**TRUNG TÂM NGHIÊN CỨU VÀ PHÁT TRIỂN ĐIỆN THOẠI DI ĐỘNG SAMSUNG VIỆT NAM SVMC**

****

**Đỗ Văn Quang**

CHATBOT TƯ VẤN TUYỂN SINH ĐẠI HỌC UNISEC-CHATBOT

**BÁO CÁO KẾT THÚC KỲ THỰC TẬP**

**Mentor: Nguyễn Thị Thanh Dung**

**Cao Trang**

**TRUNG TÂM NGHIÊN CỨU VÀ PHÁT TRIỂN ĐIỆN THOẠI DI ĐỘNG SAMSUNG VIỆT NAM SVMC**

**Đỗ Văn Quang**

**CHATBOT TƯ VẤN TUYỂN SINH ĐẠI HỌC UNISEC-CHATBOT**

**BÁO CÁO KẾT THÚC KỲ THỰC TẬP**

**Mentor: Nguyễn Thị Thanh Dung**

**Cao Trang**

# TÓM TẮT

**Tóm tắt:** 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%. Chatbot là một hình thức thô sơ của phần mềm trí tuệ nhân tạo, là một chương trình được tạo ra từ máy tính tiến hành cuộc trò chuyện thông qua các phương pháp nhập văn bản, âm thanh, cảm ứng có thể tiến hành trả lời các câu hỏi và xử lý các tình huống là một công cụ có thể giao tiếp với con người thông qua các chương trình trí tuệ nhân tạo được lập trình sẵn.

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 giáo dục chưa được sử dụng nhiều dù rằng đây là lĩnh vực 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 người dùng 24/7. 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 lựa chọn đề tài: “ Phát triển hệ thống chatbot hỗ trợ học sinh lựa chọn trường đại học Unisec-chatbot” để phần nào hỗ trợ học sinh lựa chọn trường đại học một cách tốt nhất và dễ dàng nhất. Unisec-chatbot được xây dựng với mục đích giúp những học sinh tìm hiểu về các trường đại học trong cả nước. Ứng dụng đưa ra các lời khuyên giúp học sinh lựa chọn ngành học phù hợp với nhu cầu của họ. Nó sẽ gợi ý để họ lựa chọn các phương án tốt nhất như trường đại học, chuyên ngành, địa điểm…

**Từ khóa :** Trí tuệ nhân tạo, chatbot

**MỤC LỤC**

[TÓM TẮT 1](#_Toc73888111)

[Danh mục ký hiệu và các chữ viết tắt 4](#_Toc73888112)

[Danh mục hình vẽ 5](#_Toc73888113)

[Danh mục bảng biểu 6](#_Toc73888114)

[Chương 1. Giới thiệu đề tài 7](#_Toc73888115)

[1.1. Đặt vấn đề 7](#_Toc73888116)

[1.2 . Nội dung của khóa luận 8](#_Toc73888117)

[1.3. Mục tiêu và đóng góp của khóa luận 8](#_Toc73888118)

[1.4. Cấu trúc khóa luận 8](#_Toc73888119)

[Chương 2. Cơ sở lý thuyết 10](#_Toc73888120)

[2.1. Giới thiệu về trí tuệ nhân tạo (AI – Artificial Intelligence) 10](#_Toc73888121)

[2.2. Chatbots 10](#_Toc73888122)

[2.2.1. Khái niệm 10](#_Toc73888123)

[2.2.2. Lịch sử ra đời 11](#_Toc73888124)

[2.2.3. Các thành phần cơ bản của hệ thống chatbot 14](#_Toc73888125)

[2.2.4. Phân loại chatbot 16](#_Toc73888126)

[2.2.5. Một số nền tảng phát triển chatbot 17](#_Toc73888127)

[2.2.6. Xu hướng phát triển 18](#_Toc73888128)

[2.3. Rasa framework 19](#_Toc73888129)

[2.3.1. Giới thiệu 19](#_Toc73888130)

[2.3.2. Cấu trúc chương trình của Rasa 22](#_Toc73888131)

[2.3.3. Ý định người dùng (intent) 24](#_Toc73888132)

[2.3.4. Thực thể (entity) 25](#_Toc73888133)

[2.3.5. Khung kịch bản (stories) 25](#_Toc73888134)

[2.3.6. Hành động (action) 26](#_Toc73888135)

[2.3.7. Chính sách (policies) 26](#_Toc73888136)

[2.3.8. Vị trí (slots) 27](#_Toc73888137)

[Chương 3: Xây dựng và phát triển hệ thống 28](#_Toc73888138)

[3.1. Giới thiệu bài toán 28](#_Toc73888139)

[3.1.1. Mô hình huấn luyện cho chatbot 28](#_Toc73888140)

[3.1.2. Đánh giá hiệu quả của chatbot 30](#_Toc73888141)

[3.2. Xây dựng chương trình 32](#_Toc73888142)

[3.2.1. Nguồn dữ liệu xây dựng 32](#_Toc73888143)

[3.2.2. Xây dựng ý đinh 33](#_Toc73888144)

[3.2.3. Xây dựng thực thể 34](#_Toc73888145)

[3.2.4. Xây dựng câu trả lời 36](#_Toc73888146)

[3.2.5. Xây dựng khung kịch bản (stories) 37](#_Toc73888147)

[3.3. Kết quả thực nghiệm 41](#_Toc73888148)

[3.3.1. Môi trường thực nghiệm 41](#_Toc73888149)

[3.3.2. Thiết kế 41](#_Toc73888150)

[3.3.3. Kết quả thực nghiệm 42](#_Toc73888151)

[3.4. Đánh giá 53](#_Toc73888152)

[Chương 4. Kết luận và hướng phát triển 55](#_Toc73888153)

[4.1. Kết luận 55](#_Toc73888154)

[4.2. Hướng phát triển 55](#_Toc73888155)

[Danh mục tài liệu tham khảo 57](#_Toc73888156)

# Danh mục ký hiệu và các chữ viết tắt

|  |  |  |
| --- | --- | --- |
| **Từ viết**  **tắt** | **Từ chuẩn** | **Diễn giải** |
| AI | Artificial Intelligence | Trí tuệ nhân tạo |
| API | Application Programming  Interface | Giao diện lập trình ứng dụng |
| CRF | Conditional Random Fields | Mô hình CRF |
| DL | Deep learning | Học sâu |
| DM | Dialog Management | Quản lý hội thoại |
| NLG | Natural language generation | Sinh ngôn ngữ tự nhiên |
| NLP | Natural language processing | Xử lý ngôn ngữ tự nhiên |
| NLU | Natural language understanding | Hiểu ngôn ngữ tự nhiên |

# Danh mục hình vẽ

[Hình 2.1: Tương tác giữa người dùng với chatbot Eliza 12](#_Toc73904969)

[Hình 2.2: Kiến trúc chung của chatbot 14](#_Toc73904970)

[Hình 2.3: Phân loại Chatbot 16](#_Toc73904971)

[Hình 2.4: Các thành phần của Rasa 20](#_Toc73904972)

[Hình 2.5: Chế độ học tương tác của Rasa 21](#_Toc73904973)

[Hình 2.6: Công cụ Rasa X 21](#_Toc73904974)

[Hình 2.7: Cách thức Rasa phản hồi một tin nhắn 22](#_Toc73904975)

[Hình 2.8: Cấu trúc của một chương trình Rasa 23](#_Toc73904976)

[Hình 2.9: Các ý định được khai báo trong domain.yml 24](#_Toc73904977)

[Hình 3.1: Đào tạo một mô hình 39](#_Toc73904979)

[Hình 3.2: Đào tạo cho chatbot dạng shell 39](#_Toc73904981)

[Hình 3.3: Chế độ đào tạo cho chatbot bằng Interactive Learning 40](#_Toc73904982)

[Hình 3.4: Trực quan hóa cuộc hội thoại 41](#_Toc73904984)

[Hình 3.5: Kiến trúc chung của hệ thống 41](#_Toc73904985)

[Hình 3.6: Ma trận hỗn loạn ý định người dùng 43](#_Toc73904986)

[Hình 3.7: Đánh giá mô hình Rasa Core 44](#_Toc73904987)

[Hình 3.8: Hỏi về mã của các trường đại học 45](#_Toc73904988)

[Hình 3.9: Hỏi về điểm chuẩn của các trường đại học 46](#_Toc73904990)

[Hình 3.10: Hỏi về danh sách các ngành của trường đại học 47](#_Toc73904993)

[Hình 3.11: Hỏi về khối xét tuyển của ngành của trường đại học 48](#_Toc73904994)

[Hình 3.12: Tìm kiếm các trường đại học phù hợp 49](#_Toc73904995)

[Hình 3.13: Hỏi về thông tin của các trường đại học 50](#_Toc73904996)

[Hình 3.14: Giới thiệu ngành học 51](#_Toc73904997)

[Hình 3.15: Tư vấn chọn ngành phù hợp 52](#_Toc73904998)

[Hình 3.16: Hỏi về điểm cộng của các khu vực 53](#_Toc73904999)

# Danh mục bảng biểu

[Bảng 3.1: Bảng confusion matrix 33](#_Toc69591781)

[Bảng 3.2: Bảng các ý định (intent) của chatbot 35](#_Toc69591782)

# Chương 1. Giới thiệu đề tài

## Đặt vấn đề

Công nghệ thông tin đang phát triển với tốc độ chóng mặt và đang được ứng dụng trong ngày càng nhiều lĩnh vực của đời sống. Cùng với sự phát triển đó thì tự động hóa đang ngày càng trở nên cần thiết đối với mỗi cá nhân hay tổ chức. Bởi vì mỗi cá nhân hay tổ chức luôn tìm kiếm những phương tiện hỗ trợ mới, có thể thực hiện công việc một một cách nhanh chóng và đơn giản nhất. Đó là lý do tại sao các chatbot AI xuất hiện. Chatbot nổi lên như một chủ đề nóng trong những năm gần đây và nó được nhiều công ty sử dụng trong nhiều lĩnh vực khác như: Công cụ hỗ trợ dịch vụ, hệ thống trả lời điện thoại tự động, thư điện tử… Mặc dù công nghệ này đã phát triển từ những năm 60 nhưng với sự tiến bộ của những năm gần đây trong các ứng dụng nhắn tin và công nghệ AI nên nó đã trở thành mối quan tâm hàng đầu.

Chatbot phát triển bằng cách phân tích và xác định mục đích yêu cầu của người dùng để trích xuất được các thực thể từ các kịch bản có trước và tự học trong quá trình tương tác. Người dùng sẽ tương tác với chatbot thông qua các nền tảng nhắn tin, bằng cách nhập văn bản, âm thanh, cảm ứng …Với các thông tin nhận được từ phía người dùng chatbot sẽ nhóm văn bản và trích xuất mẫu thông tin, sử dụng hệ thống hiểu ngôn ngữ tự nhiên (Natural Language Understanding – NLU) để hiểu thông tin người dùng cung cấp và chuyển đổi văn bản thành dữ liệu có cấu trúc để máy có thể hiểu được. Sau đó sử dụng các thuật toán học máy để đưa ra các loại phản hồi khác nhau, chúng sẽ dự đoán và phản hồi một cách chính xác nhất có thể. Một trong những yếu tố làm nên sức mạnh của Chatbot là khả năng tự học hỏi. Càng nhận được nhiều thông tin từ người dùng, nền tảng chatbot sẽ càng “thông minh”. Một nền tảng chatbot thông minh có khả năng tự học hỏi dựa trên dữ liệu nhận được mà không cần phải được lập trình cụ thể (đó được gọi là phương pháp học máy – Machine Learning). Chính điều này làm cho các nhà phát triển dễ dàng phát triển các chương trình trò chuyện và tự động hóa các cuộc trò chuyện với người dùng.

Theo như Bộ Giáo dục và Đào tạo cho biết (năm 2019) cả nước có tới 237 trường đại học, học viện. Với mỗi trường lại đào tạo nhiều ngành khác nhau. Hệ thống chatbot “hỗ trợ học sinh lựa chọn trường đại học Unisec-chatbot” là một hệ thống chatbot giúp những học sinh tìm hiểu về các trường đại học trong cả nước một cách nhanh chóng và dễ dàng. Ứng dụng đưa ra các lời khuyên giúp học sinh lựa chọn trường đại học phù hợp với nhu cầu của họ. Nó sẽ gợi ý để họ lựa chọn các phương án tốt nhất như trường đại học, chuyên ngành, địa điểm…

Và Rasa là một trong những công cụ hỗ trợ mạnh cho chatbot, giúp người dùng xây dựng được những hệ thống tự động và tối ưu. Xuất phát từ mục tiêu tìm hiểu và học hỏi những kiến thức mới, cũng như củng cố lại kiến thức và kỹ năng trong những năm tháng học tập tại trường, đồng thời nghiên cứu sâu hơn về lĩnh vực chabot AI. Do đó em thực hiện đồ án chọn đề tài:” Xây dựng hệ thống chatbot hỗ trợ học sinh lựa chọn trường đại học Unisec-chatbot”.

## . Nội dung của khóa luận

Hệ thống “hệ thống chatbot hỗ trợ học sinh lựa chọn trường đại học Unisec-chatbot” là hệ thống giúp học sinh giải đáp các nghi vấn về các trường đại học và tư vấn giúp học sinh lựa chọn trường đại học phù hợp với nhu cầu của họ.

Hệ thống gồm hai phần: phần ứng dụng di động. là phần tương tác với người dùng, hoạt động trên hệ điều hành android; và phần hệ thống server, là phần xử lý thông tin và tạo các phản hồi tương ứng.

Phần hệ thống server được xây dựng dựa trên framework rasa với ngôn ngữ lập trình python. Rasa là một bộ công cụ xử lý ngôn ngữ tự nhiên (Natural Language Processing-NLP) mã nguồn mở tập trung chủ yếu vào chatbot. Trên thực tế đây là một trong những công cụ hiệu quả và tiết kiệm thời gian nhất để xây dựng các chatbot phức tạp. Rasa hoạt động khá tốt và mạnh mẽ, đặc biệt trong vấn đề xác định ý định của người dùng và đối tượng được nhắc đến trong câu dù dữ liệu thu thập và cung cấp rất ít.

## Mục tiêu và đóng góp của khóa luận

Trong khóa luận này, mục tiêu chính của đề tài là nghiên cứu và tìm hiểu về hệ thống chatbots và ứng dụng của chatbots trong các lĩnh vực khác nhau của đời sống, cụ thể qua việc xây dựng hệ thống chatbots trả lời các câu hỏi hỗ trợ học sinh lựa chọn trường đại học Unisec-chatbot, hệ thống được xây dựng trên nền tảng Rasa và ngôn ngữ Python.

Khóa luận tạo ra được công cụ trả lời các câu hỏi hỗ trợ học sinh lựa chọn trường đại học theo yêu cầu theo hình thức hội thoại giữa người và máy. Giúp học sinh tìm hiểu về các trường đại học trong cả nước một cách nhanh chóng và chính xác. Khóa luận cho thấy việc xây dựng chatbot ứng dụng AI trong việc trả lời các câu hỏi hỗ trợ học sinh lựa chọn trường đại học là hoàn toàn khả thi và có thể áp dụng vào thực tiễn.

## 1.4. Cấu trúc khóa luận

Khóa luận này gồm 4 chương lớn:

Chương 1 Giới thiệu đề tài: Nêu lý do chọn đề tài, những nội dung được đề cập đến trong khóa luận.

Chương 2 Cơ sở lý thuyết : Giới thiệu về trí tuệ nhân tạo, chatbots, rasa, ý định người dùng (intent), thực thể (entity), hành động (action), chính sách (policies), NLU

Chương 3 Xây dựng và phát triển hệ thống: Trình bày về quá trình xây dựng chatbot trả lời thông tin về các ngành học và các trường đại học, thực nghiệm và đánh giá các kết quả.

Chương 4: Kết luận và hướng phát triển: Trình bày kết quả đạt được và nêu ra phương hướng phát triển đề tài.

# Chương 2. Cơ sở lý thuyết

## Giới thiệu về trí tuệ nhân tạo (AI – Artificial Intelligence)

Ngày nay cùng với sự phát triển vượt bậc của khoa học công nghệ thì trí tuệ nhân tạo đã có mặt ở khắp mọi nơi. Trí tuệ nhân tạo đã trở thành xu hướng mà các hãng công nghệ lớn đang hướng đến như Google, Facebook, Amazon, Microsoft,.. Các hãng đó đều đầu tư rất lớn vào trí tuệ nhân tạo nhằm tạo ra các sản phẩm đáp ứng được tất cả nhu cầu của con người.

Bellman(1918) định nghĩa: Trí tuệ nhân tạo là tự động hóa các hoạt động phù hợp với suy nghĩ con người, chẳng hạn các hoạt động ra quyết định, giải bài toán…[1]

Rich and Knight (1991) thì cho rằng: Trí nhân tạo là khoa học nghiên cứu xem làm thế nào để máy tính có thể thực hiện những công việc mà hiện con người làm tốt hơn máy tính [1].

Mỗi khái niệm, định nghĩa đều có điểm đúng riêng của mình, nhưng chúng ta có thể hiểu một cách đơn giản. Trí tuệ nhân tạo là một ngành thuộc lĩnh vực khoa học máy tính. Là trí tuệ do con người lập trình tạo nên với mục tiêu giúp máy tính có thể tự động hóa các hành vi thông minh như con người. Trí tuệ nhân tạo khác với việc lập trình logic trong các ngôn ngữ lập trình là ở việc ứng dụng các hệ thống học máy để mô phỏng trí tuệ của con người như: biết suy nghĩ và lập luận để giải quyết vấn đề, biết giao tiếp do hiểu ngôn ngữ, tiếng nói, biết học và tự thích nghi.

## Chatbots

### Khái niệm

Chatbot là một hình thức thô sơ của phần mềm trí tuệ nhân tạo, là một ứng dụng phần mềm dùng để quản lý một hệ thống thảo luận trực tuyến bằng văn bản hoặc văn bản chuyển thành giọng nói, thay vì cung cấp các thảo luận trực tiếp với người dùng có thật. Trong đa số trường hợp thì chatbot được sử dụng qua ứng dụng tin nhắn để nói chuyện với con người. Nó có khả năng trả lời những câu hỏi mà người dùng đề ra, ban đầu nó sẽ dựa vào các từ khóa có trong câu hỏi của con người để tìm câu trả lời phù hợp và dần dần nó sẽ học hỏi được thêm từ trải nghiệm người dùng và làm những cuộc trò chuyện trở nên cá nhân hơn và chân thật hơn.

Ví dụ 1: Nếu như bạn muốn nhắn tin tìm kiếm thông tin về một khóa học của một tổ chức hoặc một công ty nào đó. Nếu công ty đã tạo sẵn một Chatbot thì bạn chỉ cần nhập từ khóa về những thông tin bạn muốn biết, chúng sẽ lập tức tìm và hiện kết quả thông tin về khóa học đó cho bạn. Đôi khi thuật ngữ “Chatbots“ được sử dụng để chỉ các trợ lý ảo nói chung hoặc cụ thể là chỉ trương trình máy tính trả lời tự động trên nền tảng tin nhắn (Messaging Platform).

Ví dụ 2: Nếu bạn đã từng hỏi “trợ lý ảo” Cortana của Microsoft hay Siri của Apple về một vấn đề gì đó như:” Hôm nay trời có mưa không?”, thì bạn đã làm việc với một Chatbot rồi đó.

### Lịch sử ra đời

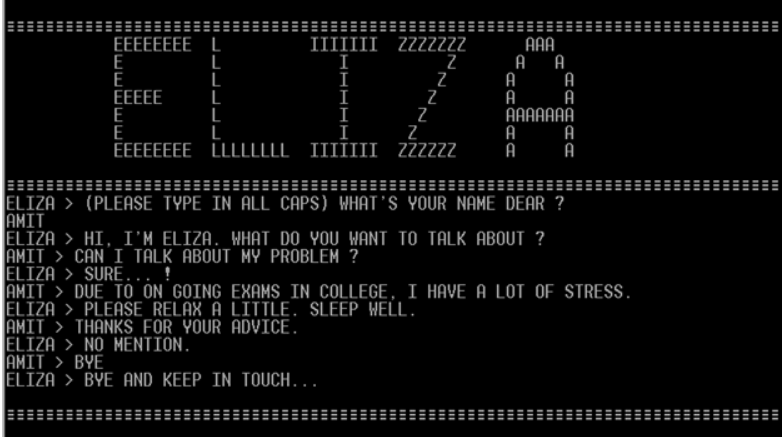
Dưới đây là tóm tắt sơ lược về lịch sử hình thành của chatbot [6], [12].

1. Turing test năm 1950

Năm Phép thử Turing là một phương pháp để xác định xem một cỗ máy có thể chứng minh trí thông minh của nó giống với não người hay không. Nếu một cỗ máy có thể tham gia vào một cuộc hội thoại với con người mà không bị phát hiện là một cỗ máy, thì nó đã thể hiện được trí tuệ của con người. Các phép thử Turing được thực hiện để xác định xem một chương trình máy tính có thể phân biệt được máy tính với con người trong một cuộc trò chuyện chỉ có văn bản thuần túy hay không. Bằng cách gõ câu hỏi cho cả hai đối tượng thử nghiệm, người thẩm vấn sẽ cố gắng xác định đối tượng nào là máy tính và đối tượng nào là con người. Máy tính sẽ vượt qua phép thử Turing nếu người thẩm vấn không thể nói sự khác biệt giữa chủ thể con người và máy tính.

1. Năm 1966 Chương trình trò chuyện đầu tiên được tạo ra

Chatbot đầu tiên ra đời năm 1960, tên là Eliza, và là một chương trình máy tính của Joseph Weizenbaum (Viện Công nghệ Massachusetts, Mỹ). Chương trình được thiết kế theo cách bắt chước cuộc trò chuyện của con người. Chatbot Eliza hoạt động bằng cách chuyển các từ mà người dùng đã nhập vào máy tính và sau đó ghép nối chúng vào danh sách các câu trả lời có kịch bản. Nó sử dụng một kịch bản mô phỏng một nhà tâm lý trị liệu. Kịch bản được chứng minh là một tác động đáng kể đến việc xử lý ngôn ngữ tự nhiên và trí thông minh nhân tạo và là một trong những chương trình đầu tiên có thể vượt qua bài kiểm tra Turing. Hình 2.1 hiển thị tương tương tác giữa người dùng và chatbot Eliza [6].



Hình 2.1: Tương tác giữa người dùng với chatbot Eliza

1. Năm 1972: Bác sĩ Kenneth Colby phát triển chatbot Parry

Parry được xây dựng bởi bác sĩ tâm thần người Mỹ Kenneth Colby vào năm 1972. Chương trình bắt chước một bệnh nhân tâm thần phân liệt. Nó là một chương trình ngôn ngữ tự nhiên tương tự như suy nghĩ của một cá nhân. Parry hoạt động thông qua một hệ thống phức tạp các giả định, phân bổ và “phản ứng cảm xúc” được kích hoạt bằng cách thay đổi trọng số được gán cho các đầu vào bằng lời nói. Trong cùng năm đó, Parry và Eliza đã “gặp” và “nói chuyện” với nhau tại Hội nghị Quốc tế về Truyền thông Máy tính ở Washington DC. Sau đó, Parry cũng đã vượt qua một phiên bản của Turing Test.

1. Năm 1981: Chatbot Jabberwacky

Được phát triển vào những năm 1980 và phát hành trực tuyến vào năm 1997, chatbot Jabberwacky được thiết kế để “Mô phỏng trò chuyện của con người tự nhiên theo cách thú vị và hài hước“. Mục đích ban đầu của dự án Chatbot Jabberwacky là tạo ra một trí tuệ nhân tạo có khả năng vượt qua Các phép thử Turing. Nó được thiết kế để bắt chước tương tác của con người và thực hiện các cuộc hội thoại với người dùng. Mục đích cuối cùng của chương trình là chuyển từ một hệ thống dựa trên văn bản sang toàn bộ hoạt động bằng giọng nói. Tác giả của nó tin rằng nó có thể được kết hợp vào các vật thể xung quanh nhà như robot, các thiết bị thông minh,… Trong khi tất cả các chatbot trước đó dựa trên cơ sở dữ liệu tĩnh để trả lời và trò chuyện, Jabberwacky thu thập cụm từ được sử dụng bởi những người tham gia trò chuyện với nó. Nó tự thêm những câu trả lời vào cơ sở dữ liệu và tự động phát triển nội dung của riêng mình. Trong năm 2008, Jabberwacky đã phát hành một phiên bản mới và đổi tên thành Cleverbot.

1. Năm 2001: SmarterChild

SmarterChild là một Chatbot có sẵn trên mạng AOL Instant Messenger và Windows Live Messenger (trước đây là MSN Messenger). AOL Instant Messenger là một chương trình tin nhắn tức thời và hiện diện do AOL tạo ra, sử dụng giao thức nhắn tin tức thời OSCAR độc quyền và giao thức TOC để cho phép người dùng đăng ký giao tiếp trong thời gian thực. SmarterChild đóng vai trò giới thiệu cho việc truy cập dữ liệu nhanh và cuộc trò chuyện được cá nhân hóa thú vị hơn. Hơn nữa, khi kết hợp với các nhà mạng, chúng trở thành một kênh tiếp thị hiệu quả và miễn phí. Chúng giúp người dùng giao tiếp nhanh chóng với hệ thống mạng bằng cách hiển thị các thông tin ngắn gọn với các lựa chọn trên bàn phím điện thoại.

1. Năm 2006: IBM Watson

IBM Watson được tạo ra với mục tiêu vượt lên và chiến thắng các thí sinh tham dự cuộc thi Jeopardy! Với khả năng chạy hàng trăm thuật toán phân tích ngôn ngữ cùng một lúc, IBM Watson sở hữu một sự thông minh ngôn ngữ đáng ngạc nhiên. IBM thiết lập cho Watson có quyền truy cập vào cơ sở dữ liệu khổng lồ về thông tin. Watson có thể nhanh chóng truy cập 200 triệu trang dữ liệu, làm cho nó trở thành một máy trả lời câu hỏi lý tưởng (hoặc, trong trường hợp của Jeopardy, Watson trở thành máy tạo câu hỏi lý tưởng).

1. Năm 2010- 2016: Thời kỳ bùng nổ của các trợ lý ảo

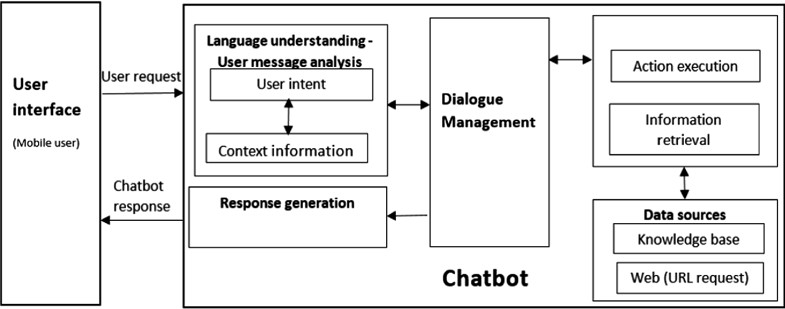
Nửa đầu thập kỷ này chứng kiến ​​sự bùng nổ của các trợ lý cá nhân ảo: Siri (2010), Google Now (2012), Alexa (2015), Cortana (2015) và Google Assistant (2016). Với khả năng phân tích và xử lý ngôn ngữ tự nhiên các trợ lý này kết nối với các dịch vụ web để trả lời các câu hỏi và đáp ứng các yêu cầu của người dùng. Gần đây, Google Home và Amazon Echo đã bắt đầu trở thành các tính năng phổ biến trong các ngôi nhà của người Mỹ. Google Assistant và Alexa là cơ sở để cho các thiết bị thông minh trong ngôi nhà tương tác với người dùng. Các thiết bị thông minh Smart Devices kết nối với nhau tạo thành hệ thống Smart Home, cho phép người dùng ra lệnh và điều khiển bằng giọng nói của họ.

1. Từ năm 2016 đến nay: Chatbot Messenger bùng nổ

Từ sau hội nghị F8 năm 2016, Facebook – mạng xã hội lớn nhất thế giới giới thiệu Messenger Platform. Một nền tảng thân thiện hơn và cho phép bất kỳ ai cũng có thể tạo cho mình một Chatbot. Ngay sau đó, các ứng dụng chat khác như LINE, WhatsApp, Telegram hay Twitter cũng đưa ra các hỗ trợ hoặc các API cho phép người dùng tạo các Chatbot trên ứng dụng nhắn tin.

### 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 2.2 giới thiệu một kiến trúc chung của chatbot [8, tr.373-383].



Hình 2.2: Kiến trúc chung của chatbot

Dưới đây trình bày chi tiết các thành phần của chatbot

* + - * **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. Các hành động tương ứng với các bước mà chatbot sẽ thực hiện khi các ý định cụ thể được kích hoạt bởi các đầu vào của người dùng và có thể có các tham số để xác định thông tin chi tiết về nó. Phát hiện ý định thường được xây dựng dưới dạng phân loại câu, trong đó các nhãn ý định đơn hoặc nhiều ý định được dự đoán cho mỗi câu.

Thực thể là một công cụ để trích xuất các giá trị tham số từ các đầu vào ngôn ngữ tự nhiên. Ví dụ, hãy xem xét câu “What is the weather in Greece?”. Mục đích của người dùng là tìm hiểu dự báo thời tiết. Giá trị thực thể là Greece (Hy Lạp). Do đó, người dùng yêu cầu dự báo thời tiết ở Hy Lạp. Các thực thể có thể do hệ thống xác định hoặc do nhà phát triển xác định.

Ngữ cảnh là các chuỗi lưu trữ ngữ cảnh của đối tượng mà người dùng đang đề cập hoặc nói đến. Ví dụ, một người dùng có thể tham chiếu đến một đối tượng đã được xác định trước đó trong câu sau của họ. Người dùng có thể nhập “Switch on the fan”. Ở đây, ngữ cảnh sẽ được lưu là fan (cái quạt) để khi người dùng nói, “Switch it off” làm đầu vào tiếp theo, ý định “tắt” có thể được gọi trên ngữ cảnh “quạt”.

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].

* + - * **DM (Quản lý hội thoại)**

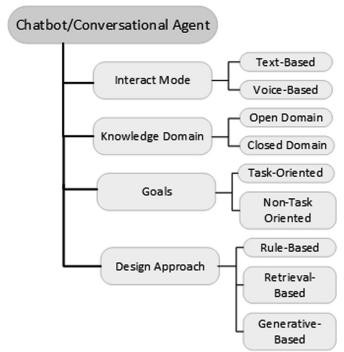
DM (Dialog Management - quản lý hội thoại): Thành phần quản lý đối thoại giữ và cập nhật ngữ cảnh của cuộc hội thoại là ý định hiện tại, các thực thể được xác định hoặc các thực thể bị thiếu cần thiết để thực hiện các yêu cầu của người dùng. Hơn nữa, nó yêu cầu thông tin thiếu, xử lý làm rõ bởi người 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. Các ngữ cảnh này phải được đối chiếu trong các kịch bản dựng sẵn (history) đã đào tạo cho bot. Thành phần này cũng đảm nhiệm việc lấy dữ liệu từ hệ thống khác qua các API/Data sources gọi trong action

* + - * **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

### 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 trong hình 2.3 [7, tr.947-954].

Hình 2.3: 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. Ví dụ: Khách hàng mua ô tô, tư vấn khách hàng mua bảo hiểm nhân thọ, dự báo thời tiết… Loại này phổ biến, dữ liệu huấn luyện trong phạm vi nhỏ nên dễ huấn luyện, độ chính xác cao.

+ 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. Rất nhiều chatbot thông minh được tạo ra. Tuy nhiên trả lời mọi vấn đề và vượt qua được Turing test thì vẫn chưa thể đạt tới.

* **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. Sự thông minh của chatbot phụ thuộc vào mức độ mà ta lập trình ra chatbot.

+ 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”. Nghĩa là ta đưa vào càng nhiều câu huấn luyện, xác suất chatbot phản hồi người dùng chính xác càng cao. Trong phương pháp này có thể chia ra thành dựa trên cơ sở sáng tạo (Generative-Based) như các mô hình theo trình tự, tạo ra trả lời phù hợp trong cuộc trò chuyện hoặc dựa trên truy xuất (Retrieval-Based) để học lựa chọn các câu trả lời từ cuộc hội thoại hiện tại từ một kho lưu trữ.

### Một số nền tảng phát triển chatbot

* + - * **Dialogfow (<https://dialogflow.com/>)**
      * Cung cấp bởi google
      * Trước đây được gọi là Api.ai và rất phổ biến rộng rãi trong cộng đồng chatbot
      * Cung cấp cho người dùng những cách mới để tương tác với sản phẩm của họ bằng cách xây dựng giao diện đàm thoại dựa trên giọng nói và văn bản hấp dẫn bằng AI
      * Kết nối với người dùng trên Google Assistant, Amazon Alexa, Facebook Messenger và các nền tảng và thiết bị phổ biến khác
      * Có khả năng phân tích và hiểu ý định của người dùng để giúp bạn phản hồi theo cách hữu ích nhất
      * **Rasa (<https://rasa.com/>)**
      * Rasa Open Source là một nền tảng để tự động hóa các trợ lý dựa trên văn bản và giọng nói sử dụng học máy
      * Có thể thực hiện các hành động mà bot có thể thực hiện bằng mã Python
      * Thay vì một loạt các câu lệnh if…else khác, logic của bot dựa trên một mô hình học máy được đào tạo trên các ví dụ hội thoại
      * **Wit.ai (**[**https://wit.ai**](https://wit.ai/)**)**
      * Được Facebook mua lại trong vòng 21 tháng kể từ khi ra mắt, nhóm wit.ai đóng góp cho công cụ NLP của Facebook trong Facebook.
      * Wit.ai giúp các nhà phát triển dễ dàng xây dựng các ứng dụng và các thiết bị mà người dùng có thể nói chuyện hoặc nhắn tin tới.
      * Có thể sử dụng wit.ai để xây dựng chatbot, tự động hóa nhà, ...
      * **Microsoft Bot Framework (**[**https://dev.botframework.com/**](https://dev.botframework.com/)**)**
      * Được cung cấp bởi Microsoft.
      * Microsoft Bot Framework có khả năng hiểu ý định của người dùng
* Có thể kết hợp LUIS để hiểu ngôn ngữ tự nhiên, Cortana cho giọng nói và API Bing cho tìm kiếm

### Xu hướng phát triển

Chabot sẽ bùng nổ và sẽ là một hiện tượng mới trong tương lai gần. Người dùng sẽ ngày càng không còn cảm thấy trở ngại khi nói chuyện với bot. Chatbot sẽ ngày càng thông minh hơn và trở nên “giống người” hơn. Khi các doanh nghiệp liên tục “sản xuất” ra lượng dữ liệu khổng lồ mỗi ngày thì việc đưa ra quyết định vừa nhanh chóng vừa chính xác đã trở thành một bài toán khó. Giờ đây, các chatbot AI sẽ là giải pháp hợp lý để vượt qua thử thách này. Bot sẽ đưa ra các phân tích cực kỳ nhanh chóng và thuận tiện để hỗ trợ ra quyết định, tăng hiệu quả kinh doanh.

Trong tương lai gần chatbot sẽ có thể thay thế các ứng dụng di động. Bắt đầu từ năm 2021, 50% các doanh nghiệp sẽ chi nhiều cho kế hoạch tạo bot và chatbot hơn so với việc phát triển các ứng dụng di động truyền thống. Giống như thị trường ứng dụng trước kia, các doanh nghiệp sẽ phát triển chiến lược bot và sử dụng nền tảng chatbot dựa trên sức mạnh của trí tuệ nhân tạo để triển khai, đào tạo và quản lý giúp việc kinh doanh trở nên tốt hơn. Vì vậy, các bot sẽ làm giảm dần các ứng dụng mà doanh nghiệp sử dụng và sẽ dần thay thế toàn bộ.

## Rasa framework

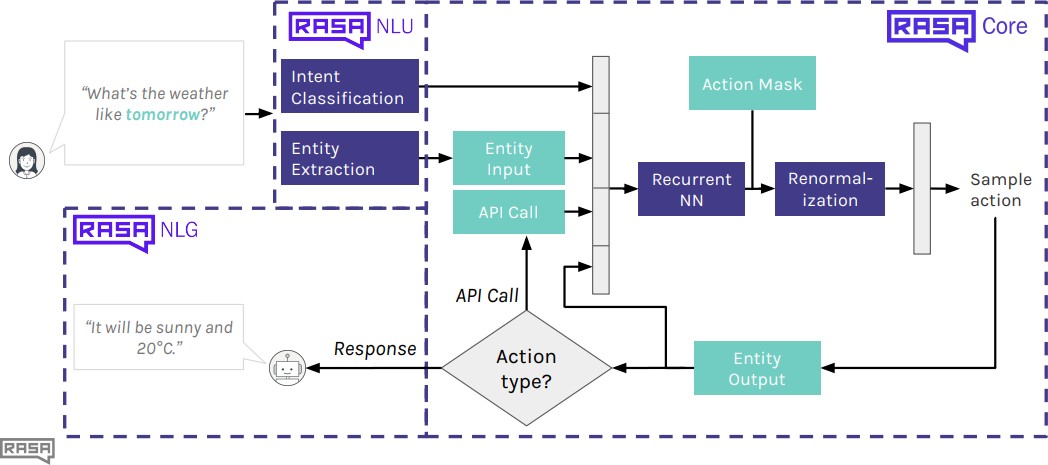
### Giới thiệu

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 rất nhiều các tính năng để tạo ra các trợ lý và chatbot dựa trên văn bản và giọng nói.

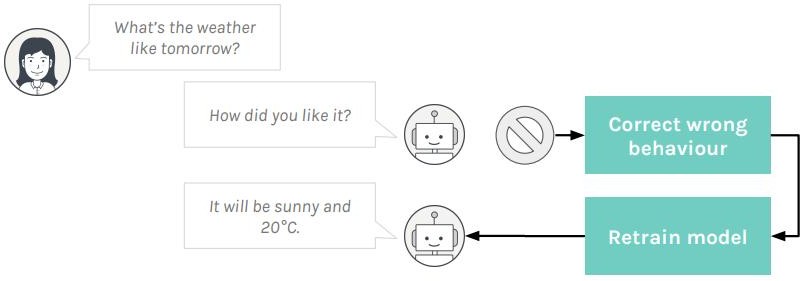
* + - * Hiểu thông điệp (Understand messages): biến văn bản dạng tự do ở bất kỳ ngôn ngữ nào thành dữ liệu có cấu trúc. Hỗ trợ đơn và đa ý định (multiple intents) và cả các thực thể được đào tạo trước và tùy chỉnh (pre-trained and custom entities).
      * Duy trì cuộc trò chuyện (Hold conversations): 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. Hình 2.5 miêu tả chế độ học tương tác của Rasa [4].
      * 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.
      * Triển khai mọi nơi (Deploy): có khả năng triển khai Docker containers và điều phối để chạy Rasa on-premise hoặc thông qua nhà cung cấp đám mây ưa thích (cloud).

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. Hình 2.4 miêu tả các thành phần của Rasa [4].



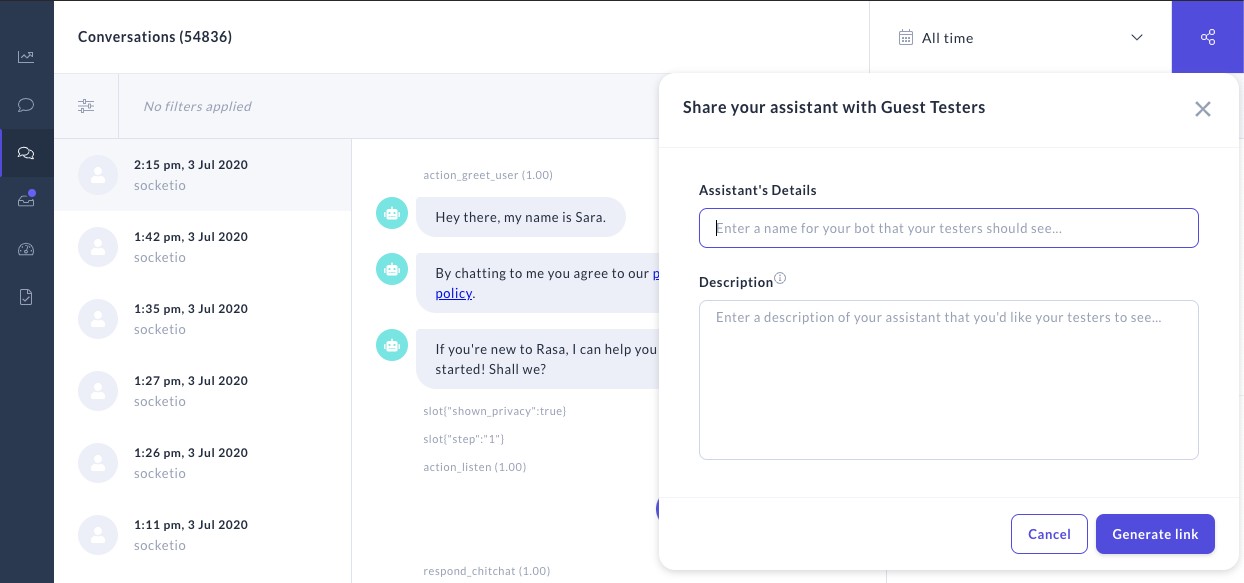
Hình 2.4: Các thành phần của Rasa

Rasa có hỗ trợ chế độ học tương tác. Trong chế độ này, ta cung cấp phản hồi cho chatbot của mình trong khi nói chuyện với nó. Đây là một cách hiệu quả để khám phá những gì chatbot có thể làm và là cách dễ nhất để sửa chữa bất kỳ lỗi nào mà nó mắc phải. Một lợi thế của đối thoại dựa trên học máy là khi chatbot chưa biết cách làm điều gì đó, ta có thể dạy cho nó. Hình 2.5 miêu tả chế độ học tương tác của Rasa [4].

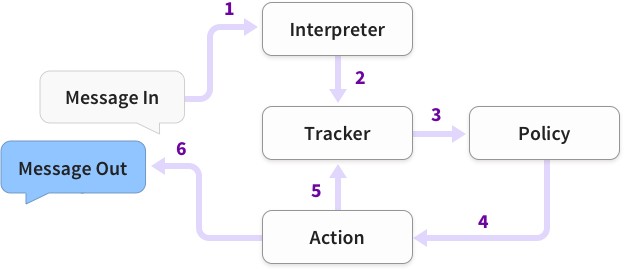


Hình 2.5: Chế độ học tương tác của Rasa

Rasa còn cung cấp công cụ Rasa X dành cho phát triển theo hướng hội thoại (Conversation-Driven Development - CDD), đây là công cụ thông qua quá trình lắng nghe người dùng và sử dụng những thông tin chi tiết đó để cải thiện trợ lý AI chatbot. Hình 2.6 là giao diện tương tác với người dùng của Rasa X [4].



Hình 2.6: Công cụ Rasa X

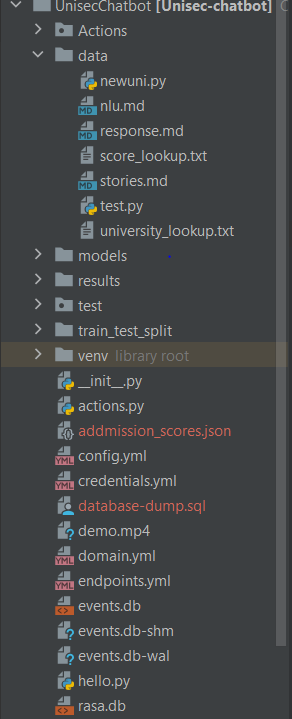
Hình 2.7 là sơ đồ hiển thị các bước cơ bản về cách một trợ lý được xây dựng bằng Rasa phản hồi một thông báo [4].

Hình 2.7: Cách thức Rasa phản hồi một tin nhắn

* + - * Tin nhắn được nhận và chuyển đến trình thông dịch (Interpreter), chuyển nó thành từ điển bao gồm văn bản gốc, ý định và các thực thể được tìm thấy. Phần này do NLU đảm nhận.
      * Tracker là đối tượng theo dõi trạng thái hội thoại. Nó nhận được thông tin rằng một tin nhắn mới đã đến.
      * Chính sách (Policy) nhận trạng thái hiện tại của Tracker.
      * Chính sách chọn hành động (Action) tiếp theo để thực hiện.
      * Hành động đã chọn được ghi lại bởi Tracker.
      * Một phản hồi được gửi đến người dùng

### Cấu trúc chương trình của Rasa

Cấu trúc của một chương trình của Rasa như hình 2.8:



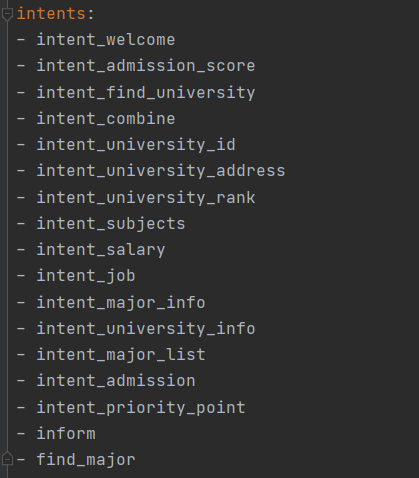
Hình 2.8: Cấu trúc của một chương trình Rasa

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 |

### Ý định người dùng (intent)

Intent: điều người dùng mong muốn chatbot thực hiện (hỗ trợ) khi đưa ra câu hội thoại. Trong Rasa thực hiện khai báo các Intent (ý định người dùng) trong domain.yml. Chẳng hạn hình 2.9 dưới đây liệt kê một số ý định của người dùng tương tác với chatbot như chào hỏi, hỏi điểm chuẩn ,hỏi trường đại học, hỏi tổ hợp môn xét tuyển, …



Hình 2.9: Các ý định được khai báo trong domain.yml

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.

### Thực thể (entity)

Thực thể được sử dụng để trích xuất các giá trị của tham số từ ngôn ngữ tự nhiên. Vai trò của tác vụ này là nhận dạng các cụm từ trong văn bản và phân loại chúng vào các nhóm đã được định trước như tên người, tổ chức, địa điểm, thời gian, loại sản phẩm, nhãn hiệu… Bất kỳ những gì mà người dùng truyền vào thông tin đều sẽ có một entity tương ứng.

Ví dụ : Trong câu nói “ Trường Đại Học Công Nghệ -ĐH Quốc Gia Hà Nội ở đâu” sẽ có “Đại Học Công Nghệ -ĐH Quốc Gia Hà Nội “ là một thực thể tương ứng là “Trường đại học”, và “ở đâu” tương ứng sẽ là “địa chỉ”.

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)

### Khung kịch bản (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:

***## story ask admission score*\*** intent\_admission\_score  
**-** form\_admission\_score  
**-** form{"name" : "form\_admission\_score"}  
**-** form{"name" : null}

* + - * Một câu chuyện bắt đầu với một cái tên đứng trước hai dấu #. Ta có thể gọi câu chuyện là bất cứ thứ gì bạn thích, nhưng nó có thể rất hữu ích để gỡ lỗi khi đặt tên mô tả cho chúng.
      * Kết thúc của một câu chuyện được biểu thị bằng một dòng mới và sau đó một câu chuyện mới bắt đầu lại với ##.
      * Tin nhắn do người dùng gửi được hiển thị dưới dạng các dòng bắt đầu bằng \* định dạng intent {"entity1": "value", "entity2": "value"}
      * Các hành động được thực hiện bởi chabot được hiển thị dưới dạng các dòng bắt đầu bằng - và chứa tên của hành động.
      * Các sự kiện được trả về bởi một hành động sẽ hiển thị ngay sau hành động đó. Ví dụ, nếu một hành động trả về một sự kiện SlotSet, điều này được thể hiện như slot{"slot\_name": "value"}.

### Hành động (action)

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.

Tất cả các hành động (cả hành động phát biểu và hành động tùy chỉnh) được thực hiện bởi chatbot được hiển thị dưới dạng các dòng bắt đầu – và theo sau là tên của hành động.

Các phản hồi cho các hành động phát biểu phải bắt đầu bằng tiền tố utter\_và phải khớp với tên của phản hồi được xác định trong domain.yml.

**utter\_ask\_entity\_province**:  
- **text**: Bạn muốn hỏi tỉnh nào?  
**utter\_ask\_entity\_combine**:  
- **text**: Khối thi của bạn là gì?  
**utter\_ask\_entity\_score**:  
- **text**: bạn dự kiến được bao nhiêu điểm?

Đối với hành động tùy chỉnh, tên hành động là chuỗi ta chọn để trả về từ phương thức name của lớp (class) hành động tùy chỉnh. Tất cả các hành động tùy chỉnh đều được khai báo trong actions.py.

### Chính sách (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. Ví dụ, trong ví dụ bên dưới, có tham số max\_history để kiểm soát lượng lịch sử đối thoại mà mô hình xem xét để quyết định hành động nào cần thực hiện tiếp theo.

**policies**:  
- **name**: TEDPolicy  
 **epochs**: 50  
 **max\_history**: 8  
- **name**: MemoizationPolicy  
 **max\_history**: 5  
- **name**: FallbackPolicy  
 **nlu\_threshold**: 0.3  
 **ambiguity\_threshold**: 0.1  
 **core\_threshold**: 0.3

### Vị trí (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).

Ví dụ: nếu người dùng đã cung cấp tên của trường đại học, ta có thể có một slot dạng text được gọi là entity\_university. Nếu người dùng hỏi về ngành học và chatbot không biết tên trường, nó sẽ phải hỏi họ. Slot chỉ cho Rasa Core biết liệu slot đó có giá trị hay không. Giá trị cụ thể của một slot (ví dụ:Đại học Bách Khoa hoặc Đại học Công Nghệ Đại Học Quốc Gia Hà Nội) không tạo ra bất kỳ sự khác biệt nào.

# Chương 3: Xây dựng và phát triển hệ thống

## 3.1. Giới thiệu bài toán

Nhiệm vụ chính trong chương này là giải quyết bài toán trả lời các thông tin liên quan đến các trường đại học, tư vấn và hỗ trợ sinh viên lựa chọn trường đại học phù hợp với yêu cầu của họ. Trên cơ sở nghiên cứu các câu hỏi thường gặp, bài toán tập trung vào một số chức năng chính của chat bot như sau.

* + - Chào hỏi
    - Tạm biệt
    - Thông tin hỏi về điểm chuẩn
    - Tìm kiếm trường đại học
    - Thông tin chỉ tiêu tuyển sinh
    - Thông tin giới thiệu ngành học
    - Thông tin hỏi về khối xét tuyển
    - Thông tin về địa chỉ của trường đại học
    - Thông tin về xếp hạng trường đại học
    - Thông tin danh sách ngành của trường đại học
    - Thông tin về khung đào tạo của ngành
    - Thông tin tham khảo mức lương của ngành
    - Thông tin tham khảo về cơ hội việc làm
    - Thông tin về mã của trường đại học
    - Thông tin về điểm cộng của từng khu vực

### 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.

Có các Components để trích xuất thực thể, để phân loại ý định, lựa chọn phản hồi, tiền xử lý và các thành phần khác. Nếu muốn thêm thành phần của riêng mình, chẳng hạn như để chạy kiểm tra chính tả (spell-check) hoặc để phân tích quan điểm (sentiment analysis), có thể thực hiện custom component.

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. Đối với ngôn ngữ tiếng Việt, các từ được phân tách bằng dấu cách , ở đây tác giả lựa chọn tách từ bằng SpacyTokenizer. Ngoài ra tác giả có thực hiện custom tokenization cho tiếng Việt với model là “vi\_spacy\_model”.

Nhằm chuẩn hóa từ đồng nghĩa bằng việc đồng nhất từ đồng nghĩa, từ địa phương, tiếng lóng về một từ chuẩn hóa, tác giả sử dụng EntitySynonymMapper.

* + - [Featurization](https://legacy-docs-v1.rasa.com/1.10.12/nlu/choosing-a-pipeline/#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. Lợi thế của việc sử dụng tính năng nhúng từ được đào tạo trước là nếu ta có một ví dụ đào tạo như: “Tôi muốn mua táo” và Rasa được yêu cầu dự đoán ý định cho “lấy lê”, thì mô hình của ta đã biết rằng từ "táo" và "lê" rất giống nhau. Điều này đặc biệt hữu ích nếu không có đủ dữ liệu đào tạo…

+ 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 phương pháp nhúng được giám sát, Rasa có khả năng huấn luyện với bất kì ngôn ngữ nào (bao gồm tiếng Việt), vì sẽ training lại mọi thứ từ đầu, chỉ phụ thuộc vào dữ liệu huấn luyện. 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 về tuyển sinh đại học 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 cùng với một số thành phần cụ thể là:

[CountVectorsFeaturizer](https://legacy-docs-v1.rasa.com/1.10.12/nlu/components/#countvectorsfeaturizer): trích xuất đặc trưng cho phân loại ý định và lựa chọn phản hồi, tạo túi từ (BoW:bag-of-words) đại diện cho tin nhắn người dùng, ý định và phản hồi.

[RegexFeaturizer](https://legacy-docs-v1.rasa.com/1.10.12/nlu/components/#regexfeaturizer): Tạo biểu diễn vectơ của thông điệp người dùng bằng cách sử dụng biểu thức chính quy(regular expressions).

[LexicalSyntacticFeaturizer](https://legacy-docs-v1.rasa.com/1.10.12/nlu/components/#lexicalsyntacticfeaturizer): Tạo các đặc trưng từ vựng và cú pháp cho tin nhắn của người dùng để hỗ trợ trích xuất thực thể.

* + - [Entity Recognition / Intent Classification / Response Selectors](https://legacy-docs-v1.rasa.com/1.10.12/nlu/choosing-a-pipeline/#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. Trong Rasa, việc nhận diện ý định thông thường sử dụng mô hình máy vector hỗ trợ (Support Vector Machines- SVM), trích xuất thông tin thực thể sử dụng mô hình trường ngẫu nhiên có điều kiện (Conditional Random Fields - CRF). Ở đây tác giả lựa chọn các thành phần như sau:

[DIETClassifier](https://legacy-docs-v1.rasa.com/1.10.12/nlu/components/#diet-classifier): 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. Đối với nhãn ý định, đầu ra bộ chuyển đổi cho CLS token và nhãn ý định được nhúng vào một không gian vectơ ngữ nghĩa duy nhất. [DIETClassifier](https://legacy-docs-v1.rasa.com/1.10.12/nlu/components/#diet-classifier) cũng hỗ trợ đa ý định (multi-intent) tách các Intent thành nhiều nhãn.

Sau khi xây dựng xong mô hình và tạo một số dữ liệu đào tạo NLU, có thể huấn luyện (train) mô hình với rasa train nlu. Sau khi quá trình huấn luyện kết thúc, ta có thể kiểm tra (test) khả năng của mô hình diễn giải các thông điệp đầu vào khác nhau qua: rasa shell nlu.

### 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.

+ Sự chính xác (accuracy):

Cách đánh giá này đơn giản tính tỷ lệ các lớp đã phân loại đúng / tổng số dự đoán.

+ Ma trận hỗn loạn (confusion matrix):

Là một phương pháp đánh giá kết quả của những bài toán phân loại với việc xem xét cả những chỉ số về độ chính xác và độ bao quát của các dự đoán cho từng lớp . Confusion matrix là một ma trận tổng quát thể hiện kết quả phân loại chính xác và kết quả phân loại sai được tạo ra bởi một mô hình phân loại. Đây là một ma trận vuông với kích thước các chiều bằng số lượng lớp dữ liệu. Giá trị tại hàng thứ i, cột thứ j là số lượng điểm lẽ ra thuộc vào class i nhưng lại được dự đoán là thuộc vào class j.

+ True/False Positive/Negative:

Xét một ma trận hỗn loạn gồm 4 chỉ số sau đối với mỗi lớp phân loại:

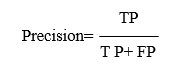
Bảng 3.1: Bảng confusion matrix

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Dự đoán (predicted)** | |
| True | False |
| **Thực tế (Actual)** | Positive | True Positive | False Negative |
| Negative | False Positive | True Negative |

* + - * TP (true positive) – mẫu mang nhãn dương được phân lớp đúng vào lớp dương.
      * FN (false negative) – mẫu mang nhãn dương bị phân lớp sai vào lớp âm.
      * FP (false positive) – mẫu mang nhãn âm bị phân lớp sai vào lớp dương.
      * TN (true negative) – mẫu mang nhãn âm được phân lớp đúng vào lớp âm.

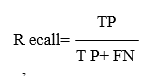
+ Precision / Recall:

* + - * Precision là tỉ lệ số điểm Positive mô hình dự đoán đúng (TP) trên tổng số điểm mô hình dự đoán là Positive (TP+FP), theo công thức:



0< Precision <=1, Precision càng lớn có nghĩa là độ chính xác của các điểm tìm được càng cao.

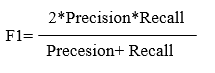
Precision = 1, tức là tất cả số điểm mô hình dự đoán là Positive đều đúng, hay không có điểm nào có nhãn là Negative mà mô hình dự đoán nhầm là Positive.

* + - * Recall là tỉ lệ số điểm Positive mô hình dự đoán đúng (TP) trên tổng số điểm thật sự là Positive (hay tổng số điểm được gán nhãn là Positive ban đầu TP+FN), chỉ số này được tính theo công thức:

Recall càng cao, tức là số điểm là positive bị bỏ sót càng ít. Recall=1, tức là tất cả số điểm có nhãn là Positive đều được mô hình nhận ra.

+ F1- Score:

Chỉ sử dụng Precision hay chỉ có Recall thì không đánh giá được chất lượng mô hình.

Khi đó F1-score được sử dụng. F1-score là trung bình điều hòa của precision và recall (giả sử hai đại lượng này khác 0). F1-score được tính theo công thức:

F1-Score có giá trị nằm trong khoảng (0, 1], F1-Score càng cao thì mô hình phân loại càng tốt.

## 3.2. Xây dựng chương trình

### 3.2.1. Nguồn dữ liệu xây dựng

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

Nguồn dữ liệu liên quan đến thông tin của các trường đại học được tham khảo từ những trang web giới thiệu về các trường đại học, những trang web của bộ giáo dục công bố về điểm chuẩn của các trường đại học… Dữ liệu được tác giả lưu dưới dạng bảng sử dụng công cụ Mongodb.

Để làm giàu cho tập dữ liệu huấn luyện, tác giả bổ sung thêm các câu hỏi mới trong mỗi ý định người dùng (intent), đảm bảo các ý định quan trọng mang tính hỏi đáp nhiều có ít nhất 10 câu hỏi, các ý định khác cũng có tối thiểu 5 câu hỏi, mỗi câu hỏi là một cách diễn đạt khác nhau với cùng một mục đích với câu hỏi ban đầu. Các câu hỏi này đã được gán nhãn cùng với các ý định của câu hỏi ban đầu.

### 3.2.2. Xây dựng ý đinh

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 các ý định như sau với nguồn dữ liệu thực nghiệm.

Bảng 3.2: Bảng các ý định (intent) của chatbot

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên ý định** | **Mô tả** | **Số lượng**  **câu hỏi** | **Số lượng**  **câu trả lời** |
| Intent\_welcome | Chào hỏi | 10 | 1 |
| Intent\_goodbye | Tạm biệt | 10 | 1 |
| Intent\_admission\_score | Ý định liên quan về điểm chuẩn của các ngành | 40 | 2 |
| Intent\_find\_unisversity | Ý định liên quan về các tìm các trường đại học | 40 | 2 |
| Intent\_combine | Ý định liên quan về khối xét tuyển | 5 | 2 |
| Intent\_university\_id | Ý định hỏi về mã của trường đại học | 5 | 2 |
| Intent\_university\_address | Ý định hỏi địa chỉ của trường đại học | 5 | 2 |
| Intent\_university\_rank | Ý định hỏi về xếp hạng của trường đại học | 5 | 2 |
| Intent\_subjects | Ý định hỏi về khung đào tạo của ngành | 5 | 2 |
| Intent\_salary | Ý định tham khảo mức lương của ngành | 10 | 2 |
| Intent\_job | Ý định tham khảo cơ hội việc làm của ngành | 10 | 2 |
| Intent\_major\_info | Ý định liên quan đến thông tin của ngành | 10 | 2 |
| Intent\_university\_info | Ý định liên quan đến thông tin của trường đại học | 10 | 2 |
| Intent\_major\_list | Ý định hỏi về danh sách các ngành của trường đại học | 20 | 2 |
| Intent\_admission | Ý định hỏi chỉ tiêu của trường đại học | 10 | 2 |
| Intent\_priority\_poin | Ý định hỏi về điểm cộng của từng vùng | 20 | 2 |
| Intent\_find\_major | Ý định tham khảo tư vấn chọn ngành phù hợp | 5 | 1 |

Ví dụ cho một ý định hỏi về chỉ tiêu tuyển sinh như sau:

***## intent:intent\_admission*-** chỉ tiêu tuyển sinh trường [đhcnmd](*entity\_university*)  
**-** chỉ tiêu tuyển sinh trường [Đại học tài chính marketing](*entity\_university*)  
**-** trường [đại học hoa lư](*entity\_university*) tuyển sinh bao nhiêu  
**-** trường [đại học nam cần thơ](*entity\_university*) lấy bao nhiêu sinh viên  
**-** chỉ tiêu tuyển sinh ngành [luật kinh tế](*entity\_major*) trường [đại học nội vụ](*entity\_university*) là bao nhiêu  
**-** ngành [công nghệ kỹ thuật xây dựng](*entity\_major*) trường [đại học quang trung](*entity\_university*) có chỉ tiêu là bao nhiêu  
**-** chỉ tiêu tuyển sinh trường [Đại học y dược hà nội](*entity\_university*)  
**-** chỉ tiêu tuyển sinh trường [đại học đà lạt](*entity\_university*)  
**-** trường [đại học thành đô](*entity\_university*) tuyển sinh bao nhiêu  
**-** trường [đại học sài gòn](*entity\_university*) lấy bao nhiêu sinh viên  
**-** chỉ tiêu tuyển sinh ngành [phân tích đầu tư tài chính](*entity\_major*) trường [đại học thương mại](*entity\_university*) là bao nhiêu  
**-** ngành [đông nam á học](*entity\_major*) trường [đại học quang trung](*entity\_university*) có chỉ tiêu là bao nhiêu

### 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 thống để dùng trong các hành động hoặc để đưa ra các câu trả lời phù hợp theo ngữ cảnh, tránh việc phải hỏi lại những thông tin mà người dùng đã cung cấp từ trước.

**entities**:  
- entity\_macro\_region  
- entity\_province  
- entity\_major  
- entity\_score  
- entity\_combine  
- entity\_university  
- entity\_subject  
- entity\_year  
- entity\_gender  
- entity\_university\_id  
**slots**:  
 **entity\_combine**:  
 **type**: list  
 **entity\_combine\_validated**:  
 **type**: list  
 **entity\_gender**:  
 **type**: list  
 **entity\_gender\_validated**:  
 **type**: list  
 **entity\_macro\_region**:  
 **type**: list  
 **entity\_macro\_region\_validated**:  
 **type**: list  
 **entity\_major**:  
 **type**: list  
 **entity\_major\_validated**:  
 **type**: list  
 **entity\_province**:  
 **type**: list  
 **entity\_province\_validated**:  
 **type**: list  
 **entity\_score**:  
 **type**: list  
 **entity\_score\_validated**:

Bài toán xây dựng 2 thực thể : trường đại học (entity\_university) và ngành học (entity\_major), các thực thể được xây dựng với các câu training như dưới.

**-** chỉ tiêu tuyển sinh trường [đhcnmd](*entity\_university*)  
**-** chỉ tiêu tuyển sinh trường [Đại học tài chính marketing](*entity\_university*)  
**-** trường [đại học hoa lư](*entity\_university*) tuyển sinh bao nhiêu  
**-** trường [đại học nam cần thơ](*entity\_university*) lấy bao nhiêu sinh viên  
**-** chỉ tiêu tuyển sinh ngành [luật kinh tế](*entity\_major*) trường [đại học nội vụ](*entity\_university*) là bao nhiêu  
**-** ngành [công nghệ kỹ thuật xây dựng](*entity\_major*) trường [đại học quang trung](*entity\_university*) có chỉ tiêu là bao nhiêu  
**-** chỉ tiêu tuyển sinh trường [Đại học y dược hà nội](*entity\_university*)  
**-** chỉ tiêu tuyển sinh trường [đại học đà lạt](*entity\_university*)  
**-** trường [đại học thành đô](*entity\_university*) tuyển sinh bao nhiêu  
**-** trường [đại học sài gòn](*entity\_university*) lấy bao nhiêu sinh viên  
**-** chỉ tiêu tuyển sinh ngành [phân tích đầu tư tài chính](*entity\_major*) trường [đại học thương mại](*entity\_university*) là bao nhiêu  
**-** ngành [đông nam á học](*entity\_major*) trường [đại học quang trung](*entity\_university*) có chỉ tiêu là bao nhiêu

### 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 người dùng hoặc nếu không thì sẽ chọn ngẫu nhiên các mẫu câu trả lời để tạo cảm giác không bị nhàm chán.

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

- [Utterance Actions](https://rasa.com/docs/rasa/core/actions/#id4):

Để xác định một hành động phát biếu (ActionUtterTemplate), ta thêm một phản hồi của chatbot vào domain.yml bắt đầu bằng utter\_:

**utter\_welcome**:  
- **text**: xin chào, bạn cần giúp gì nào?  
**utter\_goodbye**:  
- **text**: tạm biệt, mong những điều trên giúp ích được cho bạn!  
**utter\_ask\_entity\_university**:  
- **text**: Bạn muốn hỏi trường nào?  
**utter\_ask\_entity\_major**:  
- **text**: Bạn muốn hỏi ngành nào?

Thông thường, bắt đầu tên của một hành động phát biểu bằng utter\_. Nếu tiền tố này bị thiếu, ta vẫn có thể sử dụng phản hồi (response) trong các hành động tùy chỉnh của mình (custom actions), nhưng phản hồi không thể dự đoán trực tiếp như hành động của chính nó.

- [Default Actions](https://rasa.com/docs/rasa/core/actions/#id4):

Các hành động default như lắng nghe người dùng, restart lại hội thoại hoặc trả lời mặc định khi không phân loại được ý định người dùng.

**action\_default\_fallback**:  
- **text**: Tôi chưa thể giúp bạn việc này nhưng tôi vẫn đang cố gắng để hoàn thiện hơn.

- Custom actions:

Khi tập các câu trả lời mẫu không áp dụng được với các câu trả lời cần có kết quả lấy từ một nguồn dữ liệu khác thì action tùy biến được sử dụng, nó sẽ trỏ đến một hàm trong lớp action (python). Trong đây sẽ tùy biến câu trả lời như lấy dữ liệu từ database rồi điền vào tham số trong câu trả lời. Ở đây một đoạn code lấy dữ liệu từ database và trả lời cho câu hỏi chỉ tiêu của các trường đại học.

**def** submit(self, dispatcher, tracker, domain):  
 **try**:  
 entity\_university = self.get\_slot(**'entity\_university'**)[0]  
 entity\_university\_validated = self.get\_slot(**'entity\_university\_validated'**)[0]  
 **except**:  
 entity\_university = **None** entity\_university\_validated = **None  
  
 if** entity\_university\_validated != **None**:  
 **try**:  
 dt = db.admission.find\_one({ **'university\_id'**: entity\_university\_validated})  
 data = dt[**'admission'**]  
 dispatcher.utter\_message(**"sau đây là thông tin chỉ tiêu trường "** + entity\_university)  
 dispatcher.utter\_message(data)  
 **except**:  
 dispatcher.utter\_message(**"không tìm thấy thông tin tuyển sinh trường "** + entity\_university)  
 **return** [AllSlotsReset()]

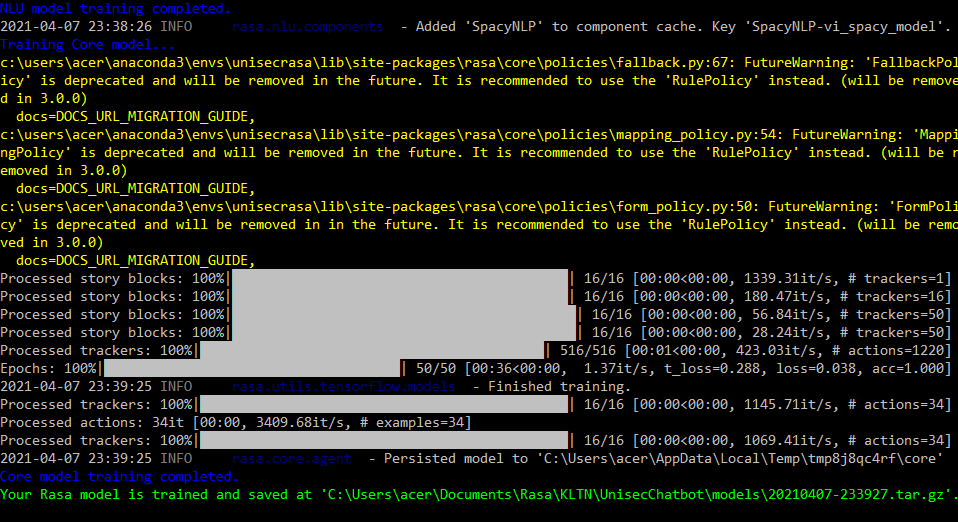
### 3.2.5. Xây dựng khung kịch bản (stories)

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

***## story\_welcome*\*** intent\_welcome  
**-** utter\_welcome  
**-** action\_welcome  
***## story\_goodbye*\*** intent\_goodbye  
**-** utter\_goodbye  
***## story\_find\_university*\*** intent\_find\_university  
**-** form\_find\_university  
**-** form{"name" : "form\_find\_university"}  
**-** form{"name" : null}  
  
***## story ask admission score*\*** intent\_admission\_score  
**-** form\_admission\_score  
**-** form{"name" : "form\_admission\_score"}  
**-** form{"name" : null}  
  
***## story\_combine*\*** intent\_combine  
**-** form\_combine  
**-** form{"name" : "form\_combine"}  
**-** form{"name" : null}  
  
***## story\_university\_address*\*** intent\_university\_address  
**-** form\_university\_address  
**-** form{"name" : "form\_university\_address"}  
**-** form{"name" : null}  
  
***## story university\_id*\*** intent\_university\_id  
**-** form\_university\_id  
**-** form{"name" : "form\_university\_id"}  
**-** form{"name" : null}

**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.



Hình 3.1: Đào tạo một mô hình

Việc xây dựng đoạn hội thoại này có thể viết bằng tay. Thông qua cửa sổ dòng lệnh người dùng có thể sử dụng để đào tạo cho chatbot.



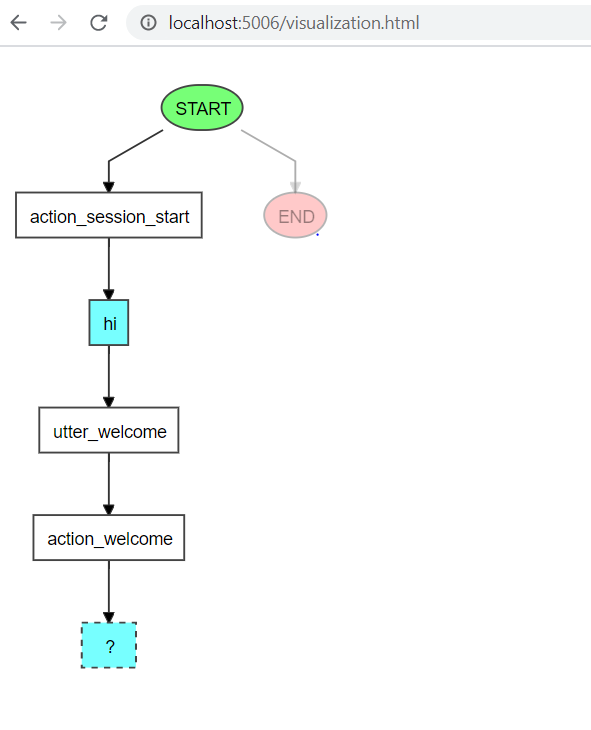
Hình 3.2: Đào tạo cho chatbot dạng shell

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

Hình 3.3: Chế độ đào tạo cho chatbot bằng Interactive Learning

Trong ví dụ trên đây, khi người dùng gõ vào từ “hi”, NLU sẽ phân loại ý định đây là dạng “welcome” (chào hỏi), và hỏi xem có đúng không (Yes/No). Xác định ý định hợp lý thì chúng ta xác nhận đúng (Yes) để tiếp tục giúp đào tạo cho chatbot đưa ra câu phản hồi cho người dùng tương ứng (utter\_welcome).

Trong quá trình học tập tương tác, Rasa sẽ vẽ sơ đồ cuộc hội thoại hiện tại và một vài cuộc hội thoại tương tự từ dữ liệu đào tạo. Ta có thể xem trực quan hóa cuộc hội thoai ngay sau khi bắt đầu học tương tác.



Hình 3.4: Trực quan hóa cuộc hội thoại

## 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 2.0.0, dựa trên ngôn ngữ lập trình python 3.7. Giao diện người dùng sử dụng nền tảng mobile/android .

### 3.3.2. Thiết kế

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



API

Front-End

NLG

Xin chào

DM

NLU

API

Xin chào, tôi có thể giúp gì cho bạn nào?

Hình 3.5: Kiến trúc chung của hệ thống

Font-end: sử dụng giao diện mobile application với mục tiêu minh họa một ứng dụng chat để người dùng có thể tương tác với bot.

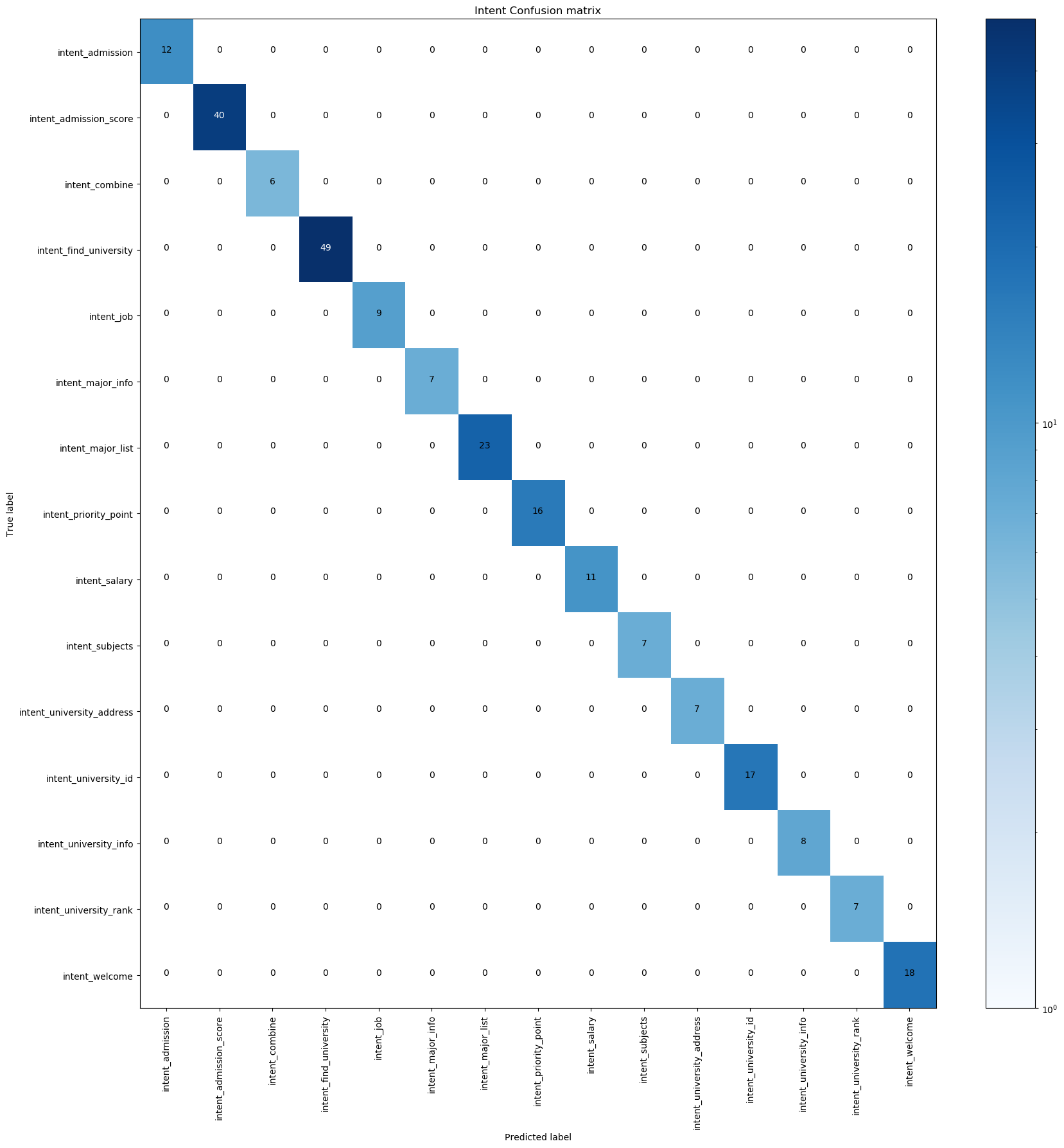
* Mỗi khi có một người dùng gửi tin nhắn cho chatbot thì nội dung tin này sẽ gửi một POST request đến webhook được sử dụng để lắng nghe sự kiện. Webhoook này sẽ chuyển tiếp đến bộ NLU của RASA.
* RASA nhận diện ý định. Sau khi đã thu được message của người dùng thì sử dụng RASA để hiểu được ý định của 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, bao gồm cả việc gọi API để lấy thông tin trả lời 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

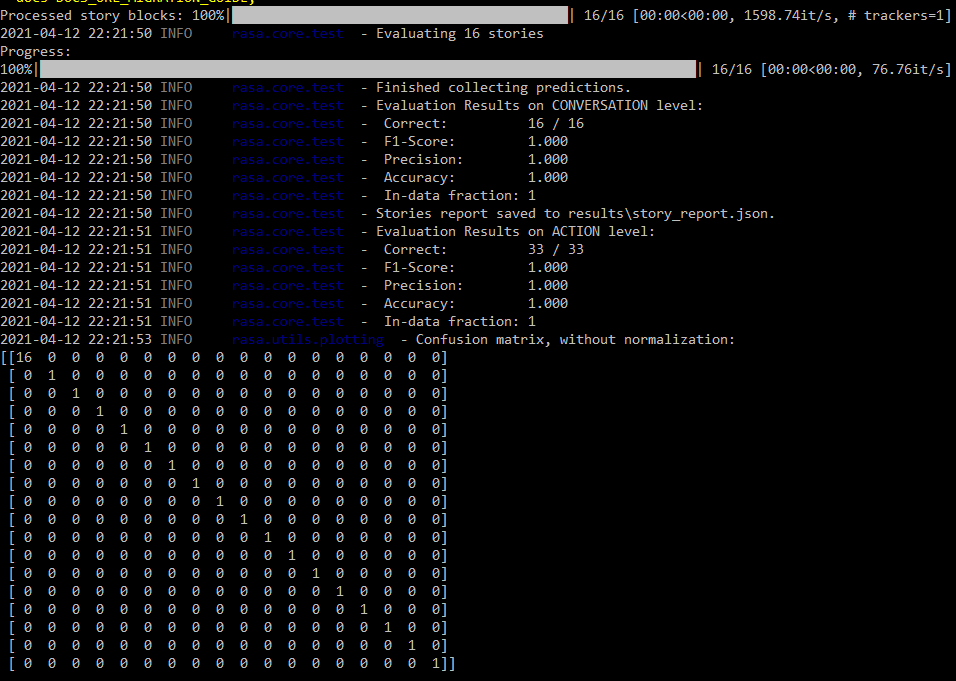
1. Kết quả thử nghiệm

Kết quả đánh giá mô hình NLU và Rasa Core sau khi thực hiện đào tạo chatbot và kiểm tra trên dữ liệu test có 316 câu dữ liệu người dùng nhập vào:

Thực hiện thử nghiệm tương tác với Chatbot với một số câu hỏi gần với kịch bản đã đào tạo cho chatbot ta thu được được ma trận hỗn loạn của các ý định như hình 3.6.



Hình 3.6: Ma trận hỗn loạn ý định người dùng

Hình 3.7: Đánh giá mô hình Rasa Core

Tính chung kết quả test trên tập dữ liệu test end-to-end cho độ chính xác 100%

Với NLU model:

Correct:316/316

F1-Score: 1.000

Precision: 1.000

Accuracy: 1000

Với Rasa core :

Correct:16/16

F1-Score: 1.000

Precision: 1.000

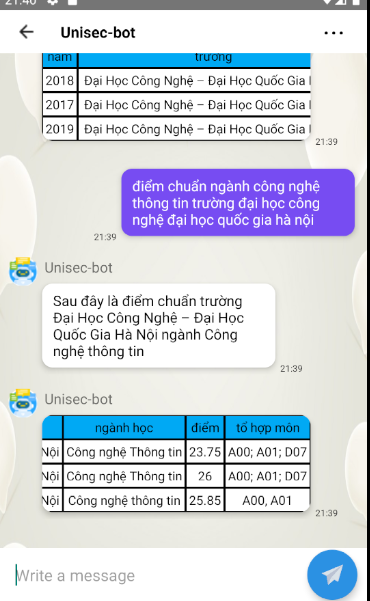
Accuracy: 1000

1. Test trên giao diện người dùng cuối



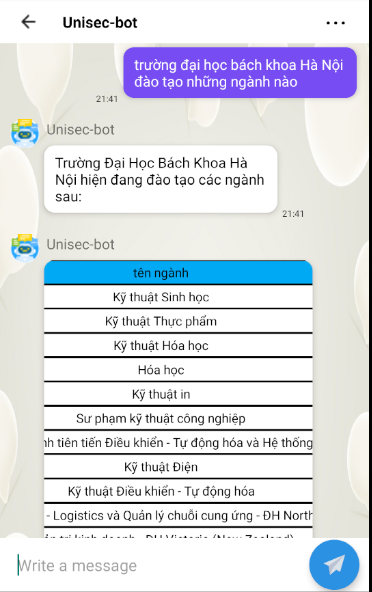
Hình 3.8: Hỏi về mã của các trường đại học

Khi người dùng nhập yêu cầu hỏi mã của trường đại học hệ thống sẽ ghi nhận ý định của người dùng theo những ý định đã được lập trình sẵn trên hệ thống và các thực thể là tên của trường đại học và mã của trường đại học. Hệ thống sau đó sẽ phản hồi lại người dùng những nội dung tương ứng dưới dạng text như hình 3.8.



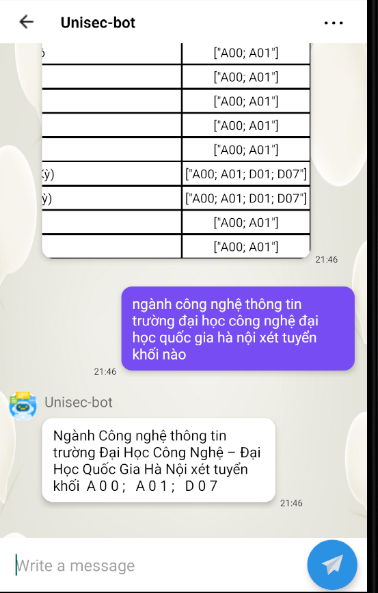
Hình 3.9: Hỏi về điểm chuẩn của các trường đại học

Tương tự khi người dùng nhập nội dung hỏi về điểm chuẩn của ngành của một trường đại học hệ thống sẽ ghi nhận ý định người dùng và các thực thể để xử lý. Như hình 3.9 các thực thể ở đây lần lượt là “ngành công nghệ thông tin” và “ trường đại học công nghệ đại học quốc gia hà nội”. Sau khi xử lý hệ thống sẽ phản hồi lại điểm chuẩn của ngành theo các năm dưới dạng bảng.



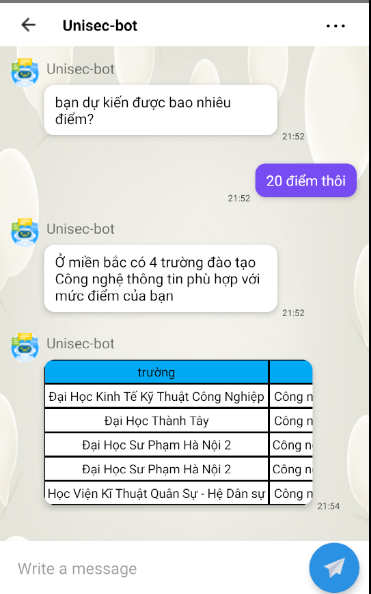
Hình 3.10: Hỏi về danh sách các ngành của trường đại học

Khi người dùng nhập nội dung hỏi về danh sách các ngành của trường đại học hệ thống sẽ ghi nhận ý định của người dùng và thực thể là tên trường đại học. Hệ thống sẽ phản hồi lại người dùng danh sách những ngành của trường đại học theo những dữ liệu được lưu trữ trên database.



Hình 3.11: Hỏi về khối xét tuyển của ngành của trường đại học

Tương tự khi người dùng hỏi về tổ hợp môn xét tuyển của từng ngành của các trường đại học. Hệ thống sẽ phản hồi lại thông tin tổ hợp môn xét tuyển tương ứng theo ngành của trường đại học dưới dạng text.



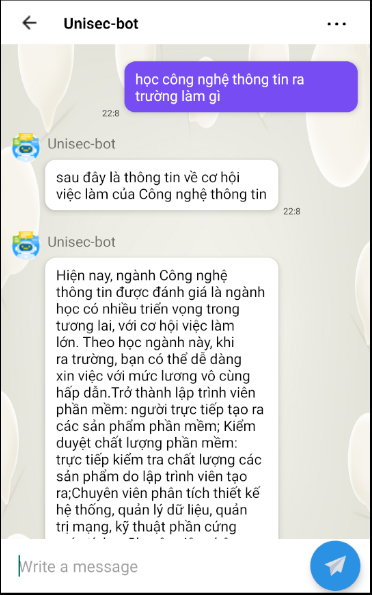
Hình 3.12: Tìm kiếm các trường đại học phù hợp

Khi người yêu cầu tìm kiếm các trường đại học với các thông tin như ngành đào tạo, khu vực hay điểm chuẩn. Hệ thống sẽ ghi nhận ý định và phản hồi lại người dùng những câu hỏi để thu thập những thông tin còn thiếu và phản hồi lại câu trả lời cuối cùng dưới dạng bảng là danh sách các trường đại học phù hợp với những thông tin mà người dùng cung cấp.



Hình 3.13: Hỏi về thông tin của các trường đại học

Tương tự khi người dùng hỏi thông tin của các trường đại học hệ thống sẽ ghi nhận lại ý định của người dùng và những thực thể từ câu hỏi của người dùng để phản hồi lại những thông tin tương ứng. Như hình 3.13 với từng ý định tương ứng là hỏi về địa chỉ, xếp hạng và chỉ tiêu tuyển sinh của trường đại học. Thực thể được ghi nhận là “ trường đại học thương mại”. Hệ thống sẽ phản hồi lại những thông tin dưới dạng text phù hợp với ý định và thực thể của người dùng đã cung cấp.



Hình 3.14: Giới thiệu ngành học

Tương tự khi người dùng yêu cầu các nội dung liên quan đến các ngành học. Hệ thống sẽ ghi nhận lại ý định của người dùng và thực thể là tên của ngành. Hệ thống sẽ phản hồi lại thông tin của ngành dưới dạng text cho người dùng tương ứng với yêu cầu.



Hình 3.15: Tư vấn chọn ngành phù hợp

Khi người dùng yêu cầu tư vấn chọn ngành phù hợp hệ thống đầu tiên sẽ phản hồi lại danh sách các lĩnh vực đã được thiết kế từ trước cho người dùng lựa chọn. Sau đó hệ thống sẽ phản hồi lần lượt là các câu hỏi để thu thập thông tin liên quan đến những điểm mạnh những môn học yêu thích và nơi làm việc mong muốn trong tương lai của người dùng. Cuối cùng hệ thống sẽ phản hồi ba chuyên ngành dựa trên những thông tin mà người dùng cung cấp.



Hình 3. 16: Hỏi về điểm cộng của các khu vực

Khi người dùng yêu cầu thông tin về điểm cộng của các vùng. Hệ thống sẽ ghi nhận ý định và thực thể là tên vùng. Hệ thống sẽ phản hồi lại lần lượt là cách tính điểm cộng của từng khu vực và danh sách các khu vực của vùng.

## 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 tuyển sinh đại học, có tính thực tiễn cao, và hoàn toàn áp dụng được ngay trong thực tiễn.

# Chương 4. Kết luận và hướng phát triển

## 4.1. 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 tư vấn tuyển sinh đại học. 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 tuyển sinh đại học 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 90%).
* Chatbot được đào tạo với lượng dữ liệu đào tạo tương đối tốt với hơn 316 câu dữ liệu mẫu người dùng nhập vào. Chatbot hỗ trợ tương đối nhiều ý định (hơn 20 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 2.3.2, ngôn ngữ lập trình python 3.7.6, sử dụng chat client dạng mobile (android). 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.

## 4.2. Hướng phát triển

Trong những phiên bản tiếp theo hệ thống sẽ khắc phục những hạn chế và có thể được mở rộng để trả lời nhiều hơn nữa những câu hỏi của người dùng.

Một số tính năng có thể được phát triển như sau:

* 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…

# Danh mục tài liệu tham khảo

1. Denis Rothman. Artificial Intelligence By Example. Develop machine intelligence from scratch using real artificial intelligence use cases. (2018).
2. Boris Galitsky. Chatbot Components and Architectures. In: Developing Enterprise Chatbots. (2019).
3. Abhishek Singh & Karthik Ramasubramanian & Shrey Shivam. Building an Enterprise Chatbot: Work with Protected Enterprise Data Using Open Source Frameworks. (2019).
4. Tom Bocklisch. Conversational AI with Rasa NLU & Rasa Core. (2018).
5. Daniel Jurafsky, James Martin. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. (2018).
6. Sumit Raj. Building Chatbots with Python Using Natural Language Processing and Machine Learning. (2019).
7. Hussain S., Ameri Sianaki O., Ababneh N. A Survey on Conversational Agents/Chatbots Classification and Design Techniques. In: Barolli L., Takizawa M., Xhafa F., Enokido T. (eds) Web, Artificial Intelligence and Network Applications. WAINA 2019. Advances in Intelligent Systems and Computing. (2019).
8. Adamopoulou E., Moussiades L. An Overview of Chatbot Technology. In: Maglogiannis I., Iliadis L., Pimenidis E. (eds) Artificial Intelligence Applications and Innovations. AIAI 2020. IFIP Advances in Information and Communication Technology. (2020).
9. Jiao Liu & Yanling Li & Min Lin. Review of Intent Detection Methods in the Human-Machine Dialogue System. Journal of Physics: Conference Series. (2019).
10. Yun-Nung (Vivian) Chen, Asli Celikyilmaz and Dilek Hakkani-Tur. Deep Learning for Dialogue Systems. Association for Computational Linguistics. (2018).
11. Allen Downey, Jeff Elkner and Chris Meyers. How to think link acomputer scientist: Learning with Python. (2002).
12. Rasa Documents: <https://rasa.com/docs/rasa/>
13. A brief history of chatbots – Timeline: <https://roboticsbiz.com/a-brief-history-of-chatbots-timeline/>
14. Underthesea (Open-source Vietnamese Natural Language Process Toolkit): <https://github.com/undertheseanlp/underthesea>
15. [Ranking Web of Universities:](Ranking%20Web%20of%20Universities:%20%20) <http://www.webometrics.info/en/asia/vietnam>