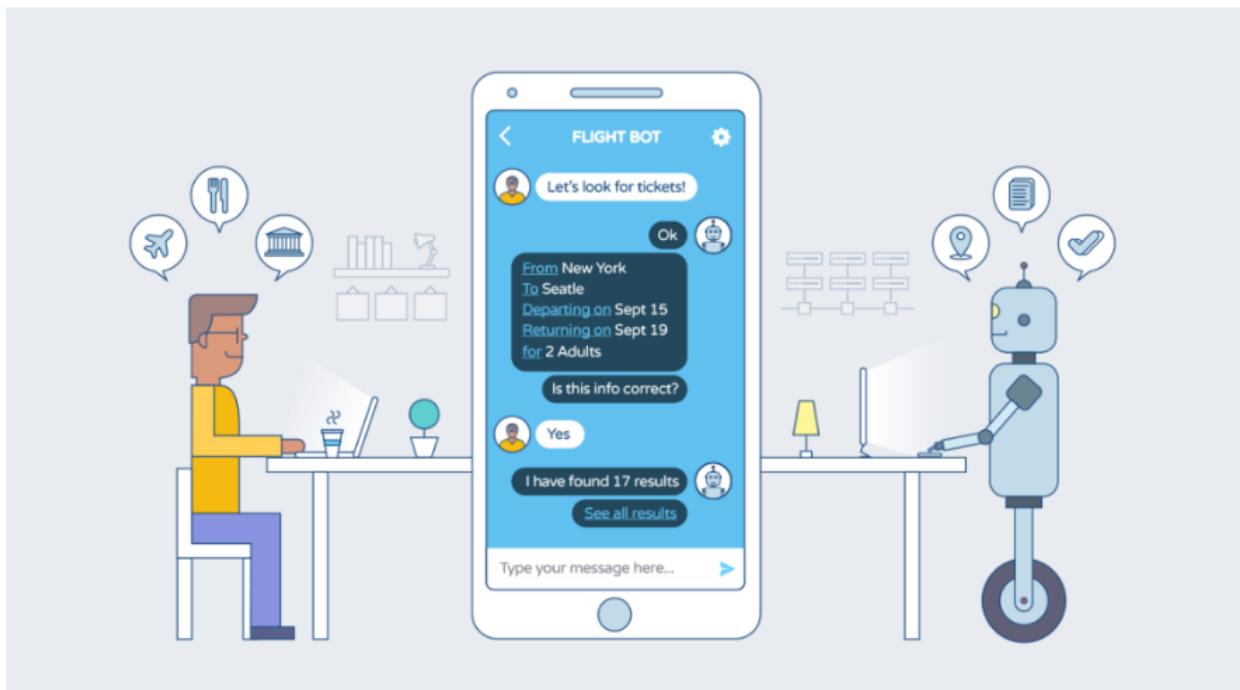
- Chương 1: Mở đầu: Chương này sẽ giới thiệu đề tài và nêu một số ứng dụng của đề tài, phân tích đề tài và xác định mục tiêu của đề tài.
- Chương 2: Giới thiệu tổng quan về Chatbot: Chương này sẽ giới thiệu những kiến thức tổng quan về hệ thống Chatbot.
- Chương 3: Xây dựng luồng hội thoại đặt tiệc tại nhà hàng.
- Chương 4: Xây dựng Chatbot.
- Chương 5: Thử nghiệm và đánh giá hệ thống.
- Chương 6: Kết luận và kiến nghị.

Chương 2

Tổng quan về Chatbot

2.1 Giới thiệu

Sự tiến bộ vượt bậc của công nghệ đã chứng kiến sự gia tăng các doanh nghiệp chuyển từ nền tảng truyền thống sang nền tảng kỹ thuật số để giao dịch và trao đổi với khách hàng. Sự tiện ích mà thông qua công nghệ đang được các công ty, doanh nghiệp triển khai và sử dụng các kỹ thuật trí tuệ nhân tạo trên nền tảng của kỹ thuật số của công ty. Một trong số đó chính là chatbot để tạo ra trợ lý ảo có thể chăm sóc, tư vấn cho khách hàng và thậm chí cho các cá nhân ở công ty. Chatbot là một phần mềm máy tính giúp phát triển cuộc trò chuyện với người dùng một cách tự nhiên. Sự phát triển không ngừng của công nghệ thông tin và truyền thông đã làm cho trí tuệ nhân tạo trở nên phức tạp và thông minh hơn.



Hình 2.1: Minh họa chatbot tương tác với con người [3]

Chatbot được hiểu đơn giản như một cái máy có thể giao tiếp tự nhiên giống con người [3]. Chatbot tương tác với con người như một hệ thống trả lời tin nhắn nhanh chóng và tự động bằng cách xây dựng các mô hình, kịch bản tương tác với con người. Hiện nay, chatbot được sử dụng trong rất nhiều các lĩnh vực dịch vụ khác nhau (như hình 2.1). Trong mỗi lĩnh vực, kịch bản được sử dụng cho chatbot phần lớn dựa vào sự đối thoại giữa người này với người khác trong mỗi môi trường khác nhau. Các kịch bản sử dụng cho chatbot được xây dựng thành các luồng hội thoại mà một người có thể sử dụng để giao tiếp.

Chatbot hoạt động bằng cách phân tích và xác định mục đích yêu cầu của người dùng để trích xuất các thực thể có liên quan, đây là nhiệm vụ quan trọng nhất của chatbot. Sau khi phân tích được thực hiện, phản hồi thích hợp sẽ được gửi đến người dùng. Chatbot mô phỏng cuộc trò chuyện của con người thông qua lệnh thoại hoặc trò chuyện bằng văn bản hoặc cả hai. Chatbot tiếp cận khách hàng vào mọi thời điểm trong ngày, trong tuần và không bị giới hạn về thời gian hoặc địa điểm. Điều này làm cho việc triển khai nó trở nên hấp dẫn đối với nhiều doanh nghiệp có thể không có nhân lực hoặc nguồn tài chính để giữ nhân viên làm việc 24/7.

Chatbot sử dụng các hoạt động của con người như đưa ra quyết định tại một thời điểm cụ thể, thực hiện các công việc hàng ngày, trả lời người dùng nhanh chóng và giải quyết các truy vấn theo cách giống như cách con người sẽ làm. Có rất nhiều tổ chức điện tử như kinh doanh, giải trí, hỗ trợ ảo và một số tổ chức khác. Chatbot sẽ sử dụng cơ sở dữ liệu, thu thập được từ phân tích câu văn từ tin nhắn từ người dùng, các câu hỏi và câu trả lời đã được huấn luyện từ trước để phản hồi lại người dùng. Chatbot được huấn luyện, cải thiện trong thời gian dài sẽ cải thiện được chính xác và độ tin cậy khi đưa ra các quyết định trả về lời nhắn đến khách hàng.

Hiện nay có rất nhiều giải pháp AI và Chatbot để phát triển để giải quyết các yêu cầu của doanh nghiệp và tổ chức. Phần lớn các công cụ phát triển Chatbot hiện nay dựa trên hai loại Chatbot chính:

- Chatbot dựa trên nguyên tắc: Chatbot này dựa trên quy tắc sử dụng logic để tạo ra các luồng hội thoại. Mô hình Chatbot này mang lại khả năng kiểm soát tinh chỉnh và linh hoạt. Loại bot này tương tác khá cụ thể và có cấu trúc nên Chatbot này có xu hướng sử dụng cho các trường hợp cụ thể trong tình huống cụ thể và thường hay tương tác cơ bản. Nên nền tảng Chatbot này được sử dụng trên các trang thông tin thương mại điện tử.
- Chatbot AI: Là mô hình dạng học sâu phức tạp hơn so với Chatbot dựa trên quy tắc và có xu hướng mang tính đối thoại, hướng dữ liệu và dự đoán nhiều hơn.

Đề tài luận văn này sẽ sử dụng và phát triển Chatbot theo mô hình dựa trên nguyên tắc trong một chủ đề đóng theo ngữ cảnh dịch vụ trong nhà hàng. Các phần tiếp theo trong luận văn sẽ mô tả các xây dựng luồng hội thoại, cách hiện thực các thành phần để tương tác với khách hàng để đạt được mục tiêu của khách hàng và nhà hàng.

2.2 Một số framework Chatbot

2.2.1 Microsoft Bot Framework

Framework này có tên khác là Azure được phát triển bởi Microsoft giúp xây dựng, kết nối và quản lý các Chatbot tương tác. Framework này hoạt động tốt nhất khi học tích cực (Active learning), thuật toán này ưu tiên dữ liệu cần được gắn nhãn để có hiệu quả cao nhất đến việc đào tạo mô hình được giám sát. Học tích cực này có thể được sử dụng trong các tình huống có được dữ liệu đào tạo quá lớn không thể gắn nhãn và cần thực hiện một số ưu tiên để gắn nhãn dữ liệu thông minh.

Framework Chatbot này cung cấp các mô hình dựng sẵn để kết nối với các kênh nền tảng như Web, Skype, .v.v. Chatbot này cung cấp cho người phát triển toàn quyền kiểm soát trải nghiệm xây dựng của bot, quyền truy cập vào các chức năng và trình kết nối khác nhau.

Hạn chế lớn của framework này là không hỗ trợ xử lý ngôn ngữ tự nhiên do đó bot sẽ không hiểu được lời nói của con người [5], nên cần phải sử dụng một bộ công cụ riêng biệt để xử lý đầu vào của người dùng trước khi đưa vào framwork này.

2.2.2 Google Dialogflow

Framework này được phát triển bởi Google nhằm giúp nhà phát triển dễ dàng hơn khi phát triển các sản phẩm có sự tương tác với con người bằng đoạn văn bản hoặc giọng nói. Framework này dễ dàng mở rộng quy mô người dùng. Về cơ bản Framework này là nền tảng đơn giản để xây dựng Chatbot nhanh, đơn giản và có thể kết hợp với nhiều nền tảng mạng xã hội.

Mặc dù Dialogflow khá trực quan nhưng nền tảng này có sự kém linh hoạt [7], khi muốn thêm một ý định với một mục đích khác thì phải thay đổi ý định hiện tại, điều này gây ra sự mất thời gian buộc người phát triển phải suy nghĩ về luồng hội thoại phân cấp trước khi hiện thực. Về việc hỗ trợ khách hàng, Chatbot này không quá mạnh về việc tương tác, hỗ trợ khách hàng trực tiếp. Ngoài ra, việc tạo Chatbot này có nhiều bước thủ công nên việc mở rộng cũng bị hạn chế. Chatbot này phần lớn chỉ mang tính chất thử nghiệm hơn là ứng dụng trong thực tế.

2.2.3 Wit.ai

Wit.ai là một framework Chatbot mã nguồn mở với khả năng xử lý ngôn ngữ tự nhiên mạnh mẽ do Facebook phát triển. Framework này cũng cung cấp khả năng tích hợp với các kênh khác như Web, ứng dụng di động, thiết bị đeo tay và các thiết bị IoT. Vì framework này là một mã nguồn mở nên có được nguồn tài nguyên lớn, người phát triển có thể dựa vào các tính năng mà cộng đồng đã phát triển được để học hỏi và sử dụng.

Nhược điểm của framework này là việc đào tạo sử dụng NLP khá khó khăn [6], phần lớn nhà phát triển đều chỉ ra rằng việc đào tạo NLP trong framework này rất khó khăn khi xây dựng Chatbot của mình

2.2.4 IBM Watson

Framework này sử dụng công cụ hiện đại như học máy, trí tuệ nhân tạo và NLP để học hỏi từ các cuộc trò chuyện trước đó của khách hàng [4]. Đây là loại Chatbot AI mang xu hướng học từ đoạn đối thoại và dự đoán. Nền tảng này cung cấp các công cụ về NLP mạnh mẽ, nên Chatbot này có thể dựa vào một số câu trả lời từ khách hàng để đưa ra ý kiến phù hợp nhất đến khách hàng.

Bên cạnh thế mạnh về công nghệ mà framework này đem lại thì nền tảng này tích hợp chậm đến các trang thương mại điện tử. Chatbot phát triển bằng framework này cần có thời gian để thu thập được nhiều dữ liệu, điều này tốn nhiều thời gian và có thể gặp nhiều nguồn dữ liệu không uy tín làm giảm khả năng dự đoán của bot. Framework này có chi phí lớn nên phù hợp với các công ty doanh nghiệp lớn.

2.2.5 Amazon Lex Framework

Chatbot mã nguồn mở này được cung cấp bởi Amazon Web Services (AWS) và sử dụng bộ Amazon AI [4]. Framework này hỗ trợ các nền tảng nhắn tin và truyền thông xã hội khác nhau, bao gồm Facebook Messenger, Kik và Twilio SMS. Amazon Lex Framework cung cấp các khả năng thay đổi quy mô tự động giúp các nhà phát triển không cần phải tăng khả năng của bot bằng cách quản lý cơ sở hạ tầng và phần cứng. Các bot mã nguồn mở này được tích hợp sẵn học máy, NLP và cho phép nhận dạng giọng nói tự động.

Nhược điểm lớn nhất của framework này là chỉ hỗ trợ cho tiếng Anh. Tích hợp vào Web khó khăn và ít kênh triển khai Chatbot hơn.

2.2.6 Rasa Framework

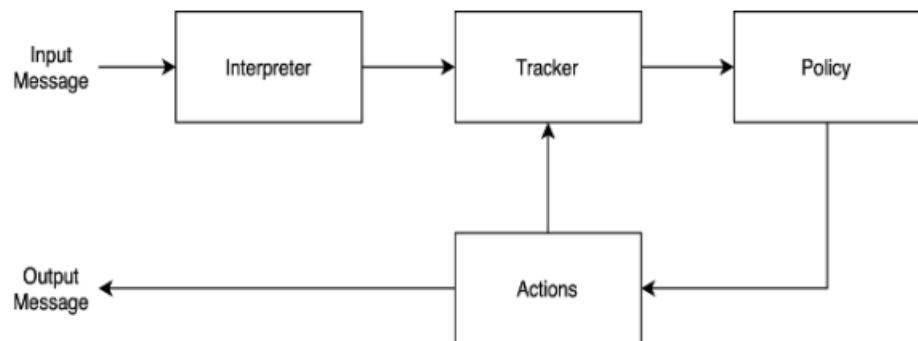
Framework nguồn mở này hoạt động tốt nhất để xây dựng các chatbot theo ngữ cảnh có thể thêm cảm giác giống người hơn vào các tương tác. Có thể xây dựng các câu chuyện để huấn luyện bot theo một chủ đề đóng mà mà phát triển muôn triễn khai. Với cộng đồng phát triển nền nhà phát triển mới có thể kế thừa và học tập các tính năng có sẵn tạo Chatbot theo nhu cầu [4]. Framework chatbot này bao gồm hai thành phần chính là Rasa NLU (hiểu ngôn ngữ tự nhiên) và Rasa Core để tạo Chatbot. Kết hợp lại, các thành phần này giúp người dùng xây dựng các bot có khả năng xử lý các yêu cầu phức tạp của người dùng.

Các tình năng trên cho thấy, framework Rasa phù hợp để sử dụng giải quyết bài toán về vấn đề đặt tiệc trong nhà hàng. Nên trong luận văn này, Chatbot dùng để giao tiếp với khách hàng được phát triển bằng framework Rasa. Framework này sẽ được phân tích kỹ ở phần tiếp theo.

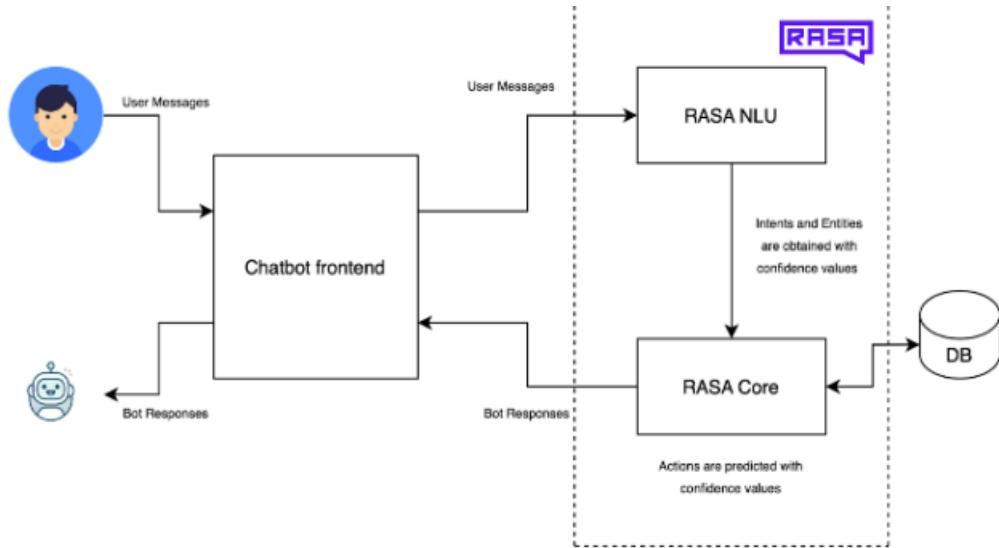
2.3 Tổng quan về Rasa Framework

Rasa là một nền tảng AI đàm thoại mã nguồn mở cho phép người phát triển có thể tổ chức các cuộc hội thoại, đồng thời kết nối với các kênh nhắn tin và hệ thống của bên thứ ba thông qua một bộ API. Nó cung cấp các khôi xây dựng để tạo trợ lý ảo (kỹ thuật số) hoặc chatbot [9]. Rasa là một thư viện NLP mã nguồn mở để phân loại mục đích trích xuất thực thể trong chatbots. Rasa gồm 3 phần:

- Rasa NLU: Thư viện để hiểu ngôn ngữ tự nhiên thực hiện phân loại ý định và trích xuất thực thể từ đầu vào của người dùng. Rasa NLU sẽ được chuẩn bị dữ liệu đào tạo cho chatbot, viết tệp cấu hình, chọn đường dẫn và đào tạo mô hình. Cuối cùng là sẽ dự đoán mục đích của một văn bản bằng cách sử dụng mô hình đào tạo. Rasa NLU sẽ phân tích cú pháp các thực thể để phân loại ý định và trích xuất thực thể từ đầu vào giúp bot hiểu những gì người dùng đang nói.
- Rasa Core: Một khung chatbot quản lý hội thoại đầu vào từ NLU để phản hồi lại cho người dùng. Phần này rất quan trọng khi người dùng có nhiều loại ý định trong chatbot của bạn và các câu hỏi hoặc câu trả lời tiếp theo. Thay vì sử dụng nhiều lệnh điều kiện trong if-else thì Rasa Core dạy mô hình tạo phản hồi lại người dùng.
- Rasa X: Là công cụ phát triển chatbot theo hướng hội thoại (Conversation-Driven Development - CDD), quá trình lắng nghe người dùng và sử dụng những thông tin chi tiết đó để cải thiện bot AI. Rasa X cho phép huấn luyện mô hình học máy bằng cách sử dụng chế độ tương tác.



Hình 2.2: Kiến trúc cơ bản của rasa [10]

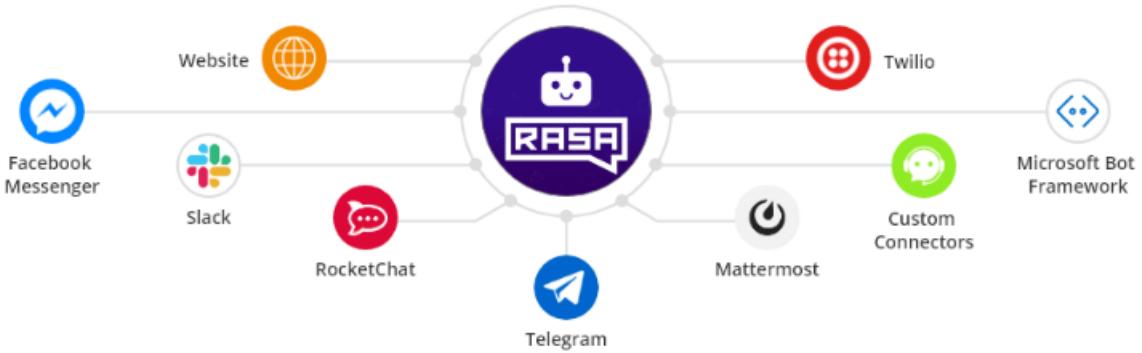


Hình 2.3: Kiến trúc Rasa [10]

Kiến trúc cơ bản của Rasa được thể hiện như hình 2.2 [10], trong kiến trúc này Rasa nhận thông điệp, tin nhắn từ bên ngoài và được đưa vào interpreter để được xử lý, phân loại ý định. Tracker giúp ghi lại tất cả dữ liệu và lưu trữ dữ liệu đó với mục đích sử dụng dữ liệu đó trong các giai đoạn tiếp theo của quy trình. Các policy có nhiệm vụ sẽ quyết định hành động nào sẽ được đưa ra đầu ra của rasa. Action có nhiệm vụ lấy các thực thể từ tracker để kết hợp với các phản hồi để gửi lại tin nhắn đầu ra cho phù hợp.

Dựa vào kiến trúc hình 2.2, rasa được phát triển và được sử dụng dưới giao diện chatbot frontend, giúp cho người dùng dễ dàng tiếp cận hơn như hình 2.3[10]. Đầu vào vẫn là thông điệp tin nhắn của người dùng được đưa vào thông qua giao diện của một khung chat bên trong một ứng dụng hoặc 1 trang web. Bên trong rasa được tách thành rasa NLU có chức năng phân tích tin nhắn người dùng để tách các intent và entity. Rasa Core có chức năng quản lý hội thoại đầu vào từ NLU để phản hồi lại cho người dùng.

Rasa là một mà nguồn mở cho phép người phát triển có thể truy cập vào toàn bộ quá trình xử lý bằng python và có thể mở rộng được khả năng của bot [11]. Rasa được phát



Hình 2.4: Các ứng dụng của Rasa [12]

triển kết hợp với nhiều API có thể được sử dụng để kết nối với các kênh nhắn tin như facebook, zalo, telegram,... Giúp doanh nghiệp, công ty có thể sử dụng để trao đổi với khách hàng một cách thuận tiện nhất như hình 2.1 [12]. Rasa giúp công ty, doanh nghiệp có thể sử dụng một cách thoải mái và giúp tiết kiệm được chi phí của công ty và doanh nghiệp.

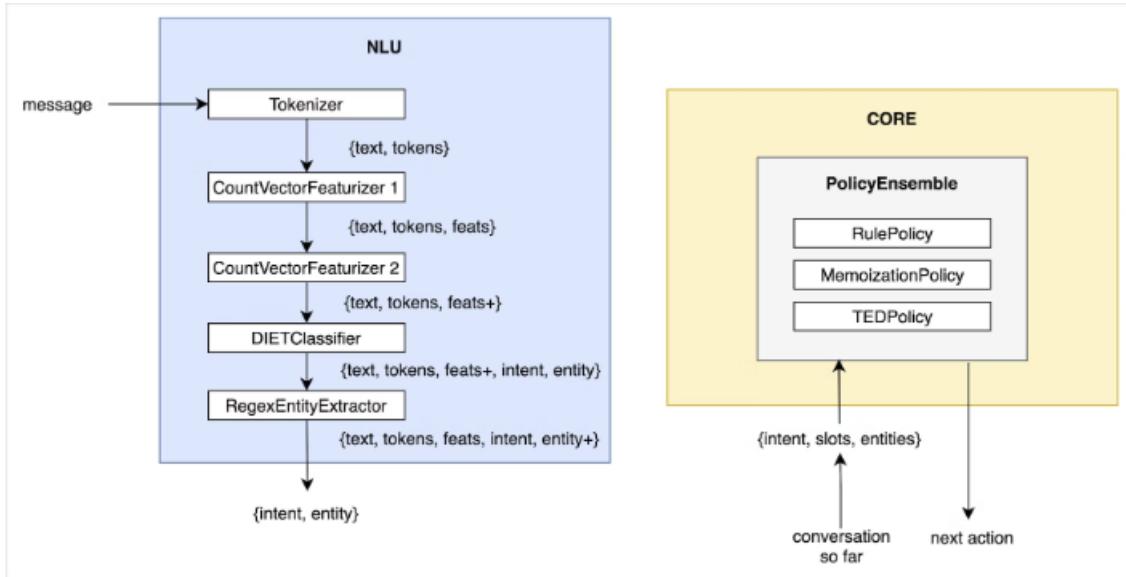
Rasa NLU có khả năng truy xuất và lưu thông tin trên chính máy chủ đang sử dụng Rasa nên giúp người quản lý có thể quản lý được thông tin thu được từ khách hàng. Rasa NLU được tích cực cập nhập và có cộng đồng lớn hỗ trợ. Rasa NLU là một nguồn mở mang tính độc lập, có thể xây dựng một chatbot từ đầu và sử dụng trên máy chủ riêng của người phát triển. Rasa NLU cung cấp cho người phát triển toàn bộ quyền phát triển chatbot theo ý định của mình.

2.3.1 Hiểu ngôn ngữ tự nhiên

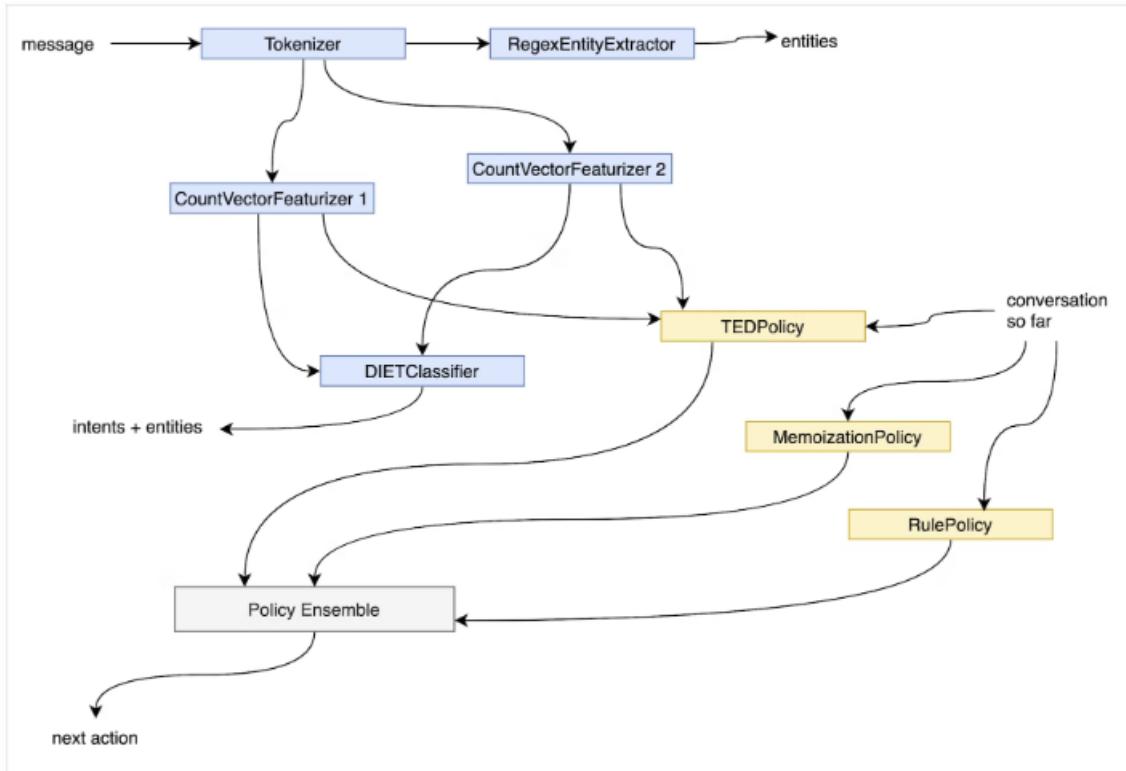
Các thành phần tạo NLU pipeline và chúng hoạt động tuần tự để xử lý đầu vào của người dùng thành đầu ra có cấu trúc. Có các thành phần để trích xuất thực thể, phân loại ý định, lựa chọn phản hồi, tiền xử lý, .v.v.

Kỹ thuật uốn đường ống: Hệ thống NLU được đào tạo độc lập với các policies trong NLU Core giống như linear pipeline (như hình 2.5) của thành phần được nối tiếp. Hình 2.5 cho thấy 2 phần riêng biệt là Rasa NLU tiếp nhận và xử lý tin nhắn của người dùng, Rasa Core có nhiệm vụ quyết định hành động để thực hiện trả về cho người dùng. Khi bot dự đoán từ hệ thống NLU sẽ đưa vào các policies, nhưng trong quá trình đào tạo model thì hai hệ thống này sẽ độc lập với nhau. Trong kịch bản end-to-end, TED cũng có thể cắt bỏ trong pipeline để đưa ra dự đoán cho các văn bản không hoàn toàn phù hợp với một ý định. Vì vậy 2 thành phần NLU/CORE được vẽ lại như một cách riêng biệt như hình 2.6. Hình 2.6 biểu diễn NLU pipeline dưới dạng đồ thị tính toán dạng cây DAG

(Directed Acyclic Graph). Mỗi nút trên đồ thị cây đại diện cho một thành phần và các cạnh đại diện cho một sự phụ thuộc.



Hình 2.5: Hệ thống độc lập linear pipeline giữa Rasa NLU và Rasa Core [13]



Hình 2.6: Đồ thị DAG biểu diễn pipeline trong Rasa [13]

Đồ thị DAG có bộ nhớ đệm khi nếu có một thành phần trong đồ thị thay đổi thì thành phần đó và các thành phần có sự phụ thuộc vào thành phần đó sẽ được đào tạo lại, còn

các thành phần khác trong pipeline không cần đào tạo lại. Quá trình hiểu ngôn ngữ tự nhiên, xác định được ý định của người dùng thì bot sẽ trải qua 4 bước là tiền xử lý dữ liệu, trích xuất đặc trưng, huấn luyện mô hình và phân lớp dữ liệu.

Tiền xử lý dữ liệu bao gồm làm sạch và phân tách các từ (Tokenizers) có chức năng chia văn bản thành các mã thông báo. 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. 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 hóa tốt thì sẽ làm tăng khả năng độ chính xác cũng như sự thông minh cho Chatbot.

Trích xuất đặc trưng (Featurizers) có chức năng xử lý dữ liệu từ những dữ liệu đã được chuẩn hóa và làm sạch để sử dụng làm đầu vào cho thuật toán. Đầu ra của quá trình này là lấy được đặc trưng từ dữ liệu đầu vào. 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 hóa tốt thì sẽ làm tăng khả năng độ chính xác cũng như sự thông minh cho Chatbot. Trong Rasa NLU có hai featurizers có thể sử dụng là:

- Sparse featurizers thường được tạo bởi CountVectorizer và trả về các vector đặc trưng với nhiều giá trị bị thiếu. Vì các đặc trưng đó thường chiếm nhiều bộ nhớ nên sparse featurizers thường lưu tập giá trị dưới dạng các đối tượng thưa thớt. Các giá trị được lưu trữ khác không và vị trí của chúng trong vector.
- Dense featurizers: Những tính năng này bao gồm nhiều nhúng được đào tạo trước. Thông thường được lấy từ SpaCyFeaturizers hoặc LanguageModelFeaturizers từ huggingface.

Huấn luyện mô hình có đầu vào là các đặc trưng quan trọng đã được trích xuất ở bước 2 và áp dụng các thuật toán học máy để tạo ra một mô hình phân lớp. Các mô hình phân lớp có thể là các quy tắc phân lớp hoặc là các vector trọng số tương ứng với các đặc trưng được trích xuất. Các mô hình huấn luyện được phát triển trong Rasa NLU bao gồm mô hình nhúng được đào tạo trước (Pre-trained Embeddings) và Nhúng được giám sát (Supervised Embeddings). Mô hình Nhúng được đào tạo trước (Pre-trained Embeddings) bao gồm:

- MitieFeaturizer: Tạo vector của tin nhắn và phản hồi của người dùng sử dụng trình

tạo dữ liệu MITIE thuộc dense featurizers, nhằm mục đích khai thác thực thể, phân loại mục đích và phân loại phản hồi. Vector câu có thể được tính theo hai cách khác nhau thông qua gán pooling là mean hoặc max.

- SpacyFeaturizer: Tạo vector cho tin nhắn và phản hồi người dùng bằng cách sử dụng trình tạo dữ liệu Spacy thuộc dense featurizers, nhằm khai thác thực thể, phân loại mục đích và phân loại phản hồi. Vector câu có thể được tính theo hai cách khác nhau thông qua gán pooling là mean hoặc max.
- ConveRTFeaturizer: Tạo vector cho tin nhắn và phản hồi người dùng bằng cách sử dụng mô hình ConveRT thuộc dense featurizers, nhằm trích xuất thực thể, phân loại mục đích và lựa chọn phản hồi.
- LanguageModelFeaturizer: Tạo vector tin nhắn và phản hồi người dùng bằng cách sử dụng mô hình ngôn ngữ được đào tạo trước thuộc dense featurizers nhằm tạo các tính năng để trích xuất thực thể, phân loại mục đích và lựa chọn phản hồi.

Mô hình Nhúng được giám sát (Supervised Embeddings):

- RegexFeaturizer: Tạo và biểu diễn vector thông điệp người dùng bằng cách sử dụng biểu thức chính quy nên đòi hỏi sử dụng các token thuộc sparse featurizers. Trong quá trình đào tạo, RegexFeaturizer tạo một danh sách các biểu thức chính quy được xác định ở dữ liệu đào tạo. Một tính năng sẽ được thiết lập để đánh dấu xem biểu thức này có được tìm thấy trong tin nhắn của người dùng hay không. Tất cả tính năng sau đó sẽ được đưa vào một trình phân loại ý định hoặc trích xuất thực thể để đơn giản hóa việc phân loại. Cấu hình bằng cách `case_sensitive` để phân biệt chữ hoa và chữ thường hay không, `use_word_boundaries` để có tách từ bởi khoảng trắng hay không.
- CountVectorizerFeaturizer: Tạo bản trình bày từng từ về thông điệp, ý định và phản hồi từ người dùng thuộc sparse featurizers. Cấu hình analyzer được đặt thành word để sử dụng số lượng token làm tính năng. Khi thiết lập analyzer thành char hoặc `char_wb` để sử dụng ký tự n-gram. Các tham số số khi thiết lập n-gram có gồm `ngram_range` và `max_ngram`. Trong quá trình đào tạo không thể đảm bảo rằng trong quá trình dự đoán, một thuật toán sẽ không gặp một từ không xác định. Để một thuật toán có thể xử lý các từ chưa biết, một số từ vựng trong dữ liệu huấn luyện có thể được thay thế bằng từ chung `OOV_token`. `OOV_token` đặt một từ khóa cho các từ không nhìn thấy nếu dữ liệu đào tạo chứa `OOV_token` dưới dạng các từ trong tin nhắn, trong quá trình đào tạo sẽ được thay thế bằng các từ đã cung cấp `OOV_token`, nếu như không có từ nào trong khai báo tại `OOV_token` thì các từ đó sẽ được bỏ qua trong quá trình đào tạo. Nếu danh sách các từ trong thiết lập `OOV_token` cần được xem như là từ vựng

thì cần thiết lập OOV_words tương tự như OOV_token.

- LexicalSyntacticFeaturizer: Tạo các tính năng từ vựng và cú pháp cho tin nhắn người dùng thuộc sparse featurizer.

Bước thứ tư là xây dựng mô hình phân lớp để có thể sử dụng để phân loại ý định và câu hỏi từ người dùng. 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 "confidential" cho từng ý định và trả về ý định có điểm số 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. Vì vậy việc xác định chính xác ý định người dùng đóng một vai trò rất quan trọng trong hệ thống Chatbot. Một số mô hình phân lớp ý định được phát triển trong Rasa NLU:

- MitieIntentClassifier: Thành phần này yêu cầu sử dụng token tin nhắn và mô hình MitieNLP nhằm phân loại mục đích.
- LogisticRegressionClassifier: Thành phần này yêu cầu sử dụng sparse_features hoặc dense_features nhằm phân loại ý định sử dụng hồi quy logistic của scikit-learn. Bộ này sẽ đào tạo nhanh hơn nhưng độ chính xác thấp.
- SklearnIntentClassifier: Thành phần này yêu cầu dense_features bộ phân loại ý định của sklearn đào tạo SVM tuyến tính được tối ưu hóa bằng cách sử dụng tìm kiếm lưỡng.
- KeywordIntentClassifier: Đây là trình phân loại mục đích đối sánh với các từ khóa đơn giản dành cho các project nhỏ với thời gian sử dụng ngắn. Đối sánh này có phân biệt chữ hoa chữ thường theo mặc định và chỉ tìm kiếm đối sánh chính xác của chuỗi từ khóa trong tin nhắn người dùng. sparse_features cho tin nhắn người dùng.
- DIETClassifier: Bộ biến đổi thực thể có ý định kép được sử dụng để phân loại ý định và trích xuất thực thể yêu cầu dense_features hoặc sparse_features cho tin nhắn từ người dùng. DIET (Dual Intent và Entity Transformer) là một kiến trúc đa tác vụ để phân loại ý định và nhận dạng thực thể. Một chuỗi các nhãn thực thể được dự đoán thông qua lớp gắn thẻ Conditional Random Field (CRF) trên đầu chuỗi đầu ra của quá trình biến đổi từ đầu vào mã thông báo. Đối với mỗi nhãn ý định, đầu tra của quá trình biến đổi cho các nhãn hoàn chỉnh và ý định được nhúng vào một không gian vector ngữ nghĩa duy nhất.

-
- FallbackClassifier: Phân loại tin nhắn từ người dùng với ý định nlu_fallback nếu điểm số phân loại mục đích của NLU không rõ ràng. Yêu cầu intent và intent_ranking từ bộ phân loại ý định trước đó sẽ cho kết quả đầu ra bao gồm entities, intent, intent_ranking. FallbackClassifier phân loại ý định tin nhắn người dùng trong trường hợp trình phân loại ý định phân loại trước đó không thể phân loại một ý định với độ tin cậy lớn hơn hoặc bằng threshold. Thành phần này cũng có thể sử dụng để dự đoán ý định dự phòng khi độ tin cậy của hai ý định được xếp hạng cao nhất gần bằng nhau. Tham số threshold đặt ngưỡng để dự đoán nlu_fallback ý định. Nếu không có ý định nào được dự đoán bởi bộ phân loại ý định trước đó có mức độ tin cậy lớn hơn hoặc bằng mức độ tin cậy threshold sẽ FallbackClassifier hêm một dự đoán về nlu_fallback ý định với độ tin cậy 1.0. Tham số ambiguity_threshold nhằm định cấu hình một ambiguity_threshold, hàm FallbackClassifier cũng sẽ dự đoán nlu_fallback ý định trong trường hợp sự khác biệt của điểm tin cậy cho hai ý định được xếp hạng cao nhất nhỏ hơn ambiguity_threshold.

Ngoài 4 bước để phân tích ý định từ khách hàng, thì bước trích xuất thực thể có chức năng truy xuất thông tin từ tin nhắn khách hàng. Đây là một kỹ thuật phân tích văn bản sử dụng NLP để tự động lấy dữ liệu cụ thể từ văn bản phi cấu trúc và phân loại dữ liệu đó theo các danh mục xác định trước [14]. Các danh mục này được đặt tên là các thực thể, các từ hoặc cụm từ đại diện cho một danh từ. Điều này bao gồm tên riêng cũng như các biểu thức số về thời gian hoặc số lượng, chẳng hạn như số điện thoại, giá trị tiền tệ hoặc ngày tháng. Việc phân tích và trích xuất thực thể trong Rasa NLU dựa vào mô hình đào tạo trước tại quá trình phân lớp ý định. Các module về trích xuất thực thể trong Rasa NLU:

- MitieEntityExtractor: Trình xuất thực thể MITIE để tìm và đưa ra kết quả thực thể (entities), yêu cầu sử dụng mô hình MitieNLP và mã thông báo (token).
- SpacyEntityExtractor: Yêu cầu sử dụng mô hình SpacyNLP để xác định và đưa ra các thực thể từ tin nhắn của người dùng. Spacy sử dụng mô hình chuyển đổi BILOU thống kê.
- CRFEntityExtractor: Trình xuất thực thể ngẫu nhiên có điều kiện (Conditional random field - CRF) để xác định và đưa ra thực thể đòi hỏi sử dụng mã thông báo và dense_features. CRF có thể được coi như một chuỗi Markov vô hướng trong đó các bước là các từ và các trạng thái là các lớp thực thể. Các đặc điểm của các từ như viết hoa, gắn thẻ POS cung cấp xác suất cho các lớp thực thể nhất định, cũng như chuyển các thẻ thực thể lân cận, tập hợp các thực thể có khả năng xảy ra cao

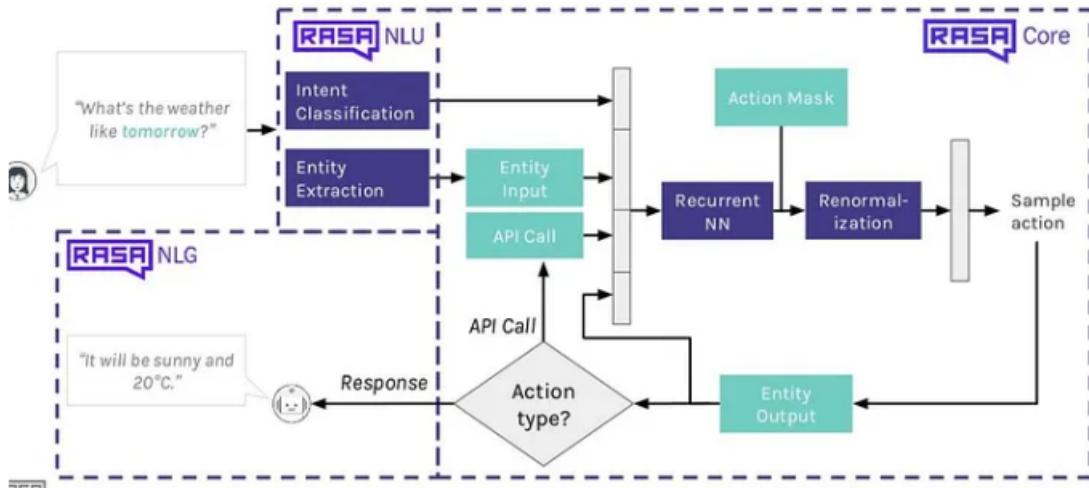
nhất sau đó được tính toán và trả về.

- DucklingEntityExtractor: Cho phép trích xuất các thực thể phổ biến như ngày tháng, số tiền, khoảng cách và các đối tượng khác bằng ngôn ngữ. Muốn sử dụng thành phần này thì người phát triển cần chạy một server duckling để cho phép nhận ra các thực thể thời gian, số, khoảng cách, các thực thể có cấu trúc khác và chuẩn hóa chúng.
- RegexEntityExtractor: Thành phần này trích xuất thực thể bằng cách sử dụng bảng tra cứu và các regex được xác định trong lúc đào tạo huấn luyện dữ liệu. Thành phần kiểm tra tin nhắn người dùng có chứa mục nhập của một trong các bảng tra cứu hoặc khớp với các regex hay không. Nếu có thì giá trị sẽ được trích xuất dưới dạng thực thể.
- EntitySynonymMapper: Thành phần này ánh xạ các giá trị thực thể đồng nghĩa với cùng một giá trị để sửa đổi các thực thể hiện có mà các thành phần trích xuất thực thể trước đưa ra. Thành phần này cho phép ánh xạ thực thể New York City và NYC tới nyc. Việc trích xuất sẽ trả về nyc mặc dù thông báo có chứa NYC. Khi thành phần này thay đổi một thực thể hiện có, nó sẽ tự gắn vào danh sách bộ xử lý của thực thể này.

2.3.2 Xử lý đối thoại

Xử lý đối thoại (Rasa Core) có chức năng dự đoán hành động nào sẽ được thực hiện từ các action được định nghĩa từ trước. Một hành động có thể là xử lý thông tin, trả về một thông điệp đến khách hàng hoặc một chức năng tùy ý thực hiện. Hình 2.7 thể hiện hoạt động của Rasa NLU và Rasa Core. Khi một hành động được thực thi, thì bot sẽ tiếp tục thực hiện hành động tiếp theo hoặc chờ đợi ý định tiếp theo của người dùng tùy theo cách luồng câu chuyện mà người phát triển thiết kế cho bot. Các hành động không thể trực tiếp thay đổi trình theo dõi, nhưng khi được thực hiện có thể trả về một danh sách các sự kiện.

Việc quyết định trả hành động cần thực hiện ở mỗi bước trong luồng câu chuyện trong Rasa sẽ dựa vào các chính sách (policies). Sau khi nhận được ý định định của người dùng, mỗi chính sách được xác định trong cấu hình sẽ dự đoán hành động tiếp theo với mức độ tin cậy nhất định. Các chính sách sẽ có một độ ưu tiên nhất định ví dụ như chính sách RulePolicy sẽ có độ ưu tiên là 6, MemoizationPolicy hoặc Augmented-MemoizationPolicy sẽ có mức độ ưu tiên là 3, UnexpectTEDIntentPolicy có độ ưu tiên là 2 và TEDPolicy có độ ưu tiên thấp nhất là 1. Độ ưu tiên của chính sách sẽ được sử dụng nếu như có nhiều hơn một chính sách có độ tin cậy dự đoán như nhau và cao nhất. Nếu trường hợp đó có hai mức độ ưu tiên đều giống nhau thì bot sẽ chọn ngẫu nhiên



Hình 2.7: Rasa NLU và Rasa Core hoạt động [15]

một trong hai chính sách đó để trả về. Điều này sẽ gây ra sự mất độ tin cậy cho người dùng nếu như hành động trả về không đúng. Vì vậy, không nên sử dụng hai chính sách có cùng mức độ ưu tiên để cấu hình. Một số chính sách được tích hợp sẵn trong Rasa:

- TED Policies: Là một kiến trúc đa tác vụ để dự đoán hành động tiếp theo và nhận dạng thực thể. Kiến trúc bao gồm một số bộ mã hóa biến đổi được chia sẻ cho cả hai nhiệm vụ. Một chuỗi các nhãn thực thể được dự đoán thông qua lớp gắn thẻ Trường ngẫu nhiên có điều kiện (Conditional random field - CRF) trên đầu đầu ra của bộ mã hóa biến áp trình tự của người dùng tương ứng với chuỗi đầu vào của mã thông báo. Đối với dự đoán hành động tiếp theo, đầu ra của bộ mã hóa biến đổi đối thoại và các nhãn hành động của hệ thống được nhúng vào một không gian vectơ ngữ nghĩa duy nhất. Đầu chấm được loại bỏ để tối đa hóa sự tương đồng với nhãn mục tiêu và giảm thiểu sự tương đồng với các mẫu tiêu cực.
- UnexpectIntentPolicy: Giúp bot xem lại các cuộc trò chuyện và cho phép bot phản hồi lại những lượt không chắc chắn. Chính sách này chỉ nên sử dụng cùng với ít nhất một chính sách khác vì hành động duy nhất được policy này kích hoạt bởi một hành động đặc biệt `action_unlikely_intent`. UnexpectIntentPolicy có kiến trúc như TEDPolicy nhưng khác nhau ở cấp độ ưu tiên. UnexpectIntentPolicy không quan tâm đến các hành động tốt nhất sẽ được kích hoạt tiếp theo mà tìm hiểu các ý định có nhiều khả năng được người dùng thể hiện trong bối cảnh hội thoại từ các câu chuyện đào tạo. Nếu mục đích mà NLU dự đoán thực sự có khả năng xảy ra trong bối cảnh hội thoại, UnexpectIntentPolicy thì không kích hoạt bất kỳ hành động nào. Nếu không, policy này sẽ kích hoạt một `action_unlikely_intent` với

confidence là 1.00. UnexpectTEDIntentPolicy nên hỗ trợ cho TEDPolicy để TED-Policy sẽ cải thiện với phạm vi bao phủ tốt hơn của các đường dẫn hội thoại duy nhất mà bot dự kiến sẽ xử lý trong dữ liệu đào tạo. UnexpectTEDIntentPolicy được gọi ngay sau câu nói người dùng và có thể kích hoạt action_unlike_intent hay không. Để xem action_unlike_intent UnexpectTEDIntentPolicy tính điểm cho ý định của người dùng trong bối cảnh đối thoại hiện tại và kiểm tra xem điểm này có dưới một điểm ngưỡng nào đó hay không. Điểm ngưỡng này được tính bằng cách thu thập đầu ra của mô hình ML trên nhiều "ví dụ tiêu cực". Những ví dụ tiêu cực này là sự kết hợp giữa ngữ cảnh đối thoại và ý định của người dùng không chính xác. UnexpectTEDIntentPolicy tạo ra những ví dụ tiêu cực này từ dữ liệu đào tạo bằng cách chọn một phần câu chuyện ngẫu nhiên và ghép nối nó với một mục đích ngẫu nhiên không xảy ra vào thời điểm này.

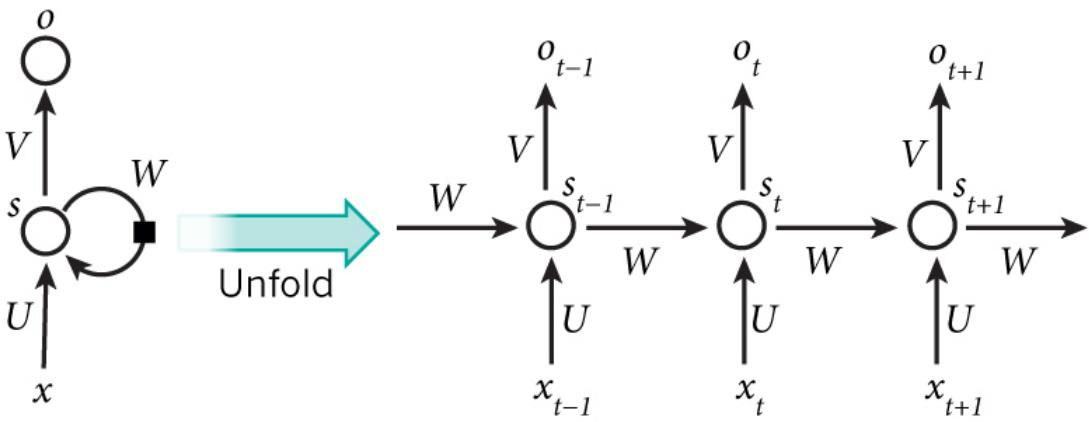
- Memoization Policy: Thành phần này kiểm tra xem cuộc trò chuyện hiện tại có khớp với những câu chuyện trong tệp stories.yml hay không. Khi có một kết quả phù hợp trong dữ liệu đào tạo thì policy sẽ tính đến giá trị max_history của cuộc trò chuyện. Một lượt bao gồm tin nhắn do người dùng gửi và bất kỳ hành động nào mà bot thực hiện trước khi chờ tin nhắn tiếp theo.
- Augmented Memoization Policy: Policy này ghi nhớ từ các câu chuyện đã tạo giống với MemoizationPolicy. Ngoài ra Policy này có cơ chế quên một số bước nhất định trong lịch sử của cuộc trò chuyện và tìm một điểm trùng khớp trong câu chuyện với lịch sử đã giảm.
- RulePolicy: Policy này xử lý các phần hội thoại tuân theo một hành vi cố định. Policy này đưa ra dự đoán dựa trên bất kỳ quy tắc có trong dữ liệu đào tạo. Các quy tắc có thể được thêm vào phần rules trong dữ liệu phần đào tạo, một quy tắc có thể được áp dụng tại bất kỳ thời điểm nào trong cuộc trò chuyện.

2.4 Thuật toán sử dụng trong NLU

2.4.1 Mạng hồi quy RNN

Mạng nơron hồi quy RNN là một loại mạng nơron nhân tạo sử dụng dữ liệu theo dạng chuỗi. RNN thực hiện cùng một nhiệm vụ cho tất cả các phần tử trong một chuỗi với đầu ra phụ thuộc vào các tính toán trước đó. Ý tưởng chính của mạng RNN là sử dụng một bộ nhớ để lưu lại thông tin từ những tính toán trước để dựa vào nó để đưa ra các dự đoán chính xác ở bước hiện tại.

Như trên sơ đồ hình 2.8 thể hiện mạng RNN đơn giản nhất. Điểm mấu chốt là sự hiện



Hình 2.8: Recurrent neural network và tính toán chuyển tiếp của mạng RNN [16]

diện vòng lặp sẽ gây ra trạng thái ẩn của mạng nơron sau khi truyền thông số đầu vào là mỗi từ trong chuỗi. Trong thực tế, RNN được sử dụng với chuỗi có độ dài hữu hạn và nên được mở rộng vòng lặp thành một mạng time-layered giống như mạng chuyển tiếp như trong hình 2.8. Dựa vào hình trên ta thấy, trong mỗi nhãn thời gian (time-stamp) có một đầu vào, đầu ra và lớp ẩn [16]. Trong thực tế có thể thiếu đơn vị đầu vào hoặc đầu ra tại bất kỳ nhãn thời gian nào. Việc thiếu đầu vào hoặc đầu ra sẽ phụ thuộc vào ứng dụng cụ thể cho việc sử dụng mạng này. Kiến trúc cụ thể trong hình 2.8 phù hợp với mô hình ngôn ngữ đặc biệt trong xử lý ngôn ngữ tự nhiên để dự đoán từ tiếp theo dựa trên lịch sử của các từ trước đó. Đưa ra một chuỗi các từ được cung cấp lần lượt từng từ cho mạng nơron. Quá trình tạm thời này tương đương với việc cung cấp các từ riêng lẻ cho đầu vào tại các dấu thời gian có liên quan như trong hình 2.8. Ví dụ như dữ liệu là một câu gồm 4 từ thì mạng sẽ được mở thành mạng nơron có 4 lớp. Các công thức để tính toán thực hiện trong mạng RNN:

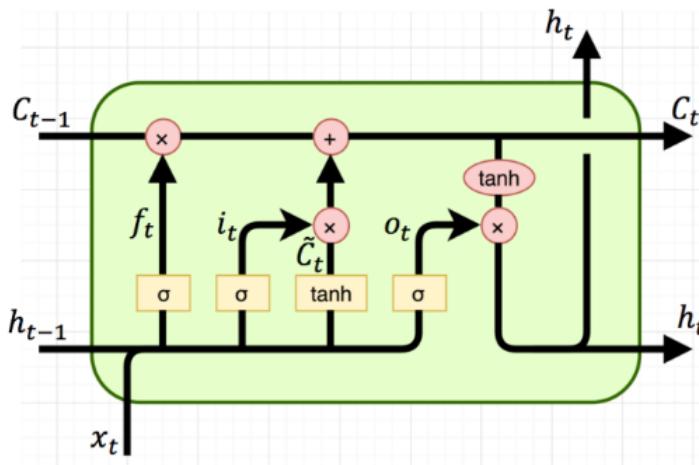
- x_t là đầu vào tại bước thời gian t . Ở mỗi đầu vào tương ứng với một từ trong câu.
- h_t là trạng thái ẩn tại bước thời gian t . Đây là bộ nhớ của mạng. h_t được tính toán dựa trên trạng thái ẩn trước đó và đầu vào ở bước hiện tại như công thức:

$$s_t = f(Ux_t + Wh_{t-1})$$
. Hàm f thường là hàm phi tuyến như tanh hoặc ReLU.
- o_t là đầu ra ở bước t .

2.4.2 Long Short-Term Memory - LSTM

Mạng RNN có các vấn đề liên quan đến độ dốc biến mất hoặc bùng nổ do mạng thực hiện liên tiếp các phép nhân ma trận liên tiếp [16]. Việc này dẫn đến việc gradient biến mất trong quá trình lan truyền ngược hoặc gradient tăng giá trị theo cách không ổn định. Dựa vào kiến trúc ở hình số 2.8, ta có thể thấy là các nhãn thời gian càng xa thì càng bị

mất thông tin. Theo lý thuyết thì mạng RNN có thể mang thông tin từ các lớp trước đến các lớp sau nhưng thực tế là thông tin chỉ mang được qua một số lượng trạng thái nhất định và sau đó sẽ dần bị mất độ dốc. Vì vậy mô hình RNN chỉ học được từ các trạng thái gần với trạng thái đó. Để giải quyết vấn đề này, dựa trên sự thay đổi phương trình truy hồi cho vector ẩn bằng cách sử dụng LSTM với việc sử dụng bộ nhớ dài hạn. Các hoạt động của LSTM được thiết kế để có quyền kiểm soát chi tiết đối với dữ liệu được ghi vào bộ nhớ dài hạn.



Hình 2.9: Kiến trúc mạng LSTM [17]

Các mạng LSTM có một số ô trạng thái theo ngữ cảnh bên trong hoạt động như các ô nhớ dài hạn hoặc ngắn hạn. Đầu ra của mạng LSTM được điều diễn bởi trạng thái của ô này. Đây là một thuộc tính rất quan trọng khi chúng ta cần dự đoán của mạng neuron phụ thuộc vào bối cảnh lịch sử của đầu vào, thay vì chỉ dựa trên đầu vào cuối cùng. Các LSTM giải quyết vấn đề độ dốc bằng cách đưa thêm một số cổng kiểm soát quyền truy cập vào trạng thái ô. Tại trạng thái thứ t của mô hình LSTM:

- Output: c_t đầu ra của ô trạng thái, h_t đầu ra của trạng thái ẩn
- Input: c_{t-1} , h_{t-1} , x_t . Trong đó x_t là input ở trạng thái thứ t của mô hình. c_{t-1} , h_{t-1} là đầu tra của lớp trước, đóng vai trò khá giống như s ở trong RNN.
- Forget gate: $f_t = \sigma(U_f * x_t + W_f * h_t + b_f)$. Thông số này quyết định xem cần lấy bao nhiêu từ cell state .
- Input gate: $i_t = \sigma(U_i * x_t + W_i * h_{t-1} + b_i)$. Thông số này sẽ quyết định lấy bao nhiêu từ input của state và hidden layer của layer trước.
- Output gate: $o_t = \sigma(U_o * x_t + W_o * h_{t-1} + b_o)$. quyết định xem cần lấy bao nhiêu từ cell state để trở thành output của trạng thái ẩn. Ngoài ra h_t cũng được dùng để tính ra output y_t cho trạng thái t.

Chương 3

Xây dựng luồng hội thoại đặt tiệc tại nhà hàng

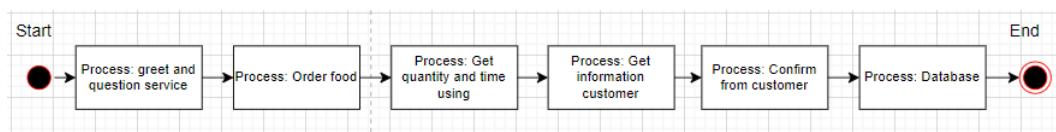
3.1 Tổng quát câu chuyện tại tương tác khách hàng

Kịch bản cơ bản của bot tại nhà hàng bao gồm các module cơ bản như đặt tiệc, gọi món, thêm món và lấy thông tin khách hàng. Các luồng câu chuyện cơ bản được thiết kế theo luồng câu chuyện như hình 3.1. Trong mỗi module là một chuỗi các hành động của bot được xây dựng để thực hiện các chức năng khi được yêu cầu. Các hành động của bot được thực hiện nhờ vào các intent từ trước đó để thực hiện hành động chính xác. Các intent được xác định từ tin nhắn của khách hàng, bot sẽ dựa vào intent đó để thực hiện tác vụ xử lý và gửi lại phản hồi cho khách hàng.

Để dễ dàng hiện thực và quản lý thì kịch bản tại nhà hàng bao gồm các mẫu chuyện nhỏ như chào hỏi, đặt đồ ăn, lấy thông tin khách hàng và xác nhận thông tin như hình 3.1.

Chào hỏi: Đây là phần mở đầu câu chuyện giữa khách hàng và bot, bot có nhiệm vụ nhận tín hiệu của người dùng muốn trao đổi với bot để sử dụng dịch vụ và bot sẽ chào hỏi lại khách hàng. Trong phần câu chuyện chào hỏi này, sau khi bot chào khách hàng thì bot sẽ hỏi về dịch vụ mà khách hàng có nhu cầu sử dụng.

Đặt đồ ăn: Tiếp theo câu chuyện trên là câu trả lời về dịch vụ mà khách hàng có nhu cầu sử dụng. Câu chuyện đặt đồ ăn được sử dụng khi khách hàng có nhu cầu đặt tiệc. Trong câu chuyện này bot sẽ hỏi về món mà khách hàng muốn sử dụng tại nhà hàng và



Hình 3.1: Luồng câu chuyện cơ bản đặt tiệc tại nhà hàng

số lượng món nhằm trích xuất thông tin từ câu trả lời để lấy tên món ăn và số lượng món ăn tương ứng. Chức năng gọi thêm món được phát triển trong mẫu chuyện này. Nếu khách hàng có nhu cầu đặt thêm món thì luồng câu chuyện sẽ thực hiện lại quá trình đặt món ăn. Nếu khách hàng không có nhu cầu đặt thêm món ăn thì bot sẽ hiển thị danh sách các món ăn bao gồm số lượng món ăn tương ứng để khách hàng kiểm tra. Kèm theo đó là câu hỏi về số lượng người tham gia trước khi kết thúc mẫu chuyện nhỏ này để chuyển qua mẫu chuyện tiếp theo.

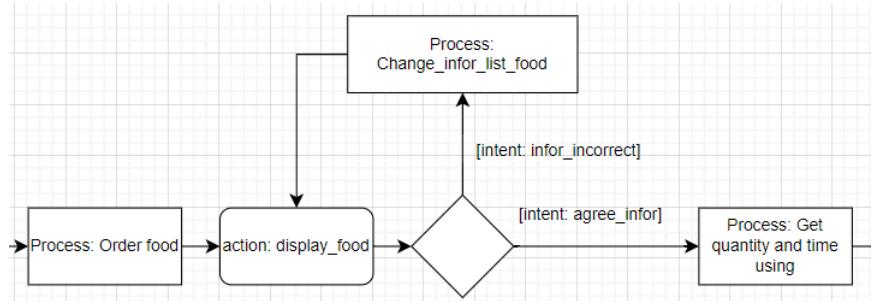
Lấy số lượng người tham gia và thời gian khách hàng sử dụng: Từ câu hỏi của mẫu **mẫu** truyện trước đó, bot cần lấy thông tin về số lượng người tham gia để người quản lý nhà hàng biết khách hàng có nhu cầu sử dụng bàn gồm bao nhiêu chỗ để chuẩn bị cho hợp lý. Số lượng người sẽ được bot trích xuất từ tin nhắn của người dùng khi trả lời bot. Và lấy thông tin về thời gian mà khách hàng có nhu cầu cần sử dụng nhà hàng. Khi nhận được tin nhắn của khách hàng, bot cần trích xuất và xử lý trước khi lưu dữ liệu. Bởi vì khách hàng có thể trả lời về thời gian bằng một câu nói thông thường cho nên bot phải có khả năng trích xuất thông tin từ câu nói đó để đưa ra các thông tin về thời gian như: giờ, ngày, tháng, năm mà khách hàng có nhu cầu sử dụng. Để đi qua một mẫu chuyện tiếp theo, bot cần đưa ra câu hỏi về để lấy tên khách hàng.

Lấy thông tin của khách hàng: Để tiếp tục mẫu chuyện này thì khách hàng cần trả lời câu hỏi trên để bot có thể trích xuất và lấy được tên khách hàng và sau đó là lấy số điện thoại của khách hàng. Thông tin này giúp người quản lý có thể nắm được chính xác thông tin của khách hàng tương ứng với các nhu cầu sử dụng tại nhà hàng để khi khách hàng đến, quản lý có thể đổi chiểu chính xác khách hàng đó đã đặt tiệc tại nhà hàng. Sau khi bot lấy được thông tin của người dùng thì bot sẽ hiển thị ra tất cả thông tin để khách hàng có thể kiểm tra lại và xác nhận. Bot sẽ đưa ra câu hỏi về xác nhận để kết thúc mẫu câu chuyện này và chuyển sang mẫu chuyện tiếp theo.

Xác nhận thông tin từ khách hàng: Mẫu câu chuyện này có chức năng lấy ý kiến xác nhận thông tin của khách hàng sau khi hiện thị thông tin mà bot đã trích xuất từ tin nhắn của khách hàng. Nếu như khách hàng xác nhận đúng các thông tin thì bot sẽ lưu lại tất cả thông tin của khách hàng vào database và cảm ơn khách hàng.

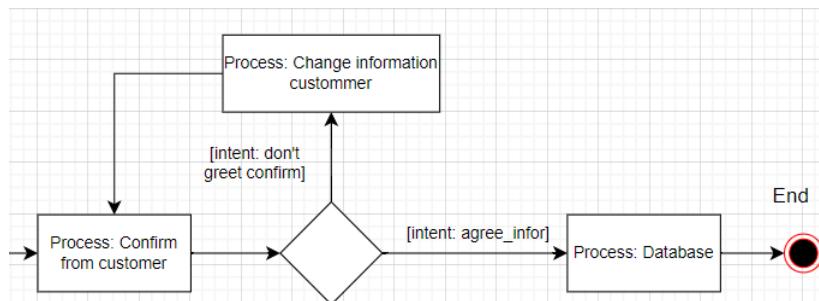
Chỉnh sửa thông tin món ăn: Luồng câu chuyện này được mở rộng sau câu chuyện đặt món ăn của khách hàng như hình 3.2. Khi hiện thị danh sách các món ăn nếu khách hàng có nhu cầu chỉnh sửa món ăn thì luồng câu chuyện sẽ chuyển qua mẫu câu chuyện chỉnh sửa món ăn. Mẫu câu chuyện này dành cho khách hàng khi có nhu cầu chỉnh sửa thông tin món ăn. Chỉnh sửa món ăn bao gồm các chức năng như thay đổi tên món ăn, thay đổi số lượng món ăn và xóa món ăn nào đó trong danh sách. Sau khi thực hiện thay

đổi thông tin trong danh sách món ăn thì bot sẽ tiến hành đưa ra danh sách món ăn đã được thay đổi để khách hàng xác nhận. Nếu như khách hàng muốn tiếp tục chỉnh sửa thông tin món ăn thì luồng câu chuyện sẽ tiếp tục thực hiện mẫu câu chuyện chỉnh sửa thông tin món ăn. Nếu khách hàng đồng ý về danh sách món ăn thì luồng câu chuyện sẽ chuyển qua mẫu câu chuyện lấy số lượng người sử dụng và thời gian sử dụng dịch vụ.



Hình 3.2: Luồng câu chuyện mở rộng chỉnh sửa món ăn của khách hàng

Chỉnh sửa thông tin khách hàng: Luồng câu chuyện này được mở rộng sau câu chuyện lấy được thông tin của khách hàng và hiển thị danh sách các thông tin để khách hàng kiểm tra như hình 3.3. Nếu như bot nhận được thông tin khách hàng có nhu cầu chỉnh sửa thông tin thì bot sẽ chuyển qua mẫu câu chuyện này. Câu chuyện này để thực hiện việc chỉnh sửa thông tin của khách hàng bao gồm tên, số điện thoại, số lượng người sử dụng và thời gian khách hàng có nhu cầu sử dụng. Sau khi chỉnh sửa thì bot sẽ quay lại mẫu câu chuyện xác nhận thông tin của khách hàng.



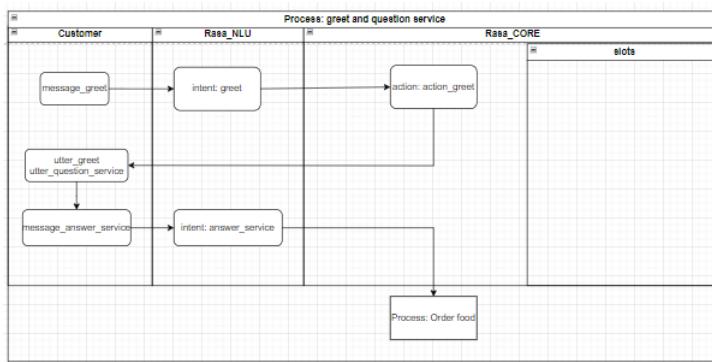
Hình 3.3: Luồng câu chuyện mở rộng chỉnh sửa thông tin của khách hàng

3.2 Các thành phần câu chuyện

3.2.1 Kịch bản chào và hỏi dịch vụ

Mở đầu luồng hội thoại giữa bot và khách hàng là khi bot nhận được câu chào từ khách hàng, sau đó bot sẽ xác nhận đó tin nhắn đó là intent greet thì bot sẽ thực hiện trả

về lời chào kèm theo câu hỏi về nhu cầu dịch vụ của khách hàng. Luồng hội thoại được thiết kế theo như hình 3.4. Sau khi bot xác định tin nhắn từ khách hàng là intent trả lời về dịch vụ thì bot sẽ chuyển sang câu chuyện tiếp theo.

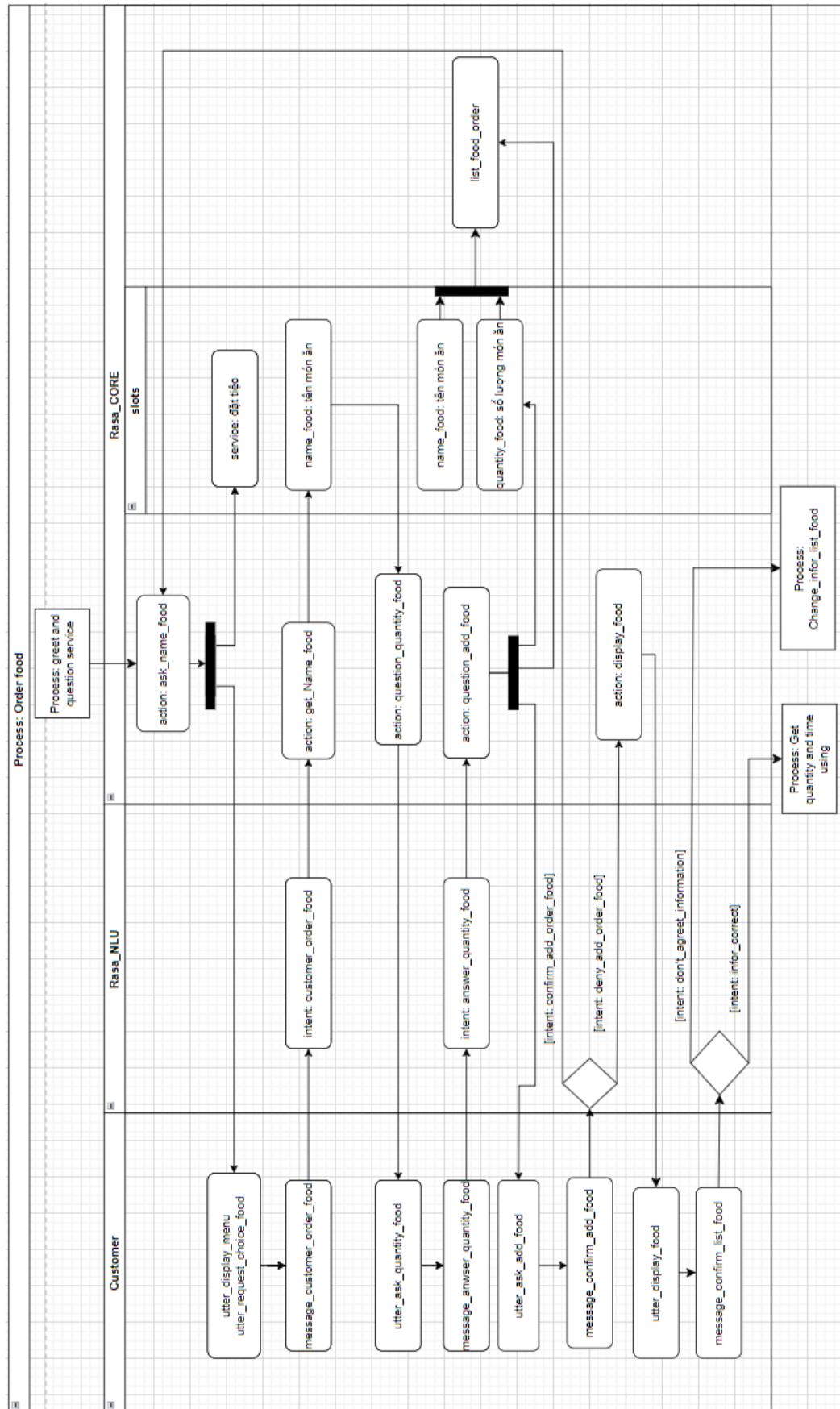


Hình 3.4: Luồng kịch bản chào và hỏi dịch vụ

3.2.2 Kịch bản đặt đồ ăn của khách hàng

Câu chuyện đặt đồ ăn với nhiệm vụ sẽ lấy được danh sách tên món ăn kèm theo số lượng từng món tương ứng trong danh sách. Chi tiết luồng hội thoại tương tác được thể hiện trong hình 3.5

- Từ intent trả lời dịch vụ trên, bot sẽ thực hiện hành động hỏi tên món ăn. Hành động này sẽ trích xuất entity service để lưu vào bộ nhớ tạm slot của rasa core đồng thời trả về tin nhắn hiện menu và hỏi khách hàng về tên món ăn.
- Từ intent trả lời tên món ăn, bot sẽ trích xuất tên món ăn và thực thể đó được lưu vào bộ nhớ tạm của bot. Sau đó tên món ăn sẽ được đưa vào câu hỏi về số lượng món ăn và trả về cho khách hàng.
- Từ intent trả lời số lượng món ăn, bot sẽ trích xuất số lượng món ăn đó, lưu cả hai tên món ăn và số lượng món ăn vào trong 1 list món ăn. Nếu khách hàng có nhu cầu đặt thêm món ăn thì luồng câu chuyện sẽ quay trở lại hỏi tên món ăn và số lượng món ăn như trên. Nếu khách hàng không có nhu cầu đặt thêm món ăn thì bot sẽ hiển thị danh sách các món ăn và số lượng tương ứng để khách hàng xác nhận.
- Từ tin nhắn xác nhận của khách hàng thì bot sẽ xác định tin nhắn đó thuộc intent đồng ý hay không đồng ý. Nếu khách hàng đồng ý danh sách đó thì luồng câu chuyện sẽ chuyển sang hỏi số lượng người sử dụng và thời gian sử dụng. Nếu không đồng ý thì luồng câu chuyện sẽ chuyển qua luồng câu chuyện thay đổi món ăn.



Hình 3.5: Luồng kịch bản đặt món ăn

3.2.3 Kịch bản thay đổi thông tin món ăn

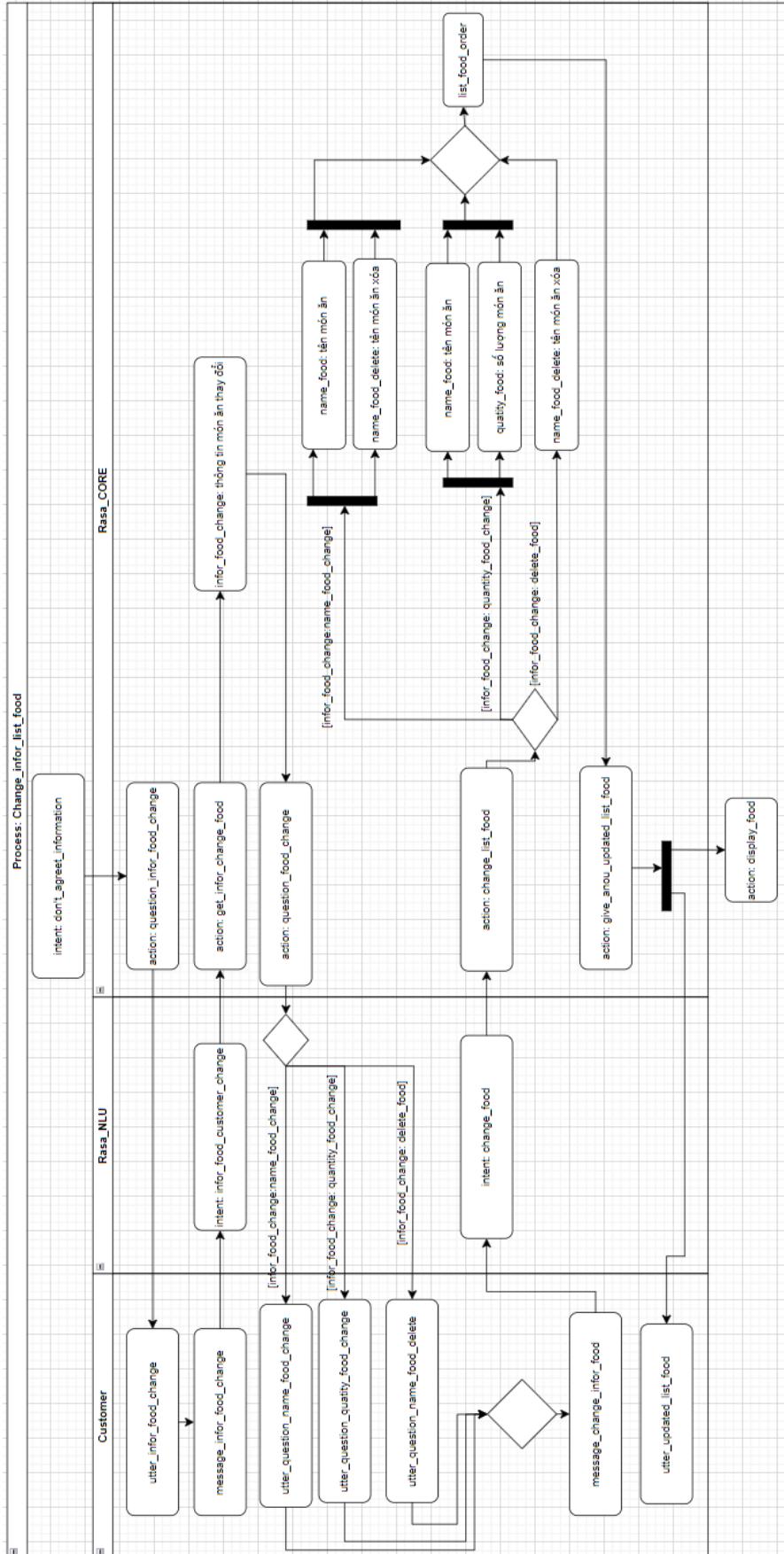
Câu chuyện này có chức năng thay đổi thông tin trong danh sách món ăn mà khách hàng đã đặt. Câu chuyện này có chức năng thay đổi các thông tin như tên, số lượng món ăn và hủy món ăn tùy thuộc vào yêu cầu của khách hàng trước khi tiến hành lấy thông tin của khách hàng. Luồng câu chuyện về chỉnh sửa danh sách món ăn được thiết kế như hình 3.6

- Sau khi khách hàng không đồng ý về danh sách món ăn thì bot sẽ phản hồi để hỏi về thông tin khách hàng cần thay đổi.
- Thông tin về sự thay đổi là xác định chức năng để bot có thể hiện thực để tra về thông tin chính xác. Dựa vào tin nhắn về thông tin thay đổi từ khách hàng bot sẽ trích xuất thông tin cần thay đổi có thể là thay đổi tên, thay đổi số lượng hoặc xóa món ăn nào đó. Sau đó bot sẽ phản hồi lại yêu cầu khách hàng nhập thông tin để thay đổi.
- Sau khi có được thông tin về thay đổi món ăn nào đó thì bot sẽ trích xuất thông tin thực thể về sự thay đổi đó nhằm cập nhật lại danh sách món ăn lại theo yêu cầu từ khách hàng.
- Sau khi danh sách được cập nhật, thì bot sẽ hiện thị lại danh sách để khách hàng kiểm tra lại. Nếu khách hàng không đồng ý hoặc có nhu cầu thay đổi thông tin trong danh sách món ăn thì bot sẽ quay lại luồng câu chuyện thay đổi đó. Nếu khách hàng đã đồng ý danh sách này thì bot sẽ chuyển qua luồng câu chuyện lấy thông tin của khách hàng.

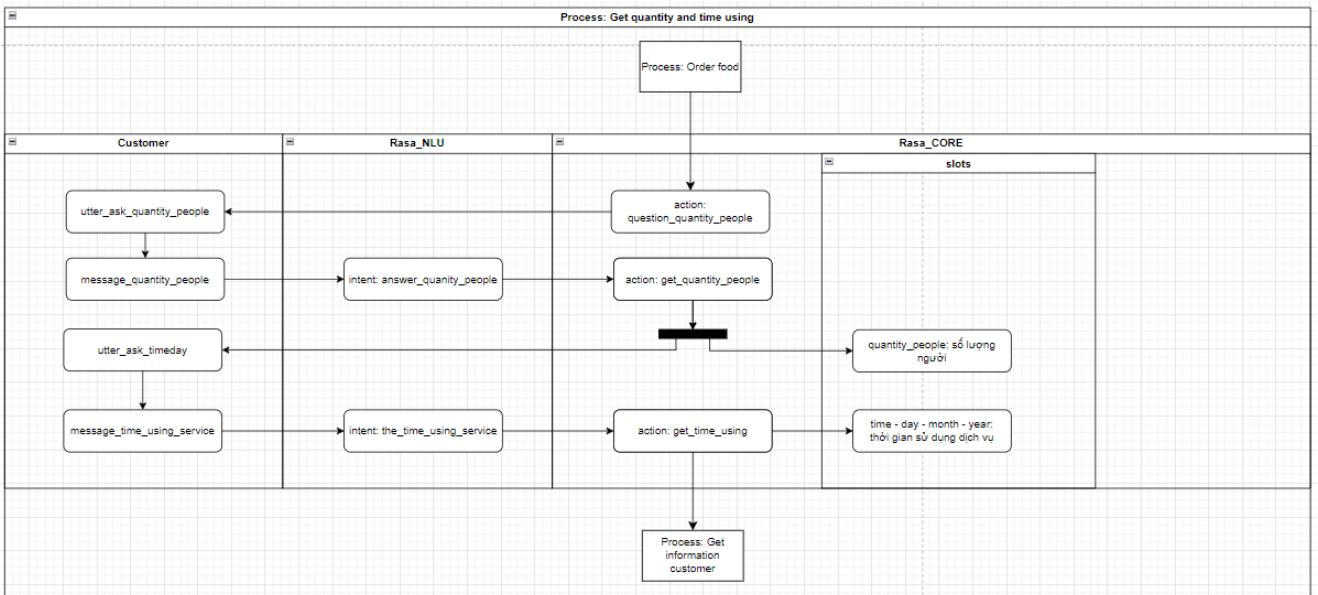
3.2.4 Kịch bản lấy thông tin về người tham gia và thời gian sử dụng

Câu chuyện này là có chức năng lấy 1 phần thông tin của người dùng bao gồm số lượng người sử dụng và thời gian sử dụng dịch vụ. Luồng câu chuyện này được thiết kế như hình 3.7.

- Sau khi đã lấy được thông tin về các món ăn mà khách hàng đã yêu cầu, bot sẽ chuyển qua luồng lấy thông tin số lượng người tham gia và thời gian sử dụng. Bot sẽ phản hồi lại để hỏi số lượng người tham gia đến khách hàng.
- Sau khi bot xác định được intent về số lượng người tham gia thì bot sẽ trích xuất thực thể người tham gia để lưu vào danh sách thông tin về khách hàng và bot sẽ phản hồi lại người dùng để hỏi về thời gian sử dụng.
- Sau khi bot xác định được intent về thời gian sử dụng thì bot sẽ tiến hành phân tích để lấy thông tin về thời gian bao gồm giờ, ngày, tháng, năm để lưu vào danh sách



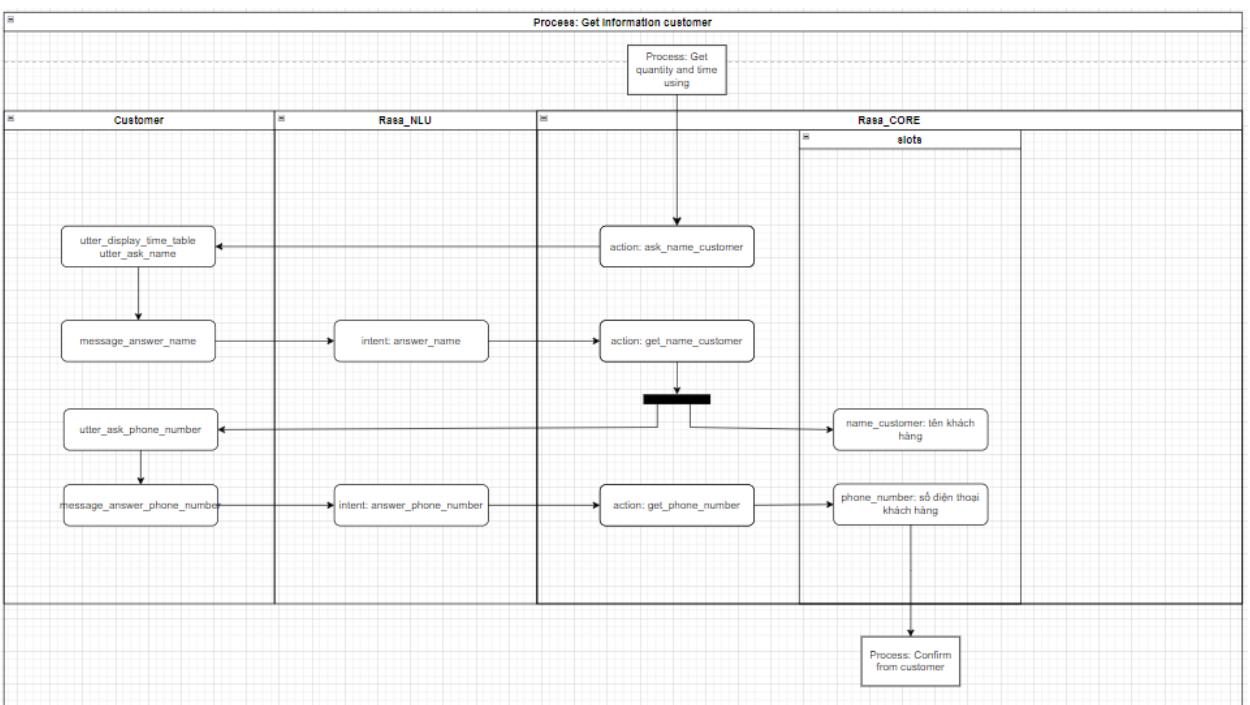
Hình 3.6: Luồng kịch bản chỉnh sửa thông tin món ăn



Hình 3.7: Luồng kịch bản lấy thông tin về người tham gia và thời gian sử dụng thông tin khách hàng. Và chuyển sang luồng câu chuyện lấy thông tin khách hàng.

3.2.5 Kích bản lấy thông tin khách hàng

Câu chuyện này có chức năng là lấy được thông tin của khách hàng bao gồm tên và số điện thoại. Luồng hội thoại của kịch bản này được thiết kế như hình ??

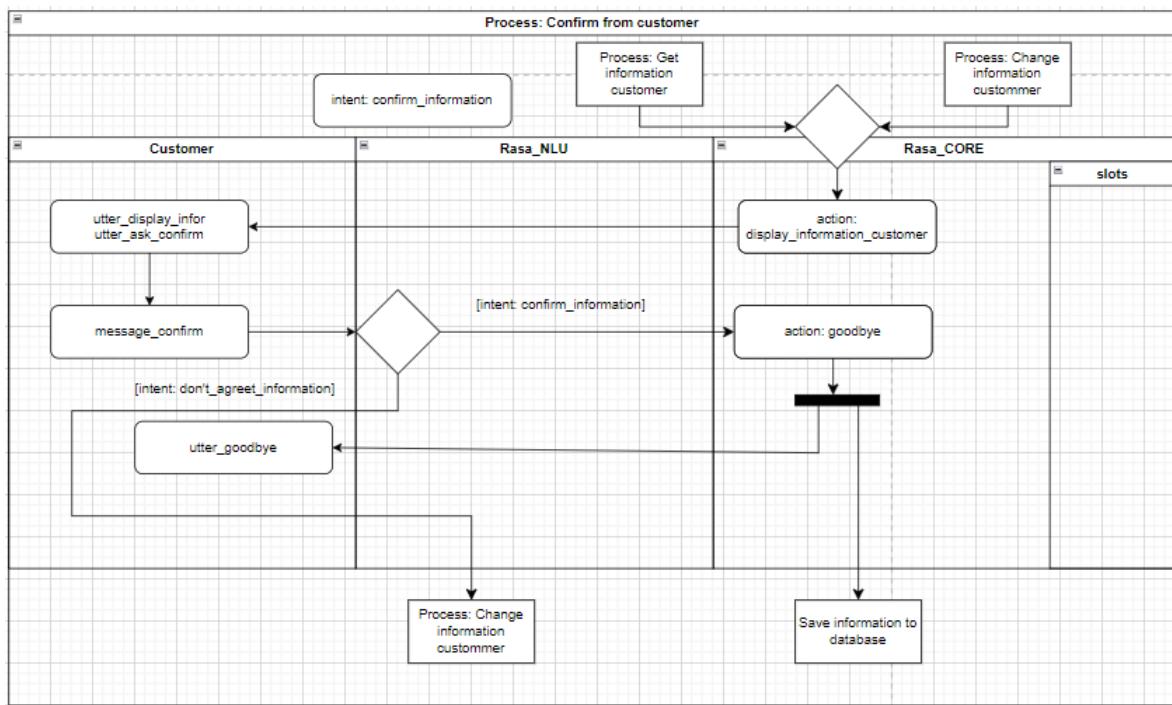


Hình 3.8: Luồng kích bản lấy thông tin về khách hàng

- Sau khi lấy được thời gian sử dụng dịch vụ tại nhà hàng thì bot sẽ phản hồi lại câu hỏi về tên nhằm lấy tên khách hàng.
- Sau khi lấy được tên khách hàng thì bot sẽ hỏi nhằm lấy số điện thoại của khách hàng.
- Sau khi lấy đầy đủ tên và số điện thoại của khách hàng thì bot sẽ chuyển qua luồng câu chuyện xác nhận tất cả thông tin của khách hàng.

3.2.6 Kịch bản xác nhận thông tin khách hàng

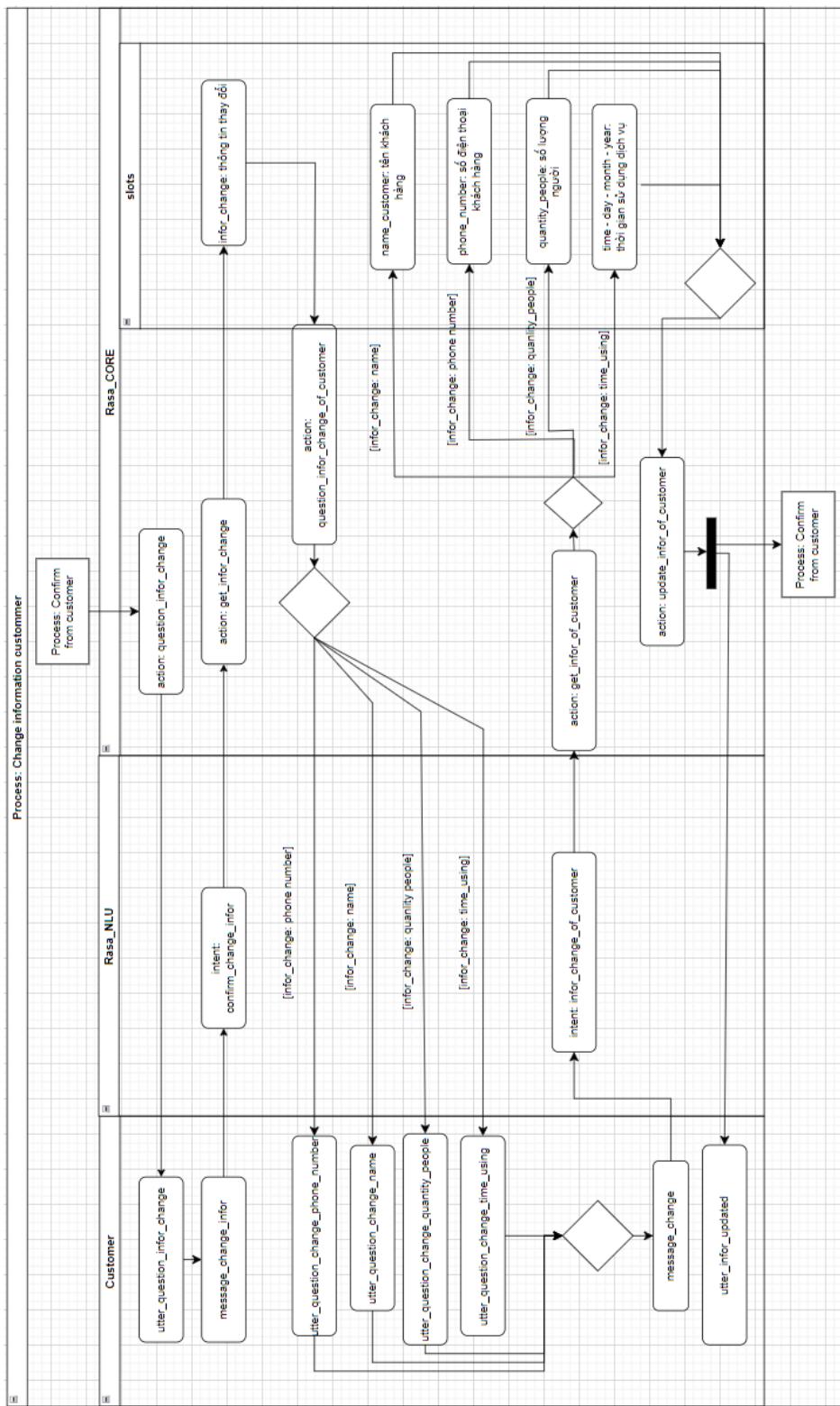
Câu chuyện này nhằm nhận được sự xác nhận từ khách hàng về thông tin mà bot lấy được từ khách hàng. Nếu khách hàng xác nhận thông tin chính xác thì bot sẽ lưu thông tin và kết thúc câu chuyện. Nếu khách hàng không chấp nhận thông tin hoặc muốn chỉnh sửa thì luồng câu chuyện sẽ chuyển qua câu chuyện thay đổi thông tin của khách hàng. Luồng câu chuyện được thiết kế như hình 3.10.



Hình 3.9: Luồng kịch bản xác nhận thông tin

3.2.7 Kịch bản chỉnh sửa thông tin khách hàng

Câu chuyện này có chức năng chỉnh sửa thông tin của khách hàng khi khách hàng có yêu cầu. Các thông tin có thể chỉnh sửa trong kịch bản này là tên, số điện thoại, số lượng người tham gia và thời gian sử dụng dịch vụ tại nhà hàng mà trước đó khách hàng đã đưa thông tin. Luồng hội thoại này được thiết kế như hình 3.10.



Hình 3.10: Luồng kịch bản chỉnh sửa thông tin khách hàng

-
- Sau khi xác định được thông tin cần thay đổi về thông tin khách hàng thì bot sẽ phản hồi lại nhằm để hỏi khách hàng cần thay đổi thông tin nào trong danh sách các thông tin của khách hàng.
 - Dựa vào câu trả lời của khách hàng về yêu cầu thông tin để thay đổi thì bot sẽ trích xuất thông tin cần thay đổi và lưu vào bộ nhớ slot của bot. Thông tin cần thay đổi tùy thuộc vào loại thông tin cần thay đổi là gì thì bot sẽ phản hồi lại câu hỏi về thông tin đó nhằm để lấy thông tin mới để chỉnh sửa.
 - Khi nhận được câu hỏi để khách hàng cung cấp thông tin mới để chỉnh sửa lại thông tin cũ thì khách hàng sẽ phản hồi lại để cung cấp thông tin mới thông qua tin nhắn được gửi cho bot.
 - Sau khi nhận được tin nhắn về cung cấp chỉnh sửa lại thông tin mới thì bot sẽ trích xuất thực thể mới đó và cùng với lại biến thông tin cần chỉnh sửa trên thì danh sách thông tin sẽ được chỉnh sửa lại theo thông tin mới. Cuối cùng thì sẽ hiển thị lại danh sách để khách hàng xác nhận lại cho bot.

Chương 4

Xây dựng Chatbot

4.1 Xây dựng câu chuyện trong Rasa

Trong file stories.yml, key stories được sử dụng để tạo kịch bản đặt tiệc nhà hàng. Trong đó, key story dùng để đặt tên cho câu chuyện cụ thể trong khi quá trình tương tác của người khách hàng và bot. Luồng hội thoại này là luồng hội thoại cơ bản mà bot dựa vào đó để xác định hành động tiếp theo cũng như xác định câu nói của khách hàng thuộc intent nào trong các intent được tạo ra. Dựa vào lưu đồ hình 3.1, hình 3.2 và hình 3.3, câu chuyện được tách thành ba mẫu chuyện chính:

- Luồng câu chuyện đặt tiệc cơ bản tại nhà hàng - story: simple order food restaurant.
- Luồng câu chuyện thay đổi thông tin danh sách món ăn - story: action change list food.
- Luồng câu chuyện thay đổi thông tin thông tin khách hàng - story: action change list information.

Các intent được sử dụng để khai báo các câu trả lời từ khách hàng. Các intent được khai báo bằng các key intent được cấu hình trong file stories.yml.

Các action được sử dụng để khai báo các hành động trả về cho người dùng từ bot khi nhận được các intent từ khách hàng. Trong file stories.yml, các action được khai báo bằng các key và được cấu hình trả về trong file action.py.

Trong từng mẫu câu chuyện - story nhỏ trong file stories.yml được khai báo liên tục action, intent để bot hành động liên tục nhận intent từ khách hàng và trả về các action về cho người dùng.

Đối với câu chuyện simple order food restaurant là một luồng câu chuyện được sử dụng để đặt tiệc nhà hàng. Khách hàng chỉ có thể đặt tiệc, gọi món và số lượng cho từng món. Khách hàng cũng cung cấp các thông tin cần thiết để lưu thông tin nhằm xác định

khách hàng là ai, thời gian sử dụng khi nào, số chỗ ngồi và danh sách các món ăn theo nhu cầu của khách hàng. Các thành phần trong câu chuyện này:

- intent: greet: Ý định lời chào được nhận từ người dùng được xem như một token để thực hiện action từ bot để phản hồi lời chào lại cho khách hàng
- action: action_greet: Hành động của bot khi nhận được lời chào từ khách hàng sẽ trả lại lời chào cho khách hàng. Đi kèm trong hành động thì bot cần hỏi thêm dịch vụ mà khách hàng có nhu cầu sử dụng. Sau khi thực hiện câu hỏi dịch vụ khách hàng muốn sử dụng thì bot sẽ đợi tin nhắn từ khách hàng trong intent: answer_service.
- intent: answer_service: Nhằm ý định lấy thông tin dịch vụ mà khách hàng yêu cầu.
- action: ask_name_food: Hành động của bot đưa ra menu nhằm ý định hỏi món ăn mà khách hàng muốn sử dụng tại nhà hàng.
- intent: customer_order_food: Lấy ý định món ăn từ câu trả lời từ khách hàng.
- action: get_name_food: Hành động của bot nhằm lấy tên món ăn từ tên hoặc chỉ số món ăn mà khách hàng đã yêu cầu.
- action: question_quantity_food: Hành động nhằm hỏi số lượng phần ăn của món ăn mà khách hàng muốn sử dụng.
- intent: answer_quantity_food: Ý định lấy số lượng món ăn từ câu trả lời của khách hàng.
- action: question_add_food: Hành động này phản hồi lại khách hàng nhằm hỏi khách hàng có ý định thêm món ăn nào khác hay không.
- intent: confirm_add_order_food: Lấy ý định xác nhận khách hàng có nhu cầu đặt thêm món ăn khác. Thì khi bot xác định tin nhắn của khách hàng thuộc intent này thì bot sẽ thực hiện lại hành động action: ask_name_food nhằm để lấy thông tin về món ăn theo nhu cầu của khách hàng.
- intent: deny_add_order_food: Lấy ý định xác nhận khách hàng không có nhu cầu đặt thêm món. Khi bot xác định tin nhắn từ khách hàng thuộc intent này thì bot sẽ thực hiện hành động action: display_food để đưa hiển thị danh sách các món ăn để khách hàng kiểm tra
- action: display_food: Hiện thông tin món ăn bao gồm tên món và số lượng món mà khách hàng đã đặt cho bot.
- intent: confirm_information_food: Lấy ý định xác nhận khách hàng đồng ý với danh sách các món ăn của khách hàng mà bot đã trích xuất ra được từ tin nhắn

của khách hàng.

- action: question_quantity_people: Hành động này sẽ trả về phản hồi đến khách hàng câu hỏi nhằm hỏi số lượng chỗ ngồi mà khách hàng có nhu cầu sử dụng.Nhằm để biết thêm thông tin số lượng khách hàng giúp người quản lý có thể đặt bàn hợp lý cho khách hàng sử dụng hợp lý. Sau khi hiện câu hỏi về số lượng người dùng thì bot sẽ đợi câu trả lời từ intent: answer_quantity_people.
- intent: answer_quantity_people: Nhằm ý định lấy thông tin số lượng người tham gia khi sử dụng dịch vụ tại nhà hàng.
- action: get_quantity_people: Hành động này bao gồm lưu thông tin số lượng người sử dụng và kèm theo đó là câu hỏi về thời gian mà khách hàng muốn sử dụng dịch vụ tại nhà hàng.
- intent: the_time_using_service: Lấy ý định thông tin về thời gian mà khách hàng có nhu cầu sử dụng tại nhà hàng.
- action: get_time_using: Hành động này lấy thông tin về thời gian từ câu trả lời từ khách hàng mà khách hàng muốn sử dụng dịch vụ tại nhà hàng.
- action: ask_name_customer: Hành động này sẽ hỏi tên khách hàng.
- intent: answer_name: Nhằm ý định yêu cầu thông tin tên khách hàng sử dụng dịch vụ từ tin nhắn của khách hàng.
- action: get_name_customer: Hành động này nhằm lấy thông tin tên khách hàng và kèm theo đó là câu hỏi nhằm lấy thông tin số điện thoại của khách hàng.
- intent: answer_phone_number: Nhằm ý định yêu cầu thông tin về số điện thoại của khách hàng.
- action: get_phone_number: Hành động này lấy thông tin về số điện thoại của khách hàng từ câu trả lời khách hàng từ intent bên trên.
- action: display_information_customer: Sẽ trả về thông tin về khách hàng bao gồm tên khách hàng, số điện thoại, tên món ăn và số lượng món ăn, thời gian mà khách hàng muốn sử dụng dịch vụ và số lượng người tham gia.
- intent: confirm_information: Nhằm ý định yêu cầu xác nhận thông tin chính xác từ khách hàng để đi tới hành động tiếp theo của bot.
- action: goodbye: Hành động này trả về tin nhắn cảm ơn khách hàng đã sử dụng dịch vụ của nhà hàng.

Đối với câu chuyện action change list food, kịch bản này có chức năng thay đổi thông tin trong danh sách các món ăn theo yêu cầu của khách hàng. Các sự thay đổi có thể hiện

thực trong luồng câu chuyện này là tên món ăn, số lượng món ăn hoặc hủy món ăn nào đó đã được đặt trong danh sách món ăn. Luồng câu chuyện bao gồm các thành phần:

- intent: don't_agree_information: Ý định này được xác định nhằm xác định thông điệp muốn thay đổi thông tin về danh sách món ăn từ khách hàng. Khi đã xác định được yêu cầu thay đổi món ăn từ khách hàng thì bot sẽ phản hồi lại yêu cầu thông tin mà khách hàng muốn thay đổi.
- action: question_infor_food_change: Hành động này sẽ trả về câu hỏi về thông tin mà khách hàng muốn thay đổi. Các thông tin này bao gồm thay đổi tên món, thay đổi số lượng món ăn và xóa món ăn.
- intent: infor_food_customer_change: Ý định này nhằm xác định được thông tin mà khách hàng muốn thay đổi. Tại đây, thực thể về thông tin cần chỉnh sửa trong danh sách món ăn sẽ được lấy ra từ tin nhắn của khách hàng.
- action: get_infor_change_food: Hành động này sẽ dựa vào thông tin được lấy từ intent: infor_food_customer_change để xác định câu phản hồi lại đến khách hàng.
- action: question_food_change: Sau khi lấy được thông tin cần thay đổi, bot sẽ quyết định trả về câu hỏi để xác nhận tên món ăn cần thay đổi và thay đổi như thế nào.
- intent: change_food: Ý định này sẽ xác định được món ăn và sự thay đổi về thông tin của món ăn đó. Các thông tin sẽ được trích xuất từ tin nhắn của khách hàng.
- action: change_list_food: Dựa vào các thực thể mà bot đã trích xuất được tại intent: change_food, bot sẽ thực hiện thay đổi thông tin danh sách món ăn.
- action: give_anou_updated_list_food: Sau khi danh sách đã được thay đổi thì bot sẽ phản hồi lại thông báo cho khách hàng về sự thay đổi thông tin của món ăn.

Đối với câu chuyện chỉnh sửa thông tin của khách hàng, kịch bản cũng được thiết kế tương tự như kịch bản chỉnh sửa thông tin trong danh sách món ăn. Các thông tin khách hàng có thể thay đổi là tên, số điện thoại, số người tham gia hoặc thời gian sử dụng. Luồng câu chuyện bao gồm các thành phần:

- intent: don't_agree_information: Ý định này được xác định nhằm xác định thông điệp muốn thay đổi thông tin về khách hàng. Khi đã xác định được yêu cầu thay đổi món ăn từ khách hàng thì bot sẽ phản hồi lại yêu cầu thông tin mà khách hàng muốn thay đổi.

-
- action: question_infor_change: Hành động này sẽ trả về câu hỏi về thông tin mà khách hàng muốn thay đổi.
 - intent: confirm_change_infor: Ý định này nhằm xác định được thông tin mà khách hàng muốn thay đổi. Tại đây, thực thể về thông tin cần chỉnh sửa trong danh sách thông tin của khách hàng sẽ được lấy ra từ tin nhắn của khách hàng.
 - action: get_infor_change: Hành động này sẽ dựa vào thông tin được lấy từ intent: confirm_change_infor để xác định câu phản hồi lại đến khách hàng.
 - action: question_infor_change_of_customer: Sau khi lấy được thông tin cần thay đổi, bot sẽ quyết định trả về câu hỏi để xác nhận thông tin cần thay đổi và thay đổi như thế nào.
 - intent: infor_change_of_customer: Ý định này sẽ xác định được thông tin và sự thay đổi về thông tin đó. Các thông tin sẽ được trích xuất từ tin nhắn của khách hàng.
 - action: get_infor_of_customer: Dựa vào các thực thể mà bot đã trích xuất được tại intent: infor_change_of_customer, bot sẽ thực hiện thay đổi thông tin của khách hàng.
 - action: update_infor_of_customer: Sau khi danh sách đã được thay đổi thì bot sẽ phản hồi lại thông báo cho khách hàng về sự thay đổi thông tin của khách hàng.

4.2 Xây dựng quy tắc trong Rasa

Nội dung quy tắc được cấu hình trong file rules.yml. Các quy tắc trong rasa cung cấp một cách để mô tả các đoạn hội thoại ngắn luôn diễn ra theo một cách. Các quy tắc là một khuôn mẫu mà bot phải tuân theo.

greet user
- intent: greet
- action: action_greet

Trong rule Greet user, mỗi khi người dùng đưa ra câu chào bằng intent: greet thì bot đều sử dụng action: action_greet để thực hiện lại lời chào cho khách hàng bất kể luồng câu chuyện đang ở thời điểm nào trong cuộc trò chuyện.

ask_quantity_food
- intent: customer_order_food
- action: get_name_food
- action: question_quantity_food

Trong quy tắc `ask_quantity_food`, khi người dùng nhập chỉ số món thì bot cần phải ngay lập tức lưu tên món trước khi thực hiện câu hỏi số lượng món ăn đó trong action tiếp theo. Điều này làm tránh trường hợp tên món bị thiếu trong câu hỏi về số lượng món ăn mà khách hàng yêu cầu. Trong nguyên tắc này, thì - `action: get_name_food` cần được thực hiện ngay khi khách hàng gửi chỉ số món cho bot.

get_time
- intent: the_time_using_service
- action: get_time_using
- action: ask_name_customer

Quy tắc `get time`, khi khách hàng gửi thông tin về thời gian về cho bot trong `intent: the_time_using_service` thì bot cần thực hiện ngay lập tức `action: get_time_using` để sử lý và lấy thông tin về giờ, ngày, tháng, năm mà khách hàng có nhu cầu sử dụng dịch vụ.

confirm and display_food
- intent: deny_add_order_food
- action: display_food

Quy tắc `confirm and display_food`, khi khách hàng gửi thông điệp về từ chối thêm món khi bot hỏi khách hàng về việc thêm món ăn thì bot sẽ lập tức thực hiện hành động hiển thị danh sách món ăn để khách hàng kiểm tra.

Các quy tắc trong khi cấu hình cần được chọn lọc những intent và action cần thiết. Điều này quan trọng, luồng quy tắc cấu hình thiếu, thì thông tin cần thiết để bot có thể lấy entity từ tin nhắn của khách hàng bị thiếu, hoặc chưa thể xử lý để lấy các entity cần thiết thì câu trả lời của bot sẽ bị thiếu thông tin. Nếu thông tin chưa đầy đủ, khi action thực hiện lệnh gọi lấy thông tin từ entity bị thiếu có thể gây ra lỗi cho bot và bot sẽ không xử lý được. Nếu cấu hình quy tắc quá nhiều, các intent và các action không cần thiết nếu cấu hình quy tắc sẽ làm cho training model tốn nhiều thời gian. Vì vậy, trong khi xem xét để cấu hình quy tắc cho bot, cần phải chọn lọc những quy tắc bao gồm intent và action thực sự cần thiết để cấu hình, để tiết kiệm thời gian đào tạo model cũng như đảm bảo

luồng hội thoại không bị sai lệch.

4.3 Truy xuất và lưu thông tin từ người dùng

Các entity cần được khai báo trước khi được sử dụng để truy xuất thực thể tương ứng từ tin nhắn của người dùng. Các entity này được cấu hình trong file domain.yml. Khi quyết định cần trích xuất thực thể nào để có thể sử dụng cho bot thì cần khai báo tên thực thể tương ứng trước khi sử dụng.

Trong câu chuyện này, các entity quan trọng cần được trích xuất từ tin nhắn người dùng như:

- **service**: Được sử dụng để trích xuất thông tin thực thể về nhu cầu khách hàng muốn sử dụng dịch vụ nào của nhà hàng.
- **name_food**: Được sử dụng để trích xuất thông tin thực thể về món ăn theo nhu cầu của khách hàng.
- **name_food_delete**: Được sử dụng để trích xuất thông tin về món ăn mà khách hàng muốn thay đổi hoặc xóa.
- **quantity_food**: Được sử dụng để lấy thông tin về số phần của món ăn mà khách hàng đã đặt ở trên.
- **quantity_people**: Được sử dụng để trích xuất thông tin về số lượng người tương ứng với số lượng chỗ mà khách hàng đặt.
- **name_customer**: Được sử dụng để trích xuất tên của khách hàng.
- **phone_number**: Được sử dụng để trích xuất số điện thoại của khách hàng.
- **time, day, month, year**: Được sử dụng để truy xuất thời gian bao gồm giờ, ngày, tháng, năm mà khách hàng có nhu cầu sử dụng dịch vụ tại nhà hàng.
- **infor_change**: Được sử dụng nhằm trích xuất về thông tin thay đổi theo yêu cầu của khách hàng.
- **infor_food_change**: Được sử dụng nhằm trích xuất về thông tin về món ăn thay đổi theo yêu cầu của khách hàng.

Sau khi truy xuất các thực thể từ tin nhắn người dùng, bot cần sẽ lưu thông tin của người dùng dưới dạng dictionay. Bot sẽ được khai báo và lưu các entity vào các slots. Các slot này được cấu hình trong file domain.yml. Các slot sẽ có tên khóa được khai báo bằng **slot_name** và giá trị sẽ được ánh xạ từ thực thể khi được trích xuất từ tin nhắn của khách hàng.

Từ 13 thực thể được trích xuất bên trên, 13 slot tương ứng sẽ được tạo để lưu thông tin từ 13 thực thể trên. Các slot được sử dụng để lưu thông tin:

- **service**: Được sử dụng để lưu thông tin về dịch vụ mà khách hàng muốn sử dụng tại nhà hàng. Slot này được lưu dưới dạng text và được ánh xạ đến thực thể service tương ứng. Đối với slot này, tùy vào từng loại dịch vụ mà khách hàng muốn sử dụng tại nhà hàng mà luồng hội thoại của bot sẽ đi theo mỗi hướng khác nhau. Vì vậy, **influence_conversation**: true cần sử dụng ảnh hưởng đến luồng hội thoại.
- **name_food**: Được sử dụng để lưu thông tin dưới dạng text của tên món ăn mà khách hàng đã chọn.
- **name_food_delete**: Được sử dụng để lưu thông tin dưới dạng text của tên món ăn mà khách hàng muốn hủy.
- **quantity_food**: Được sử dụng để lưu thông tin dưới dạng float về số phần của món ăn mà khách hàng đã đặt ở trên.
- **quantity_people**: Được sử dụng để lưu thông tin dưới dạng float trích xuất thông tin về số lượng người tương ứng với số lượng chỗ mà khách hàng đặt.
- **name_customer**: Được sử dụng để lưu thông tin dưới dạng text về tên của khách hàng.
- **phone_number**: Được sử dụng để lưu thông tin dưới dạng float về số điện thoại của khách hàng.
- **time, day, month, year**: Được sử dụng lưu thông tin dưới dạng text về thời gian bao gồm giờ, ngày, tháng, năm mà khách hàng có nhu cầu sử dụng dịch vụ tại nhà hàng.
- **infor_change**: Được sử dụng để lưu thông tin dưới dạng text của thông tin mà khách hàng yêu cầu thay đổi.
- **infor_food_change**: Được sử dụng để lưu thông tin dưới dạng text của thông tin mà khách hàng yêu cầu thay đổi.

Ví dụ cấu hình slots trong Rasa bao gồm những thông số cơ bản như kiểu dữ liệu, thông số influence_conversation ý định thực thể này có ảnh hưởng đến luồng hội thoại hay không. Nếu có thì được gắn thông số là true biểu hiện tùy vào từng dữ liệu được lưu trong slot này sẽ đi đến một luồng hội thoại khác nhau. Kiểu ánh xạ và vị trí ánh xạ sẽ được cấu hình đến vị trí thực thể đã được trích xuất từ thông tin khách hàng.

slots:

service:

type: text

```
influence_conversation: true
mappings:
- type: from_entity
  entity: service

name_food:
type: text
mappings:
- type: from_entity
  entity: name_food

name_food_index:
type: text
mappings:
- type: from_entity
  entity: name_food_index

quantity_food:
type: float
mappings:
- type: from_entity
  entity: quantity_food
```

Bộ nhớ slot trong Rasa không phải ở dạng danh sách nên khi bot lập lại hành động trích xuất thực thể lần thứ hai thì thực thể này sẽ ghi đè lên thực thể đã lưu lần thứ nhất. Nên khi khách hàng có nhu cầu gọi món ăn khác thì món ăn thứ hai sẽ được lưu lại bộ nhớ slot và số lượng món ăn cũng sẽ bị ghi đè lên dữ liệu lần thứ nhất. Vì vậy, để giải quyết vấn đề này thì danh sách các món ăn sẽ được tạo trong Rasa action, danh sách này sẽ lưu tên món ăn cùng với số lượng món ăn sau khi bộ nhớ slot về tên món ăn cùng với số lượng món ăn của món ăn tương ứng được trích xuất đầy đủ. Sau mỗi lần khách hàng gọi thêm món thì danh sách các món ăn sẽ thêm dữ liệu tại hai slot tên món ăn và số lượng món ăn.

4.4 Dữ liệu đào tạo NLU

Dữ liệu này được cấu hình trong file nlu.yml, dữ liệu này bao gồm các ví dụ về cách nói chuyện của người dùng được phân loại thành các intent. Các ý định này bao gồm các những câu nói xin chào bình thường hoặc các câu nói có chứa các entity để bot có thể truy xuất và lưu vào các slot. Sau khi phân tích cú pháp bot có thể hiểu được các ý định của người dùng nhằm thực hiện action tiếp theo một cách chính xác.

intents:

- greet
- answer_service
- customer_order_food
- anwser_quantity_food
- confirm_add_order_food
- deny_add_order_food
- answer_quantity_people
- the_time_using_service
- answer_name
- answer_phone_number
- confirm_information
- don't_agreet_information
- confirm_information_food
- confirm_change_infor
- infor_change_of_customer
- infor_food_customer_change
- change_food

Trong intent greet nhằm ý định chào hỏi bot, tạo ra tín hiệu là cần chat với bot nên ở intent này cần thêm một số ví dụ câu chào hỏi.

- intent: greet

examples: |

- hello
- hi
- chào bot
- chào
- xin chào

Trong intent tiếp theo là answer_service, intent này bot cần truy xuất thực thể service từ câu trả lời của khách hàng nên các thực thể cần được chú thích trong ví dụ đào tạo với tên thực thể. Ví dụ trong intent này, thực thể service này muốn truy xuất từ đặt tiệc từ tin nhắn người dùng và sau đó lưu thực thể này vào trong slot service tương ứng để bot thực hiện hành động tiếp theo.

- intent: answer_service

examples: |

- [đặt tiệc](service)
- anh muốn [đặt tiệc](service)
- tôi muốn [đặt tiệc](service)
- chị muốn [đặt tiệc](service)

Bởi vì slot service được cấu hình là influence_conversation: true có nghĩa là thực thể này ảnh hưởng đến hội thoại nên tùy vào từng dữ liệu từ slot service mà luồng hội thoại sẽ đi đến một chủ đề khác nhau. Trong câu chuyện này, câu chuyện chỉ xây dựng cho chủ đề đặt tiệc trong nhà hàng, nên slot này được chỉ có dữ liệu là đặt tiệc và sau đó sẽ đi đến kịch bản chọn món tại nhà hàng.

Trong dữ liệu về chọn món, dữ liệu khách hàng nhập vào có thể là chỉ số món hoặc tên món dựa trên menu mà bot đã hiện ra trước khi yêu cầu khách hàng chọn món. Dữ liệu khách hàng mà bot có thể trích xuất ra được sau đó được map tới danh sách từ đồng nghĩa của tên món ăn để được tên món ăn chính xác. Trong danh sách các từ đồng nghĩa thì tên là tên của món ăn và danh sách là bao gồm các trường hợp mà khách hàng sử dụng để gọi món ăn đó có thể là chỉ số món trong menu hoặc có thể là tên của món ăn đó.

- intent: customer_order_food

examples: |

- Cho tôi món [thứ 1](name_food)
- Cho tôi gọi món [Baba hầm hải vị](name_food)
- Cho tôi món [Bánh tiramisu](name_food)
- Món [số 2](name_food)
- Món [Số một](name_food)
- món [số mười bốn](name_food)
- ...

Các tên của món ăn sẽ được tạo một bản tìm kiếm nhằm khi tên món ăn được trích xuất sẽ tra về đúng tên món ăn đó. Các món ăn sẽ được tạo thành một danh sách các từ

đồng nghĩa về món ăn đó có thể là chỉ số hoặc tên. Các từ hoặc các câu từ đồng nghĩa là những từ mà khách hàng có thể nhập vào. Nên từ câu nói của khách hàng, bot sẽ lấy từ đồng nghĩa đó để truy xuất đến tên trong bản tìm kiếm và cuối cùng sẽ trả về tên chính xác cho món ăn theo nhu cầu của khách hàng.

- lookup: name_food

examples: |

- Baba hầm hải vị kèm bánh mì
- Bào ngư hầm nấm đồng cô và cải thìa
- Bánh cam chiên với nước cốt dừa
- Bánh kem mềm chanh dây, trái cây, vụn hạt dẻ, xốt rượu Malibu
- Bánh mochi
- ...

- synonym: Baba hầm hải vị kèm bánh mì

examples: |

- Số 15
- số 15
- Số mươi lăm
- thứ mươi năm
- Thứ 15
- thứ 15
- Baba hầm hải vị kèm bánh mì
- baba hầm hải vị kèm bánh mì
- Baba
- baba
- Baba hầm hải vị
- baba hầm hải vị

Sau khi có được slot name_food bot sẽ sử dụng thực thể này để kết hợp với các câu hỏi thoại từ bot để phản hồi lại cho khách hàng một tin nhắn hoàn chỉnh và đúng ý nghĩa.

Đối với dữ liệu về yêu cầu số lượng món ăn được khai báo tương tự như dữ liệu về chỉ số món ăn bao gồm khai báo intent về câu trả lời về số lượng món ăn và khai báo một biểu thức chính quy về thực thể số lượng món ăn. Dữ liệu về xác nhận và từ chối thêm món ăn từ khách hàng, dữ liệu này bao gồm các câu hoặc các từ để xác nhận ý định của khách hàng có hoặc không. Khi câu nói của khách hàng mà bot xác nhận là đồng ý thì bot sẽ thực hiện hành động quay lại để hỏi về món ăn, nếu bot xác nhận về câu nói của

khách hàng không có nhu cầu thêm món thì bot sẽ thực hiện hành động hiển thị danh sách món ăn.

Tiếp theo cấu hình dữ liệu về số lượng người sử dụng để truy xuất và lưu thông tin về số lượng người sử dụng dịch vụ tại nhà hàng. Cách cấu hình cũng tương tự như cấu hình về chỉ số món ăn và số lượng món ăn đã cấu hình phía trên chỉ thay đổi tên của intent tương ứng cho phù hợp.

Cấu hình dữ liệu về thời gian mà khách hàng muốn sử dụng dịch vụ tại nhà hàng. Đôi với câu nói nhu cầu về thời gian của khách hàng và thông tin chuẩn về thời gian thì rất khác biệt. Hầu như câu nói ám chỉ thời gian mà khách hàng yêu cầu thì chỉ chứa đựng về giờ và ngày bằng những câu nói ví dụ như:

- 1 giờ rưỡi chiều ngày mai
- 7 giờ tối mốt
- 8 giờ sáng mai
- 7 giờ rưỡi tối mai
- 5 giờ rưỡi chiều hôm nay
- ...

Đối với những câu nói thông thường không có cấu trúc về dữ liệu như trên thì trước khi lưu vào slot thì cần phải thông qua một bước xử lí của bot trước khi lưu vào slot đầy đủ thông tin về giờ, ngày, tháng, năm. Trước tiên, thời gian sử dụng dịch vụ cần được cấu hình để cho bot có thể nhận dạng và trích xuất được một số thực thể từ câu nói của người dùng. Các thực thể được trích xuất và được xử lý thông qua file timehandle.py để đưa về dạng giờ, ngày, tháng, năm. Ví dụ một số câu nói và khai báo thực thể có thể được trích xuất từ câu nói từ khách hàng:

- [Trưa] (time) [nay] (day)
- [Chiều] (time) [mai] (day)
- [Tối] (time) [mốt] (day)
- [Cuối ngày] (time) [hôm nay] (day)
- [6 giờ 20] (time)
- ...

Thông thường trong câu nói của khách hàng trong khi đặt tiệc dịch vụ thì chỉ truy xuất được 2 thực thể về thời gian là time và day. Để chính xác hơn thì các tên thực thể này sẽ được sử dụng bảng tìm kiếm để đối chiếu thực thể trích xuất được và bảng tìm kiếm này.

-
- lookup: time

examples: |

- 1 giờ chiều
- 1 giờ rưỡi
- 2 giờ rưỡi
- 1 giờ rưỡi chiều
- ...

- lookup: day

examples: |

- mai
- ngày mai
- một
- ngày một
- hôm nay
- ...

Trong phần thời gian này, để tránh trường hợp người dùng trả lời ngày trong tuần là số hoặc chữ ví dụ như thứ 2 và thứ hai làm cho dữ liệu training nhiều, bảng về từ đồng nghĩa sẽ đưa dạng chữ về dạng số. Để thực hiện việc này cần sử dụng bảng từ đồng nghĩa như ví dụ bên dưới:

- synonym: thứ 2

examples: |

- thứ hai
- ...

Đối với dữ liệu về việc xác nhận dữ liệu chấp nhận hoặc có yêu cầu cần chỉnh sửa thì dữ liệu này bao gồm các câu từ mang ý nghĩa đồng ý hay không, cần chỉnh sửa dữ liệu hay không.

Trong phần dữ liệu `infor_food_customer_change`, dữ liệu được tạo bao gồm thực thể thông tin của sự thay đổi `infor_food_change` nhằm để lấy thông tin mà khách hàng yêu cầu thay đổi. Vì vậy, trong câu nói thuộc ý định này sẽ chứa thông tin của sự thay đổi đó. Các thông tin thuộc 3 thông tin chính là `name_food_change`, `quantity_food_change`, `delete_food` tương ứng với 3 chức năng chỉnh sửa thông tin của danh sách món ăn như thay đổi tên món ăn, số lượng món ăn và xóa món ăn. Các

thông tin này sẽ được tạo một danh sách các từ đồng nghĩa mà khách hàng có thể sử dụng. Khi bắt được các thông tin này thì bot sẽ trả về câu hỏi về dữ liệu thay đổi.

- intent: infor_food_customer_change

examples: |

- tôi muốn thay đổi [tên món ăn] (infor_food_change)
- [hủy món] (infor_food_change) này giúp tôi
- cho chị đổi [món ăn] (infor_food_change)
- ...

- lookup: infor_food_change

examples: |

- name_food_change
- quantity_food_change
- delete_food

- synonym: name_food_change

examples: |

- tên món ăn
- tên món
- món ăn

Đối với dữ liệu thay đổi thông tin về danh sách món ăn change_food, dữ liệu sẽ chứa các dữ liệu của sự thay đổi về thông tin đó. Các thực thể sử dụng trong các câu này bao gồm name_food_delete, name_food, quantity_food tương ứng với thực thể tên món ăn được thay đổi hoặc xóa, tên món ăn thay thế và số lượng món ăn theo yêu cầu chỉnh sửa từ khách hàng.

- intent: change_food

examples: |

- tôi muốn thay đổi món [Bánh mochi] (name_food_delete) thành món [Bánh su với kem phomai hạt dẻ] (name_food)
- tôi muốn xóa món [Baba hầm hải vị kèm bánh mì] (name_food_delete)
- thay đổi món [Bánh mochi] (name_food) thành [5] (quantity_food)
- phần
- ...

Trong phần dữ liệu confirm_change_infor, dữ liệu được tạo bao gồm thực thể thông tin của sự thay đổi infor_change nhằm để lấy thông tin mà khách hàng yêu cầu thay đổi. Vì vậy, trong câu nói thuộc ý định này sẽ chứa thông tin của sự thay đổi đó. Các thông tin thuộc 4 thông tin chính là name_customer, phone_number, time_using, quantity_people tương ứng với 4 dữ liệu lấy từ khách hàng là tên, số điện thoại, thời gian sử dụng và số lượng người sử dụng nhà hàng. Các thông tin này sẽ được tạo một danh sách các từ đồng nghĩa mà khách hàng có thể sử dụng. Khi bắt được các thông tin này thì bot sẽ trả về câu hỏi về dữ liệu thay đổi.

- intent: confirm_change_infor

examples: |

- tôi muốn chỉnh sửa [tên] (infor_change)
- tôi cần thay đổi [thời gian sử dụng] (infor_change)
- tôi muốn thay đổi [time_using] (infor_change)
- ...

- lookup: infor_change

examples: |

- name_customer
- phone_number
- time_using
- quantity_people

- synonym: name_customer

examples: |

- name
- tên
- tteen
- teen
- tên của tôi
- teen của tôi
- tên tôi

Tiếp theo đó là phần dữ liệu infor_change_of_customer về lấy thông tin thay đổi nhằm cập nhập lại danh sách của thông tin khách hàng. Các dữ liệu của ý định này chứa thông tin về thực thể mà khách hàng cần cập nhật. Khi bot trích xuất được thực thể này thì bot sẽ cập nhật lại danh sách thông tin của khách hàng.

-
- intent: infor_change_of_customer
 - examples: |
 - tên của tôi là [An] (name_customer)
 - số lượng người sử dụng chỉ có [9] (quantity_people)
 - số điện thoại của chị là [013644897] (phone_number)
 - ...

Biểu thức chính quy được sử dụng trong dữ liệu đào tạo nhằm giảm sự nhầm lẫn của các thực thể khi bot dự đoán thực thể. Nên một số thực thể có thể được sử dụng biểu thức chính quy để quy định thêm về kiểu dữ liệu như số lượng món ăn có thể 1 đến 2 chữ số, số điện thoại có thể 9 đến 11 số, .v.v.

- regex: quantity_food
 - examples: |
 - \d{1,2}

4.5 Tạo mẫu câu phản hồi người dùng

Đối với các mẫu câu được bot sử dụng để trả về cho người dùng được cấu hình trước trong file domain.yml hoặc cấu hình trong file action khi có nhu cầu sử dụng để phản hồi lại khách hàng. Các mẫu câu được tạo dưới tên responses tiếp theo là key của mẫu câu được đưa vào các action để phản hồi lại khách hàng. Key của phản hồi sẽ trích xuất đoạn tin nhắn để phản hồi lại khách hàng như ví dụ bên dưới:

- utter_greet:
 - text: "Nhà hàng Bot_Rasa xin chào quý khách."

Ví dụ mẫu câu phản hồi đơn giản này, mẫu câu này sẽ được rasa sử dụng để phản hồi lại bằng action: action_greet để phản hồi lại người dùng sau khi khách hàng chào rasa khi có nhu cầu sử dụng nhà hàng. Cấu hình câu trả lời được định dạng theo file yml nên cấu trúc viết câu phải theo định dạng file yml để tránh bị lỗi khi tiến hành training. Mẫu câu đơn sử dụng - text: "Lời nhắn". Nếu tin nhắn có độ dài phải xuống dòng để phù hợp trong giao diện tin nhắn thì phải thay đổi theo định dạng như ví dụ bên dưới về hiện menu của nhà hàng:

- utter_display_menu:

- text: |

Khai vị:

-
1. Càng cua bách hoa
 2. Gỏi hải sâm kèm bánh phồng tôm
 3. Súp hải sản hoàng kim
 4. Sushi thập cẩm
 5. Salad hoàng đế
- ...

Đối với những mẫu câu cần sử dụng thêm thông tin từ khách hàng đã được truy xuất và được lưu vào các slot thì cần phải thêm {entity} trong đoạn text của tin nhắn như ví dụ bên dưới.

utter_ask_quantity_people:

- text: "Quý khách đặt tiệc cho bao nhiêu người?"

4.6 Cấu hình hành động Rasa

Các hành động của Rasa cần được khai báo trong file domain.yml để sử dụng trong file action.py. Phần này, chỉ định nghĩa các tên action mà bot sử dụng như bên dưới:

actions:

- action_greet
- ask_name_food
- get_name_food
- question_quantity_food
- question_add_food
- display_food
- question_quantity_people
- get_quantity_people
- get_time_using
- ask_name_customer
- get_name_customer
- get_phone_number
- display_information_customer
- question_infor_change
- get_infor_change
- question_infor_change_of_customer
- get_infor_of_customer

-
- update_infor_of_customer
 - question_infor_food_change
 - get_infor_change_food
 - question_food_change
 - change_list_food
 - give_anou_updated_list_food
 - goodbye

Để Rasa có thể thực hiện một hành động bất kì trong luồng câu chuyện được cấu hình trong file stories. Hành động của Rasa cần được cấu hình trong file action.py. Trong file này, mỗi hành động được cấu hình bằng một class ví dụ muốn Rasa thực hiện hành động action: action_greet để phản hồi lại cho người dùng khi được chào hỏi cấu hình như bên dưới:

```
class ActionGreet(Action):  
    def name(self) -> Text:  
        return "action_greet"  
  
    def run(self, dispatcher: CollectingDispatcher, tracker: Tracker,  
           domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:  
  
        user_id = (tracker.current_state())["sender_id"]  
  
        dispatcher.utter_message(response = "utter_greet")  
        dispatcher.utter_message(response = "utter_question_service")  
  
        return []
```

Trong class để thực hiện hành động action_greet, hai hàm được cấu hình là name và run. Hàm def name(self) sẽ trả về tên hành động mà bot muốn trả về cho khách hàng, tên này trùng với tên trong luồng câu chuyện được cấu hình trong phần 4.1. Đối với hàm run được truyền 3 tham số:

- dispatcher: Tham số này là một phương thức được sử dụng để thêm một tin nhắn sử dụng để phản hồi lại người dùng. Như ví dụ trên, tham số dispatcher sẽ gọi phương thức utter_message với tham số truyền vào là một response được lấy từ file domain.yml để phản hồi lại người dùng.

-
- **tracker:** Tham số này là một phương thức được sử dụng để truy cập vào bộ nhớ slot của bot. Phương thức này lấy các thông tin từ các tin nhắn của khách hàng để truy xuất và lưu thông tin vào slot của bot.
 - **domain:** Được sử dụng để load file domain.yml, tham số này sẽ cung cấp quyền truy cập vào dữ liệu được cấu hình trong file domain.yml và định dạng dữ liệu dạng dictionary.

Đối với mỗi người dùng gửi tin nhắn đến bot đều sẽ có một id duy nhất để xác định và tiếp tục câu chuyện nhằm đúng với luồng hội thoại được cấu hình để sử dụng cho khách hàng đó. Khi khách hàng gửi tin nhắn đầu tiên thì bot sẽ tạo cho khách hàng đó một id. Trước khi gửi tin nhắn nào đó cho khách hàng, bot sẽ load đúng id hiện tại của khách hàng để gửi phản hồi lại đúng khách hàng. Id này sẽ được load từ tham số tracker theo hàm `user_id = (tracker.current_state())["sender_id"]`

Trong class cấu hình hành động Rasa, các slot của bot có thể được load để sử dụng bằng phương thức `get_slot("entity")`. Bởi vì trong slot, bot lưu thông tin dưới dạng dictionary nên khi lấy dữ liệu từ slot thì chỉ cần gọi key sẽ trả về giá trị của slot đó. Ví dụ lệnh lấy dữ liệu từ slot tên service từ bot:

```
service = tracker.get_slot("service")
```

Các hành động trả về tin nhắn đến khách hàng có thể sử dụng các câu response đã được tạo sẵn trong file domain.yml hoặc sử dụng trực tiếp câu được tạo ra như trong hai mẫu dưới:

```
dispatcher.utter_message(response="utter_greet")
dispatcher.utter_message(text="Xin chào quý khách")
```

Các action này được hiện thực ngay sau khi bot xác định được ý định của khách hàng, action này thực hiện các chức năng như lấy thực thể lưu vào bộ nhớ slot hoặc danh sách được người phát triển tạo ra. Chức năng của các action của bot được tạo ra để tương tác với khách hàng:

- **action_greet:** Có chức năng phản hồi lại lời chào đến khách hàng.
- **ask_name_food:** Có chức năng hiển thị menu và hỏi tên món ăn mà khách hàng có nhu cầu.
- **get_name_food:** Lấy tên món ăn và lưu vào bộ nhớ tạm slot của bot.
- **question_quantity_food:** Có chức năng đặt câu hỏi về số lượng món ăn mà khách hàng đã chọn.

-
- question_add_food: Có chức năng lấy tên món ăn và số lượng món ăn lưu vào danh sách các món ăn được tạo trong phần action này, đồng thời hỏi về nhu cầu có đặt thêm món hay không.
 - display_food: Có chức năng hiển thị danh sách các món ăn mà khách hàng đã chọn, cùng lời yêu cầu khách hàng kiểm tra và xác nhận lại thông tin.
 - question_infor_food_change: Có chức năng hiển thị câu hỏi về thông tin mà khách hàng có yêu cầu thay đổi về danh sách món ăn.
 - get_infor_change_food: Hành động này bot sẽ lấy và lưu thông tin về sự thay đổi món ăn vào biến để bot xử lý ở hành động tiếp theo.
 - question_food_change: Tùy thuộc vào thông tin về sự chỉnh sửa mà hành động này sẽ trả lời đến khách hàng câu hỏi tương ứng.
 - change_list_food: Sau khi được trích xuất được thông tin thay đổi thì bot sẽ thực hiện sự thay đổi của danh sách các món ăn của khách hàng.
 - give_anou_updated_list_food: Gửi thông báo đến khách hàng thông tin trong danh sách món ăn đã được thay đổi.
 - question_quantity_people: Gửi câu hỏi về số lượng người tham gia đến khách hàng.
 - get_quantity_people: Lấy thông tin về số lượng người tham gia của khách hàng để lưu và danh sách thông tin của khách hàng, đồng thời hỏi khách hàng về thời gian tham gia theo nhu cầu của khách hàng.
 - get_time_using: Lấy thông tin về thời gian để lưu vào danh sách thông tin về thời gian của khách hàng.
 - ask_name_customer: Có chức năng hiển thị thời gian sử dụng nhà hàng và câu hỏi về tên, nhằm lấy thông tin về tên của khách hàng.
 - get_name_customer: Có chức năng lấy tên của khách hàng và lưu vào danh sách thông tin của khách hàng, đồng thời đưa ra câu hỏi về số điện thoại của khách hàng.
 - get_phone_number: Có chức năng lấy số điện thoại của khách hàng và lưu vào danh sách thông tin của khách hàng.
 - display_information_customer: Hiển thị tất cả thông tin của khách hàng và yêu cầu khách hàng kiểm tra danh sách về thông tin đó.
 - question_infor_change: Hỏi về thông tin khách hàng muốn thay đổi ở danh sách thông tin khách hàng khi khách hàng có yêu cầu.
 - get_infor_change: Có chức năng lấy thông tin mà khách hàng cần thay đổi để thực hiện hành động tiếp theo.

-
- `question_infor_change_of_customer`: Dựa vào thông tin mà khách hàng cần thay đổi để phản hồi lại câu hỏi về sự thay đổi đó.
 - `get_infor_of_customer`: Sau khi có được thực thể mới và thông tin thay đổi để thực hiện thay đổi về thành phần trong danh sách thông tin khách hàng.
 - `update_infor_of_customer`: Gửi thông báo đến khách hàng là thông tin khách hàng đã được thay đổi.
 - `goodbye`: Hành động này được thực hiện sau khi khách hàng đã đồng ý tất cả thông tin mà bot đã lấy được từ khách hàng. Chức năng của hành động này là lấy tất cả thông tin để đưa vào cơ sở dữ liệu tại nhà hàng và tạm biệt khách hàng.

4.7 Cấu hình pipeline xử lý tin nhắn từ người dùng

Các thành phần pipeline được sử dụng để xử lý tin nhắn từ người dùng. Tin nhắn của người dùng sẽ được đưa qua hệ thống pipeline này để phân tích và trích xuất thực thể có thể được chứa trong câu nói từ khách hàng. Từ tin nhắn của khách hàng, hệ thống sẽ xác nhận được tin nhắn đó thuộc ý định trong luồng câu chuyện được tạo trong bot, từ đó sẽ thực hiện hành động để xử lý và trả về lời nhắn đến khách hàng một cách chính xác. Các thành phần pipeline được sử dụng:

- `WhitespaceTokenizer`: Từ tin nhắn người dùng, thành phần này sẽ phân tách thành các token, các token tạo ra là các từ riêng biệt khi thành phần này lấy ký tự khoảng trắng làm phân cách. Các token này được lấy và sử dụng cho 3 thành phần tiếp theo.
- `RegexFeaturizer`: Từ các token được tách ra từ tin nhắn khi sử dụng thành phần trên, thành phần này sẽ tạo một danh sách các biểu thức chính quy đã được cấu hình trong dữ liệu đào tạo.
- `LexicalSyntacticFeaturizer`: Thành phần này sẽ sử dụng các token từ thành phần đầu tiên để tạo từ vựng và cú pháp để hỗ trợ trích xuất thực thể.
- `CountVectorsFeaturizer`: Thành phần này cũng sử dụng các token được tách ra từ thành phần đầu tiên, thành phần này sẽ sử dụng các token để xác định các ý định từ tin nhắn người dùng. Thành phần này sẽ cấu hình thêm thông số để sử dụng phân tích token. Thông số analyzer: "char_wb" sẽ sử dụng mô hình n-gram để xác định theo tần suất từ ký tự xuất hiện. Mô hình này cần thêm hai thông số về số lượng ký tự `min_ngram`: 1 và `max_ngram`: 4, thông số này sẽ đặt giới hạn về ký tự lấy để xác định cấu trúc tin nhắn và ý định người dùng.
- `DIETClassifier`: Các dữ liệu được trích xuất để từ 3 thành phần trên sẽ được sử dụng để phân loại các thực thể và ý định của người dùng. Dựa vào các dữ liệu này,

bot có thể xác định được ý định của người dùng dựa vào tin nhắn của người dùng và có thể truy xuất được các thực thể trong tin nhắn người dùng.

4.8 Cấu hình policies quyết định hành động trả về

Sau khi trích xuất thực thể và phân tích ý định của người dùng, Rasa cần cấu hình các chính sách (policies) để quyết định hành động trả về cho người dùng. Các chính sách được cấu hình trong bot này:

- MemoizationPolicy: Chính sách này sẽ kiểm tra câu chuyện được cấu hình trong file stories.yml có khớp với luồng câu chuyện hiện tại hay không. Chính sách này sẽ dựa vào tin nhắn hiện tại và hành động trước đó để xác định ý định thu được từ tin nhắn trên có trùng khớp với luồng tin nhắn hay không để đưa ra hành động tiếp theo.
- TEDPolicy: Chính sách này được sử dụng để dự đoán hành động tiếp theo. Trong chính sách này, thông số epochs là số lần thuật toán xem dữ liệu sau khi đào tạo. Thông số này càng lớn thì thời gian đào tạo càng lâu nhưng có thể tăng được hiệu suất dự đoán của mô hình. Thông số max_history sẽ kiểm soát số lượng lịch sử đoạn đối thoại nhằm quyết định hành động tiếp theo. Thông số constrain_similarities được đặt thành true, điều này giúp khai quát hóa mô hình tốt hơn cho các bộ thử nghiệm thực tế.
- RulePolicy: Chính sách này sẽ xử lý các phần hội thoại để quyết định hành động tiếp theo dựa vào các rules đã được cấu hình trong phần rule. Thông số core_fallback_action_name này sẽ quyết định response sẽ được trả về khi độ tin cậy - confidence thấp hơn mức ngưỡng nào đó. Điều này có nghĩa là bot không biết thực hiện hành động nào tiếp theo sẽ trả về response cho người dùng.

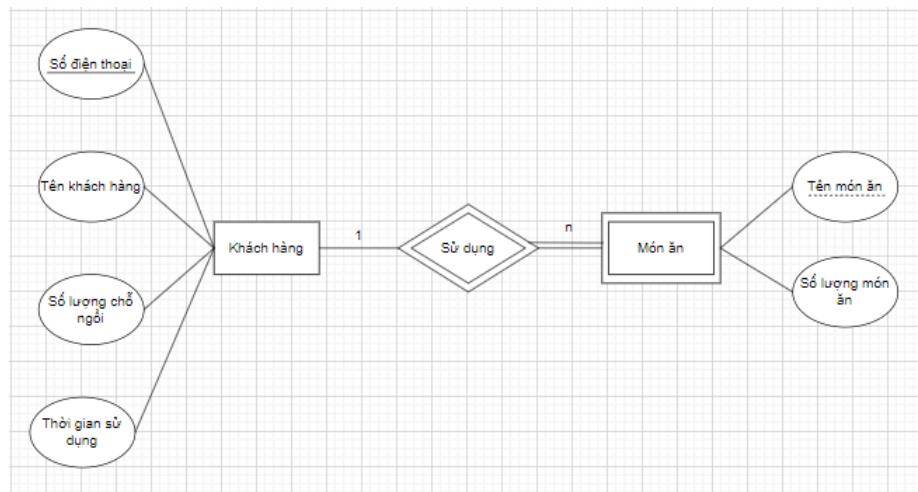
4.9 Xây dựng cơ sở dữ liệu

Cơ sở dữ liệu được sử dụng trong bài toán này để lưu trữ dữ liệu các thông tin của khách hàng. Mô hình dữ liệu được thiết kế như hình 4.1 và 4.2 bao gồm 2 bảng chứa thông tin khách hàng và các món ăn theo nhu cầu của khách hàng. Các thực thể được sử dụng để lưu các thông tin của khách hàng cần lưu bao gồm:

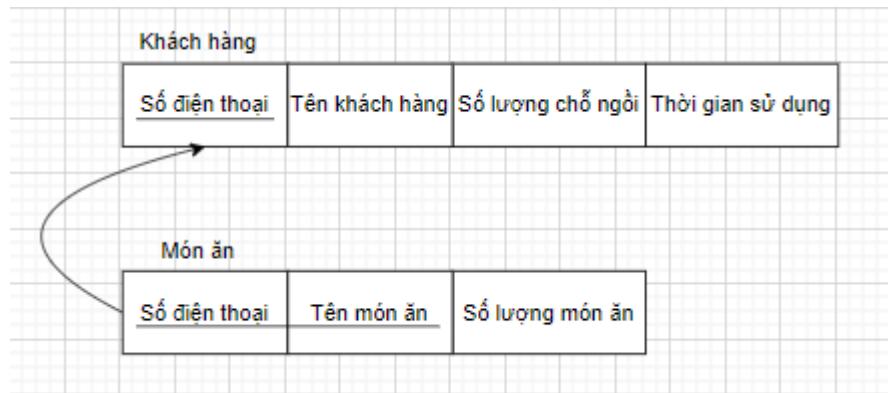
- Số điện thoại: Cột thông tin này được sử dụng để lưu thông tin về số điện thoại của khách hàng. Kiểu thông tin này ở dạng chuỗi. Mỗi người có một số điện thoại được sử dụng để đặt tiệc nên thông tin này được chọn làm khóa chính trong bảng cơ

sở dữ liệu.

- **Tên khách hàng:** Cột thông tin này được sử dụng để lưu thông tin tên khách hàng. Kiểu thông tin này ở dạng chuỗi.
- **Số lượng chỗ ngồi:** Cột thông tin này được sử dụng để lưu thông tin về số lượng chỗ ngồi mà khách hàng có nhu cầu sử dụng tại nhà hàng. Kiểu dữ liệu được sử dụng để lưu thông tin này ở dạng số.
- **Thời gian sử dụng:** Cột thông tin này được sử dụng để lưu thông tin về thời gian mà khách hàng có nhu cầu sử dụng. Kiểu dữ liệu để lưu thông tin này ở dạng Date-time.



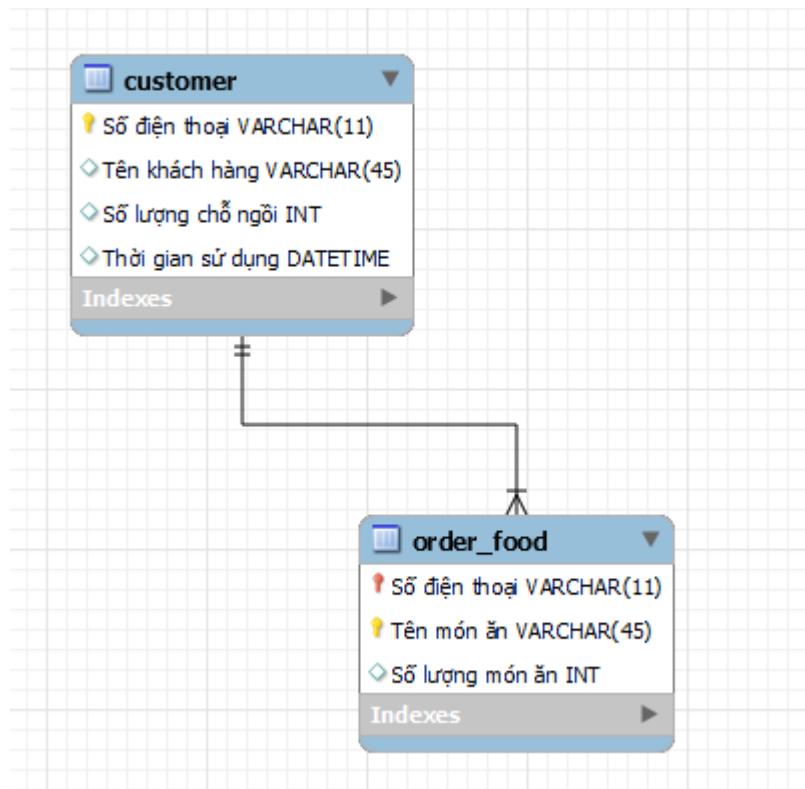
Hình 4.1: Mô hình dữ liệu thực thể - liên kết giữa thông tin khách hàng và món ăn



Hình 4.2: Ràng buộc lược đồ cơ sở dữ liệu quan hệ của thông tin khách hàng

Bảng thứ hai là bảng thông tin các món ăn của khách hàng. Bảng thông tin các món ăn này sẽ được ràng buộc về lược đồ quan hệ với thông tin khách hàng như hình 4.3. Đối với bảng thông tin món ăn này bao gồm các thông tin:

- **Số điện thoại:** Cột thông tin này được sử dụng làm khóa chính được ràng buộc quan hệ với bảng thông tin khách hàng.



Hình 4.3: Sơ đồ lớp dữ liệu thông tin khách hàng và các món ăn

- **Tên món ăn:** Cột thông tin này được sử dụng để lưu thông tin về tên của món ăn. Thông tin này được đặt làm khóa chính của bảng thông tin về món ăn.
- **Số lượng món ăn:** Cột thông tin này được sử dụng để lưu thông tin về số lượng món ăn tương ứng.

Hình 4.4 là ví dụ các thông tin của khách hàng được lưu lại trong cơ sở dữ liệu của nhà hàng. Tên các món ăn và số lượng món ăn tương ứng của khách hàng sẽ được lưu như trong bảng hình 4.4.

	Số điện thoại	Tên khách hàng	Số lượng chỗ ngồi	Thời gian sử dụng
▶	0123456789	Dung	4	2023-05-16 18:00:00
	0987654321	Anh	4	2023-05-16 18:00:00
*	NULL	NULL	NULL	NULL

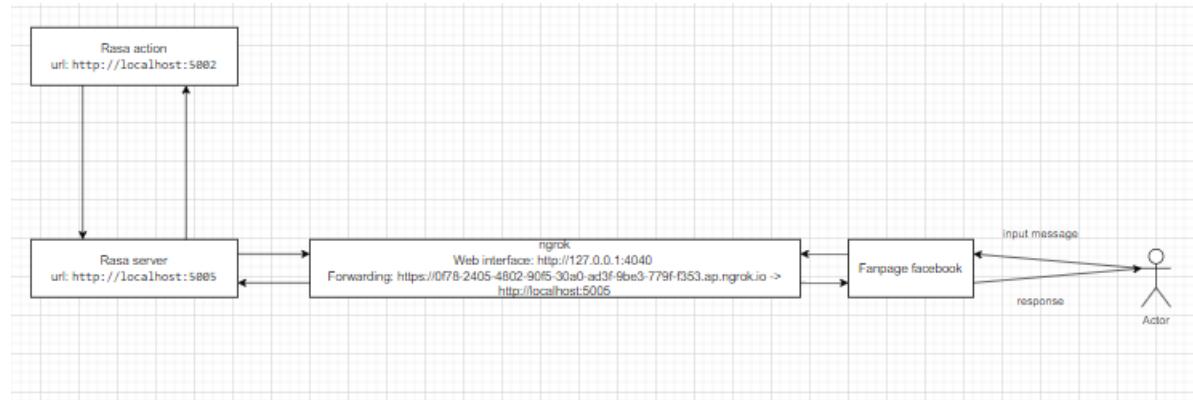
Hình 4.4: Ví dụ về thông tin khách hàng được lưu trên cơ sở dữ liệu

Số điện thoại	Tên món ăn	Số lượng món ăn
0123456789	Gà hấp đông cỗ sốt bào ngư	6
0123456789	Tôm càng nướng wasabi	4
0123456789	Tôm hùm nứa con đút lò kiểu Pháp	5
▶ 0987654321	Gà hấp đông cỗ sốt bào ngư	8
0987654321	Thạch xoài ăn kèm sữa dừa	3
0987654321	Tôm càng nướng wasabi	7
* NULL	NULL	NULL

Hình 4.5: Ví dụ về các món ăn của khách hàng

4.10 Kết nối chatbot Rasa với fanpage facebook

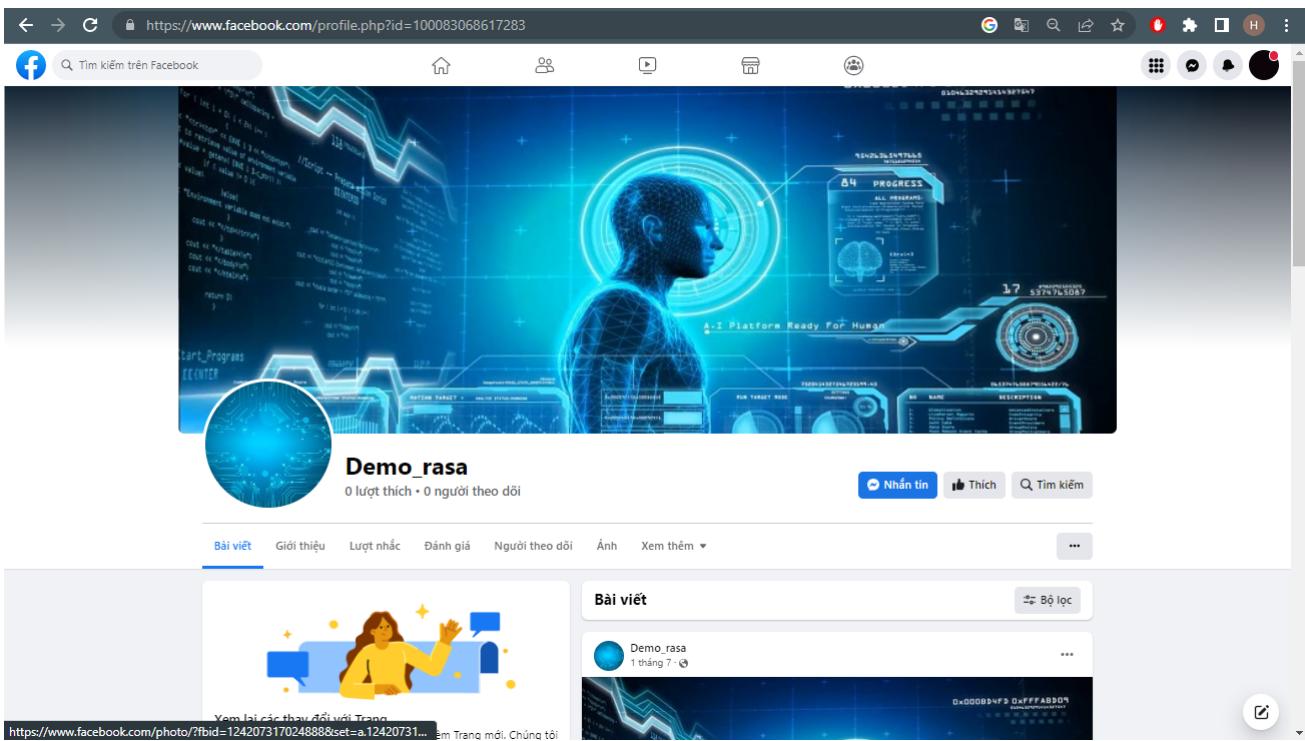
Rasa có thể kết nối với một số ứng dụng nhắn tin như zalo, facebook, telegram, etc. Trong phần này, chatbot Rasa sẽ được kết nối với fanpage của facebook thông qua công cụ tạo đường hầm kết nối ngrok. Đường hầm này sẽ cho phép trao đổi dữ liệu giữa localhost và internet thông qua domain của ngrok. Sau khi kết nối Rasa và ứng dụng khác thì luồng dữ liệu tin nhắn của người dùng sẽ được thể hiện như hình 4.6. Trong kiến trúc này, người dùng sẽ nhắn tin bằng các ứng dụng được kết nối, ứng dụng này được liên kết với Rasa server thông qua ngrok. Dữ liệu tại đây sẽ được xử lý như đã phân tích ở phần trên và dựa vào dữ liệu đó thì Rasa action sẽ trả về người dùng một response về Rasa server. Dữ liệu trả về sẽ thông qua ngrok để đi đến ứng dụng để hiển thị cho người dùng.



Hình 4.6: Kiến trúc luồng dữ liệu đi qua các cổng kết nối

Các bước để kết nối Rasa với fanpage facebook:

- Bước 1: Tạo fanpage facebook trong Meta development như hình 4.7
- Bước 2: Cấu hình file credentials.yml các thông số như verify: "mã xác định", thông số này được cấu hình bằng 1 chuỗi ký tự tùy ý và được sử dụng để nhập chuỗi xác nhận trong fanpage. Thông số secret: "chuỗi khóa" được sử dụng để xác



Hình 4.7: Tạo fanpage facebook

thực. Thông số `page-access-token`: "chuỗi token" được sử dụng để ủy quyền hành động cho Rasa. Khi cung cấp đầy đủ 3 thông số này, Rasa có thể lấy được tin nhắn của người dùng nhấn trong fanpage và Rasa có thể gửi phản hồi đến người dùng thông qua fanpage. Thông số này được lấy bằng cách tạo mã trong phần cài đặt message của fanpage như hình 4.8. Khóa bí mật được lấy như hình 4.9

- Bước 3: Khởi động Rasa action và Rasa server bằng lệnh

```
python -m rasa run actions và
```

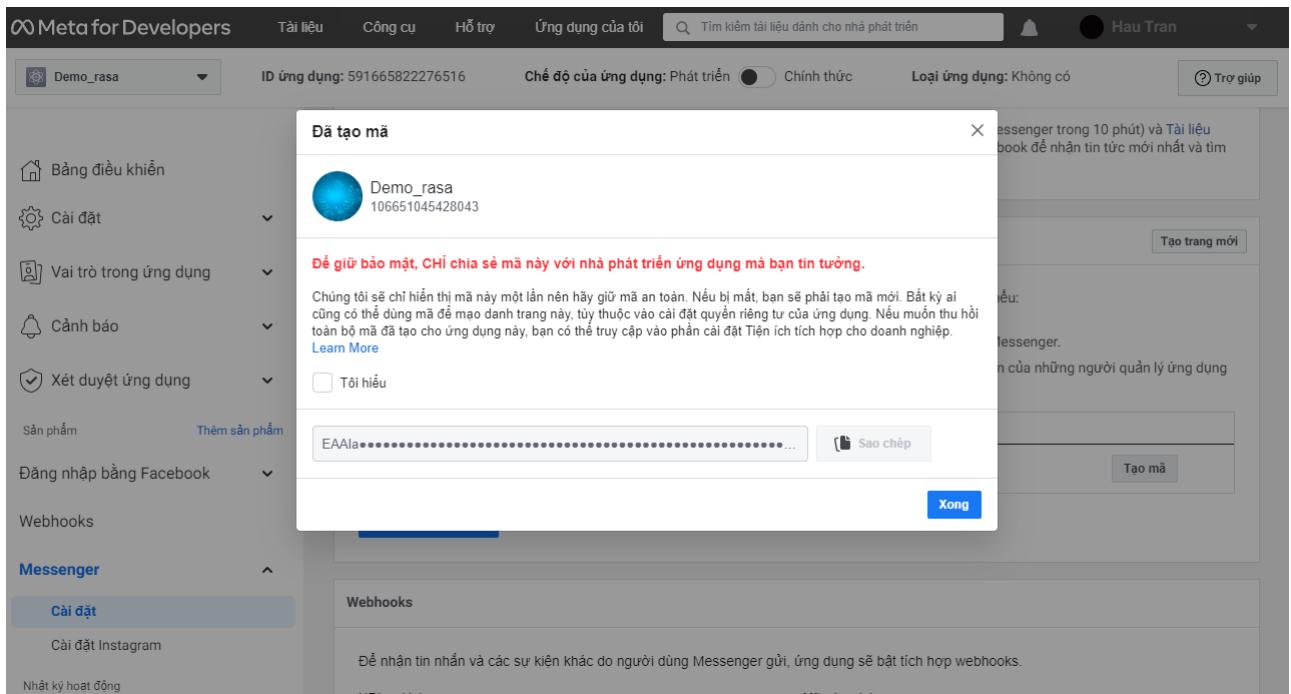
python -m rasa run --enable-api --cors "*" trong terminal. Sau khi khởi động thì Rasa action chạy trên port: 5055 và Rasa server sẽ chạy trên port: 5005.

- Bước 4: Sau khi Rasa server được chạy trên port: 5005, thì cần sử dụng ngrok để tạo đường hầm kết nối port: 5005 bằng lệnh `ngrok http 5005` trong ngrok. Kết quả thể hiện như hình 4.10

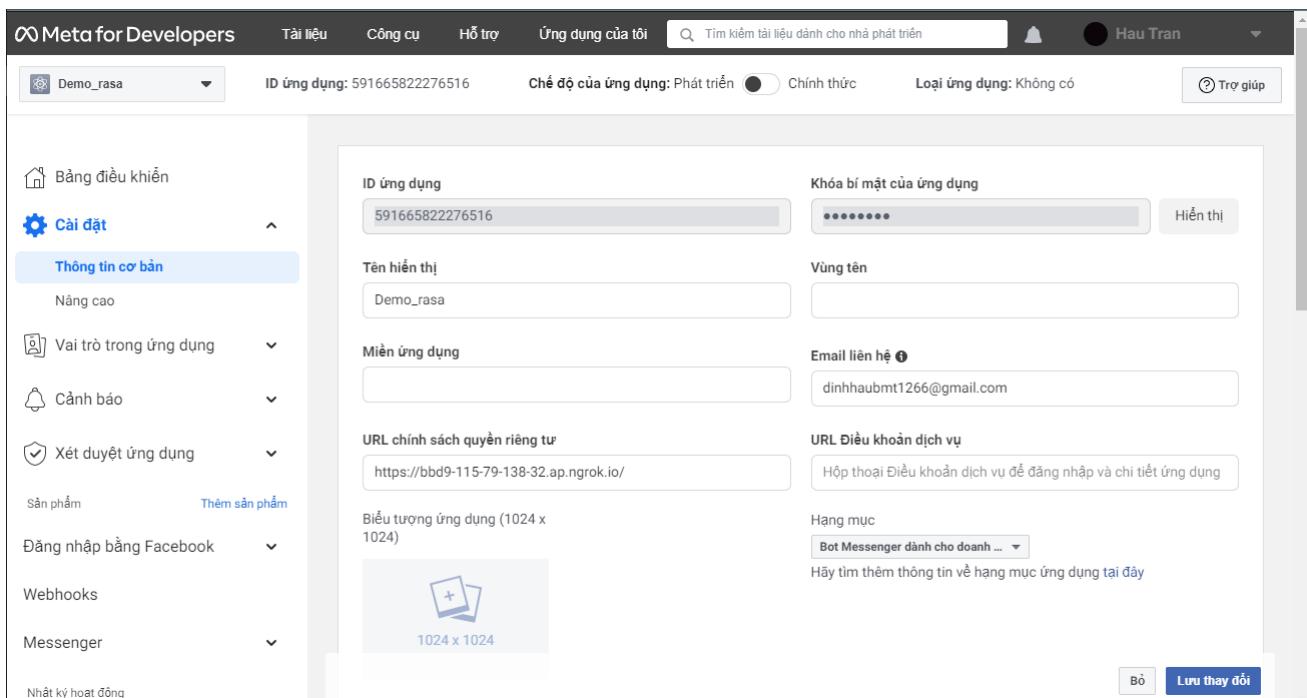
- Bước 5: Khi đường hầm được tạo có forwarding sẽ được sử dụng cho URL gọi lại cùng với chuỗi xác nhận được cấu hình trong Rasa như hình 4.11.

- Bước 6: Cấu hình trường trong fanpage như hình 4.12

- Bước 7: Sau khi cấu hình xong, trong ngrok xuất hiện trạng thái 200 OK là đã kết nối thành công như hình 4.13. Tiến hành nhấn tin để test fanpage.



Hình 4.8: Lấy mã fanpage facebook



Hình 4.9: Lấy khóa bí mật của fanpage facebook

```
C:\Users\Hau Mon\Desktop\ngrok-v3-stable-windows-amd64 (1)\ngrok.exe -ngrok http 5005
ngrok
Try our new native Go library: https://github.com/ngrok/ngrok-go

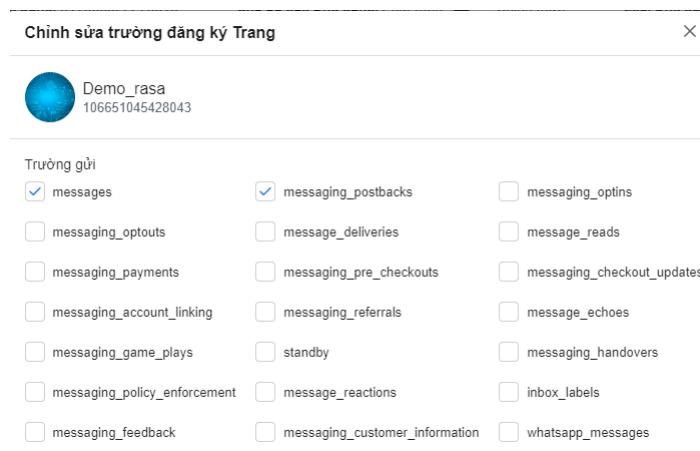
Session Status      online
Account            dinhhaubmt1266@gmail.com (Plan: Free)
Version            3.1.0
Region             Asia Pacific (ap)
Latency            29ms
Web Interface     http://127.0.0.1:4040
Forwarding         https://6222-2405-4802-90f5-30a0-ad3f-9be3-779f-f353.ap.ngrok.io -> http://localhost:5005

Connections        ttl     opn     rti     rt5     p50     p90
                    0       0       0.00   0.00   0.00   0.00
```

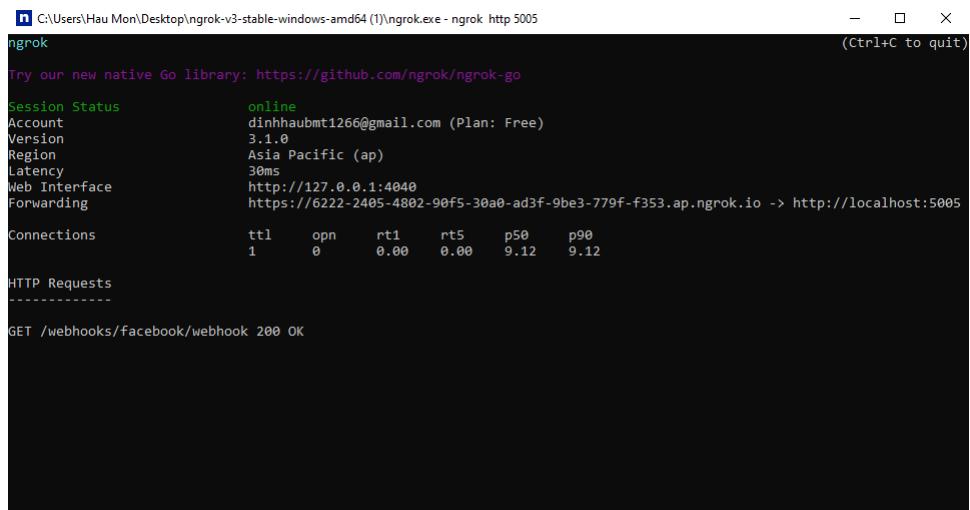
Hình 4.10: Khởi động đường hầm trong ngrok



Hình 4.11: Sử dụng URL gọi lại cho fanpage



Hình 4.12: Cấu hình trường trong fanpage



```
n C:\Users\Hau Mon\Desktop\ngrok-v3-stable-windows-amd64 (1)\ngrok.exe - ngrok http 5005
ngrok
Try our new native Go library: https://github.com/ngrok/ngrok-go
Session Status      online
Account            dinhnaubmt1266@gmail.com (Plan: Free)
Version            3.1.0
Region             Asia Pacific (ap)
Latency            30ms
Web Interface     http://127.0.0.1:4040
Forwarding         https://6222-2405-4802-90f5-30a0-ad3f-9be3-779f-f353.ap.ngrok.io -> http://localhost:5005
Connections        ttl     opn     rt1     rt5     p50     p90
                   1       0     0.00    0.00   9.12   9.12
HTTP Requests
-----
GET /webhooks/facebook/webhook 200 OK
```

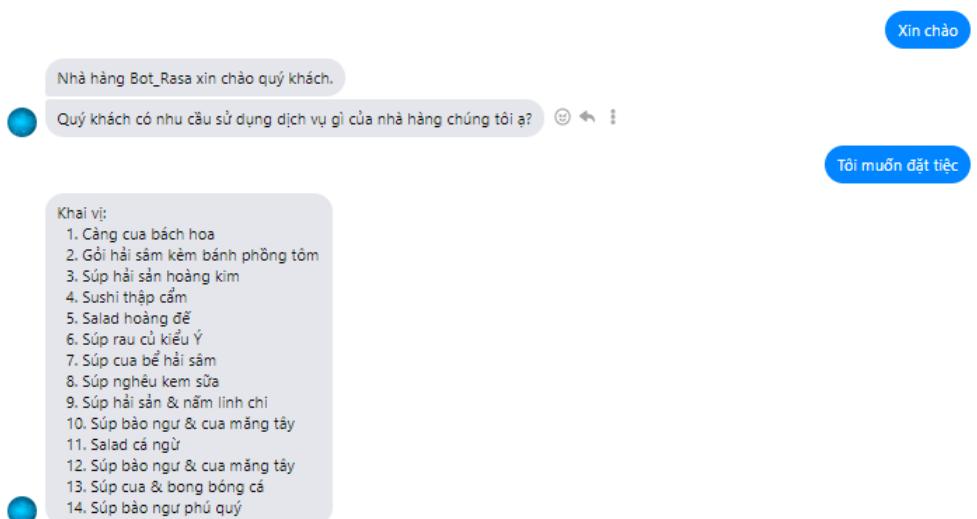
Hình 4.13: Trạng thái kết nối fanpage thành công

Chương 5

Thử nghiệm và đánh giá hệ thống

5.1 Thử nghiệm

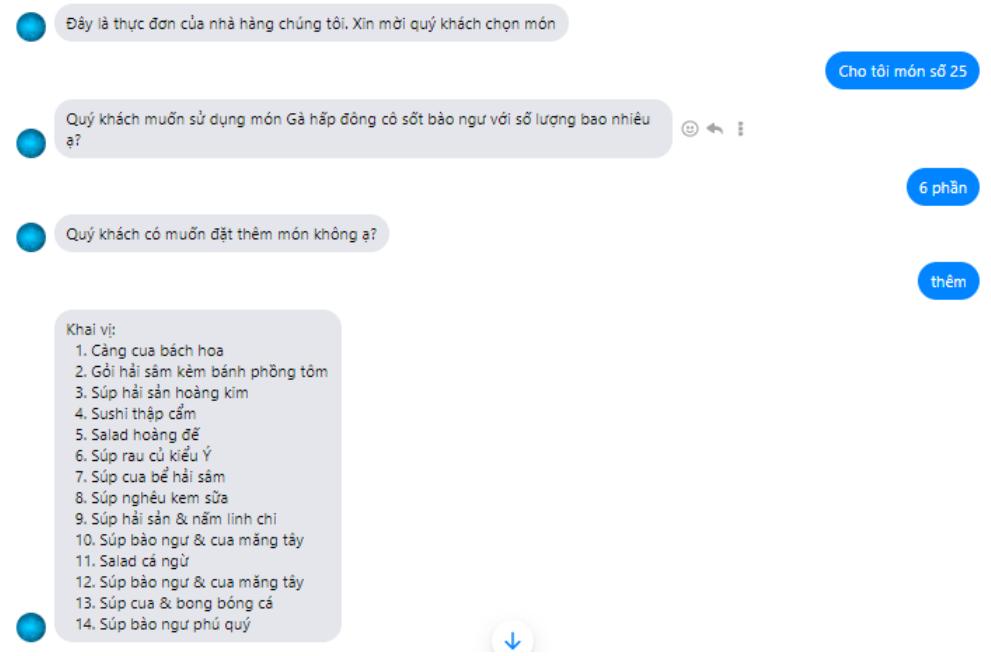
Sau khi Rasa được cấu hình và thử nghiệm Chatbot thông qua fanpage facebook thì Rasa có thể phản hồi lại cho người dùng như hình 5.1, 5.2, 5.3, 5.4, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10 và cơ sở dữ liệu được cập nhập sau khi khách hàng xác nhận đúng thông tin như hình 5.11, 5.12.



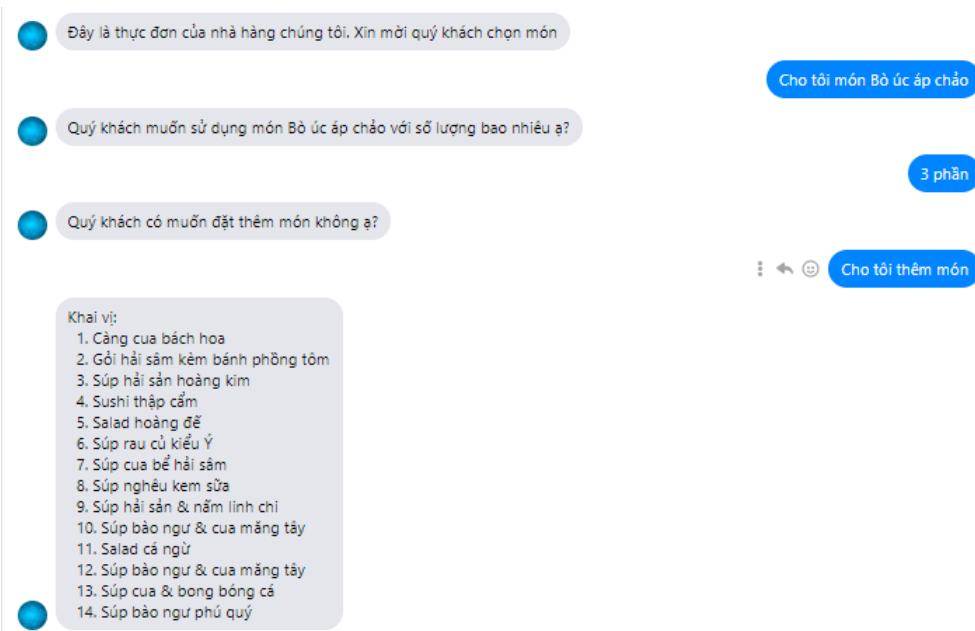
Hình 5.1: Kiểm tra lời chào và hỏi dịch vụ

Khi bot nhận được thông tin là khách hàng có nhu cầu sử dụng dịch vụ thông qua lời chào thì bot sẽ phản hồi lại lời chào cũng như câu hỏi về dịch vụ như hình 5.1.

Đối với các luồng gọi món được xảy ra khi bot xác định được là khách hàng có nhu cầu đặt tiệc tại nhà hàng. Quá trình đặt món ăn bao gồm lấy tên món ăn và số lượng của món ăn đó. Trong quá trình này cũng bao gồm chức năng thêm món ăn, khi khách hàng có nhu cầu đặt thêm món ăn thì bot sẽ quay lại quy trình để hỏi tên món ăn và số lượng món ăn đó như hình 5.2, 5.3, 5.4, 5.5. Tên món ăn có thể được gọi theo nhiều tên khác

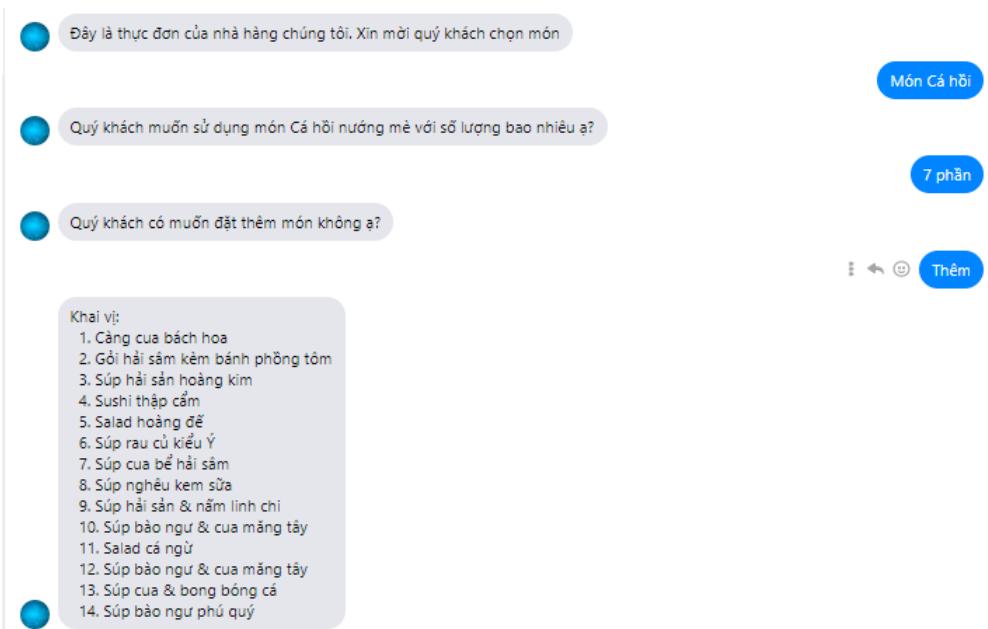


Hình 5.2: Kiểm tra đặt món ăn theo chỉ số

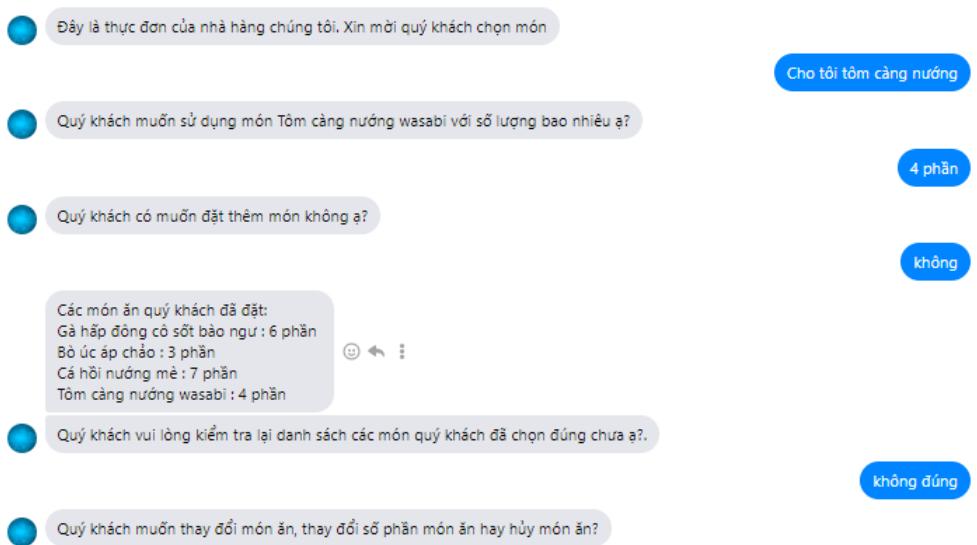


Hình 5.3: Kiểm tra đặt món ăn theo tên

nhau như chỉ số món ăn như trên menu, tên món ăn hoặc có thể là tên không đầy đủ. Nếu như gọi theo tên không đầy đủ thì bắc buộc không được để tên đó là tên không đầy đủ của hai hoặc nhiều món ăn. Ví dụ như không thể gọi tên món ăn là súp bào ngư được bởi vì trong menu có hai món là "súp bào ngư & cua măng tây" và "súp bào ngư phú quý" nên sẽ bị trùng tên và không thể gọi được.



Hình 5.4: Kiểm tra đặt món ăn theo tên không đầy đủ



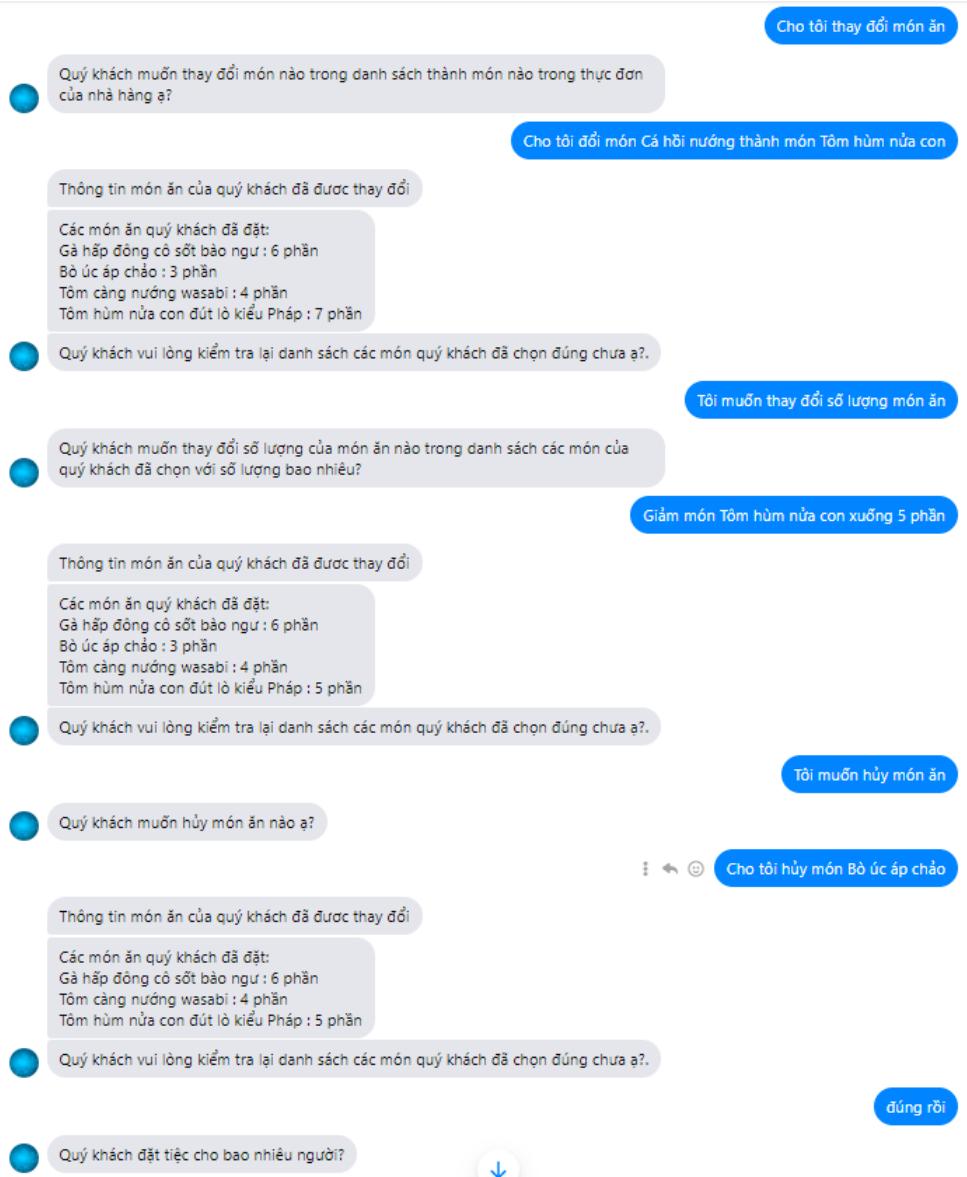
Hình 5.5: Kiểm tra đặt món ăn và xác nhận thay đổi danh sách món ăn

Tiếp theo là thử nghiệm thay đổi các thành phần trong danh sách món ăn như hình 5.6. Việc thay đổi các thành phần món ăn bao gồm tên món ăn, số lượng món ăn và hủy món ăn tùy theo lựa chọn của khách hàng.

Đối với thông tin khách hàng bao gồm các thông tin cơ bản như tên, số điện thoại, số lượng người tham gia và thời gian sử dụng như hình 5.7.

Khi khách hàng có nhu cầu thay đổi các thông tin liên quan đến khách hàng thì tùy thuộc vào thông tin nào cần thay đổi thì bot sẽ hỏi câu hỏi nhằm về thông tin cần cập nhập đó như hình 5.8, 5.9

Sau khi danh sách được cập nhật theo đúng thông tin yêu cầu của khách hàng thì

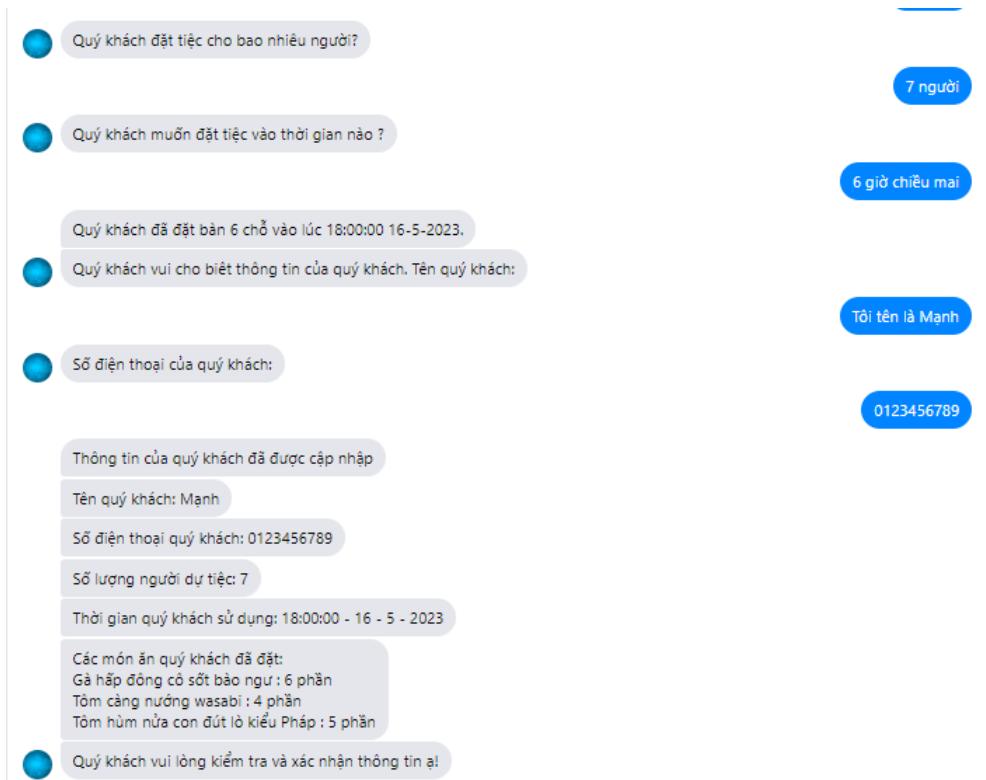


Hình 5.6: Kiểm tra thay đổi các thành phần trong danh sách món ăn

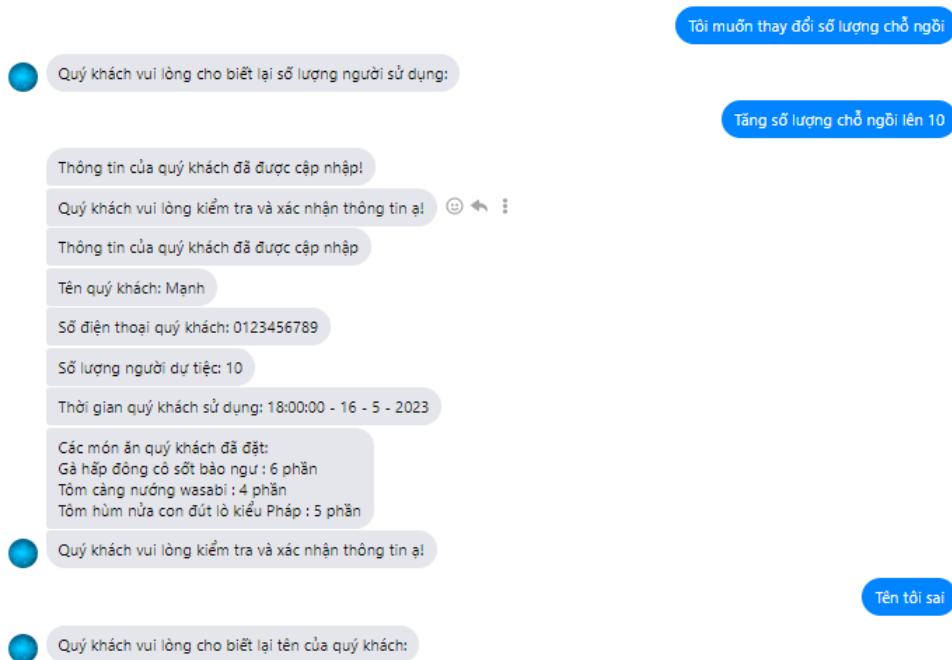
bot sẽ hỏi nhằm để xác định khách hàng đã đồng ý với tất cả danh sách về thông tin hay chưa. Nếu khách hàng có nhu cầu thay đổi thì bot sẽ quay lại luồng bên trên. Khi khách hàng đã đồng ý thì bot sẽ xác nhận và cảm ơn khách hàng. Trong quá trình này, khi nhận được thông tin xác nhận thì bot sẽ lưu tất cả thông tin thu thập được từ khách hàng để lưu vào cơ sở dữ liệu như hình 5.10, 5.11 và 5.12.

Rasa cũng có thể được kết nối với các kênh khác để chat được với người dùng. Nhờ sự hỗ trợ của ngrok mà Rasa có thể hoạt động ổn định với fanpage của facebook.

Luồng dữ liệu cơ bản đã đi đúng với hướng đã được cấu hình trong file stories.yml. Các quy tắc được cấu hình trong file rules.yml đã được sử dụng chính xác để định đúng hành động sau khi nhận tin nhắn từ người dùng. Các tin nhắn từ người dùng Rasa có thể



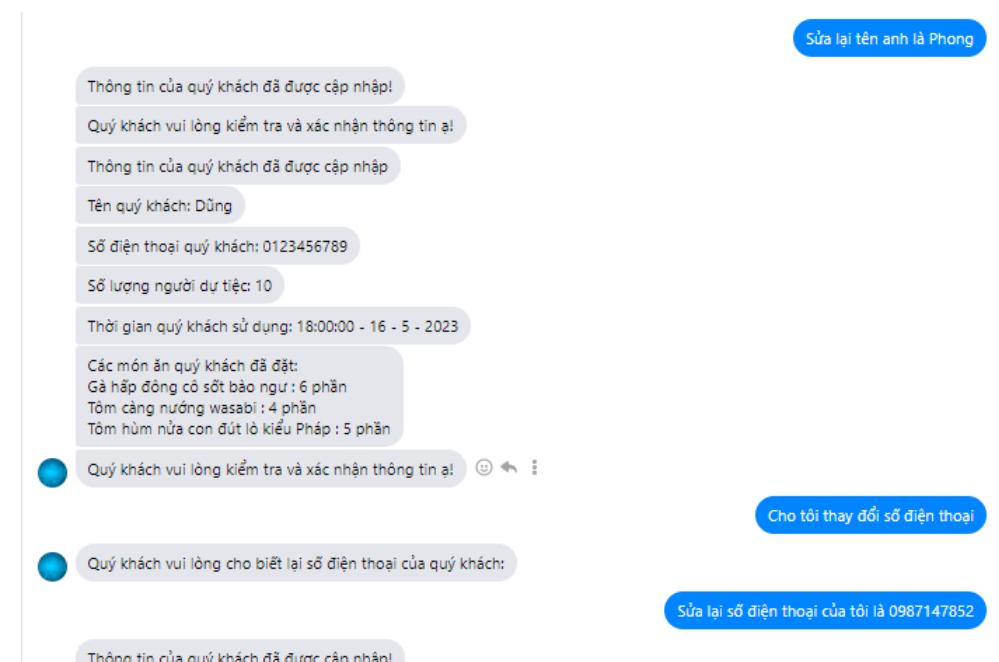
Hình 5.7: Kiểm tra lấy thông tin của khách hàng



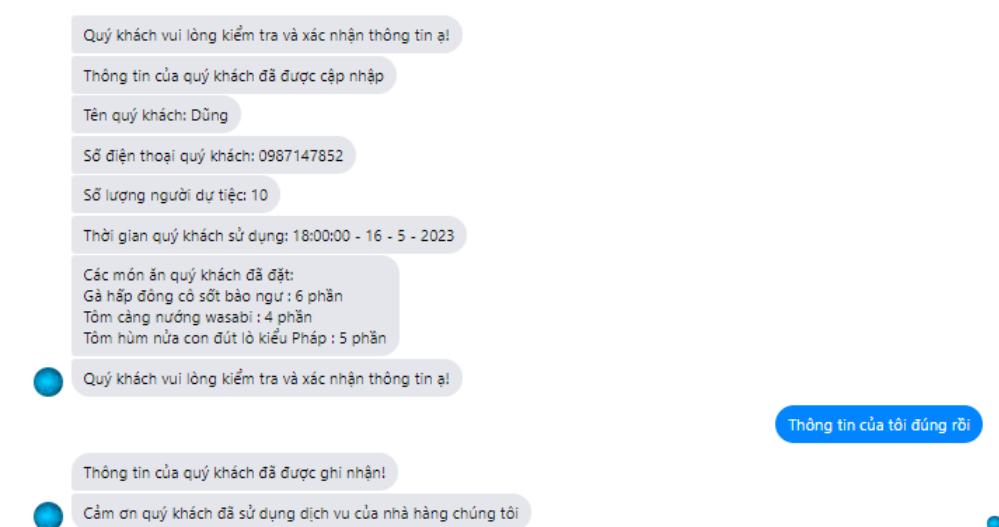
Hình 5.8: Kiểm tra thay đổi thông tin về số lượng người sử dụng

trích xuất được các thực thể để lưu vào bộ nhớ của bot.

Dựa vào tin nhắn người dùng đơn giản mà bot có thể xác định được ý định từ tin nhắn đó để thực hiện đúng được hành động tiếp theo. Các action đã thực hiện đúng chức năng được cấu hình là lấy tên món ăn từ chỉ số món ăn mà khách hàng đã chọn. Phần thời



Hình 5.9: Kiểm tra thay đổi thông tin về tên và số điện thoại



Hình 5.10: Kiểm tra xác nhận thông tin chính xác

	Phone_number	Name	Quantity_people	Time_using	Food_order
	0112345678	Anh	6	2023-04-12 18:00:00	Trái vải hành nhân lạnh : 8 phần Tôm còng...
	01126554879	Anh	6	2023-04-10 13:30:00	Mon 8: 2 phần
	0213456789	Bảo	6	2023-04-11 18:00:00	Thạch xoài ăn kèm sữa dừa : 3 phần Trái ...
▶	0987147852	Dũng	10	2023-05-16 18:00:00	Gà hấp đồng cốt sốt béo ngư : 6 phần Tôm...
*	NULL	NULL	NULL	NULL	NULL

using_service1 x Apply

Hình 5.11: Kiểm tra bản dữ liệu cập nhật

The screenshot shows a Rasa NLU test interface with the following data:

- Phone_number:** 0987147852
- Name:** Dũng
- Quantity_people:** 10
- Time_using:** 2023-05-16 18:00:00
- Food_order:**
 - Gà hấp đồng cỗ sốt béo ngậy : 6 phần
 - Tôm còng nướng wasabi : 4 phần
 - Tôm hùm nứa con đút lò kiểu Pháp : 5 phần

At the bottom left is the text "using_service1" and at the bottom right is a button labeled "Apply".

Hình 5.12: Kiểm tra các thông tin khách hàng trong bản dữ liệu

gian mà người dùng muôn sử dụng dịch vụ tại nhà hàng thì đã xử lý được, bot đã truy xuất được thông tin thời gian từ tin nhắn của khách hàng để đưa ra được dạng chuẩn giờ, ngày, tháng, năm giúp quản lý có thể kiểm soát được thời gian của các khách hàng.

Có thể sử dụng thêm lệnh python -m rasa interactive để quan sát các entity được truy xuất từ tin nhắn người dùng và luồng câu chuyện đi trong một cuộc trò chuyện.

Bộ nhớ slot của bot đã ánh xạ đúng vào thực thể trích xuất được trong tin nhắn của người dùng. Các phản hồi đã gửi đến chính xác cho người dùng trong mỗi hành động đã cấu hình. Các thực thể đã sử dụng đúng chỗ khi được sử dụng để phản hồi lại khách hàng.

5.2 Đánh giá hệ thống

5.2.1 Phương pháp đánh giá

Đây là Chatbot theo ý định nên việc đánh giá nhằm xác định độ hiệu quả trong quá trình NLU phân tích, xác định ý định và trích xuất đúng ý định của người dùng. Sau khi đào tạo mô hình bằng phương pháp học máy để phân loại các ý định của người dùng thì việc đánh giá sẽ xác định mô hình này có đáng tin cậy để chỉ ra đúng ý định từ người dùng hay không. Việc đánh giá mô hình sẽ dựa vào 4 thông số như hình 5.13 với 2 lớp dự đoán và 2 lớp thực tế dựa vào dữ liệu thực tế gồm positive (đúng) và negative (sai). Từ 2 thông số của dự đoán và thực tế sẽ có 4 thông số: True Positive (TP), False Positive (FP), False Negative (FN) và True Negative (TN). Phần phân tích ý định và trích xuất thực thể được thực hiện trong phần NLU nên việc đánh giá NLU dựa vào 3 thông số:

- Precision: Thông số này nhằm đánh giá về tỷ lệ về cái đúng được xác định trên tập dữ liệu kiểm soát có bao nhiêu dữ liệu được mô hình dự đoán đúng. Thông số này cao thì tương đương với việc độ chính xác của các điểm đúng xác định được là cao.

Công thức để tính thông số này:

$$Presicion = \frac{TP}{TP+FP}$$

- Recall: Thông số này xác định tỉ lệ có bao nhiêu cái đúng trong số những điểm thực sự là đúng. Thông số này lớn khi khả năng bỏ sót các điểm đúng thực sự thấp. Công thức để tính thông số này:

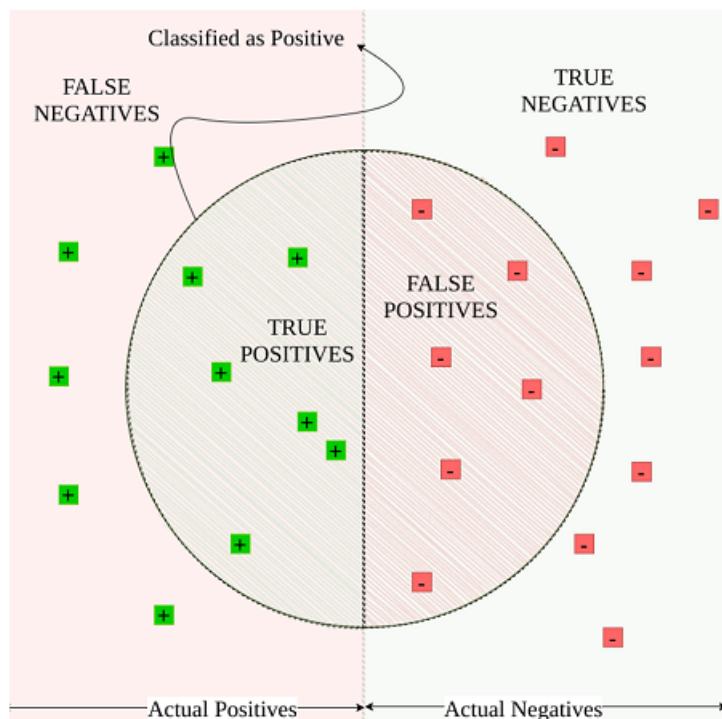
$$Recall = \frac{TP}{TP+FN}$$

- F1-Score: Thông số này được tính theo hàm điều hòa (harmonic mean) của hai thông số Precision và Recall. Khi một trong hai hoặc cả hai thông số thấp thì giá trị F1-Score này sẽ thấp. Nên thông số F1-Score này sẽ thể hiện một cách khách quan về mô hình học máy. Công thức tính F1-Score:

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

- Accuracy: Thông số này được định nghĩa là tỷ lệ phần trăm dự đoán đúng trên tổng dữ liệu thử nghiệm. Thông số này được tính theo công thức:

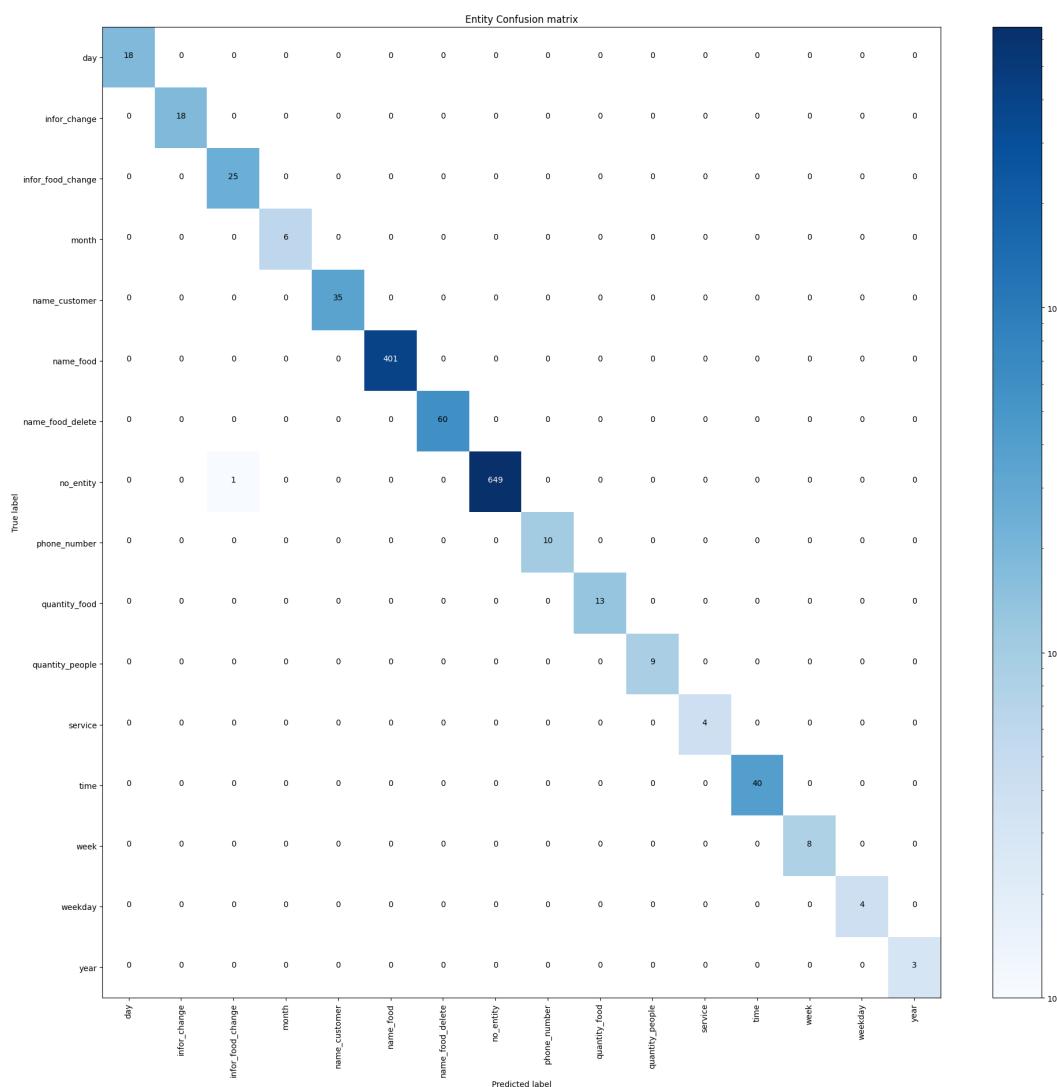
$$Accuracy = \frac{Correct\ predictions}{All\ predictions}$$



Hình 5.13: Các chỉ số để đánh giá mô hình [18]

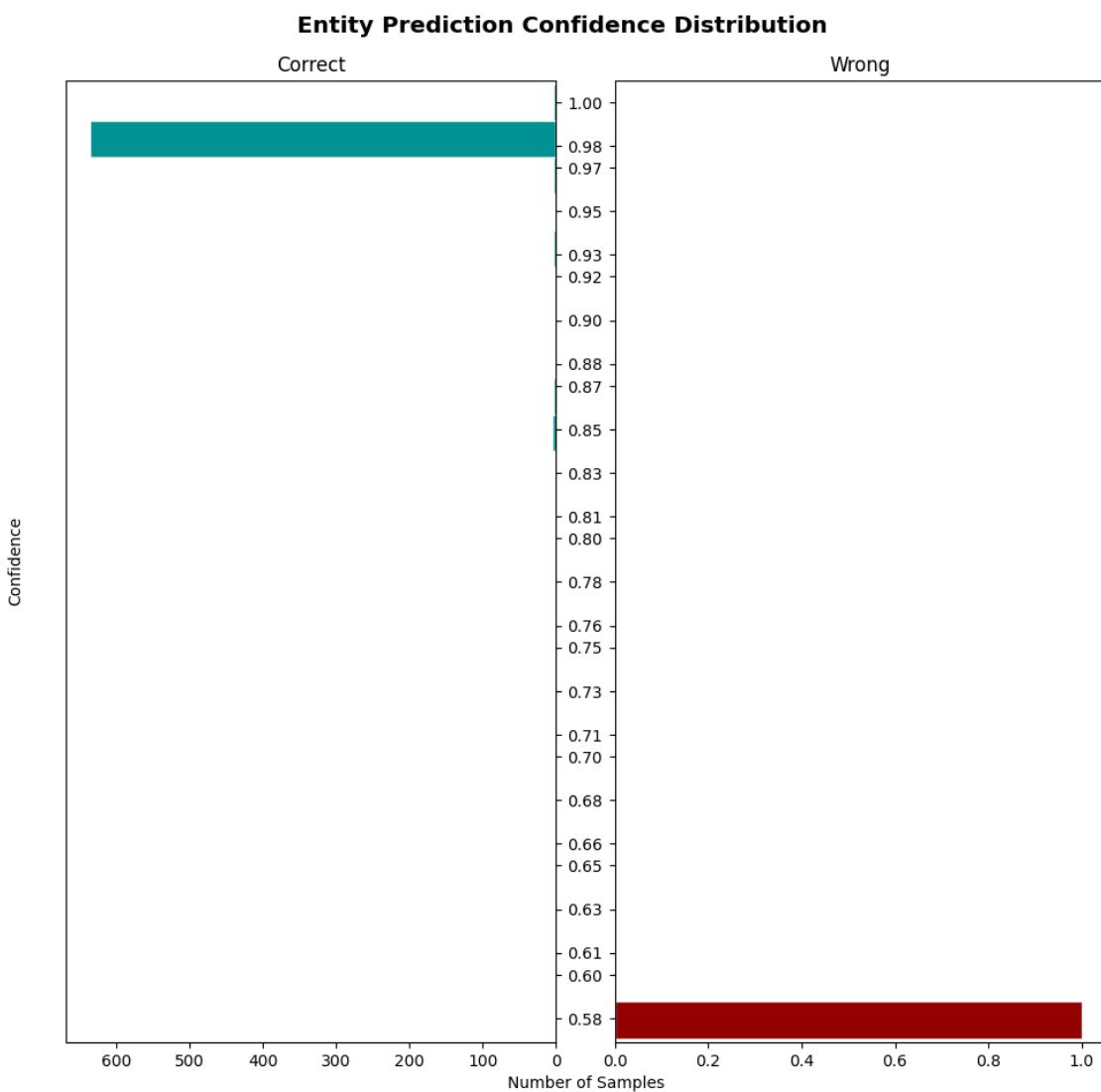
5.2.2 Đánh giá dự đoán thực thể

Việc đánh giá về dự đoán thực thể của NLU sẽ giúp đánh giá được khả năng trích xuất thực thể của mô hình đào tạo. Trong phần dữ liệu nlu.yml sẽ tách ra thành 2 file sử dụng để đào tạo và đánh giá model sau khi train với tỷ lệ 80% sử dụng để train và 20% sử dụng để đánh giá. Việc đánh giá dự đoán thực thể để đánh giá độ hiệu quả của pipeline được cấu hình có đánh tin cậy để trích xuất thực thể từ tin nhắn của người dùng hay không. Việc trích xuất thực thể quan trọng trong việc lấy thông tin của khách hàng nên việc đánh giá này là cần thiết để đánh giá độ hiệu quả của Chatbot.



Hình 5.14: Ma trận nhầm lẫn dự đoán thực thể

Ma trận nhầm lẫn và biểu đồ độ tin cậy của các trích xuất thực thể được thể hiện trong hình 5.14, 5.15. Biểu đồ thể khả năng lấy thông tin và gán nhãn thông tin và về khả năng trích xuất và gán nhãn cho thông tin đó. Khả năng trích xuất thông tin còn phụ thuộc vào ý định được dự đoán. Bởi vì trong dữ liệu đã định cấu hình về thực thể có thể được



Hình 5.15: Biểu đồ độ tin cậy các dự đoán thực thể

Entity	Độ tin cậy			
	Precision	Recall	F1-score	Support
day	1.0	1.0	1.0	1
name_customer	1.0	1.0	1.0	32
infor_food_change	1.0	1.0	1.0	25
quantity_food	0.9090	1.0	0.9524	10
quantity_people	1.0	1.0	1.0	6
week	1.0	1.0	1.0	12
service	1.0	1.0	1.0	4
year	1.0	1.0	1.0	6
month	1.0	1.0	1.0	8
phone_number	1.0	1.0	1.0	5
name_food_delete	1.0	1.0	1.0	64
infor_change	1.0	1.0	1.0	18
time	1.0	1.0	1.0	23
name_food	1.0	0.9973	0.9986	368
Weighted Average	0.9985	0.9983	0.9984	593
Accuracy	0.9983			

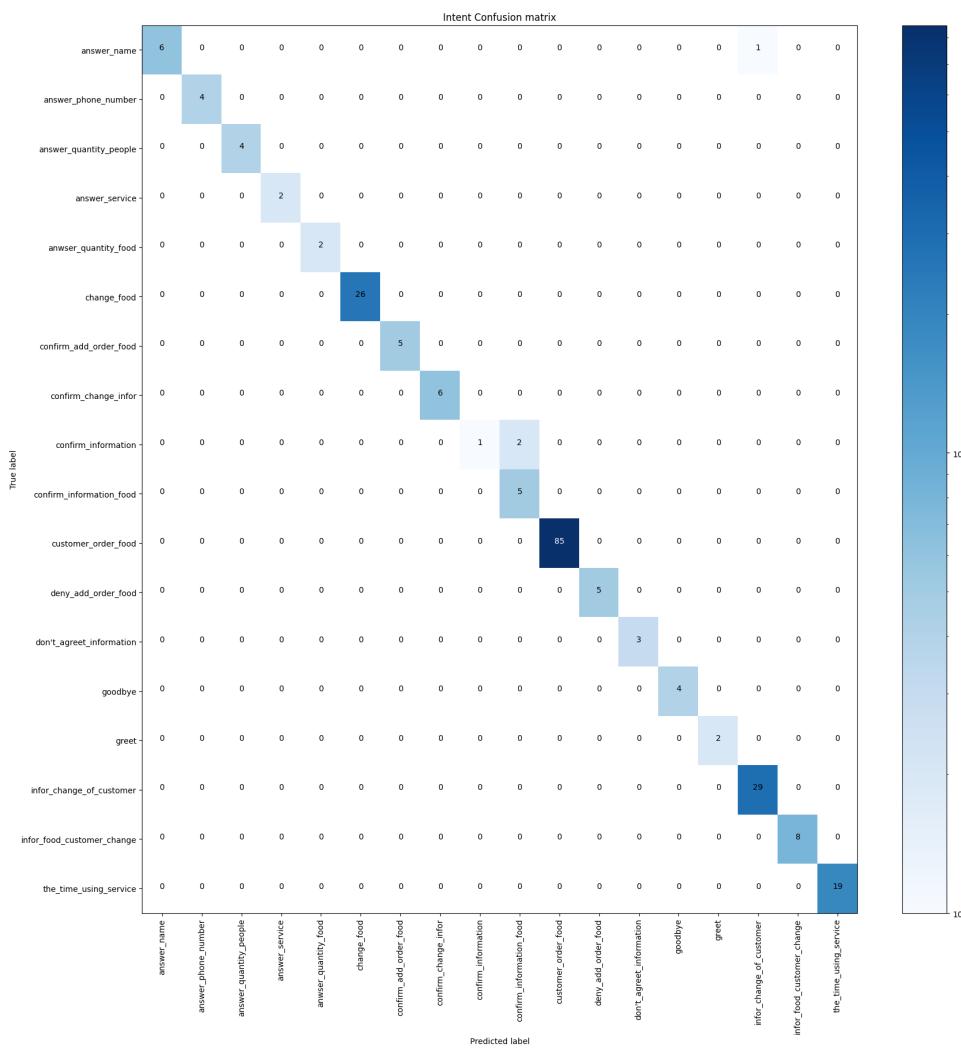
Bảng 5.1: Bảng ước lượng độ chính xác dự đoán từng thực thể

xác định tại ý định nào nên việc xác định đúng ý định làm cho bot có tăng khả năng gán nhãn cho thực thể. Dựa vào đồ thị hình 5.15 và bảng 5.1 trên ta thấy khả năng nhầm lẫn khi trích xuất thông tin và gán nhãn là rất thấp. Vì vậy việc tạo lập dữ liệu là bước cực kỳ quan trọng để tăng khả năng chính xác của mô hình dự đoán. Khi thêm các giới hạn về tập luật của dữ liệu để gán nhãn thông tin thì sẽ tăng được độ tin cậy về dự đoán nhãn và trích xuất thông tin đó từ những câu nói của người dùng. Nếu như thông tin thực thể khai báo giống nhau về kiểu dữ liệu và một số thông tin dữ liệu thì có thể bot sẽ xác định nhầm lẫn dữ liệu như trong ma trận nhầm lẫn mà bot đã gán nhãn sai một thực thể về tên món ăn và số lượng người tham gia. Việc này được giải thích trong khi tạo mẫu câu dữ liệu thì tên món ăn có thể được chọn theo chỉ số món ăn trong menu đó và số lượng người tham gia đều ở kiểu số, cộng thêm việc dữ liệu đó có giới hạn là 1 đến hai chữ số nên bot đã nhầm trong việc gán nhãn dữ liệu đó. Nên việc thêm tập luật về kiểu dữ liệu thực thể sẽ giúp làm giảm khả năng dán nhãn sai khi trích xuất thông tin của người dùng. Như việc giới hạn về số ký tự kiểu số trong hai tên thực thể là số điện thoại, số người tham gia. Do số ký tự là khác nhau nên bot đã không nhầm lẫn trong các câu trả lời về số điện thoại và số người tham gia. Tuy nhiên, trong một số trường hợp kiểu dữ liệu, luật dữ liệu có thể là giống nhau như ví dụ ở trên về tên món ăn và số lượng người tham gia. Nên để giải quyết trường hợp này thì việc xác định đúng ý định của người dùng thì việc

nhầm lẫn sẽ được giảm.

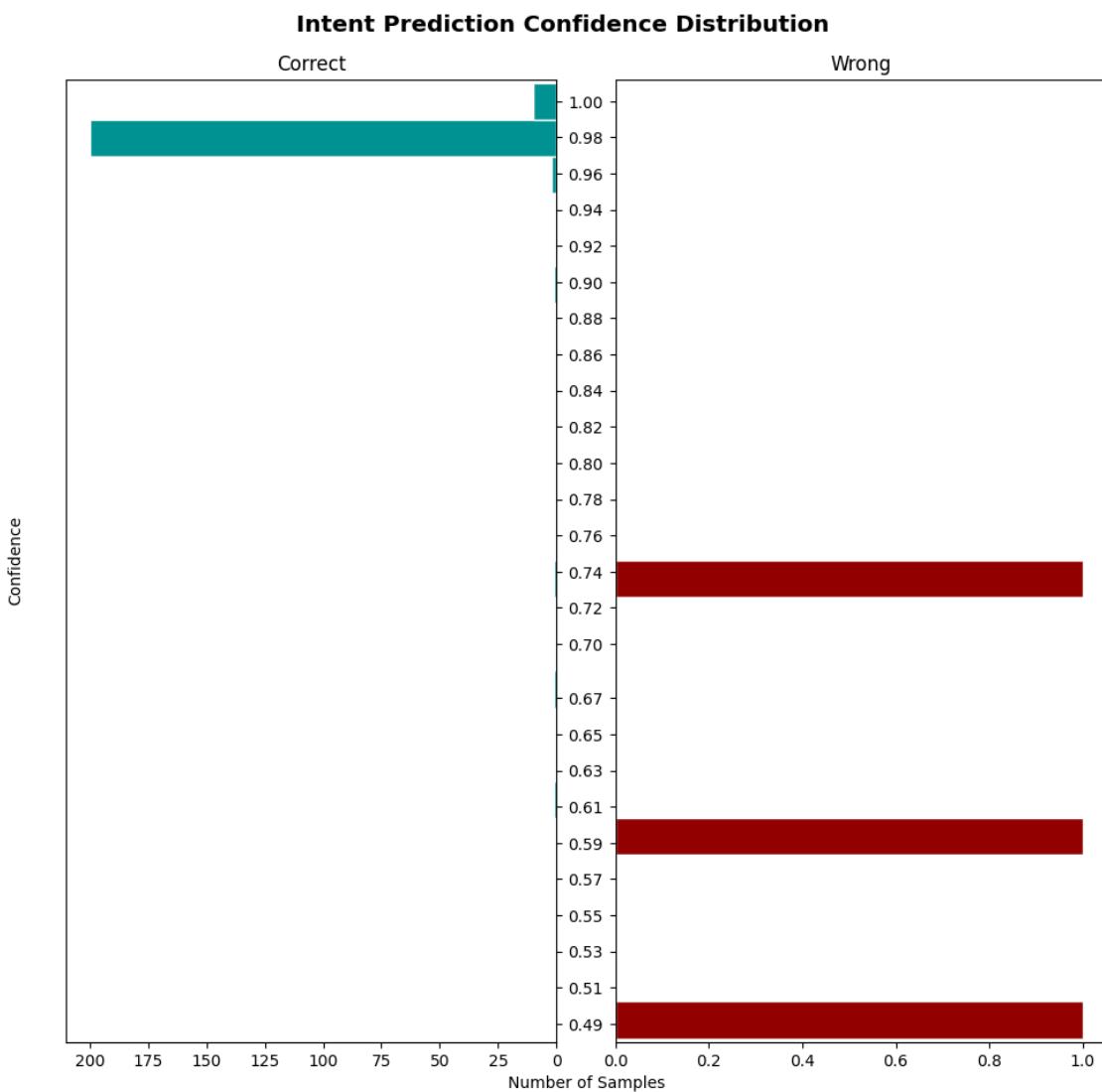
5.2.3 Đánh giá dự đoán ý định

Việc đánh giá dự đoán ý định của NLU sẽ xác định được độ tin cậy về việc xác định ý định từ tin nhắn người dùng. Trong phần dữ liệu nlu.yml sẽ tách ra thành 2 file sử dụng để đào tạo và đánh giá model sau khi train với tỷ lệ 80% sử dụng để train và 20% sử dụng để đánh giá. Sau khi thử nghiệm với 20% dữ liệu đánh giá thì bot đạt độ chính xác khoảng 99.52%.



Hình 5.16: Ma trận nhầm lẫn dự đoán ý định

Với ma trận nhầm lẫn, biểu đồ ước lượng về dự đoán ý định hình 5.16 và hình 5.17 thì ta thấy, các câu nói từ người dùng được NLU phân tích ít bị nhầm lẫn với nhau. Như trên ma trận nhầm lẫn về dự đoán ý định, chỉ có 1 dữ liệu bị dự đoán nhầm lẫn về nhãn ý định về xác nhận thông tin món ăn và xác nhận thông tin về khách hàng. Nhưng khi phân tích về câu nói nhầm ý định hai nhãn này thì câu văn có thể là như nhau, điều này làm cho



Hình 5.17: Biểu đồ độ tin cậy các dự đoán ý định

phần phân tích NLU có đã xảy ra nhầm lẫn khi 2 ý định này được xác định trong cùng một câu văn. Nhưng trong luồng dữ liệu bot, luồng dữ liệu sẽ đi theo hướng đã được tạo lập sẵn trong câu chuyện của bot. Nên bot vẫn sẽ xác định được đúng ý định của khách hàng khi dựa vào câu chuyện được cấu hình. Vì vậy dữ liệu này vẫn có thể chấp nhận được trong trường hợp này. Không chỉ trong ý định được gán nhãn này mà còn trong một số nhãn khác như trả lời về số điện thoại, trả lời về tên khách hàng, trả lời về thời gian sử dụng và trả lời về số lượng người có thể bị nhầm lẫn về ý định thông tin để chỉnh sửa cho khách hàng khi phân tích về câu nói thực thể mà khách hàng có thể nói khi muốn thực hiện một ý định nào đó. Các xung đột này đều có thể được giải quyết trong phần tạo câu chuyện về luồng hội thoại. Điều này làm tăng tính hiệu quả về việc dự đoán ý định từ người dùng.

Biểu đồ về độ tin cậy cho tất cả các dự đoán như hình 5.17, bảng 5.2 với dự đoán

Intent	Độ tin cậy			
	Precision	Recall	F1-score	Support
anwser_quantity_food	1.0	1.0	1.0	3
deny_add_order_food	1.0	1.0	1.0	4
customer_order_food	1.0	1.0	1.0	85
the_time_using_service	1.0	1.0	1.0	15
answer_service	1.0	1.0	1.0	2
answer_phone_number	1.0	1.0	1.0	2
confirm_information_food	0.7142	1.0	0.8888	5
infor_change_of_customer	0.9667	1.0	0.9831	29
confirm_add_order_food	1.0	1.0	1.0	5
answer_name	1.0	0.8571	0.9231	7
confirm_change_infor	1.0	1.0	1.0	6
answer_quantity_people	1.0	1.0	1.0	5
infor_food_customer_change	1.0	1.0	1.0	8
change_food	1.0	1.0	1.0	26
goodbye	1.0	1.0	1.0	4
greet	1.0	1.0	1.0	4
confirm_information	1.0	0.3333	0.5	3
don't_agreet_information	1.0	1.0	1.0	2
Weighted Average	0.9890	0.9863	0.9846	219
Accuracy		0.9952		

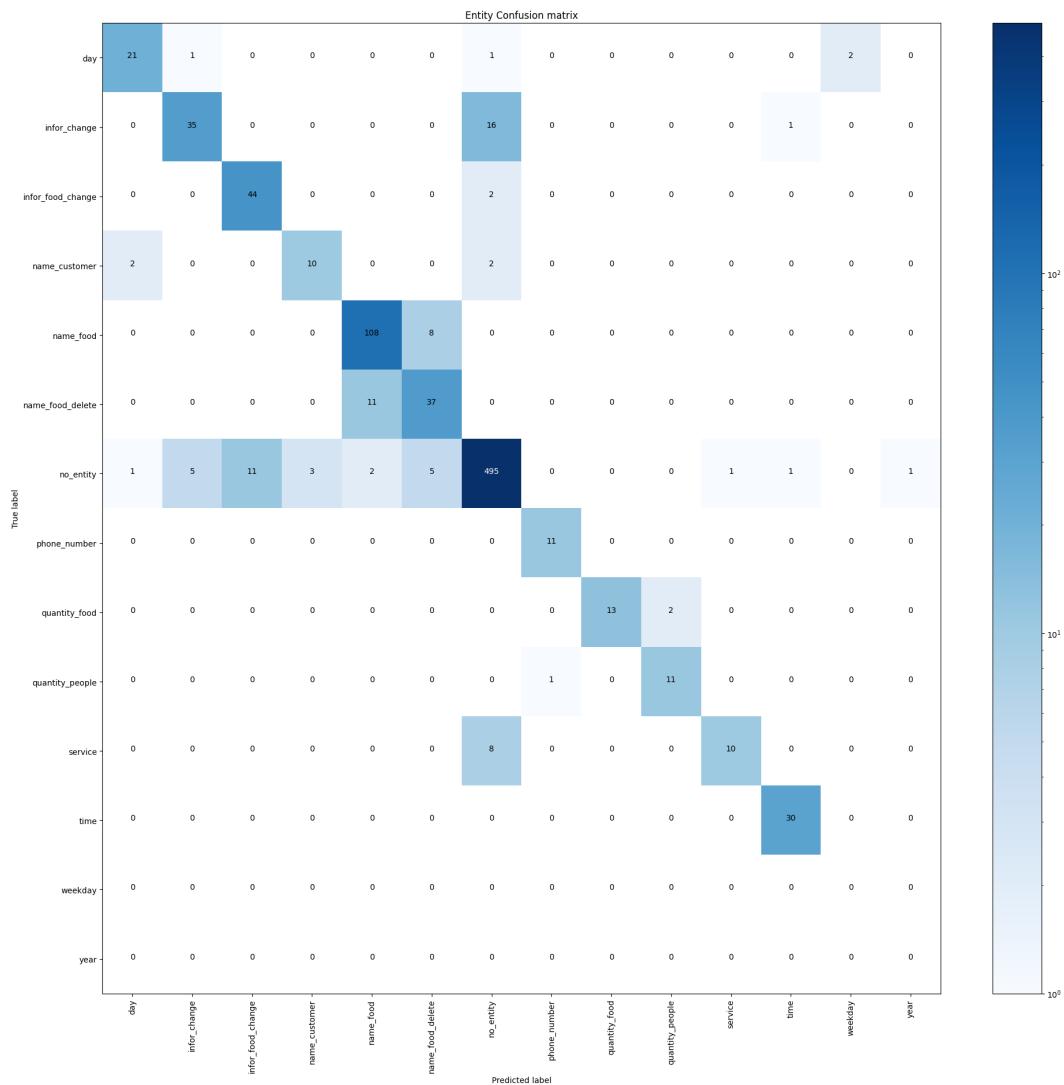
Bảng 5.2: Bảng ước lượng độ chính xác dự đoán từng ý định

đúng được hiển thị bằng thanh màu xanh và dự đoán sai được thể hiện bằng thanh màu đỏ. Việc thêm những dữ liệu có độ tin cậy cao và ít xung đột sẽ làm giảm thanh màu đỏ và tăng thanh màu xanh bên trái đồ thị. Ngược lại, khi dữ liệu có nhiều xung đột sẽ gây ra nhiều sự nhầm lẫn về dự đoán thì thanh màu đỏ xuất hiện càng nhiều và giá trị càng cao, có thể gây ra sự nhầm lẫn cho bot. Mặc dù luồng câu chuyện có thể giải quyết được vấn đề này với độ chênh lệch về độ tin cậy của hai hoặc nhiều ý định dự đoán cao phải thấp. Vì vậy, tăng độ tin cậy dựa vào tránh xung đột về dữ liệu là cần thiết để huấn luyện cho bot.

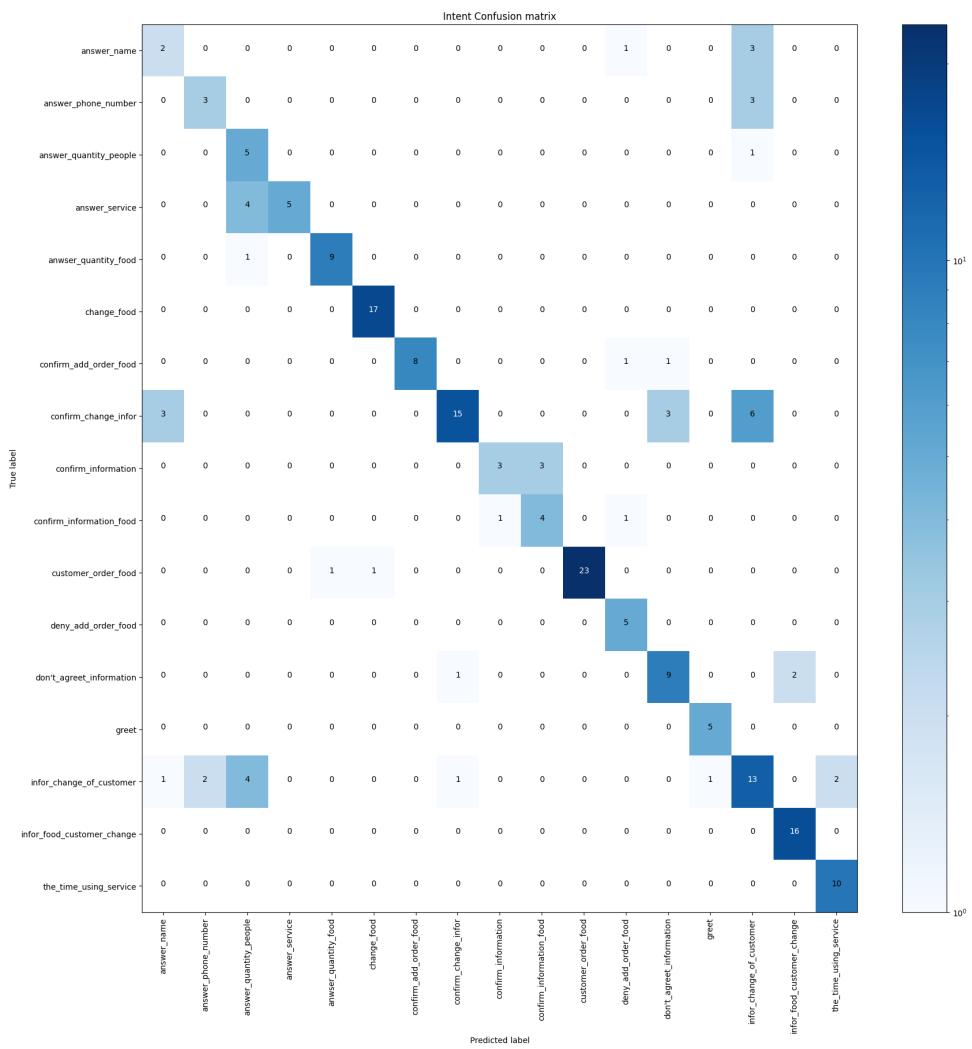
5.2.4 Đánh giá mô hình hiểu ngôn ngữ

Việc đánh giá mô hình hiểu ngôn ngữ này được thực hiện với các câu ngẫu nhiên do người dùng thực hiện. Đánh giá bao gồm việc xác định đúng ý định và trích xuất đúng thực thể yêu cầu. Quá trình thực nghiệm này được thực hiện với 5 người dùng tương tác với bot qua từng câu chuyện và được ghi lại để đánh giá việc dự đoán của bot. Tổng số

câu được ghi nhận và sử dụng cho việc đánh giá này bao gồm 200 câu bao gồm hết tất cả các ý định được hiện thực trong luồng câu chuyện. Hình 5.18 và 5.19 biểu thị ma trận trực quan về độ nhầm lẫn của việc dự đoán ý định và trích xuất thực thể từ các mẫu câu tương tác với khách hàng. Độ chính xác cuối cùng về trích xuất dữ liệu là 85.27% và xác định đúng ý định là 76% như trong bảng 5.3 và 5.4.



Hình 5.18: Ma trận nhầm lẫn trích xuất thực thể từ người dùng



Hình 5.19: Ma trận nhầm lẫn dự đoán ý định từ người dùng

Việc đánh giá dựa vào tin nhắn của khách hàng có thể được xem là khách quan khi dữ liệu được kiểm tra được lấy từ tin nhắn của khách hàng thực tế. Việc đánh giá này giúp người phát triển có thể đa dạng về mẫu câu để tạo dữ liệu đào tạo bot. Sự đa dạng hóa được người phát triển bot lấy mẫu câu của người dùng thực tế để tiến hành phân tích các thực thể xuất hiện trong câu và xác định câu đó thuộc ý định nào. Ngoài ra, việc đánh giá này có thể giúp nhà phát triển bot có thể tối ưu dữ liệu đào tạo như bỏ các thực thể không cần thiết, đa dạng hóa dữ liệu của bảng tìm kiếm và danh sách từ đồng nghĩa từ đó giúp dữ liệu đào tạo bot thêm đáng tin cậy. Như trong hình 5.18 dựa vào tin nhắn người dùng thì thực thể weekday và year không xuất hiện trong câu nên trong dữ liệu đào tạo ta có thể bỏ hai thực thể này để giảm độ phức tạp của dữ liệu khi đào tạo bot.

Entity	Độ tin cậy			
	Precision	Recall	F1-score	Support
day	0.875	0.84	0.8571	25
name_customer	0.7692	0.7143	0.7407	14
infor_food_change	0.8	0.9565	0.8713	46
quantity_food	1.0	0.8667	0.9285	15
quantity_people	0.8461	0.9167	0.8799	12
service	0.9091	0.5555	0.6896	18
phone_number	0.9166	1.0	0.9565	11
name_food_delete	0.74	0.7708	0.7551	48
infor_change	0.8536	0.6731	0.7527	52
time	0.9375	1.0	0.9677	30
name_food	0.8925	0.9310	0.9114	116
Weighted Average	0.8594	0.8527	0.8512	387
Accuracy		0.8527		

Bảng 5.3: Bảng ước lượng độ chính xác dự đoán từng thực thể dựa trên dữ liệu từ khách hàng

Intent	Độ tin cậy			
	Precision	Recall	F1-score	Support
anwser_quantity_food	0.9	0.9	0.9	10
deny_add_order_food	0.625	1.0	0.7692	5
customer_order_food	1.0	0.92	0.9583	25
the_time_using_service	0.8333	1.0	0.9091	10
answer_service	1.0	0.5555	0.7143	9
answer_phone_number	0.6	0.5	0.5454	6
confirm_information_food	0.5714	0.6666	0.61538	6
infor_change_of_customer	0.5	0.54166	0.52	24
confirm_add_order_food	1.0	0.8	0.8888	10
answer_name	0.3333	0.3333	0.3333	6
confirm_change_infor	0.8823	0.5555	0.6818	27
answer_quantity_people	0.3571	0.8333	0.5	6
infor_food_customer_change	0.8888	1.0	0.9412	16
change_food	0.9444	1.0	0.9714	17
greet	0.8333	1.0	0.9090	5
confirm_information	0.75	0.5	0.6	6
don't_agreeet_information	0.6923	0.75	0.71999	12
Weighted Average	0.7935	0.76	0.7621	200
Accuracy		0.76		

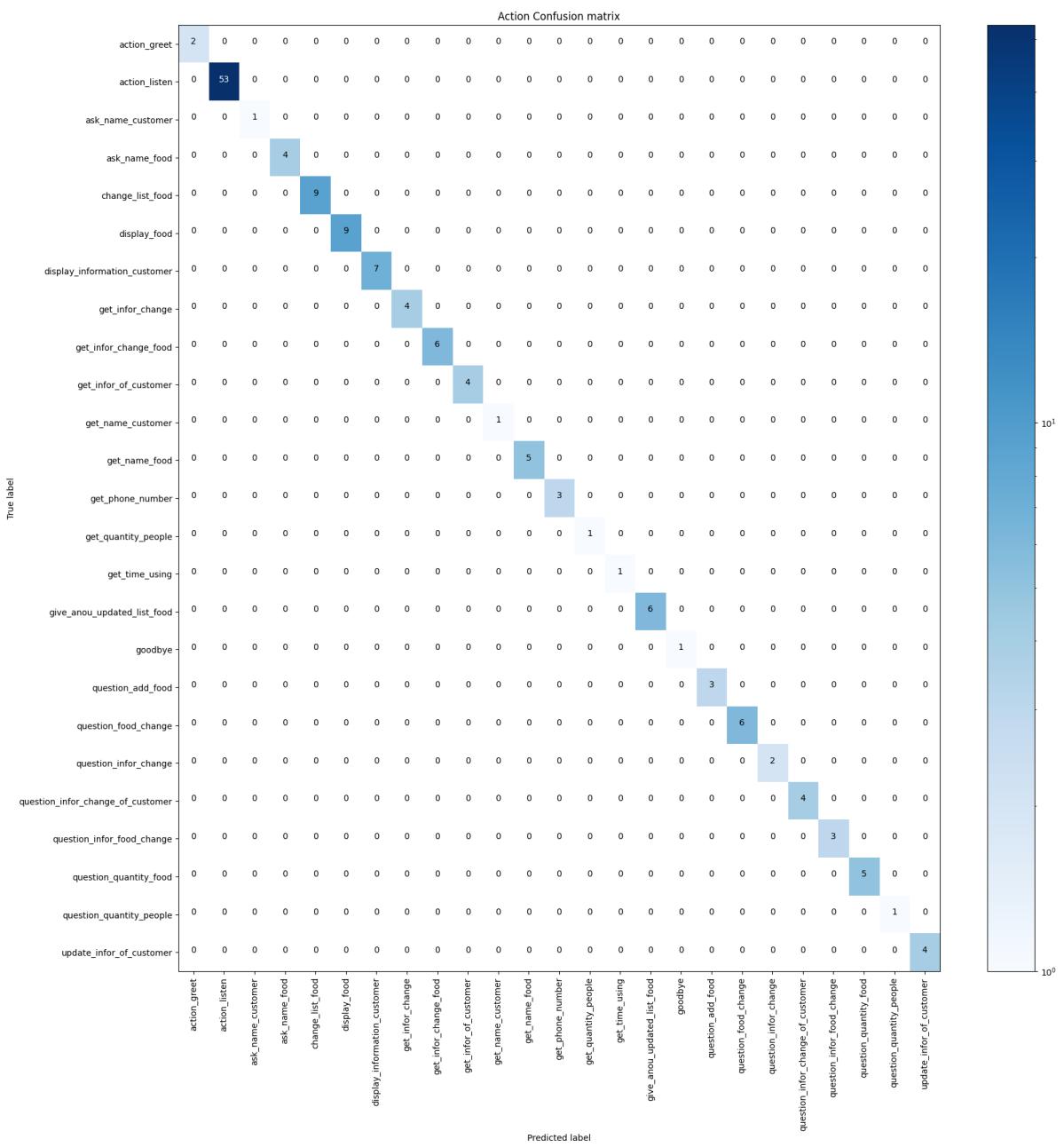
Bảng 5.4: Bảng ước lượng độ chính xác dự đoán từng ý định dựa trên dữ liệu của khách hàng

5.2.5 Đánh giá mô hình đối thoại

Việc đánh giá mô hình đối thoại có chức năng đánh giá được hành động được thực hiện khi xác định được ý định của khách hàng. Cách thức để đánh giá mô hình đối thoại này được thực hiện thông qua câu chuyện mẫu được viết qua để đi qua tất cả các luồng được thiết kế. Bất kỳ hành động nào được hiện thực sai so với thiết kế trong luồng đều sẽ được ghi lại. Mẫu truyện thử nghiệm sẽ cung cấp cho toàn bộ cuộc hội thoại để kiểm tra với ý định người dùng thì bot có dự đoán và quyết định đúng hành động được theo luồng mong muốn được viết trong mẫu chuyện thử nghiệm hay không. Trong mẫu chuyện thử nghiệm sẽ bao gồm tin nhắn người dùng, ý định được xác định và hành động thực hiện tiếp theo. Một câu chuyện được xem là sai khi có bất kỳ hành động nào sai được thực hiện. Khi đó, luồng câu chuyện sẽ bị sai lệch và sẽ gây ra việc mất đi sự hỗ trợ dự đoán ý định từ luồng hội thoại thiết kế.

Action	Độ tin cậy			
	Precision	Recall	F1-score	Support
question_food_change	1.0	1.0	1.0	6
get_time_using	1.0	1.0	1.0	1
display_information_customer	1.0	1.0	1.0	7
question_quantity_food	1.0	1.0	1.0	5
question_infor_change	1.0	1.0	1.0	2
change_list_food	1.0	1.0	1.0	9
question_quantity_people	1.0	1.0	1.0	1
goodbye	1.0	1.0	1.0	1
get_name_food	1.0	1.0	1.0	5
ask_name_food	1.0	1.0	1.0	4
get_name_customer	1.0	1.0	1.0	1
question_infor_food_change	1.0	1.0	1.0	3
get_infor_change	1.0	1.0	1.0	4
action_greet	1.0	1.0	1.0	2
display_food	1.0	1.0	1.0	9
give_anou_updated_list_food	1.0	1.0	1.0	6
ask_name_customer	1.0	1.0	1.0	1
question_infor_change_of_customer	1.0	1.0	1.0	4
get_quantity_people	1.0	1.0	1.0	1
get_phone_number	1.0	1.0	1.0	3
update_infor_of_customer	1.0	1.0	1.0	4
get_infor_of_customer	1.0	1.0	1.0	4
get_infor_change_food	1.0	1.0	1.0	6
action_listen	1.0	1.0	1.0	53
question_add_food	1.0	1.0	1.0	3
Weighted Average	1.0	1.0	1.0	145
Accuracy			1.0	

Bảng 5.5: Bảng ước lượng độ chính xác dự đoán các hành động



Hình 5.20: Ma trận nhầm lẫn các dự đoán hành động

Hình 5.20 biểu thị sự nhầm lẫn của các dự đoán và đưa ra quyết định hành động của bot. Quá trình đánh giá này sẽ đưa ra mức độ chính xác về sự quyết định hành động của bot. Quá trình đánh giá và chỉnh sửa luồng câu chuyện này sẽ được thực hiện đi thực hiện lại cho tới khi sẽ chắc chắn đưa ra hành động chính xác theo như đã thiết kế khi xác định đúng ý định của khách hàng. Dựa vào hình 5.20 và bảng 5.5 ta thấy, khả năng dự đoán và quyết định hành động là chính xác theo thiết kế thiết kế. Trong đó, hành động `action_listen` là hành động bot sẽ chờ đợi để nhận được tin nhắn khách hàng.

Chương 6

Tổng kết

6.1 Kết luận

Chatbot có thể thay thế con người để nhắn tin, trò chuyện với khách hàng. Chatbot có thể được sử dụng trong công ty và doanh nghiệp để tăng hiệu xuất và giảm chi phí cho doanh nghiệp một cách hiệu quả. Chatbot có thể hoạt động 24/7 nên có thể tăng cường khả năng tương tác với khách hàng.

Chatbot Rasa hoạt động trò chuyện theo nguyên tắc, từ tin nhắn của khách hàng bot sẽ phân tích và xác định ý định của người dùng. Đối với Chatbot sử dụng cho các doanh nghiệp về dịch vụ cho khách hàng, thì Chatbot theo nguyên tắc là giải pháp có thể đáp ứng được cho nhu cầu dịch vụ khách hàng. Rasa framework là một mã nguồn mở nên có thể được sử dụng để tạo dữ liệu nhằm phù hợp với chủ đề từ doanh nghiệp.

Rasa có thể kết nối với nhiều kênh tin nhắn làm có thể làm tăng lượng tương tác với người dùng. Đây là một trong những thế mạnh của Rasa khi có thể giao tiếp với khách hàng qua nhiều kênh mạng xã hội giúp tăng lượng khách hàng cho doanh nghiệp. Chatbot hầu như phản hồi ngay lập tức khi nhận được tin nhắn từ khách hàng bất kì nên hạn chế sự không hài lòng cho khách hàng. Chatbot có thể được sử dụng với nhiều mục đích và loại dịch vụ mà công ty có nhu cầu sử dụng.

Việc chat trực tiếp có thể giúp khách hàng có trải nghiệm tốt nhất, có cảm nhận như đang nhắn tin trực tiếp với nhân viên. Với khả năng hoạt động 24/7 nên có thể đem có thể đáp ứng khách hàng trong mọi khung giờ trong ngày. Hơn nữa, Chatbot có thể giao tiếp cùng lúc với nhiều khách hàng trong một lúc, vì vậy sẽ tránh được tình trạng phản hồi chậm do số lượng khách hàng lớn.

Việc trích xuất và lưu lại các thông tin quan trọng của khách hàng từ tin nhắn của khách hàng giúp cho doanh nghiệp nắm bắt được các thông tin quan trọng cũng như các nhu cầu của khách hàng đang quan tâm đến những dịch vụ của doanh nghiệp. Nên việc xác định đúng ý định có ý nghĩa rất quan trọng đối với chủ đề được tạo ra.

Việc khai báo các kiểu dữ liệu về thực thể sẽ làm giảm khả năng về việc nhầm lẫn giữa các tên thực thể về giá trị thực thể đó đối với các thực thể có kiểu dữ liệu cụ thể. Các bảng tìm kiếm cũng có thể được sử dụng để tăng khả năng trích xuất thực thể từ câu nói của khách hàng. Đối với các thực thể có cùng kiểu dữ liệu thì sự xác định đúng tên thực thể có thể dựa vào ý định của câu nói đó để xác định tên thực thể đúng.

Việc xây dựng luồng câu chuyện sẽ cải thiện và tăng khả năng dự đoán ý định người dùng. Đối với các câu có độ tin cậy dự đoán về hai hay nhiều ý định đều cao thì bot sẽ căn cứ vào luồng câu chuyện để quyết định câu nói của khách hàng thuộc ý định nào.

Trong quá trình xây dựng Chatbot, quá trình tạo dữ liệu là quá trình quan trọng để huấn luyện Chatbot một cách chính xác về dự đoán ý định và trích xuất thông tin cần thiết có trong câu nói của người dùng. Vì vậy, việc phân tích và nắm rõ chủ đề của Chatbot là cần thiết để xây dựng dữ liệu của Chatbot. Việc đào tạo Chatbot dựa trên tập dữ liệu có độ tin cậy cao sẽ giúp tăng độ chính xác cho mô hình dự đoán.

Chatbot Rasa có thể được cấu hình đa dạng các phản hồi để phản hồi lại người khách hàng một cách hợp lý. Nên nhà phát triển có thể cấu hình nhiều cách phản hồi khác nhau để tăng mức độ hài lòng của khách hàng.

Qua bài toán thử nghiệm về chủ đề đặt tiệc nhà hàng thì có thể thấy việc áp dụng Chatbot Rasa cho các chủ đề được xác định là khả thi và có thể triển khai một cách hợp lý để giảm chi phí và tăng hiệu quả trong việc chăm sóc khách hàng.

6.2 Các hướng phát triển

Trong menu đặt tiệc của nhà hàng, việc kết hợp và thiết kế một menu hoàn chỉnh sẽ giúp cho khách hàng tăng thêm mức độ hài lòng đến khách hàng.

Có thể phát triển thêm các tính năng và kịch bản cho Chatbot như kiểm trả lại thông tin, hủy dịch vụ, .v.v.

Tạo thêm nhiều dữ liệu để giúp Chatbot tăng khả năng dự đoán ý định về khách hàng. Tạo dữ liệu và thêm phần xử lý đối với các câu xuất hiện nhiều thực thể cùng một lúc ví dụ như: Gọi nhiều món cùng một lúc, gọi món đi kèm với số lượng, khách hàng có thể trả lời 1 lúc nhiều thông tin, .v.v.

Chatbot này có thể tích hợp speech to text và text to speech để có thể giao tiếp với khách hàng qua giọng nói.

Tài liệu tham khảo

- [1] Marc Mercir. *Three Key Advantages of Using a Framework.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://botpress.com/blog/three-key-advantages-of-using-a-framework>
- [2] Beata Stefanowicz. *9+ Best Open Source Chatbot Frameworks Compared.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://www.tidio.com/blog/chatbot-framework/>
- [3] REACH DIGITAL. *Talk Bot-y To Me: Boost Conversions with Chatbots.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://reachdigital.ca/talk-bot-y-to-me-boost-conversion-with-chatbots/>
- [4] MOUSUMI. *13 Best AI Chatbot Development Frameworks Platforms.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://www.kommunicate.io/blog/chatbot-framework-platform/>
- [5] Software Engineer with profession. *Microsoft Bot Framework: The limitations you should know about.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://jd-bots.com/2022/04/10/microsoft-bot-framework-the-limitations-you-should-know-about/>
- [6] SBuilding Bots with Wit.ai. <https://discover.bot/bot-talk/guide-to-bot-buiding-frameworks/wit-ai/>. Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://discover.bot/bot-talk/guide-to-bot-buiding-frameworks/wit-ai/>
- [7] Chatbots.org. *Dialogflow Review of Pricing, Features Pros Cons, and Bot Builders Expert Advices.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://www.chatbots.org/dialogflow#section3>
- [8] Artificial Solutions. *Types of Chatbot Technology.* Cập nhập lần cuối year=5/4/2023, Truy cập từ <https://reachdigital.ca/talk-bot-y-to-me-boost-conversion-with-chatbots/>

-
- [9] GitHub CI. *Rasa Open Source Document*. Cập nhập lần cuối year=16/12/2022, Truy cập từ <https://rasa.com/docs/rasa/>
- [10] Harith L. K, K. Vadhi Raja, Dr. G. S. Mamatha. *University chatbot with database integration using RASA*. year=2021, volume=7, Truy cập từ <https://www.ijaraiit.com/manuscripts/v7i3/V7I3-2193.pdf>, ISSN=2454-132X.
- [11] Rakesh Patel. *What is Rasa Framework? (10 Reasons to Choose Rasa Framework)*. Cập nhập lần cuối year=2022, volume=7, Truy cập từ <https://www.spaceo.ca/blog/ai-chatbot-development-using-rasa-reasons/>.
- [12] Naveen Kumar. *Understanding the Basics of Rasa - Open source conversational AI*. Cập nhập lần cuối year=2020. Truy cập từ <https://smazee.com/blog/basics-of-rasa>.
- [13] Vincent Warmerdam. *Bending the ML Pipeline in Rasa 3.0*. Cập nhập lần cuối year=2021. Truy cập từ <https://rasa.com/blog/bending-the-ml-pipeline-in-rasa-3-0/>.
- [14] Inés Roldós. *Introduction to Entity Extraction: What Is It And How It Works*. Cập nhập lần cuối year=5/4/2023. Truy cập từ <https://monkeylearn.com/blog/entity-extraction/>.
- [15] Kunal Bhashkar. *Conversational AI chatbot using Rasa NLU Rasa Core: How Dialogue Handling with Rasa Core can use LSTM by using Supervised and Reinforcement Learning Algorithm*. Cập nhập lần cuối year=5/4/2023. Truy cập từ <https://bhashkarkunal.medium.com/conversational-ai-chatbot-using-rasa-nlu-rasa-core-how-dialogue-handling-with-rasa-core-can-use-331e7024f733>.
- [16] Charu C. Aggarwal (2018). *Neural Networks and Deep Learning*. Truy cập từ <https://doi.org/10.1007/978-3-319-94463-0>.
- [17] Saurabh Rathor. *Simple RNN vs GRU vs LSTM :- Difference lies in More Flexible control*. Cập nhập lần cuối year=19/5/2023. Truy cập từ <https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-\protect\@normalcr\relaxcontrol-5f33e07b1e57>.
- [18] TAPIT. *Trí tuệ nhân tạo: Các phương pháp đánh giá một mô hình phân loại*. Cập nhập lần cuối year=5/4/2023. Truy cập từ <https://tapit.vn/cac-phuong-phap-danh-gia-mot-mo-hinh-phan-loai/3>.