HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CHU LÊ LONG

NGHIÊN CỨU, XÂY DỰNG CHATBOT HỎI ĐÁP THÔNG TIN KHÁCH SẠN SỬ DỤNG RASA FRAMEWORK

CHUYÊN NGÀNH: **KHOA HỌC MÁY TÍNH**
MÃ SỐ: **8.48.01.01**

TÓM TẮT LUẬN VĂN THẠC SĨ

HÀ NỘI - 2020

Luận văn được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: **PGS.TS. NGUYỄN MẠNH HÙNG**

Phản biện 1: …….....

Phản biện 2: …….....

Luận văn sẽ được bảo vệ trước Hội đồng chấm luận văn thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông

Vào lúc: giờ ngày tháng năm 2020

Có thể tìm hiểu luận văn tại:

- Thư viện của Học viện Công nghệ Bưu chính Viễn thông

MỞ ĐẦU

Ngày nay, cùng với sự phát triển của khoa học kỹ thuật, Chatbot đang được ứng dụng phổ biến và mạnh mẽ trong nhiều lĩnh vực, tạo nên một cơn sốt công nghệ khi có nhiều hãng công nghệ nổi tiếng thế giới tham gia như Google, Facebook, Microsoft, IBM...Theo Grand View Research, thị trường Chatbot dự kiến sẽ đạt khoảng 1,25 tỷ đô la trên toàn cầu vào năm 2025. Hơn nữa, các chuyên gia dự đoán rằng thị trường này sẽ tăng trưởng với tốc độ tăng trưởng gộp hàng năm hơn 24%. Đặc biệt là xu hướng chuyển dịch phát triển AI chatbot có khả năng hội thoại, xử lý những tương tác phức tạp hơn với khách hàng. Ở Việt Nam, chatbot đã bắt đầu được áp dụng ở trong một số lĩnh vực như chăm sóc khách hàng, mua sắm trực tuyến, trả lời thông tin ngân hàng, y tế… Đối với lĩnh vực du lịch, khách sạn chatbot chưa được sử dụng nhiều dù rằng đây là lĩnh vực rất phù hợp cho các ứng dụng chatbot. Chatbot có thể thay thế con người trong việc trả lời các câu hỏi có tính lặp đi lặp lại, hỗ trợ khách hàng 24/7, tiếp thị quảng cáo cho doanh nghiệp…

Với mong muốn hiểu sâu hơn về chatbot và các kỹ thuật giúp chatbot trả lời câu hỏi xử lý theo ngôn ngữ tự nhiên (NLP), em quyết định chọn đề tài "Nghiên cứu, xây dựng Chatbot hỏi đáp thông tin khách sạn sử dụng Rasa Framework" làm đề tài luận văn thạc sĩ. Qua đề tài em muốn nâng cao sự hiểu biết về AI Chatbot, NLP (xử lý ngôn ngữ tự nhiên) và nghiên cứu khả năng áp dụng thực tiễn tại Việt Nam.

Nội dung luận văn được chia ra làm 3 phần như sau:

Chương 1: Giới thiệu tổng quan về hệ thống chatbot, kiến trúc high-level và các thành phần cơ bản của AI chatbot, một số nền tảng và ứng dụng của chatbot.

Chương 2: Nghiên cứu một số kĩ thuật được sử dụng trong chatbot, tìm hiểu về Rasa Framework.

Chương 3: Trình bày về quá trình xây dựng chatbot trả lời thông tin khách sạn, thực nghiệm và đánh giá các kết quả.

CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN VỀ CHATBOT

1.1. Khái niệm

Theo từ điển Cambridge, chatbot là một một chương trình máy tính được thiết kế để trò chuyện với con người, đặc biệt là qua internet.

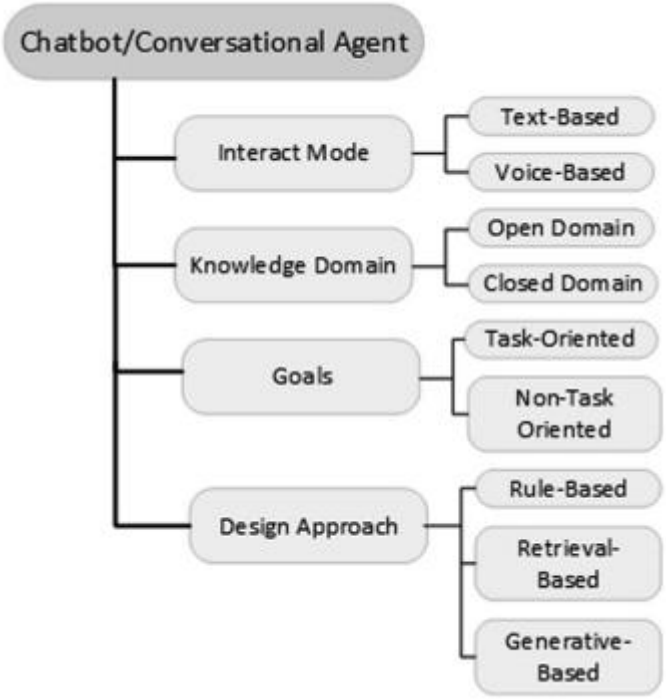
Chatbot thường trao đổi với người dùng qua hình thức tin nhắn hoặc âm thanh. Do được thiết kế để mô phỏng cách trò chuyện với con người, các hệ thống chatbot thường phải điều chỉnh và thử nghiệm liên tục.

Chatbot thường được sử dụng trong các hệ thống hội thoại cho các mục đích khác nhau bao gồm dịch vụ khách hàng, định tuyến yêu cầu hoặc để thu thập thông tin. Mặc dù một số ứng dụng chatbot sử dụng các phương pháp phân loại từ (word-classification), xử lý ngôn ngữ tự nhiên (NLP) và trí tuệ nhân tạo (AI), một số ứng dụng khác chỉ cần quét các từ khóa chung và tạo phản hồi bằng các cụm từ phổ biến thu được từ thư viện hoặc cơ sở dữ liệu liên quan.

1.2. Lịch sử ra đời

1.3. Phân loại chatbot

Chatbots có thể được phân loại thành nhiều loại khác nhau dựa trên một số tiêu chí. Các phân loại có thể được thực hiện dựa trên các tiêu chí sau.



Hình 1.1: Phân loại Chatbot

- Theo chế độ tương tác (Interact Mode):

+ Dựa trên văn bản (Text-Based)

+ Dựa trên giọng nói (Voice-Based)

- Theo miền (Domain):

+ Miền đóng/miền cụ thể (Closed Domain): Phạm vi của chatbot chỉ giải quyết một số vấn đề trong phạm vi nhất định.

+ Miền mở (Open Domain): Loại này là mục tiêu của trí tuệ nhân tạo. Một chatbot biết mọi thứ và có thể trả lời mọi vấn đề.

- Theo mục tiêu (Goals):

+ Các chatbot hướng nhiệm vụ (Task-Oriented): được thiết kế cho một nhiệm vụ cụ thể và được thiết lập để có thời gian ngắn các cuộc hội thoại, thường là trong một miền đóng.

+ Các chatbot không hướng nhiệm vụ (Non Task-Oriented): có thể mô phỏng cuộc trò chuyện với một người và thường thực hiện chat cho mục đích giải trí trong các miền mở.

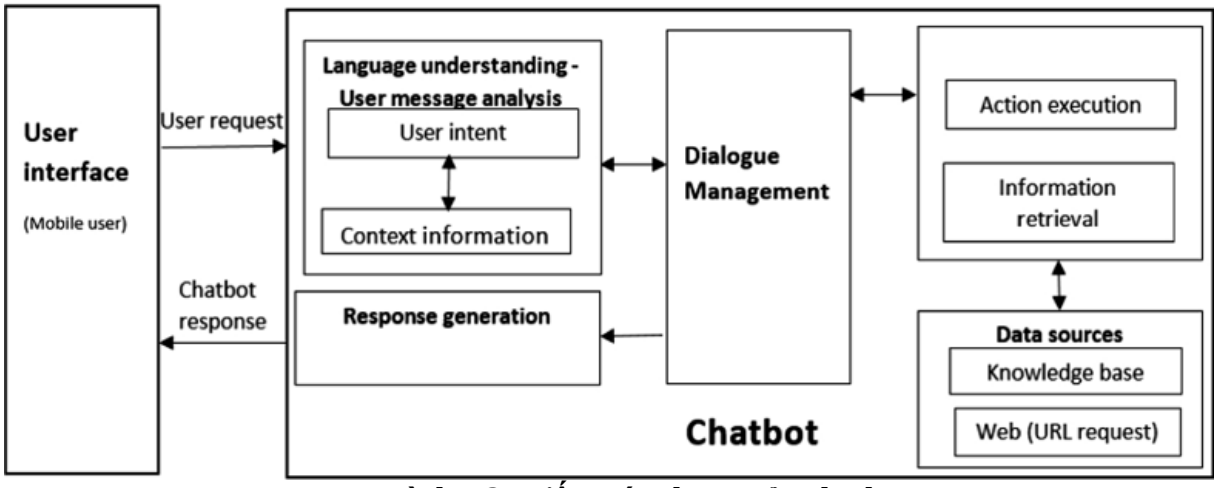
- Theo Phương pháp thiết kế (Design Approach):

+ Dựa theo luật (Rule- Based): Loại chatbot này khả năng rất hạn chế. Chỉ có khả năng phản hồi chính xác những lệnh cụ thể mà ta đã xác định từ trước hoặc người dùng không được phép tùy ý phản hồi mà phải lựa chọn các phản hồi do lập trình viên tạo ra.

+ Dựa theo trí tuệ nhân tạo (AI): Loại này có khả năng "hiểu" ngôn ngữ. Nghĩa là chatbot không bị giới hạn bởi tập các luật xác định từ trước, mà có thể hiểu ở phạm vi rộng hơn. Tất nhiên chatbot vẫn phải được "học" từ dữ liệu có sẵn, nhưng nó có khả năng "đoán" được ý nghĩa và ngữ cảnh của những lệnh chưa từng gặp. Một khả năng nữa của chatbot dựa trên AI là khả năng "học thêm".

1.4.Các thành phần cơ bản của hệ thống chatbot

Bước đầu tiên trong việc thiết kế bất kỳ hệ thống nào là chia nó thành các bộ phận cấu thành theo một tiêu chuẩn để có thể tuân theo cách tiếp cận phát triển mô đun. Trong hình dưới giới thiệu một kiến trúc chatbot chung của chatbot.



Hình 1.2: Kiến trúc chung của chatbot

Dưới đây trình bày chi tiết về các thành phần của chatbot.

1.4.1. NLU (Hiểu ngôn ngữ tự nhiên)

NLU (Natural Language Understanding - hiểu ngôn ngữ tự nhiên): bao gồm việc xử lý ngôn ngữ tự nhiên (NLP) có nhiệm vụ xác định được ý định câu hỏi (intent classification) và trích chọn thông tin (slots filter).

NLU nhằm mục đích trích xuất ngữ cảnh (context) và ý nghĩa từ đầu vào của người dùng bằng ngôn ngữ tự nhiên, mà có thể không có cấu trúc và phản hồi một cách thích hợp theo ý định của người dùng (user intent). Nó xác định mục đích của người dùng và trích xuất các thực thể (entities) theo miền cụ thể. Cụ thể hơn, một ý định đại diện cho một ánh xạ giữa những gì người dùng nói và hành động (action) nên được thực hiện bởi chatbot.

Đúc kết lại, khi người dùng gõ một câu "What is the meaning of environment?" trong một chatbot sử dụng ứng dụng nhắn tin như Facebook, Slack, WhatsApp, WeChat hoặc Skype. Sau khi chatbot nhận được yêu cầu của người dùng, thành phần hiểu ngôn ngữ tự nhiên sẽ phân tích nó để suy ra ý định của người dùng và thông tin liên quan (ý định: dịch, thực thể: [từ: environment].

1.4.2. DM (Quản lý hội thoại)

DM (Dialog Management - quản lý hội thoại): Thành phần quản lý đối thoại giữ và cập nhật ngữ cảnh của cuộc hội thoại là ý định hiện tại, các thực thể được xác định hoặc các thực thể bị thiếu cần thiết để thực hiện các yêu cầu của người dùng. Hơn nữa, nó yêu cầu thông tin thiếu, xử lý làm rõ bởi người dùng và đặt câu hỏi tiếp theo. Ví dụ: chatbot có thể phản hồi câu hỏi trên lại bằng câu: "Would you like to tell me as well an example sentence

with the word environment?". Quản lý hội thoại cũng có nhiệm vụ xác định được hành động (action) tiếp theo dựa vào trạng thái hành động trước đó hay ngữ cảnh hội thoại.

1.4.3. NLG (Sinh ngôn ngữ tự nhiên)

NLG (Natural Language Generator - Sinh ngôn ngữ tự nhiên): là thành phần sinh ngôn ngữ dựa vào chính sách (policy) và hành động được xác định trong DM thông qua các tập hội thoại.

Khi phản hồi, NLG chuẩn bị phản hồi giống ngôn ngữ tự nhiên cho người dùng dựa trên ý định và thông tin ngữ cảnh. Các câu trả lời thích hợp được tạo ra bởi một trong các mô hình thiết kế theo luật hoặc theo AI.

1.5. Một số nền tảng phát triển chatbot

- Dialogflow (https://dialogflow.com/)
- Rasa (https://rasa.com/)
- Wit.ai (https://wit.ai)
- Microsoft Bot Framework (https://dev.botframework.com/)
- Woebot (https://woebot.io/)
- Chatfuel (https://chatfuel.com/)

1.6. Một số ứng dụng của chatbot

- Thương mại điện tử
- Nhà hàng và dịch vụ ăn uống
- Tài chính và ngân hàng
- Phương tiện truyền thông tin tức
- Y tế
- Hàng không

1.7. Giới thiệu chatbot trả lời thông tin du lịch, khách sạn

- The Cosmopolitan of Las Vegas
- Marriott Hotels
- Hyatt Hotels

1.8. Kết luận chương

Qua các ví dụ trên, ta có thể nhận thấy bài toán chatbot trả lời thông tin du lịch, khách sạn chỉ cần tập trung vào trả lời các câu hỏi đối thoại liên quan đến một miền cụ thể xoay

xung quanh các thông tin về khách sạn, nhằm cố gắng để đạt được các mục tiêu thông tin rất cụ thể. Đây là bài toán trong miền đóng (Closed domain).

Ngoài ra, dù có nhiều nền tảng hỗ trợ xây dựng chatbot nhưng trong luận văn này, tác giả lựa chọn Rasa framework để xây dựng thực nghiệm công cụ chatbot minh họa với các lý do sau:

- Hiện tại hầu hết các nền tảng lớn như Dialogflow (Google), wit.ai, Microsoft Bot Framework… do các hãng lớn cung cấp, có nhiều công cụ xây dựng và tích hợp thông qua API nhưng các thuật toán và dữ liệu người dùng sẽ lưu trên các nền tảng này, khó làm chủ hệ thống trong khi đó việc sử dụng mã nguồn mở Rasa để xây dựng hệ thống Chatbot giúp nắm rõ hơn các công nghệ phía sau chatbot, làm chủ dữ liệu và thông tin người dùng.

- Rasa thực sự dễ tiếp cận cho người mới bắt đầu, ngay cả người không biết gì về chatbot hay NLP cũng có thể làm quen sử dụng. Hầu hết công việc của người sử dụng là tập trung xây dựng nội dung kịch bản, khả năng của chatbot chứ không cần thiết phải quan tâm đến công nghệ xây dựng.

- Rasa hoạt động khá tốt và mạnh mẽ, đặc biệt trong vấn đề xác định ý định người dùng (intent) và xác định thực thể (entity).

- Mã nguồn của Rasa là mã nguồn mở, do đó Rasa giúp ta biết chính xác được đang làm gì với chatbot của mình, thậm chí có thể tùy biến chatbot theo mong muốn của bản thân, ví dụ như: tùy biến hành động (action), quyết định nền tảng tin nhắn (messenger) của chatbot…

CHƯƠNG 2: GIỚI THIỆU MỘT SỐ KỸ THUẬT SỬ DỤNG TRONG CHATBOT VÀ RASA FRAMEWORK

2.1. Một số kỹ thuật sử dụng trong chatbot

2.1.1 Xác định ý định người dùng

Intent: điều người dùng mong muốn chatbot thực hiện (hỗ trợ) khi đưa ra câu hội thoại.

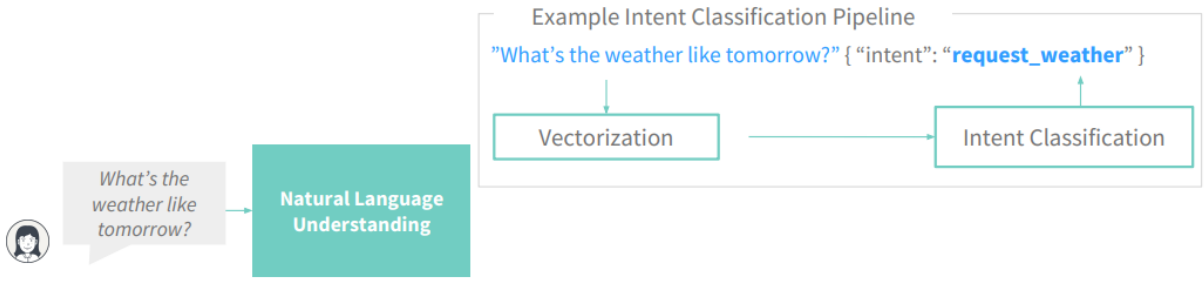
Ví dụ: Khi người dùng muốn hỏi về thông tin các loại phòng của khách sạn:

Khách sạn mình có những loại phòng nào vậy ad?

Khách sạn mình có phòng đơn không?

Khách sạn mình có phòng twin không?

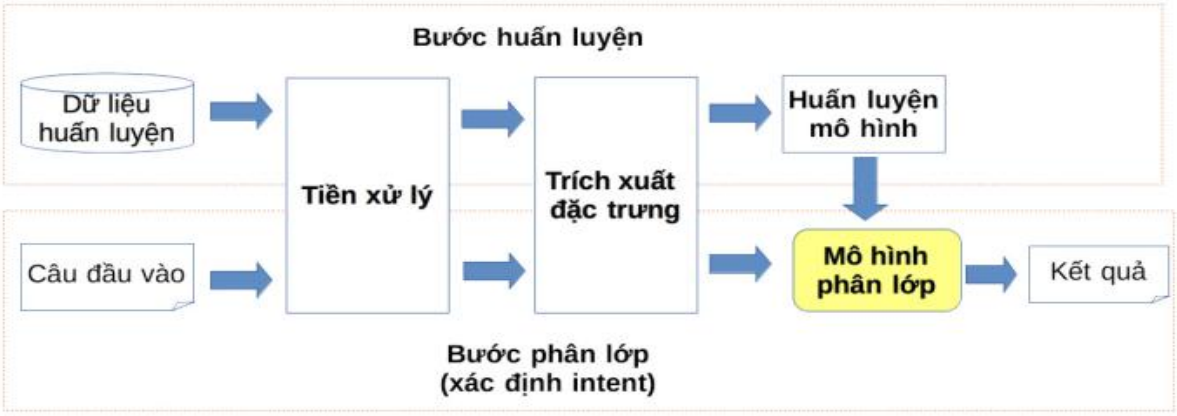
Hoặc khi một người dùng hỏi "What's the weather like tomorrow?" thì chatbot cần hiểu được ý định của họ là hỏi về thời tiết (request weather).



Hình 2.1 Xác định ý định người dùng

Intent được xác định sẽ quyết định cấu trúc (frame) và kịch bản (script) của đoạn hội thoại tiếp theo. Việc xác định ý định là rất quan trọng đối với chatbot. Nếu chatbot xác định sai intent sẽ dẫn đến phản hồi không thích hợp dẫn đến người dùng không hài lòng và có thể rời bỏ hệ thống.

Các bước xác định ý định dựa trên học máy được minh họa như hình dưới.



Hình 2.2: Xác định ý định dựa trên học máy

Hệ thống phân lớp ý định người dùng có một số bước cơ bản:

- Tiền xử lý dữ liệu:

Bước tiền xử lý dữ liệu chính là thao tác "làm sạch" dữ liệu như: loại bỏ các thông tin dư thừa, chuẩn hoá dữ liệu và chuyển các từ viết sai chính tả thành đúng chính tả, chuẩn hoá các từ viết tắt… Bước tiền xử lý dữ liệu có vai trò quan trọng trong hệ thống chatbot. Nếu dữ liệu đầu vào có xử lý ở bước này thì sẽ làm tăng khả năng năng độ chính xác cũng như sự thông minh cho bot.

Một số kỹ thuật tiền xử lý:

+ Tách từ (word segmentation): như tách câu thành các token.

+ Xử lý các từ đồng nghĩa

+ Xử lý từ gõ sai chính tả (ví dụ mạng chaamj)

+ Xử lý từ viết tắt (ví dụ: gõ ip thay vì iphone)

- Trích xuất đặc trưng:

Tiếp đến là bước trích xuất đặc trưng (feature extraction hay feature engineering) từ những dữ liệu đã được làm sạch. Trong mô hình học máy truyền thống (trước khi mô hình học sâu được áp dụng rộng rãi), bước trích xuất đặc trưng ảnh hưởng lớn đến độ chính xác của mô hình phân lớp. Để trích xuất được những đặc trưng tốt, chúng ta cần phân tích dữ liệu khá tỉ mỉ và cần cả những tri thức chuyên gia trong từng miền ứng dụng cụ thể. Đây là bước biểu diễn ngôn ngữ loài người dưới dạng số sao cho có nghĩa và machine có thể hiểu được.

Một số kỹ thuật trích xuất đặc trưng:

+ Word2Vec

+ One-hot Encoding

+ Bag of Words

+ TD/IDF…

- Huấn luyện mô hình và Mô hình phân lớp:

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

Một số kỹ thuật phân lớp:

Support Vector Machines (SVM)

Random Forest

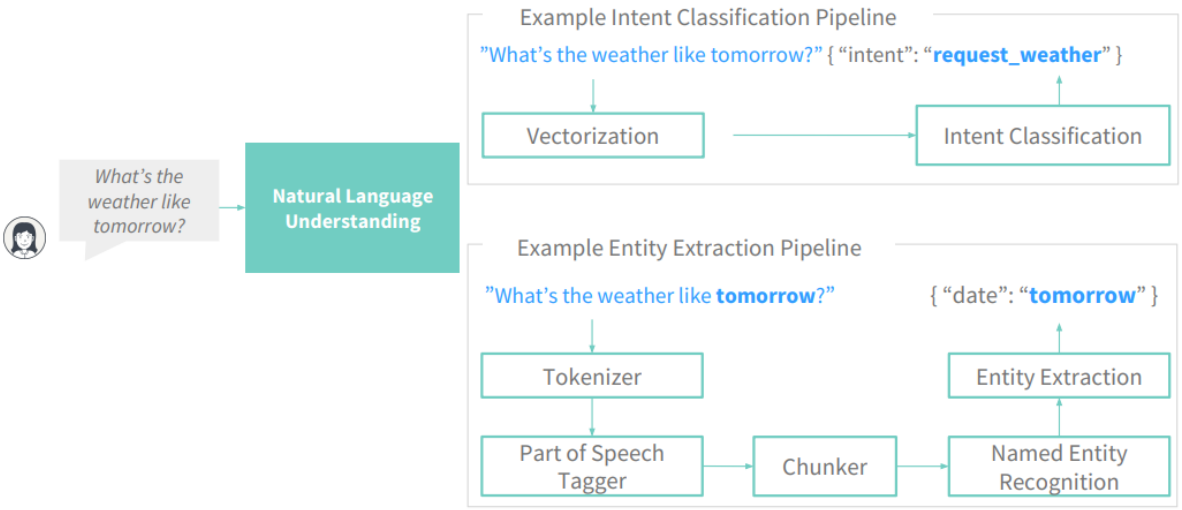
Neural Networks (LSTM)…

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

2.1.2 Trích xuất thông tin

Named Entity Recognition (NER): đây là tác vụ cơ bản trong lĩnh vực xử lý ngôn ngữ tự nhiên. Vai trò chính của tác vụ này là nhận dạng các cụm từ trong văn bản và phân loại chúng vào trong các nhóm đã được định trước như tên người, tổ chức, địa điểm, thời gian, loại sản phẩm, nhãn hiệu, ...

Vẫn ở ví dụ đề cập ở trên, khi người dùng hỏi "What's the weather like tomorrow?" thì chatbot ngoài cần hiểu được ý định của họ là hỏi về thời tiết, còn cần xác định được thực thể xác định ở thời gian (date) là tomorrow (ngày mai) [4].



Hình 2.3: Trích xuất thông tin thực thể

Các loại thực thể mà chatbot thường hỗ trợ:

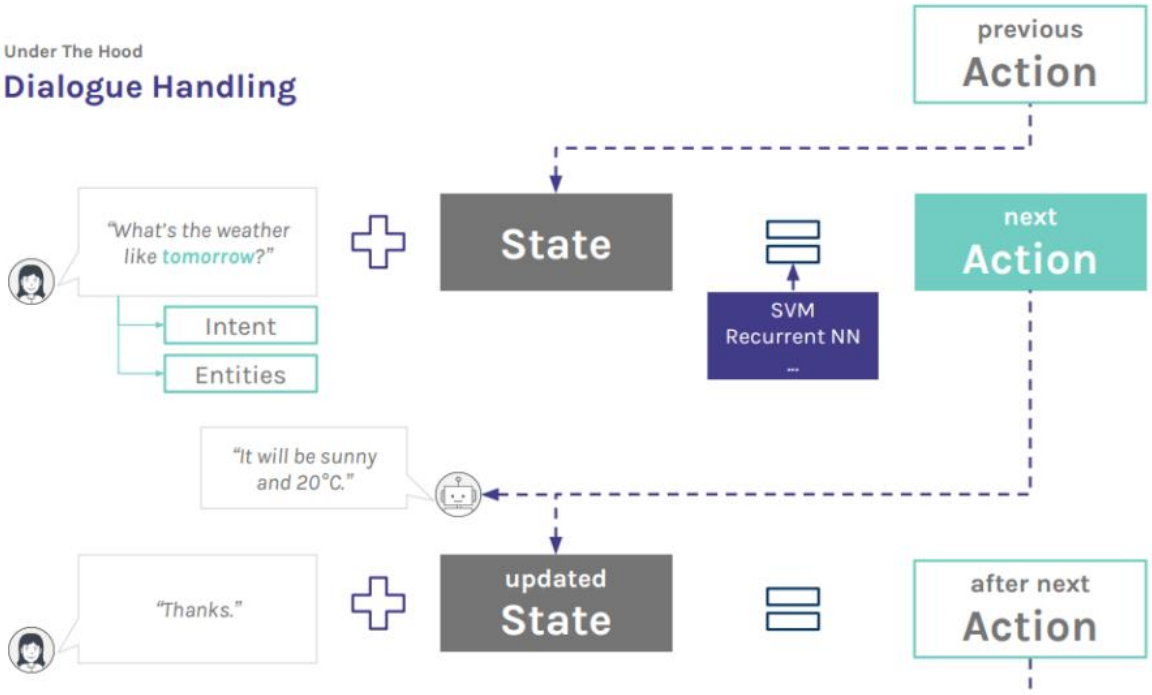
- Vị trí (Location)
- Thời gian (Datetime)
- Số (Number)
- Địa chỉ liên lạc (Contact)
- Khoảng cách (Distance)
- Khoảng thời gian (Duration)
- …

Các kỹ thuật thường được sử dụng:

- Hidden Markov Model
- Maximum Entropy
- Conditional Random Fields – CRFs…

2.1.3 Quản lý hội thoại

Trong các phiên trao đổi dài (long conversation) giữa người và chatbot, chatbot sẽ cần ghi nhớ những thông tin về ngữ cảnh (context) hay quản lý các trạng thái hội thoại (dialog state). Vấn đề quản lý hội thoại (dialoge management) khi đó là quan trọng để đảm bảo việc trao đổi giữa người và máy là thông suốt. Vẫn ở ví dụ về hỏi về thời tiết ở trên, chatbot cần có thông tin về trạng thái hội thoại, để có thể phản hồi đúng thông tin cần hỏi "It will be sunny and $20^{o}C$".



Hình 2.4: Quản lý hội thoại

Chức năng của thành phần quản lý hội thoại là nhận đầu vào từ thành phần NLU, quản lý các trạng thái hội thoại (dialogue state), ngữ cảnh hội thoại (dialogue context), và truyền đầu ra cho thành phần sinh ngôn ngữ (Natural Language Generation - NLG).

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

Hiện nay, các chatbot thường dùng mô hình máy trạng thái hữu hạn (Finite State Machines – FSM), mô hình Frame-based (Slot Filling) hoặc kết hợp hai mô hình này.

2.1.4 Mô hình sinh hội thoại cho chatbot

Các chatbot mô hình dựa trên quy tắc (rule-based): Đây là là kiểu kiến trúc mà hầu hết các chatbot đầu tiên đã được xây dựng. Chúng chọn câu trả lời của hệ thống dựa trên một tập hợp các quy tắc được xác định trước cố định, dựa trên việc nhận ra dạng từ vựng của văn bản đầu vào mà không tạo bất kỳ câu trả lời văn bản mới nào. Kiến thức được sử dụng trong chatbot được con người viết mã bằng tay (hard-coded) và được sắp xếp và trình bày bằng các mẫu hội thoại. Cơ sở dữ liệu quy tắc toàn diện cho phép chatbot trả lời nhiều loại đầu vào của người dùng hơn. Tuy nhiên, loại mô hình này không mạnh đối với các lỗi chính tả và ngữ pháp trong đầu vào của người dùng.

Mô hình dựa trên truy xuất (retrieval-based): Một chút khác biệt so với mô hình dựa trên quy tắc là mô hình dựa trên truy xuất (retrieval-based), cung cấp tính linh hoạt hơn vì nó truy vấn và phân tích các tài nguyên có sẵn bằng cách sử dụng các API. Một chatbot dựa trên truy xuất lấy một số lựa chọn phản hồi từ một chỉ mục trước khi nó áp dụng phương pháp matching cho lựa chọn phản hồi.

Mô hình sáng tạo (generative-based): tạo ra câu trả lời theo cách tốt hơn so các mô hình còn lại, dựa trên các tin nhắn hội thoại của người dùng hiện tại và trước đó. Các chatbot này giống con người hơn và sử dụng các thuật toán máy học (machine learning) hoặc kỹ thuật học sâu (deep learning) nên linh hoạt hơn. NLG nâng cao cho phép dự đoán khả năng xuất hiện từ này đến từ khác và sửa các lỗi ngôn ngữ, chẳng hạn như lỗi chính tả. Các thuật toán được sử dụng trong NLG nâng cao cũng tốt hơn trong việc xử lý các từ và biểu thức mới không có trong các mẫu đào tạo ban đầu.

Phương pháp dựa trên AI này dựa trên một công cụ NLP nâng cao để hỗ trợ ngôn ngữ tự nhiên và đáp ứng yêu cầu dựa trên các thuật toán ML và tích hợp hệ thống để truy xuất thông tin động. Độ chính xác của chatbot thấp hơn khi bắt đầu và tăng lên theo thời gian.

2.2. Rasa framework

2.2.1. Giới thiệu

Rasa Open source là một nền tảng học máy để tạo ra các trợ lý ảo dựa trên văn bản và giọng nói. Tính đến 8/2020, Rasa đã được download hơn 3 triệu lần, có cộng đồng diễn đàn hơn 10.000 thành viên và trên 450 người đóng góp vào mã nguồn. Các chức năng chính của Rasa:

Rasa Open Source là một nền tảng có khả năng hiểu ngôn ngữ tự nhiên, quản lý đối thoại và tích hợp. Rasa X là bộ công cụ miễn phí được sử dụng để cải thiện các trợ lý theo ngữ cảnh được xây dựng bằng Rasa Open Source. Cùng với nhau, chúng bao gồm tất cả các tính năng để tạo ra các trợ lý và chatbot dựa trên văn bản và giọng nói.

- Hiểu thông điệp (Understand messages): biến văn bản dạng tự do ở bất kỳ ngôn ngữ nào thành dữ liệu có cấu trúc. Hỗ trợ đơn và đa ý định (multiple intents) và cả các thực thể được đào tạo trước và tùy chỉnh (pre-trained and custom entities).

- Duy trì cuộc trò chuyện (Hold conversations): ghi nhớ ngữ cảnh bằng cách sử dụng quản lý hội thoại dựa trên máy học.

- Học tập tương tác (Interactive learning): tạo dữ liệu đào tạo bằng cách nói chuyện với chatbot của bạn và cung cấp phản hồi khi nó mắc lỗi.

- Kết nối với các nền tảng nhắn tin thường dùng (Connect): tích hợp chatbot của bạn trên Slack, Facebook, Google Home, …

- Tích hợp các lệnh gọi API (Integrate): sử dụng các hành động tùy chỉnh của Rasa để tương tác với các API và các hệ thống khác.

- Xem và chú thích cuộc hội thoại (View and annotate conversations): lọc, gắn cờ và sửa các cuộc trò chuyện để liên tục cải thiện chatbot của bạn.

Rasa có đầy đủ các thành phần cơ bản của hệ thống chatbot bao gồm: NLU (hiểu ngôn ngữ tự nhiên), Dialogue Management (Quản lý hội thoại) và NLG (sinh ngôn ngữ tự nhiên).

2.2.2. Cấu trúc chương trình của Rasa

Cấu trúc của một chương trình của Rasa như hình dưới:

Các thành phần chính trong chương trình được diễn giải như sau:

__init__.py	một file trống giúp python tìm thấy hành động của chatbot
actions.py	mã cho các hành động tùy chỉnh của chatbot
config.yml '*'	cấu hình NLU và các mô hình Core của chatbot
credentials.yml	chi tiết để kết nối với các dịch vụ khác
data/nlu.md '*'	dữ liệu đào tạo NLU của chatbot
data/stories.md '*'	các stories
domain.yml '*'	miền của chatbot
endpoints.yml	chi tiết để kết nối với các kênh như fb messenger
models/<timestamp>.tar.gz	mô hình(model) ban đầu của bạn

2.2.3. Intent

Trong Rasa, thực hiện khai báo các Intent (ý định người dùng) trong domain.yml. Chẳng hạn một số ý định của người dùng tương tác với chatbot như chào hỏi, tạm biệt, từ chối, đặt phòng, …

2.2.4. Entity

Trong Rasa, thực hiện khai báo các Entity (thực thể) trong domain.yml. Chẳng hạn một số thực thể như số, vị trí, kiểu phòng.

2.2.5. Stories

Rasa stories là một dạng dữ liệu đào tạo được sử dụng để đào tạo các mô hình quản lý hội thoại của Rasa.

Một story là sự trình bày cuộc trò chuyện giữa người dùng và trợ lý AI, được chuyển đổi thành một định dạng cụ thể trong đó thông tin đầu vào của người dùng được thể hiện dưới dạng ý định (intents) tương ứng (và các thực thể nếu cần) trong khi phản hồi của chatbot được thể hiện dưới dạng tên hành động (action) tương ứng.

Một ví dụ đào tạo cho hệ thống đối thoại Rasa Core được gọi là một câu chuyện. Đây là ví dụ về một đoạn hội thoại ở định dạng câu chuyện Rasa:

2.2.6. Actions

Trong khi viết các story, ta sẽ gặp hai loại hành động: hành động phát biểu (utterance actions) và hành động tùy chỉnh (custom actions). Hành động phát biểu là các thông điệp được cố định (hardcoded) mà bot có thể phản hồi. Trong khi đó, các hành động tùy chỉnh liên quan đến mã tùy chỉnh (custom code) được thực thi.

2.2.7. Policies

Trong file config.yml có khóa policies mà ta có thể sử dụng để tùy chỉnh các chính sách mà chatbot của mình sử dụng.

2.2.8. Slots

Slots là bộ nhớ của chatbot. Chúng hoạt động như một kho lưu trữ khóa-giá trị (key-value) có thể được sử dụng để lưu trữ thông tin mà người dùng cung cấp (ví dụ: thành phố của họ) cũng như thông tin thu thập được về thế giới bên ngoài (ví dụ: kết quả của một truy vấn cơ sở dữ liệu).

2.3. Kết luận chương

Trong chương này luận văn đã giới thiệu các kỹ thuật quan trọng nhất được sử dụng trong chatbot, ngoài ra cũng đề cập đến các thành phần cơ bản của Rasa framework. Đây là các cơ sở để áp dụng xây dựng bài toán chatbot trả lời thông tin khách sạn.

CHƯƠNG 3: XÂY DỰNG CÔNG CỤ HỎI ĐÁP THÔNG TIN KHÁCH SẠN

3.1. Giới thiệu bài toán

Trong chương này tác giả lựa chọn bài toán trả lời thông tin khách sạn nhằm cung cấp thông tin, tư vấn và hỗ trợ bán hàng cho khách sạn, với khả năng hoạt động liên tục 24/7, hỗ trợ con người trong việc trả lời các câu hỏi liên quan đến khách sạn. Trên cơ sở nghiên cứu các câu hỏi thường gặp, bài toán tập trung vào một số chức năng chính của chatbot như sau:

- Chào hỏi
- Tạm biệt
- Thông tin về các loại phòng khách sạn
- Thông tin về các câu hỏi (FAQ) trong khách sạn
- Thông tin chung của khách sạn
- Đặt phòng

3.1.1. Mô hình huấn luyện cho chatbot

Trong Rasa, các messages được xử lý bởi một chuỗi các thành phần (components). Các thành phần này được thực thi lần lượt trong "pipeline" được xác định trong file config.yml. Việc lựa chọn một NLU pipeline cho phép ta tùy chỉnh mô hình của mình và kết hợp nó trên tập dữ liệu của mình.

Một pipeline thường bao gồm ba phần chính:

- Tokenization:

Tách mỗi câu thành một danh sách các từ tố (token), mỗi câu được tách ra thành một danh sách các từ có nghĩa.

- Featurization:

Ta cần quyết định xem có nên sử dụng các thành phần cung cấp tính năng nhúng từ được đào tạo trước (pre-trained word embeddings) hay nhúng được giám sát (Supervising Embeddings).

+ Pre-trained Embeddings: phân loại ý định người dùng sẽ dựa trên các tập dữ liệu được lọc trước, sau đó được sử dụng để thể hiện từng từ trong thông điệp người dùng dưới dạng từ nhúng hay biểu diễn ngôn ngữ dưới dạng vector.

+ Supervised Embeddings: Với phương pháp nhúng được giám sát này thì ta sẽ tự tạo tập dữ liệu training riêng của mình từ đầu. Với việc khó tìm ra được mô hình đào tạo trước cho ngôn ngữ tiếng Việt, cùng với bài toán trong một miền lĩnh vực đóng như trả lời thông tin tin khách sạn thì nó sẽ đảm bảo tính chính xác hơn nhiều và tránh dư thừa dữ liệu. Do đó, ở đây tác giả lựa chọn phương pháp này.

- Entity Recognition / Intent Classification / Response Selectors:

Tùy thuộc vào dữ liệu, ta có thể chỉ muốn thực hiện phân loại ý định, nhận dạng thực thể hoặc lựa chọn phản hồi. Hoặc ta có thể muốn kết hợp nhiều nhiệm vụ đó. Rasa hỗ trợ một số thành phần cho mỗi nhiệm vụ. Ở đây tác giả lựa chọn các thành phần như sau:

DIETClassifier: DIET (Dual Intent và Entity Transformer) là một kiến trúc đa tác vụ để phân loại ý định và nhận dạng thực thể. Kiến trúc dựa trên một bộ chuyển đổi được chia sẻ cho cả hai nhiệm vụ. Một chuỗi các nhãn thực thể được dự đoán thông qua một lớp gắn thẻ trường ngẫu nhiên có điều kiện (Conditional Random Field - CRF) tương ứng với chuỗi đầu vào của tokens.

ResponseSelector: Thành phần này có thể được sử dụng để xây dựng mô hình truy xuất phản hồi nhằm dự đoán trực tiếp phản hồi của bot từ một tập hợp các phản hồi.

3.1.2. Đánh giá hiệu quả của chatbot

Dưới đây là một số phương pháp đánh giá các mô hình phân loại của Rasa.

+ Accuracy

+ Confusion matrix

+ True/False Positive/Negative

+ Precision / Recall

+ F1- Score

3.2. Xây dựng Chương trình

3.2.1. Nguồn dữ liệu xây dựng

Nguồn dữ liệu thực nghiệm, tác giả đã thu thập chính từ bộ câu hỏi, câu trả lời, là những câu hỏi thường gặp của một số khách sạn, ngoài ra có tham khảo thêm một số chatbot facebook fanpage.

Để làm giàu cho tập dữ liệu huấn luyện, tác giả bổ sung thêm các câu hỏi mới trong mỗi Intent, đảm bảo các Intent quan trọng mang tính hỏi đáp nhiều có ít nhất 10 câu hỏi, các intent khác cũng có tối thiểu 5 câu hỏi.

3.2.2. Xây dựng ý định

Nhiệm vụ xây dựng các tập ý định sẽ theo nguyên tắc là những mẫu câu hỏi mà người dùng khi có ý định đó hay sử dụng nhất có thể. Cần định nghĩa các ý định khớp với ngôn ngữ tự nhiên nhất, trong hệ thống này tác giả định nghĩa 50 ý định với nguồn dữ liệu thực nghiệm.

3.2.3. Xây dựng thực thể

Entities là các thực thể thông tin đặc trưng quan trọng được trích xuất theo các ý định người dùng. Slots là các thông tin được trích chọn trong câu nói của người dùng được hệ thống lưu lại trong bộ nhớ hệ thông để dùng trong các hành động hoặc để đưa ra các câu trở lời phù hợp theo ngữ cảnh, tránh việc phải hỏi lại những thông tin mà người dùng đã cung cấp từ trước. Tác giả xây dựng 2 thực thể về vị trí và tên loại phòng khách sạn.

3.2.4. Xây dựng câu trả lời

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

Để tạo tính tự nhiên và phù hợp với độ tuổi trong cuộc hội thoại thì ta có thể xây dựng nhiều tập mẫu câu trả lời để Chatbot lựa phù hợp với lứa tuổi, giới tính khi có được thông tin về khách hàng hoặc nếu không thì sẽ chọn ngẫu nhiên các mẫu câu trả lời để tạo cảm giác không bị nhàm chán. Đối với từng ý định tác giả xây dựng 1-2 câu trả lời dựa vào bộ dữ liệu mẫu.

3.2.5. Xây dựng khung kịch bản

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

3.2.6. Đào tạo cho chatbot

Tiến hành train cho một model sử dụng NLU data và các kịch bản (stories), mô hình được đào tạo sẽ được lưu dưới dạng thư mục /models.

3.2.7. Kiểm tra chatbot

Rasa Open Source cho phép ta kiểm tra các cuộc hội thoại từ đầu đến cuối bằng cách chạy các cuộc hội thoại kiểm tra để đảm bảo rằng cả NLU và Core đều đưa ra dự đoán chính xác.

Để làm điều này, ta cần một số stories ở định dạng end-to-end, bao gồm cả đầu ra của NLU và văn bản gốc. Theo mặc định, Rasa lưu các bài hội thoại kiểm tra vào

tests/conversation_tests.md. Ta có thể kiểm tra chatbot của mình bằng cách chạy lệnh: $ rasa test

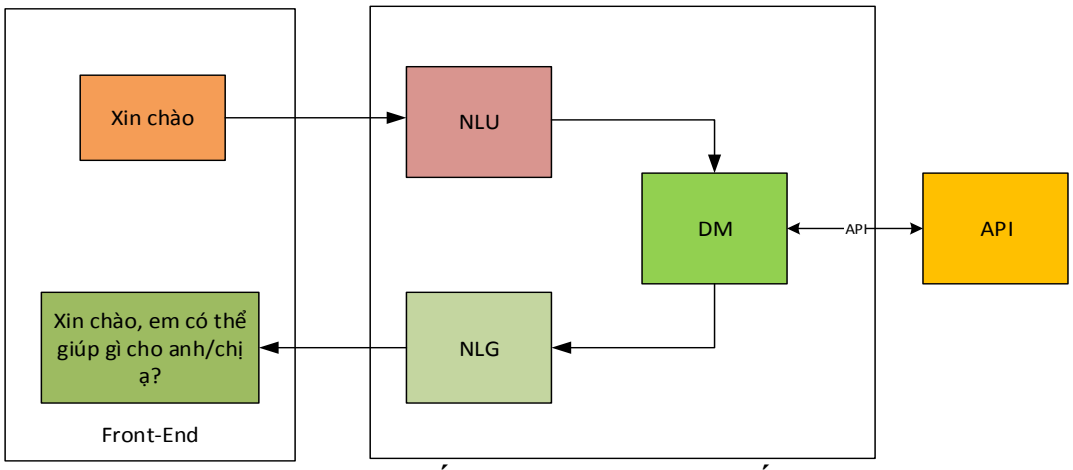
3.3. Kết quả thực nghiệm

3.3.1. Môi trường thực nghiệm

Chương trình thử nghiệm được thiết kế, xây dựng và thực hiện trên môi trường hệ điều hành Windows với nền tảng framework Rasa 1.10.12, dựa trên ngôn ngữ lập trình python 3.7. Giao điện người dùng sử dụng nền tảng web/ứng dụng chat Telegram.

3.3.2. Thiết kế

Hình dưới đây minh hoạ kiến trúc chung của bài toán.



Hình 3.1: Kiến trúc chung của hệ thống

Front-end: sử dụng giao diện web hoặc các trình nhắn tin phổ biến (Facebook Messenger/Telegram…). Với mục tiêu minh họa, ở đây sử dụng một giao diện web và chat Telegram.

- Mỗi khi có một người dùng gửi tin nhắn cho chatbot thì nội dung tin này sẽ gửi một POST request đến webhook được sử dụng để lắng nghe sự kiện.

- RASA nhận diện ý định. Sau khi đã thu được message của người dùng thì sử dụng RASA để hiểu được ý định của người dùng cùng các thông tin thực thể.

- Thông tin này tiếp tục được chuyển đến DM của Rasa, tại đây tùy theo ý định và thông tin thực thể cùng với các thông tin theo dõi của cuộc trò chuyện đã xảy ra cho đến nay, để dự đoán một phản ứng thích hợp.

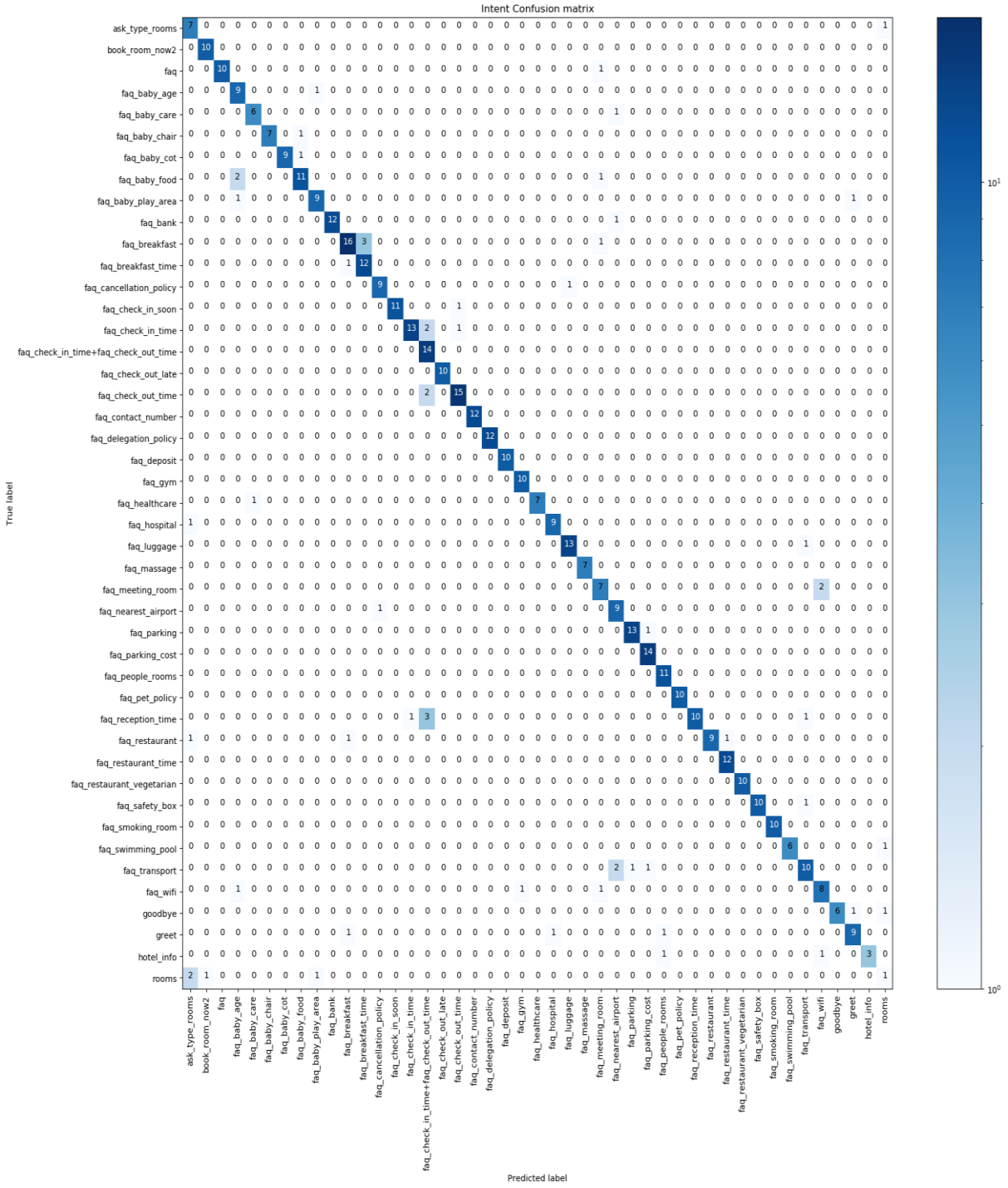
- NLG sinh ra câu trả lời dựa vào dữ liệu từ thành phần DM theo các mẫu câu template đã được xây dựng trước hoặc là kết quả của API.

- Gửi tin nhắn qua phản hồi trả về cho người dùng.

3.3.3. Kết quả thực nghiệm

a) Kết quả thử nghiệm

Kết quả đánh giá NLU model và Rasa Core cho kết quả như dưới đây.



Hình 3.2: Intent Confusion matrix

Bảng 3.1: đánh giá trích chọn thông tin thực thể (entity)

	precision	recall	f1-score	support
room_type	0.85	0.75	0.79	8
location	1.00	0.58	0.74	17

Bảng 3.2: đánh giá mô hình Rasa Core

	precision	recall	f1-score	support
utter_faq_restaurant_time	1.00	0.64	0.78	11
utter_faq_safety_box	1.00	0.75	0.86	4
utter_faq_bank	1.00	1.00	1.00	8
utter_rooms	0.80	0.80	0.80	5
utter_faq_contact_number	1.00	1.00	1.00	10
utter_faq_deposit	1.00	0.67	0.80	9
utter_ask_type_rooms	0.86	1.00	0.92	6
utter_faq_hospital	1.00	1.00	1.00	9
utter_faq_baby_food	1.00	1.00	1.00	14
utter_faq_check_out_late	1.00	0.78	0.88	9
utter_faq_transport	1.00	0.83	0.91	6
utter_faq_restaurant	1.00	0.75	0.86	8
utter_faq_reception_time	0.70	1.00	0.82	14
utter_faq_pet_policy	1.00	1.00	1.00	8
utter_deluxe_details	1.00	1.00	1.00	2
utter_faq_check_in_time	0.71	0.94	0.81	16
utter_faq_massage	1.00	1.00	1.00	13
utter_faq_healthcare	1.00	1.00	1.00	5
utter_hotel_info	0.92	1.00	0.96	11
utter_faq_luggage	1.00	0.90	0.95	10
utter_faq_smoking_room	1.00	1.00	1.00	8
book_room_form2	1.00	0.95	0.97	37
utter_greet	1.00	1.00	1.00	7
utter_faq_baby_age	1.00	0.91	0.95	11
utter_faq_nearest_airport	1.00	1.00	1.00	5
utter_faq_meeting_room	1.00	0.88	0.93	8
utter_faq_baby_chair	1.00	1.00	1.00	8
utter_faq_people_rooms	1.00	1.00	1.00	4
utter_faq_baby_play_area	0.89	1.00	0.94	8
action_listen	0.98	1.00	0.99	401
utter_faq_delegation_policy	1.00	1.00	1.00	8
utter_faq_breakfast_time	0.78	0.64	0.70	11

utter_faq_check_in_soon	1.00	0.57	0.73	14
utter_faq_swimming_pool	1.00	1.00	1.00	5
utter_faq_cancellation_policy	1.00	1.00	1.00	9
utter_suite_details	1.00	1.00	1.00	2
utter_faq_baby_cot	1.00	1.00	1.00	8
utter_family_details	1.00	0.50	0.67	8
utter_faq_parking	1.00	0.80	0.89	10
utter_faq_check_out_time	0.62	0.59	0.61	17
utter_faq_parking_cost	0.73	1.00	0.85	11
utter_faq_breakfast	0.89	1.00	0.94	8
utter_faq_restaurant_vegetarian	1.00	1.00	1.00	7

Tính chung, kết quả test trên tập dữ liệu test end-to-end cho độ chính xác khoảng 89%.

Correct: 731 / 814

F1-Score: 0.906

Precision: 0.926

Accuracy: 0.899

b) Test trên giao diện người dùng cuối

Thực hiện thử nghiệm tương tác với Chatbot với một số câu câu hỏi gần với kịch bản đã đào tạo cho Chatbot.

3.4. Đánh giá

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

- Xác định đúng được ý định (intent) có ý nghĩa quan trọng nhất đối với chatbot. Đối với bài toán trong miền đóng cần xác định rõ ràng các intent, xây dựng tập dữ liệu đủ lớn, gán nhãn và tiến hành training.

- Xây dựng dữ liệu đào tạo, training cho chatbot với các kịch bản là rất cần thiết để cho độ chính xác cao của chatbot.

- Chatbot ứng dụng AI có khả năng đáp ứng tốt với các kịch bản dựng sẵn, và được đào tạo. Đối với các kịch bản nằm ngoài kịch bản dựng sẵn, có thể tăng cường khả năng cho chatbot bằng cách điều hướng người dùng về các câu mặc định hoặc các dạng giao diện menu lựa chọn.

- Việc xác định và phản hồi đa ý định có thể thực hiện bằng việc kết hợp các ý định.

- Qua bài toán thực nghiệm có thể thấy rằng áp dụng bài toán Chatbot cho việc hỗ trợ trả lời thông tin khách sạn là khả thi, có tính thực tiễn cao, và hoàn toàn áp dụng được ngay trong thực tiễn.

3.5. Kết luận chương

Trong chương này luận văn đã mô tả các bước chính trong xây dựng chương trình thực nghiệm từ việc lựa chọn nguồn dữ liệu, xây dựng ý định, thực thể và các kịch bản trả lời. Kết quả của việc huấn luyện và test dữ liệu cho kết quả khá cao trên tập dữ liệu huấn luyện và test. Huấn luyện càng nhiều dữ liệu thì độ chính xác sẽ càng cao hơn.

KẾT LUẬN

Đề tài thực hiện nghiên cứu một số kiến thức bao gồm kiến trúc và nhiệm vụ các thành phần chatbot, một số thuật toán cơ bản áp dụng vào việc xây dựng chatbot để giải quyết các bài toán theo hướng tiếp cận miền đóng, cụ thể là lĩnh vực khách sạn du lịch. Dựa vào đó ta có thể áp dụng xây dựng chatbot giải quyết các bài toán hỗ trợ người dùng trong nhiều lĩnh vực thực tế.

Các vấn đề mà luận văn đã đạt được:

- Nghiên cứu và tìm hiểu, trình bày một cách khái quát nhất về hệ thống chatbot, các thành phần, kỹ thuật được áp dụng trong chatbot.

- Tìm hiểu lựa chọn mô hình chatbot phù hợp, từ đó có thể quyết định xây dựng bot theo mô hình nào, phương pháp nào phù hợp hơn cho từng yêu cầu bài toán. Cụ thể với bài toán trả lời thông tin khách sạn sử dụng phương pháp tiếp cận dạng chat văn bản (text-based), trong miền đóng (closed-domain), hướng mục tiêu (task- oriented), thiết kế dựa trên AI.

- Khảo sát, nghiên cứu, xây dựng chatbot thực nghiệm xử lý ngôn ngữ tự nhiên (NLP) bằng tiếng Việt sử dụng Rasa framework. Thử nghiệm và đánh giá mô hình RASA trên bộ dữ liệu xây dựng cho kết quả độ chính xác khá cao (khoảng 89%). Chatbot hỗ trợ tương đối nhiều ý định (50 intents) và cũng đã có kịch bản minh họa hỗ trợ đa ý định (multi-intent).

- Cài đặt và triển khai ứng dụng trên môi trường hệ điều hành Windows với nền tảng Rasa framework 1.10.12, ngôn ngữ lập trình python 3.7, sử dụng chat client dạng web hoặc telegram. Sản phẩm demo có được sẽ làm tiền đề cho việc phát triển, hoàn thiện sản phẩm trong thời gian tới. Sản phẩm cũng cho thấy khả năng xây dựng chatbot ứng dụng AI trong các lĩnh vực khác là hoàn toàn khả thi.

Một số hạn chế:

- Chưa xây dựng được đa dạng hóa câu trả lời

- Chatbot mới chỉ hỗ trợ dạng hội thoại text, chưa hỗ trợ voice.

- Chưa có khả năng trả lời các câu hỏi phức tạp.

- Chưa xây dựng được giao diện quản trị để tự tổ chức xây dựng intent, entity, kịch bản… nhằm mục đích cung cấp cho nhiều khách sạn khác nhau.

Định hướng nghiên cứu tiếp theo:

- Xây dựng thêm đa dạng hóa các câu trả lời ngẫu nhiên theo các ý định

- Tích hợp speech to text và text to speech cho chatbot để hỗ trợ voice.

- Xây dựng bot có khả năng trả lời các câu hỏi phức tạp hơn.

- Xây dựng giao diện quản trị tự tổ chức và quản lý intent, entity, kịch bản… để hỗ trợ cho các khách sạn tự xây dựng chatbot của riêng mình mà không cần hiểu nhiều về lập trình.