Conversational AI for Primary Healthcare Support Advice

Vishesh Bhagat
210010283

Supervisors: Dr Alice Toniolo , Dr Phong Le

Master of Science (MSc) in Artificial Intelligence
School Of Computer Science
16 August 2022

University of
St Andrews

FOUNDED
1413

Abstract

Conversational AI for Primary Healthcare Support Advice

For common illnesses, citizens of the internet often search information on the internet manually, read through complicated medical blogs and understand the appropriate suggested treatment as per their capacity. The information available on the internet may not be in a format that is easy to search, interpret and consume. With this project, we aim to develop a prototype of an AI-based conversational agent commonly known as a chatbot which will be trained with information about common illnesses and corresponding symptoms. Users will be able to have a natural communication with the agent similar to what they would have with a healthcare representative during their initial screening at a medical establishment. Users will input the symptoms they are observing and an agent will respond with one or more possible illnesses that provided symptoms listed against the illness. With this ready-to-consume knowledge in simplified form, users will be saved from the hassle to go through complicated online documentation. This will also reduce the chances of users reading or interpreting information incorrectly.

We intend to make use of state-of-the-art NLP techniques for developing the conversational agent to demonstrate how the above-mentioned problem can be tackled. We will make use of transformers and transfer learnings for core NLP tasks and will orchestrate the conversation using the open-source conversational AI platform. With the use of NLP techniques, an agent will be intelligently able to identify the user's requirement and generate a dynamic response as opposed to a state machine-based conversation.

Keywords

NLP, Transformer, Conversational AI, Transfer Learning

Acknowledgements

I would like to thank my supervisor, Dr Alice Toniolo and Dr Phong Le, for their unwavering enthusiasm regarding this project and the invaluable guidance and support given to me throughout the project.
I thank you for enabling me to choose a topic of my interest and providing new and interesting exploration into the field of natural language processing.

I would also like to thank my family for their unconditional support and encouragement throughout my education and especially during the master’s degree at the University of St Andrews where we were not able to meet for extended periods.

Declaration of Authorship

Date: 26st August 2022

Vishesh Bhagat

Table of Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
NLP	Natural Language Processing
NLU	Natural Language Understanding
NLG	Natural Language Generation
RNN	Recurrent Neural Network
NN	Neural Network
ML	Machine Learning
DL	Deep Learning
HMM	Hidden Markov Model
POS	Part-of-speech
LSTM	Long Short-Term Memory
BERT	Bidirectional Encoder Representations from Transformers
GPT	Generative Pre-trained Transformer
CNN	Convolutional Neural Network
FAQ	Frequently Asked Questions
ASR	Automated Speech Recognition
DM	Dialogue Management
TTS	Text to speech
NER	Named Entity Recognition

1 Introduction

Conversational Artificial Intelligence (AI) is a system that primarily makes use of techniques from the natural language processing field of artificial intelligence and machine learning techniques to generate humanly interactions with the system. When combined with state-of-the-art interaction techniques, interacting with these systems is as if users are interacting with other humans.

Kong (2021) argues that voice-based and conversation-based interactions are poised to replace the traditional web-based or command-line interfaces in many applications. Interactions with digital systems are becoming more natural. Applications of these intelligent systems are pervasive and getting adopted rapidly.

A Conversational AI agent also referred to as a chatbot is a system that uses full conversational dialogue to accomplish one or more tasks. Command interpreters use enough dialogue to interpret and execute a single command. Event classifiers just read the message and perform an action based on the content. Enhancing customer buying experience using guided conversation (chatbots), using natural language or voice interface to interact with devices (command interpreter) or sorting emails into folders (event classification) are representative examples of how AI assistants are changing the world around us (Freed, 2021).

Conversational AI provides convenient interactions, is more effective and efficient and hence is important to adopt.

1.1 Importance of the project

"Don't become a mere recorder of facts, but try to penetrate the mystery of their origin."

— *Ivan Pavlov*

As discussed by (Palash Goyal, 2018), the initial work on chatbot date back to 1966 when ELIZA was introduced by Joseph Weizenbaum. However, the recent state of the art in natural language processing and advancements in computing technologies has given a great thrust to the field of conversational AI. As per the research on insider intelligence(Dolan, 2022), 72.3 per cent of the time on mobile devices is was spend on smartphones. This is a significant amount of time and conversational AI techniques can greatly simplify interactions between customers and companies. The use of advanced language modes and transformers is showing promising results in building chatbots that are much more than just a state machine. The project aim to explore these techniques to build an intelligent chatbot.

Heatly life is the basis of a prosperous life. In the healthcare domain, there is a good amount of reliable knowledge available on the internet from authentic sources such as government websites and healthcare research groups. Despite this, common users find it difficult to navigate through the complex healthcare jargon and understand and use this information. Demands for healthcare professional has risen significantly and continues to grow. Many healthcare systems are under tremendous pressure, and this limits the number of patients that can be treated on time. Patients find it difficult to get treatment on time even for simple and mild illnesses. This delay in early treatment exaggerates the symptoms and can lead to further health complications. Having a chatbot-like system can come to the rescue of patients and healthcare services as well. Patients can

ask about their health issues to the chatbot in a fashion like how they will interact with doctors and nurses. They don’t have to be experts in the medical field to interact with chatbots.
Chatbots will be intelligent enough to understand a user’s problem, search its knowledge base for possible solutions and present it to the user in an understandable format. Chatbots can be trained to handle mundane administrative tasks and reduce the work pressure on healthcare services.
Advanced chatbots can be equipped with voice-based conversation capabilities. With such advanced features, these bots can virtually replace any front desk presents in hospitals and medical services. A chatbot can complete patient registration and data gathering in a timely fashion without waiting for any person. Chatbots can also be integrated with local pharmacies for ordering medicines and medical supplies.

Healthcare chatbots are AI-powered conversational solutions that help patients and healthcare service providers to connect easily. Chatbots can play a critical role in making first-line service available to everyone. Patients can use chatbot systems round the clock and get their queries answered. This project aims at providing easy to use interface with which a common user can extract enough information about the possible illness. There are endless possibilities about what chatbots can do. Some of the interesting capabilities can be symptoms checker and triage, self-care advice, health risk assessment, chronic condition monitoring, appointment booking, medication reminder and tracker, healthcare tracker, and much more. Chatbots can make vast medical knowledge available to patients in need. Patients don’t have to wait for doctors’ availability for basic illnesses.

Due to the sensitive nature of data in the healthcare domain, healthcare agencies need to be careful while making technological choices. Data protection rights have the utmost importance and failing to comply with the law and privacy may result in significant damage to an organization (financial, reputational and many other damages). We aim to explore conversational AI platforms that give developers and organizations more control and freedom over the system and its components rather than integrating the data and logic into prebuilt less customizable products.

1.2 Objectives

With this project, we aim to build a prototype of a conversational AI for disease and illness matching by following the objectives mentioned below

1.2.1 Primary objective

The primary objectives focus on the development of a basic conversational AI, similar to an interactive Q&A system, where the user mentions all the symptoms in a single interaction and receives information on the most likely illness matching these symptoms. The following objectives aim at building a conversation agent with these capabilities:

- Prepare data for training the natural language understanding unit
- Develop the natural language understanding unit. The system should be able to read question input and extract appropriate entities and intent
- Prepare scenarios for natural language generations
- Develop the natural language generation part where an agent can respond to queries
- Data processing for symptom and disease matching

1.2.2 Secondary objective

The conversation agent’s dialogue management can be extended for a more coherent conversation with the end users. The below objectives aim to enhance dialogue management of the agent:

- Gradual information gathering of symptoms from the user
- Extending the dialogue with back and forth exchange of information to achieve a more humanly conversation

1.2.3 Tertiary objective

User experience can be enhanced with a user interface and better techniques for conversational agents' knowledge representation. Following objective aim to enhance user experience:

- Web-based graphical user interface (GUI) for the chatbot
- Improving conversational features by exploring techniques like Knowledge representation using a knowledge graph

2 Context Survey

In this chapter, we will review the state-of-the-art of NLP techniques and conversational platforms available in the market. We will discuss the scholarly and industrial chatbot developments in the domain of healthcare.

2.1 Natural Language Processing

As defined by (Wikipedia, 2022),
"Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data."

Freed (Freed, 2021) argues that every intelligent application involving human language has some NLP behind it. The below table summarizes various NLP tasks and corresponding popular applications identified by him.

Table 1:NLP Tasks and Applications

NLP Task	General Applications
Text classification	Spam classification
Information Extraction	Calendar Event Extraction
Conversational Agent	Personal Assistance
Information Retrieval	Search Engines
Question Answering System	Legal entity Extraction

Language is a structured system of communication that involves complex combinations of its constituent components, such as characters, words, sentences, etc. Linguistics is the systematic study of language. To study NLP, it is important to understand some concepts from linguistics about how language is structured. We can think of human language as composed of four major building blocks: phonemes, morphemes and lexemes, syntax, and context. NLP applications need knowledge of different levels of these building blocks, starting from the basic sounds of language (phonemes) to texts with some meaningful expressions or context (Vajjala et al., 2020).

Table 2: Language blocks and NLP Applications

Blocks of Language	Applications
Context (meaning)	Summarization Topic Modelling Sentiment Analysis
Syntax (phrases and sentences)	Parsing Entity Extraction Relation Extraction
Morphemes and Lexemes (words)	Tokenization Word embedding POS tagging
Phonemes (speech and sounds)	Speech to text Speaker Identification Text to speech

2.2 Challenges in NLP

Language has inherent ambiguity in a general sense as well. (Perera et al., 2013) have discussed some of the common challenges that NLP faces in clinical settings and below are the challenges for most of the common NLP tasks as described by (Vajjala et al., 2020)

Ambiguity: Most human languages are inherently ambiguous. Many times sentence has multiple meanings and the meaning is decided by the context around the sentence. We can draw multiple meanings from the sentence "not bad" depending on the context used.

Common knowledge: Humans use common knowledge all the time to understand and process any language. One of the key challenges in NLP is how to encode all the things that are common knowledge to humans in a computational model.

Creativity: Humans are creative, and language is no exception for creativity. Various styles, dialects, genres, and variations are used in any language. Making machines understand creativity is a hard problem not just in NLP, but in AI in general.

Diversity across languages: For most languages in the world, there is no direct mapping between the vocabularies of any two languages. This makes porting an NLP solution from one language to another hard.

2.3 Machine Learning, Deep Learning, and NLP

The boundaries and overlaps between machine learning, deep learning and NLP are well discussed by (Mukhopadhyay, 2018).

He argues that Artificial intelligence (AI) is a subfield of computer science that tries to create systems that can do activities that would normally need human intelligence. Machine learning (ML) is a field of artificial intelligence that focuses on the creation of algorithms that can learn to do tasks automatically based on a large number of instances without the need for hand-crafted rules. Deep learning (DL) is a type of machine learning that uses artificial neural network designs to learn.

He also highlights that While NLP, ML, and DL have some overlap, they are also quite independent fields of study. Rules and heuristics were also used in early NLP applications. However, in recent decades, ML approaches have had a significant effect on the development of NLP applications. More recently, DL has been widely developed and applied to natural language processing (NLP) systems.

2.4 Approaches to NLP

A variety of approaches have been identified for solving NLP tasks. As argued by (Vajjala et al., 2020), the approaches can be categorized as follow:

2.4.1 Heuristics-Based NLP

Early attempts at constructing NLP systems, like other early AI systems, were based on creating rules for the task at hand. This necessitated the developers having some domain knowledge in ways to construct rules that could be put into a system. Such systems also needed dictionaries and thesauruses. More extensive knowledge bases have been constructed to facilitate NLP in general and rule-based NLP in particular, in addition to dictionaries and thesauruses. Wordnet (Miller, 1995) (Miller, 1995), for example, is a database of words and the semantic ties that exist between them. More recently, common-sense world knowledge has been included in knowledge bases such as Open Mind Common Sense (Singh et al., 2002), which supports rule-based systems. Regexes are a common paradigm for creating rule-based systems, and NLP software like StanfordCoreNLP contains a framework for developing them. CFG stands for context-free grammar and is a sort of formal grammar used to model natural languages. Grammar languages like JAPE (Java Annotation Patterns Engine) may be used to model more sophisticated rules.

2.4.2 Machine Learning for NLP

For many NLP applications, supervised machine learning approaches such as classification and regression algorithms are widely used. The extraction of features from the text, the use of the feature representation to develop a model, and the evaluation and improvement of the model are all typical phases in any machine learning technique for NLP. Some of the commonly used ML algorithms as listed by (Vajjala et al., 2020) are Naive Bayes and support vector machine (SVM) for classification tasks, hidden Markov model (HMM) conditional random field (CRF) for part-of-speech (POS) tagging.

2.4.3 Deep Learning for NLP

The artificial intelligence field has seen a big increase in the use of neural networks to deal with complicated tasks in recent years. Language is naturally unstructured and complicated. Neural network (NN) models are better at representing the complexity of language and producing better outcomes.

Recurrent neural networks (RNNs) are specifically intended to keep such sequential processing and learning in mind since language is fundamentally sequential. RNNs have neural units that can remember what they've processed previously. This memory is temporal, and when the RNN reads the next word in the input, it stores and updates the information at each time step.

The problem of forgetting memory is a challenge that RNNs face. To address this problem, long short-term memory networks (LSTMs), a form of RNN, were developed. LSTMs get around this difficulty by ignoring irrelevant information and memorising just the parts of it that are important to the job at hand. This alleviates the burden of memorising a large amount of information in a single vector representation. Because of this solution, LSTMs have largely replaced RNNs in many applications. GRUs are a type of RNN that is mostly utilised in language generation.

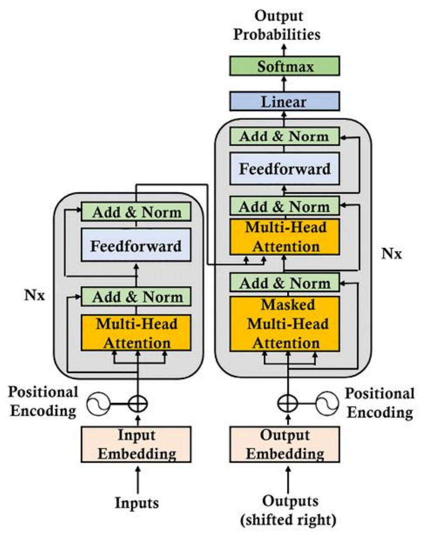
Convolutional neural networks (CNNs) are widely employed in computer vision applications such as image classification and video recognition, among others. CNN has also shown promise in NLP, particularly in text categorization. The capacity of CNNs to use a context window to look at a collection of words together is their major benefit (Vajjala et al., 2020).

2.5 Transformers

Transformers (Wolf et al., 2020) is the most recent addition to the league of deep learning NLP models. The transformer model was released in 2017 (Vaswani et al., 2017), and it performed amazing results on machine translation tasks. In the last two years, Transformer models have surpassed state-of-the-art in practically all key NLP tasks. They model the textual context, but not in the order in which it appears. Rather they prefer to look at all the words surrounding it known as self-attention and represent each word in its context when given a word in the input.

According to (Vajjala et al., 2020), the transformer's huge success has sparked the interest of numerous NLP researchers. They've created even more fantastic Transformer-based models. Generative Pre-trained Transformer (GPT) and Bidirectional Encoder Representations from Transformers (BERT) are two of the most well-known and essential of these models. GPT is entirely made up of the decoder layer of the Transformer, whereas BERT is entirely made up of the encoder layer of the Transformer. The purpose of GPT is to create text that appears to be written by a human. BERT's purpose is to give a better language representation to aid a variety of downstream activities (sentence-pair classification tasks, single-sentence classification tasks, question-answering (QA) tasks, and single-sentence tagging tasks) in achieving better outcomes.

Figure 1: The transformer - model architecture



Note: From (Vaswani et al., 2017)

Figure 1 shows the original transformer architecture proposed by Vaswani et al. (2017) which is based on a self-attention mechanism relating to different positions of a single sequence (positional encoding) to make sense of the entire sequence. The original transformer model was employed for the language translation task.

Each encoder structure converts an input sequence of tokens into a sequence of embedding vectors, often called the hidden state or context and the structure consists of two layers: multi-head self-attention mechanisms layer and position-wise fully connected fee forward layer.

Each decoder structure uses the encoder's hidden state to iteratively generate an output sequence of tokens, one token at a time. The structure has an additional layer to perform multi-head attention over the output of the encoder stack.

Only the lowest level of the stack contains the embedding sublayer. The encoded input is guaranteed to remain stable through all other layers because there isn't an embedding layer in the other layers.

The self-attention which is a token-to-token operation has replaced the recurrence function that is present in RNN, LSTM or CNN. The attention mechanism will determine how each word in a sequence, including the word being processed, relates to every other word in the sequence which results in finding a deeper relationship between the words and producing better results. This layer applies three independent linear transformations to each embedding to generate the query, key, and value vectors. These transformations project the embeddings, and each projection has a unique set of parameters that can be learned. This enables the self-attention layer to concentrate on various semantic facets of the sequence. Having several heads allows the model to focus on several aspects at once and that is achieved with multi-head attention.

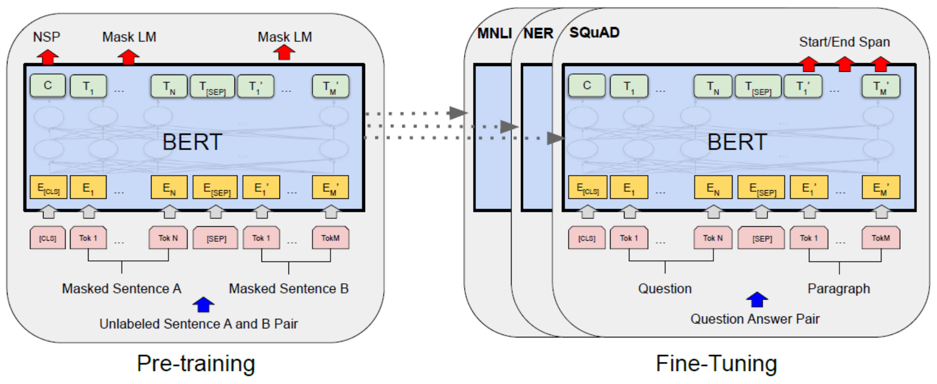
The positional encoding layer adds positional information to input embedding which helps in establishing relationships between words which are semantically closely related but far apart in the sentence (positionally).

2.6 BERT

Bidirectional Encoder Representations from Transformers (BERT) is an encoder-only architecture of transformer. As argued by Devlin et al.(2018) BERT is pre-trained with the two objectives of predicting masked tokens in texts (masked language modelling task or MLM)and determining if one text passage is likely to follow another(next sentence prediction or NSP).

BERT has two steps: pre-training and fine-tuning. During pre-training, the model is trained on unlabelled over different pre-training tasks. For finetuning, the BERT model is first initialized with the pre-trained parameters and then these parameters are fined tune using labelled data. Each downstream task has separate fine-tuned models though they were initialized with the same pre-trained parameters. During fine-tuning, task-specific inputs and outputs are plugged into BERT and all the parameters are fine-tuned. Compared to pre-training, fine-tuning is relatively inexpensive.

Figure 2: Overall pre-training and fine-tuning procedures for BERT



The above figure shows overall pre-training and fine-tuning procedures for BERT(Devlin et al., 2018).

2.7 Transfer Learning

Transfer Learning (TL) is an AI approach in which information obtained while addressing one problem is used for a related but different problem. Pre-trained transformers are embedded. Large transformers have recently been employed in the transfer learning of smaller downstream activities. TL needs less training data. In a chatbot domain, there usually is not much training data. With TL, we can achieve much better performance on the limited amount of training data. TL makes training faster and only needs a few training epochs to fine-tune a model for a new task. Generally, it is much faster and more efficient than the traditional ML method (Tunstall et al., 2022).

2.8 The Hugging Face Hub

The Hugging Face (Hugging Face, 2022) provides a standardized interface to a wide range of transformer models as well as code and tools to adapt these models to new use cases. TensorFlow, PyTorch and JAX are the three main deep learning frameworks that the Hugging Face presently supports, and it makes it simple to transition between them. It also gives task-specific heads so you can fine-tune transformers on downstream tasks like text classification, named entity recognition, and question answering. Because of this, it takes less time to train and test a variety of models. The ecosystem consists of mainly two parts: a family of libraries and the Hub. The libraries provide the code while the Hub provides the pre-trained model weights, datasets, scripts for the evaluation metrics, and more.

https://github.com/heyunh2015/diseaseBERT

2.9 Chatbots

A chatbot is a computer programme that can converse with people via text or speech. Chatbots are divided into two categories based on their goals: task-oriented bots and chitchat bots. Task-oriented bots aim to do certain tasks through engaging with humans, such as purchasing a flight for someone, whereas chitchat bots are more like real beings—their purpose is to answer users' messages easily, exactly like in natural chitchat. Some example scenarios in which a chatbot may have an advantage are Hospital reception or medical consulting, Online shopping customer service, After-sales service, Investment consulting, and Bank services.

The standard method for creating a chatbot has been established. Developing a chatbot generally comprises five distinct parts, which are shown below:

- ASR to convert user speech into text
- NLU to interpret user input
- DM to make decisions on the next action concerning the current dialogue status
- Natural-language generation (NLG) to generate text-based responses to the user
- TTS to convert text output into voice

2.9.1 Need for chatbot:

- Waiting time
- Require huge human resources
- Investment / skilled employees
- Infrastructure cost
- Commonly asked questions

Add a layer of a computer program that can take inputs in a natural language, process the information and generate an appropriate response.

2.9.2 Types of chatbots:

- Rule-based
  - Question
  - answer
- Conversation-based – virtual assistance
  - Question
  - Answer
  - Question referring to the above question
  - Answer

2.9.3 Conversational AI Frameworks

There are two types of solutions for creating chatbots: closed-source solutions and open-source solutions. The downsides of closed source systems include high costs, vendor lock-in, the possibility of data leaking, and the inability to develop personalised functionality. These issues do not exist with open-source solutions.

Microsoft

Microsoft offers separate Azure Cognitive Services: Language Understanding Intelligent Service for natural language understanding and Bot Framework for dialogue and response (Microsoft, 2022).

Amazon

Amazon Lex is the primary service used for building AI assistants and integrates easily with Amazon's other cloud-based services as well as external interfaces (Amazon Web Services, 2019).

Google

Google's Dialogflow is a service by Google cloud for building Lifelike conversational AI with state-of-the-art virtual agents (Google, 2022).

IBM

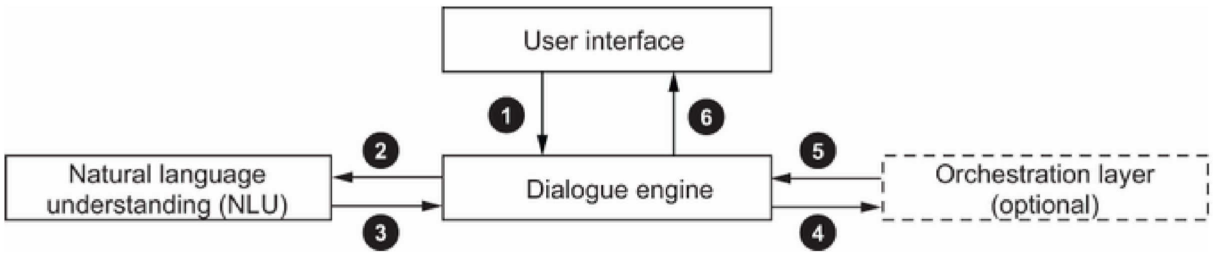
Watson Assistant is IBM's AI assistant platform for building conversational AI experiences for businesses. (IBM, 2022).

RASA

Rasa is an open-source solution with all the industry-standard features: built-in enterprise-grade concurrency capabilities, rich functions covering all the needs of chatbots, rich documents and tutorials, and a huge global community. Rasa provides you with complete control over the applications that you deploy. Other platforms allow you to control the classifier by changing the training data, but Rasa allows you to customize the entire classification process.

2.10 Closes Look at Generalized AI (In progress)

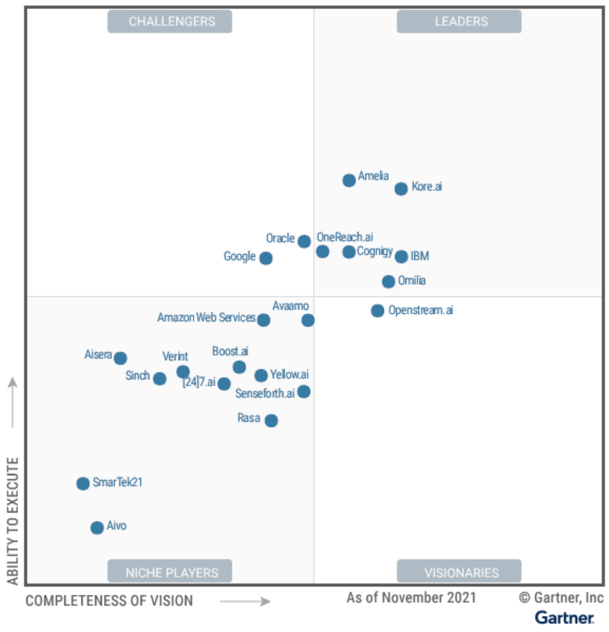
Figure 3: Generalized AI Assistant architecture and control flow


User interface
1
6
2
5
Natural language understanding (NLU)
Dialogue engine
Orchestration layer (optional)
3
4

2.11 Rasa

The chatbot market has seen increased interest and rapid adoption over the last five years. During that time, the products and platforms enabling the building of chatbots have been extremely diverse and have evolved considerably. These solutions can be divided into two types: closed source solutions and open source solutions. Closed source solutions come with a high cost, vendor lock-in, risk of data leakage, and the inability to implement custom functions. Open source solutions have an advantage in this regard. A downside of open source solutions is that users need to be careful in choosing the framework, make sure it is scalable and concurrent and has wide community support.

Rasa is the only open-source, industry-grade conversational AI framework that meets these requirements. Many companies have successfully built their chatbots using Rasa. Rasa has been recognized as a Niche Player in Gartner®Magic Quadrant™(Magnus Revang). For the privacy-conscious health care domain, the open source nature of Rasa offers complete control over all the aspects of chatbot development and operations.



2.12 Related Work

The dataset required for a typical healthcare chatbot is proposed in a paper titled HealFavor(Ur Rahman Khilji et al.). The paper talks about data sourcing, data quality, pre-processing, and representation. Researchers have suggested prototype system architecture and proposed user experience surveys as an evaluation criterion for the system. (Sheth et al., 2019) focuses on Contextualization and Personalization of Patient's Data. discuss how existing chatbot systems can be extended by a whole ecosystem of the Internet of things for better and personalized health tracking of individuals. It has mentioned the usefulness of knowledge graphs in data representation. A paper on mental healthcare highlights the usefulness of BiLSTM (Bi-directional LSTM) and Sequence-to-Sequence (Seq2Seq) encoder-decoder architecture and has used the Bilingual Evaluation Understudy (BLEU) score for model evaluation. A paper by IIT Delhi researchers (Pandey et al.) has worked on studying the Q&A support system for maternal and child health in rural India. A paper published by researchers at Digital Health (Nadarzynski et al., 2019) has conducted an in-depth study of the Acceptability of artificial intelligence (AI)-led chatbot services in healthcare. They investigated participants' willingness to interact with AI-powered health chatbots. Researchers at MDPI have studied the feasibility of developing a rule-based virtual caregiver system using a mobile chatbot for elderly people. A paper written by students and professors at Vishwakarma Institute of Technology Pune, India has studied using deep learning for developing contextual chatbots (Kandpal et al.).

A paper by Yu et al. (2020) has studied the use of a bi-directional transformer for financial service chatbots. They have shown how the BERT model outperformed other methods for common NLP tasks like intent classification, sentence completion, information retrieval and question answering. A paper from Microsoft researchers (Damani et al., 2020) has discussed optimized transformers for

FAQ answering. A paper by hugging face researchers (Wolf et al., 2020) has a detailed explanation of how transformers have reshaped state-of-the-art natural language processing. A paper published by MODUL Technology GmbH (Brasoveanu & Andonie) focuses on explaining Transformer architectures through visualizations. Gillioz et al. have provided an Overview of the Transformer-based Models for NLP Tasks.

3 Requirements specification

3.1 Purpose

The system will be used by a patient who wants to understand their symptoms and disease in more detail. This is a prototype and should not be considered as a replacement for any medical advice or diagnosis.

3.2 Functional Requirements

The system must be able to achieve the following functionalities:

- The agent should introduce itself and its purpose
- The system should be able to accept text inputs from the user
- The agent should be able to determine if patients want to discuss their symptoms or just informing about their good health
- When a patient is not well, the agent should be able to ask for symptoms, collect and try to match those symptoms with its disease dataset to determine possible disease
- The agent should be able to answer specific questions outside the current conversation context
- The agent should be able to ask for clarifications when information shared by the user is not clear

3.3 Non-functional Requirements

As an end user of the system, the patient should be able to interact with the system via easy to use interface. Complexities of the system should be transparent to the user. Users need not be healthcare experts to make complete use of the system.

4 Software engineering process

4.1 Planning

The initial planning of the project was done in the first two weeks of the project work. A Gantt chart with details of tasks and allocated efforts for each task were prepared. The chart is attached in the appendix.

4.2 Development

I followed the agile development methodology. During the chatbot development, we worked on incremental feature delivery. This helped us in breaking the bot development process into smaller workable tasks and focusing on working on the most important task at a time.

4.3 Tracking

Similar to sprint review meetings, a weekly meeting helped us in reviewing the project progress. Thursday meetings were used for all the critical discussions, brainstorming and tracking of the work done in the week and work items for the coming week. MS Teams channel was used for maintaining the backlog items.

4.4 Testing

Manual Testing: Initial bot testing was performed manually using rasa interactive command line features.

Automated Testing: We plan to use the RASA Testing framework to perform chatbot automated testing.

4.5 Version Control

The GitHub repository was used for maintaining the code base of the project. It is a private repository and access can be provided on a need basis until it is made publicly available.

The Github repository link: https://github.com/RightWrite/HealthAgent.git

4.6 Final Artefacts

The RASA train creates trained models in the archive format with a naming convention `<timestamp>.tar.gz`.

User interface artefacts are static and do not need compilations. Detailed instructions on using the artefacts are mentioned in the README.md hosted in the version control.

5 Ethics

Ethics evaluation is done following the artefact evaluation form. Throughout the development and evaluation, no personal information was collected. The system is a prototype and is not supposed to replace any existing healthcare advice mechanisms.

5.1 System Access:

The evaluation will be performed in line with the artefact evaluation form by demonstrating the chatbot to the staff or students at the University. We will only use dummy scenarios with hypothetical medical conditions to demonstrate the capabilities of the system. We will collect anonymous feedback on the quality of the artefact only using the Qualtrics university survey system. Users will get access to only pre-recorded behaviour of the system.

5.2 Data Usage and Accessibility:

The data used in the research is available publicly Only the researcher will have access to the raw evaluation data and this data will be destroyed at the end of the project. Aggregated data and results will be published in the final dissertation report.

5.2.1 Pre-trained data:

The Spacy language model used for Disease NER is trained on BC5CDR corpus and is made available under Apache License 2.0. The corpus is published by the National Library of Medicine of the National Centre for Biotechnology Information. The corpus has 6,892 diseases mentioned. More information about the corpus and datasets can be found on https://www.ncbi.nlm.nih.gov/CBBresearch/Dogan/DISEASE/

5.2.2 Disease dataset for symptom matching:

The data used for symptom matching is sourced from the Kaggle platform (https://www.kaggle.com/datasets/priya1207/diseases-dataset). It contains information about diseases, symptoms and medical treatments scrapped from MayoClinic (https://www.mayoclinic.org/). The dataset does not contain any personally identifiable information (PII).

6 Design

The proposed prototype of an AI-enabled conversational virtual agent named Health Agent is built on top of an open-source conversational AI platform called Rasa.

6.1 Conversation flow design

Below is an example of dialogue that was covered for the primary objective.

The 👤 symbol represents a human while the 🤖 symbol represents the chatbot.

6.1.1 Conversation scenario of a healthy patient:

👤 Hi

🤖 Hello, I am your health advisor, how are you feeling today?

👤 I am not feeling well.

🤖 What symptoms do you observe

👤 I am suffering from diarrhoea and Vomits.

🤖 These are the possible Symptopons of Rotavirus

6.1.2 Conversation scenario of an unhealthy patient:

👤 Hi

🤖 Hello, I am your health advisor, how are you feeling today?

👤 I am fine.

🤖 Great! Keep doing well

6.1.3 Advanced Conversation scenario of an unhealthy patient: future work

🤖 Hello, I am your health advisor, how are you feeling today?

👤 I am not feeling well.

🤖 What symptoms do you observe

👤 diarrhea and Vomits

🤖 Any other Symptom?

👤 I am feeling tired

🤖 What else you feel?

👤 dehydration, dry mouth

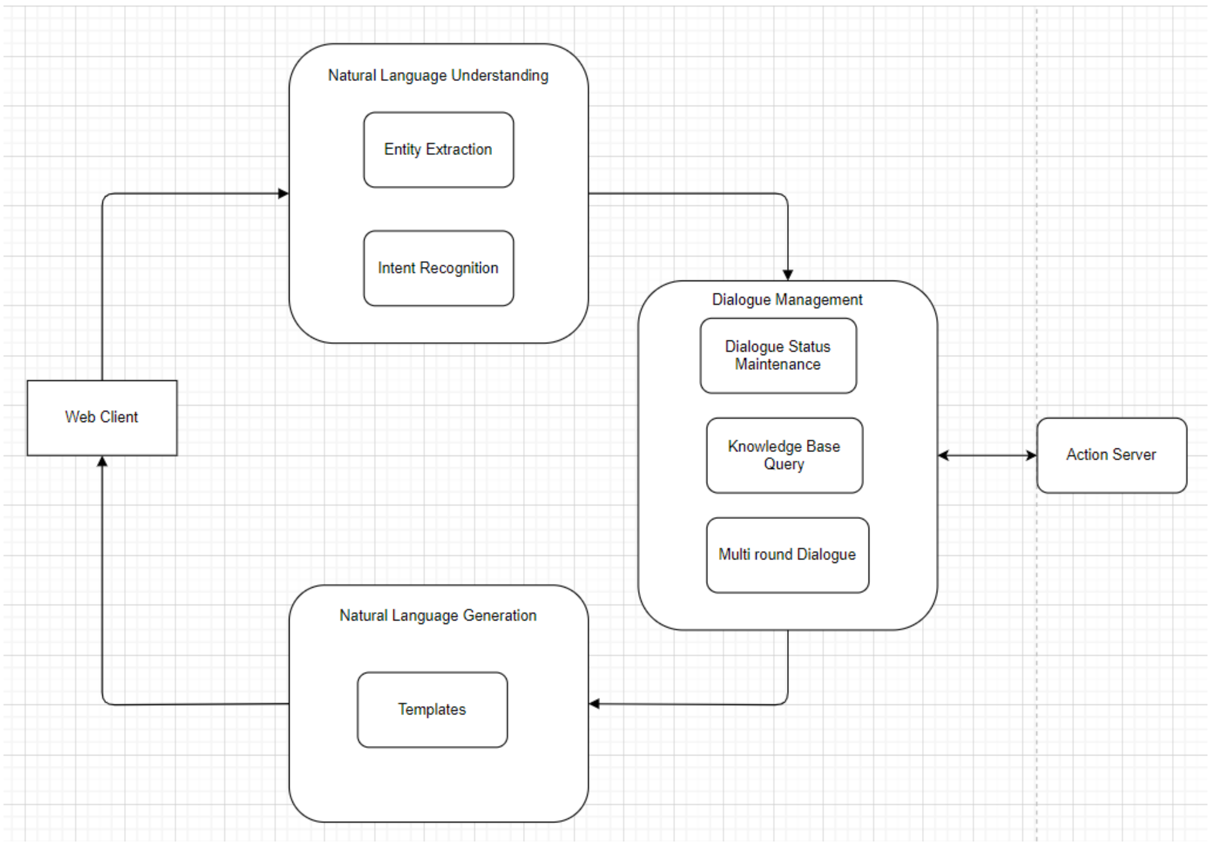
Anything else?

No

These are the possible Symptopons of Rotavirus.

6.2 Architecture

Health Agent is a Rasa-powered chatbot that interacts with users through a web-based interface and manages dialogue using state-of-the-art (SOTA) techniques in natural language processing.

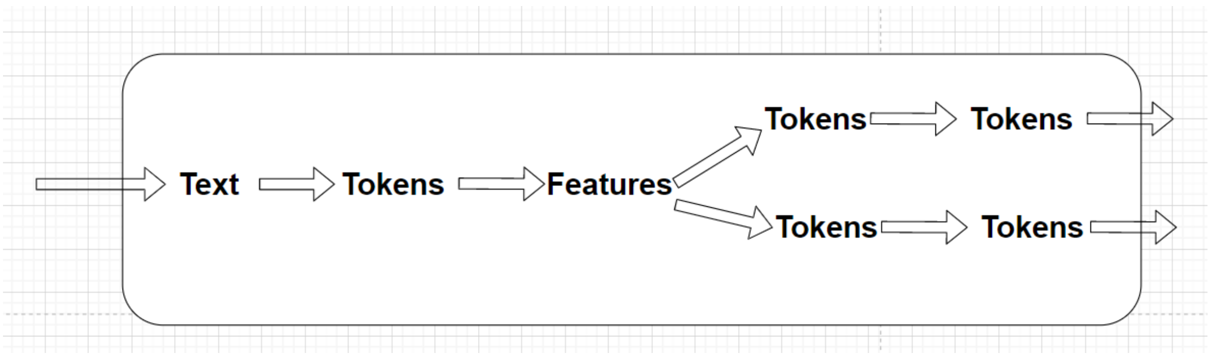


<ADD USER/SYSTEM messages to the diagram>

This architecture includes the following primary components:

6.2.1 Natural Language Understanding Unit (NLU)

NLU interprets text-based user inputs. With Rasa, NLU is a data processing pipeline that converts unstructured user messages into intents and entities. The pipeline consists of a series of configurable and customizable components.
In Rasa, config.yml file defines steps and components of the NLU pipeline. The pipeline takes text as input and processes it to generate intents and entities as output.



There are different types of components that you can expect to find in a pipeline. The main ones are:

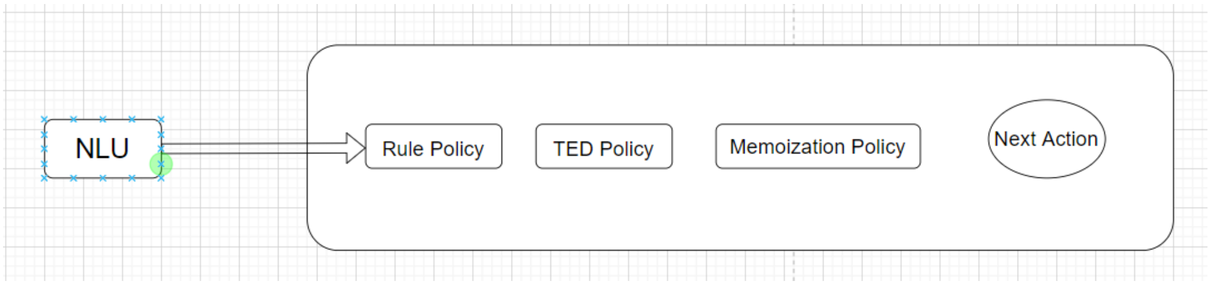
- **Tokenizers**: Splits text into tokens
- **Featurizers:** Returns sequence features and sentence features
- **Intent Classifiers:** assign one of the predefined intents to incoming user messages
- **Entity Extractors:** extracts entities from the user message

NLU also has a response selector component used to predict a bot response from a set of candidate responses (Bocklisch et al., 2017).

Choice of language model: accuracy vs speed

6.2.2 Dialogue Management (DM)

The main task of the DM module is to coordinate and manage the whole conversation flow and is particularly important for multi-turn task-oriented dialogue. It also predicts the next best action that the agent needs to perform.



Rasa provides rule-based, machine learning based and hybrid models for DM. The health agent makes use of rule-based and machine learning-based models. Rule-based models are useful for handling strict conversational behaviour where the next action depends on the current turn. Machine learning-based models consider the entire dialogue context, the previous dialogue turns, extracted entities and slots for dialogue management. TED policy and Memoization policy are used in the ML-based approach. DM module also provides fallback capabilities in case of ambiguous intent (Rustamov et al., 2021). If the intent is not clear, DM instructs the agent to either repeat the question or send fall back response to let the user know about the ambiguity.

6.2.3 Natural Language Generation (NLG)

NLG is almost the last challenging mile in human-machine interaction. Once the next action is chosen by DM, NLG generates the text of the response to the user. In other words, NLG converts the agent's response into human-readable text. There are broadly two ways of doing this: template-based method and deep learning (DL) based method. The template-based method creates a response without much flexibility or variety. DL-based methods are capable of generating a quite dynamic response, however, it is difficult to control the quality and stability of the result.

In the case of task-oriented chatbots like Health Agent, users need accurate and concise responses to their queries. Hence we have used a template-based NLU in our architecture. We can add some level of flexibility to the agent’s response by creating a pool of templates.

6.2.4 Action Server / Backend:

The DM module may require interactions with databases, APIs or third-party integration to get extra information to generate responses to user queries or complete the intended task. DM might be interested in implementing custom actions which might be complicated as compared to built-in response generation. For all such scenarios, Rasa provides an action server that runs custom actions for a Rasa Open Source conversational assistant.

When your assistant predicts a custom action, the Rasa server sends a POST request to the action server with a JSON payload including the name of the predicted action, the conversation ID, the contents of the tracker and the contents of the domain. When the action server finishes running a custom action, it returns a JSON payload of responses and events. The Rasa server then returns the responses to the user and adds the events to the conversation tracker.

6.2.5 Web Client

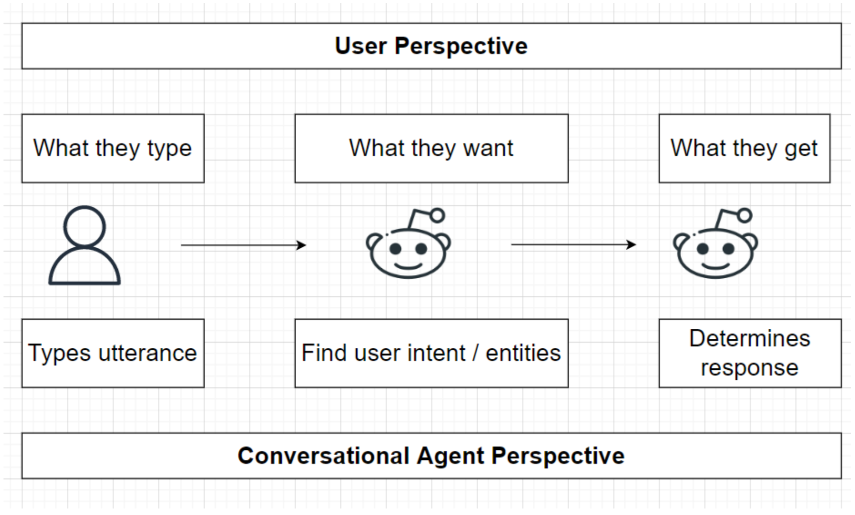
The web client is responsible for collecting text input from the user and delivering it to Rasa NLU. It also renders the response generated by NLG and presents it to the user. Rasa open source does not provide a built-in GUI client. Developers can integrate Rasa open source with a channel or web interface of their choice.

6.3 Conversational AI Terminologies:

Refwerce book here

- **Utterance**: The user interacts with the agent through natural language. Whatever user types in the web client is an utterance.
- **Response:** A response is whatever the assistant returns to the user. It can be textual or multimedia.
- **Intent:** An intent is a normalization of what the user means by their utterance or what they want to do.
- **Entity:** An entity is a noun-based term or phrase. An entity can be any important detail that your assistant could use later in a conversation
  See Figure 1

Figure 4 Conversation Perspectives



6.4 Additional Rasa concepts:

Metioned the rasa referece

- **Stories:** This is training data that is used to train agents' dialogue management systems. It is represented in the form of a conversation between user and agent. Stories are made up of intents, actions, entities and forms. Based on the stories, models can generalize to unseen conversations.
- **Rules:** Rule is also part of the dialogue management unit's training data. They cover the strict conversation path between user and agent.
- **Slots:** The slots act as long-term memory of the agent. Information stored in slots is generally used later in the decision-making process. Slots are generally filled using entities but not mandatorily.
- **Forms:** Slot filling is a common conversation pattern used to collect pieces of information from a user to do something. Rasa recommends the use of rasa forms for slot filling. It is a controlled way of slot filling from extracted entities or text (and is customizable).

6.5 Rasa Data Files

Rasa uses YAML as a way to manage all the training data. The training data consists of NLU data, stories and rules.

NLU data defines all the intents that the agent needs to use for intent classification. It also has information about extracting entities from user text.

Stories and Rules are the representation of a conversation between user and assistant.

6.6 Rasa Config Files

- **config.yml:** Defines components as well as policies that the Rasa trained model will use for making predictions after accepting user inputs.
- **credentials.yml:** It contains credentials for voice and chat platforms that the agent will be integrated with.

- **domain.yml:** It holds all the information on which agent needs to operate. It specifies intents, entities, slots, responses, forms, actions and session configurations.
- **endpoints.yml:** It contains all the API endpoints and their configuration that the agent can use.

6.7 Rasa policies

TBD

6.8 Spacy Language models

Transformer models:

6.9 DIET

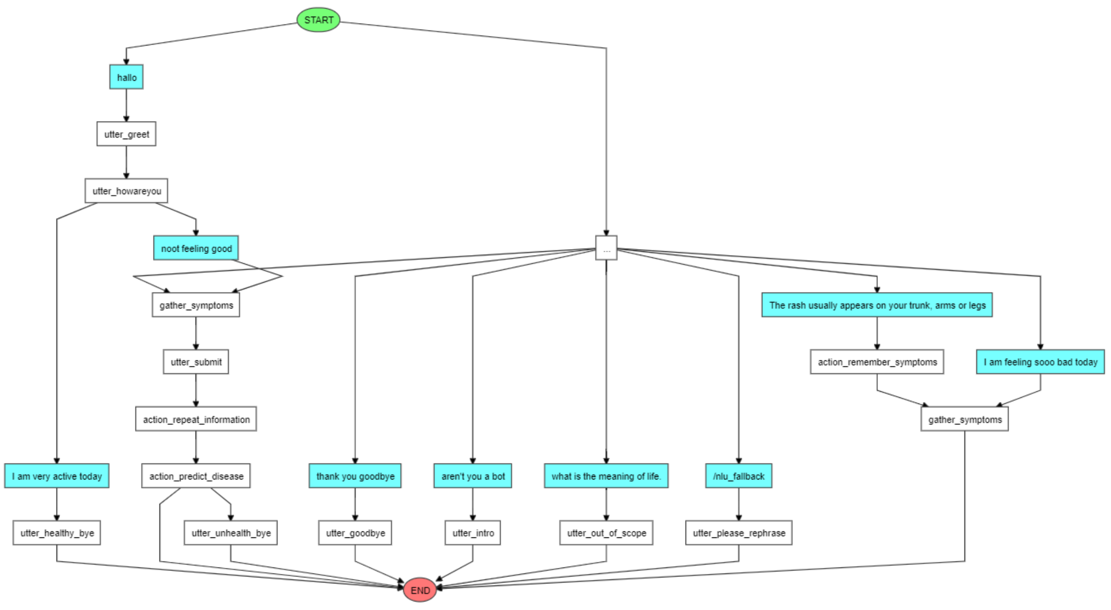
7 Implementation

The health agent (HA) is developed using the open-source conversational AI framework developed by Rasa. The HA is a task-oriented and closed domain chatbot.

What we had to do to get the system to run. What is the structure…what we need to provide to rasa to run the functionality

The figure 5 shows the visual representation of stories that consititues conversationa between

Figure 5: Health Agent Conversation Flow Graph



7.1 NLU data generation

NLU is responsible for understanding the user's intent and the information user is entering.

The NLU data consists of intents that the agent can understand and sentence patterns corresponding to the intent. Rasa intents may contain patterns for entity extraction but this is optional.

7.1.1 Intents

The table below summarizes the intents that the bot can process:

Table :3 NLU Intent data

Intent	Function	Example	Number of examples
greet	A user wants to start a conversation	Hey	
goodbye	A user wants to end a conversation	I'm done	
healthy_patient	A user wants to inform the bot that he is healthy	I am perfectly fine	
unhealthy_patient	A user wants to inform the bot that he is unhealthy	I am sick	
inform_symptoms	A user wants to inform symptoms	Seems like I have	

		a headache	
bot_challenge	A user wants to know about the agent	are you real?	

We also use NLU data to train the model for entity extraction. Below is an example pattern that Rasa use to identify predefined entity types from the input sentence.

- I am observing **[chest pain] (DISEASE)**

As described in (Kong et al., 2021), In the training dataset, entities are annotated and written as the Markdown URL expression, namely **[entity value] (entity type)**.

7.1.2 Entities

Entities are used to extract specific information from the text entered by the user. The chatbot has a **DISEASE** entity for extracting symptoms from the user's input text. The NLP pipeline is equipped with Spacy's en_ner_b5cdr_md model for disease data extraction.

7.2 Response data generation

NLG is responsible for the generation of responses. The response is generated based on the user's input.

The table below summarizes the responses that the bot can generate:

Table 4: NLG Resposne data

Response	Function
utter_greet	Bot greeting to the user
utter_healthy_bye	Bot saying bye to healthy user
utter_unhealth_bye	Bot saying bye to unhealthy user
~~utter_goodbye~~	Bot saying bye
utter_ask_disease	Bot asking for symptom information
utter_repeat_information	Bot repeating collected information
utter_iamabot	Bot answering who it is
utter_out_of_scope	Bot answering out-of-scope questions
utter_please_rephrase	Bot asking the user to paraphrase ambiguous text
utter_submit	Bot informing the user about submitting data for processing
action_predict_disease	Bot informing information about disease prediction
~~utter_ask_more_symptoms~~	Bot asking the user for more symptoms

7.3 Dialogue Data Generation

Dialogue management data consists of stories and rules for understanding conversation flow between user and bot.

The stories should vaguely cover most of the conversational scenarios that are expected to be covered by the system. Stories and Rules combined with policies are used to predict the next action(s) in each turn.

7.3.1 Stories

The table below summarizes the flexible conversation flows that the bot can handle:

Table 5: Story data for DM

Story	Function
Health patient	Healthy patient conversation flow
Unhealthy patient informing symptoms	Unhealthy patient conversation flow

7.3.2 Rules

The table below summarizes the strict conversation flows the bot can handle:

Table 6: *Rule Data for DM*

Rule Name	Conversational Flow
Utter goodbye anytime the user says goodbye	User's goodbye
Utter bot information anytime user asks about bot	A user asking about the bot
Handle out-of-scope intents	A user asking anything outside the bot's scope
Ask the user to rephrase the ambiguous text	A user entering ambiguous text
Activate gather_symptoms	Gathering symptoms from a user
Submit gather_symptoms	Submitting symptoms collected from a user
Add symptom to symptoms list slot	A user wants to inform symptoms

7.3.3 Slots

Table 7

Slot Data for DM

Slots	Function
disease	Store a symptom (if any) informed in the last input text
symptoms	Store all the symptoms informed

7.3.4 Forms

Dialogue management also has a rasa form called **gather_symptoms** for the systematic gathering of symptom entities.

7.4 Training the model

In NLU, incoming messages are processed by a sequence of components and enriched to a machine-understandable format. As per rasa documentation (Rasa Technologies Inc., 2022), training the NLU model is controlled by components defined in the configuration file. The model can be finetuned by configuring parameters in the configuration file. The Botfront (Botfront, n.d.) has discussed some of the pipeline optimization techniques for better intent and entity extraction.

7.4.1 NER Language model

We have used the ScispaCy Language model en_ner_bc5cdr_md (Neumann et al., 2019) which is a fined finer-grained NER model trained on BC5CDR dataset for diseases and chemicals. It loads pre-trained models with pre-trained word vectors in the NLU pipeline. All the other spacy components used in the pipeline relied on the spaCy structures. The model is configured to be case insensitive.

7.4.2 Tokenizer

What Why How

Tokenizers split received text into tokens. As the system is using the spaCy language model, the pipeline is configured with SpacyTokenizer as recommended by rasa(Rasa Technologies Inc., 2022).

7.4.3 Featurizer

The NLU pipeline is configured with SpacyFeaturizer, RegexFeaturizer, and LanguageModelFeaturizer.

SpacyFeaturizer is used by the Spacy NLP pipeline for entity extraction.

LanguageModelFeaturizer is based on HuggingFace pre-trained model LaBSE developed by Rasa. It creates features for entity extraction, intent classification, and response selection using the configured pre-train model. Tokenizer components should appear before featureziers (Rasa Technologies Inc., 2022).

WHY?

7.4.4 Entity extractor

The pipeline makes use of SpacyEntityExtractor. Although the spacy language model *en_ner_bc5cdr_md* can perform NER for disease and chemical entities, the system only makes use of DISEASE dimension of the model.

Dual Intent Entity Transformer (DIET) based DIETClassifier is also capable of extracting the entities and is used along with the SpacyEntityExtractor for disease NER.

WHY?

7.4.5 Intent classifier

The system uses DIETClassifier for intent identification and classification purpose. DIET makes use of pre-trained word embedding made available through LanguageModelFeaturizer. DIET exposes many hyperparameters available to tune the model. As discussed by Botfront (Botfront, n.d.), some of the key parameters have been set.

<Add sample NLU response showing intent and entity extraction>

A FallbackClassifier is configured to handle cases where the NLU intent classification score becomes ambiguous. The threshold parameter of the classifier ensures that if no intent is classified with a confidence level greater than the threshold, a fallback intent is triggered. The ambiguity threshold parameter is used to maintain enough confidence score difference between the highest ranked intents.

7.5 Action Server

As documented by rasa (Rasa Technologies Inc., 2022) the action server acts as a backend system of the chatbot and executes custom actions and any other business logic.

7.5.1 Custom actions

The chatbot system is using the action server for the following two reasons:

- To collect the symptoms entered by the user
  A rasa custom action is used to extract all the DISEASE entities identified by the NLU and add them to the 'symptoms' slot
- To call the backend service for identifying the disease matching the collected symptoms
  This rasa custom action sends the collected symptoms to a disease matching service and renders the response of the service to the user.

7.5.2 Disease and symptom matcher

This is a backend service and is a part of the action server. It received a list of symptoms and matches it with the symptom list of diseases present in the database. It rates each matched disease based on the number of matches symptoms.

<Add the example response>

7.6 Data preprocessing

The disease data obtained from the Kaggle (https://www.kaggle.com/datasets/priya1207/diseases-dataset) required some preprocessing.

The original dataset contained the following columns:

- name
- link
- symptoms
- causes
- risk_factor
- overview
- treatment
- medication
- home_remedies

Of these, 'name' and 'symptoms' were considered for further processing. An additional column for the symptom list was created by extracting symptoms from the 'symptoms' column of the dataset We applied Spacy en_ner_bc5cdr_md mode for disease NER (Neumann et al., 2019).

8 Evaluation

In the NLU pipeline, the two important tasks are intent classification and entity extraction.

As explained by Géron (Géron, 2019), the main performance metrics for classification tasks are based on the following measures:

- True Positive (TP): model correctly predicts the positive class
- True Negative (TN): model correctly predicts the negative class
- False Positive (FP): model incorrectly predicts positive class
- False Negative (FN): model incorrectly predicts negative class

Equation 1 - Precision

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

Equation 2 - Recall

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

Equation 3 – F_1 score

$$F_1 = 2 \times \frac{precision \ \times recall}{precision + recall}$$

Equation 4 - Accuracy

$$Accuracy = \frac{TP + TN}{TP + TF + \ FP + FN}$$

Intent extraction is a multi class classification problem and we apply above metrics to evaluate how good the system performacen.

8.1 Experimental Setup

Table 8 details the configuration of the system that was used for conducting the experiments.

Table 8: *Experimental Setup*

Specification	**Details**
System Type	Laptop Running Windows 10
Processor and Cores	Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz
Memory (RAM)	20.0 GB

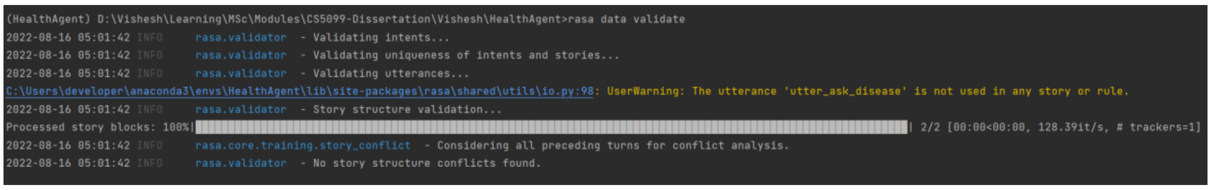
Operating System	Windows 10 Home Single Language (Version 21H2)
Python Version	Python 3.8
Python Environment Manager	Conda Environment
Development IDE	PyCharm 2022.1.3 (Professional Edition)

8.2 Data validation

There should not be inconsistencies in the domain, NLU data, story data or rule data.

Rasa data validation utility ensures there are no mistakes between domain, NLU and story data. Th

Figure 6: Rasa Data Validation



```
(HealthAgent) D:\Vishesh\Learning\MSc\Modules\CS5099-Dissertation\Vishesh\HealthAgent>rasa data validate
2022-08-16 05:01:42 INFO     rasa.validator  - Validating intents...
2022-08-16 05:01:42 INFO     rasa.validator  - Validating uniqueness of intents and stories...
2022-08-16 05:01:42 INFO     rasa.validator  - Validating utterances...
C:\Users\developer\anaconda3\envs\HealthAgent\lib\site-packages\rasa\shared\utils\io.py:98: UserWarning: The utterance 'utter_ask_disease' is not used in any story or rule.
2022-08-16 05:01:42 INFO     rasa.validator  - Story structure validation...
Processed story blocks: 100%|████████████████████████████████████████████████████████████████| 2/2 [00:00<00:00, 128.39it/s, # trackers=1]
2022-08-16 05:01:42 INFO     rasa.core.training.story_conflict  - Considering all preceding turns for conflict analysis.
2022-08-16 05:01:42 INFO     rasa.validator  - No story structure conflicts found.
```

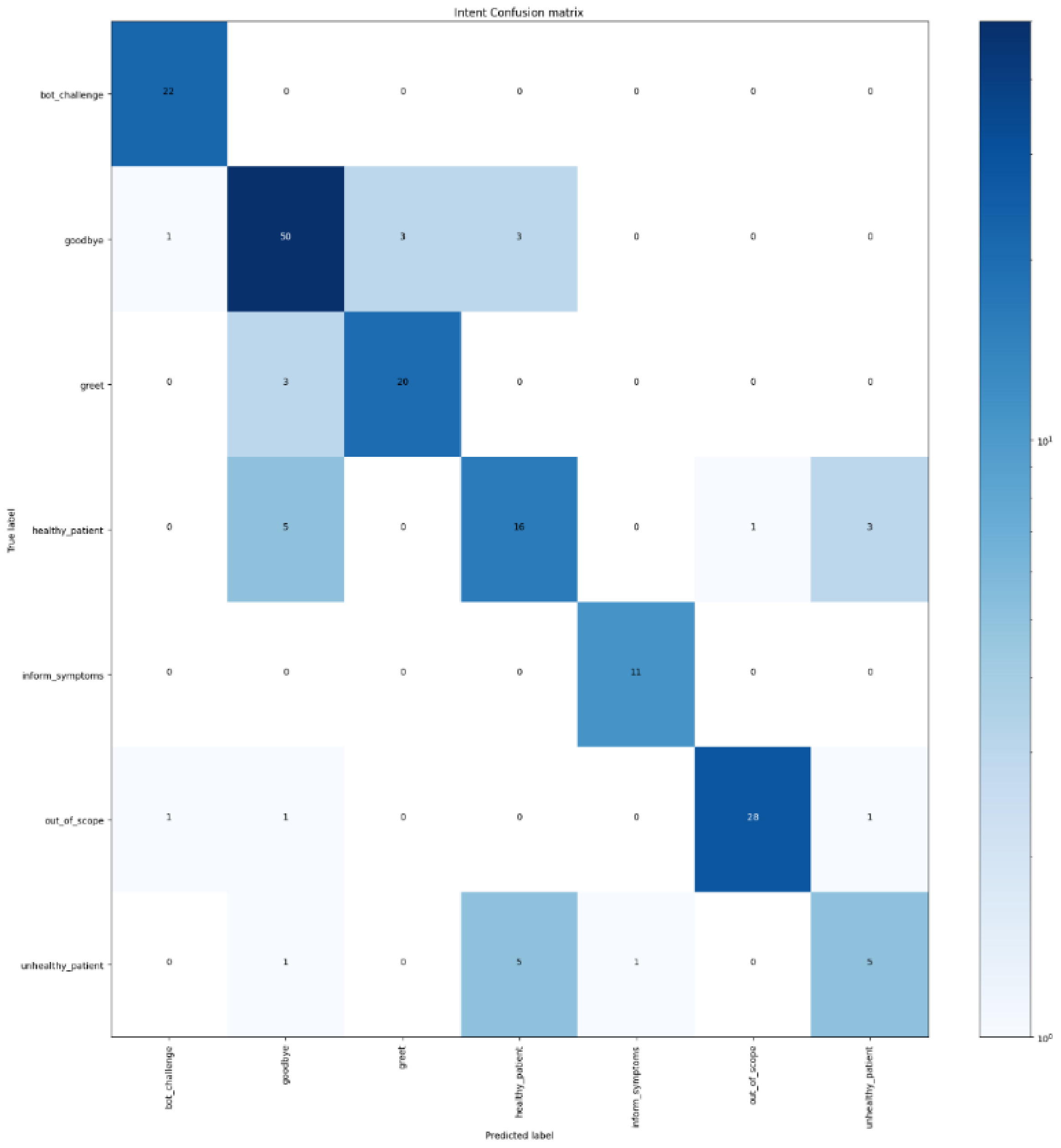
If there is any inconsistency between rules and stories, it gets validated in the training data and training gets aborted. No data issues were reported during the training.

8.3 Evaluating NLU Model

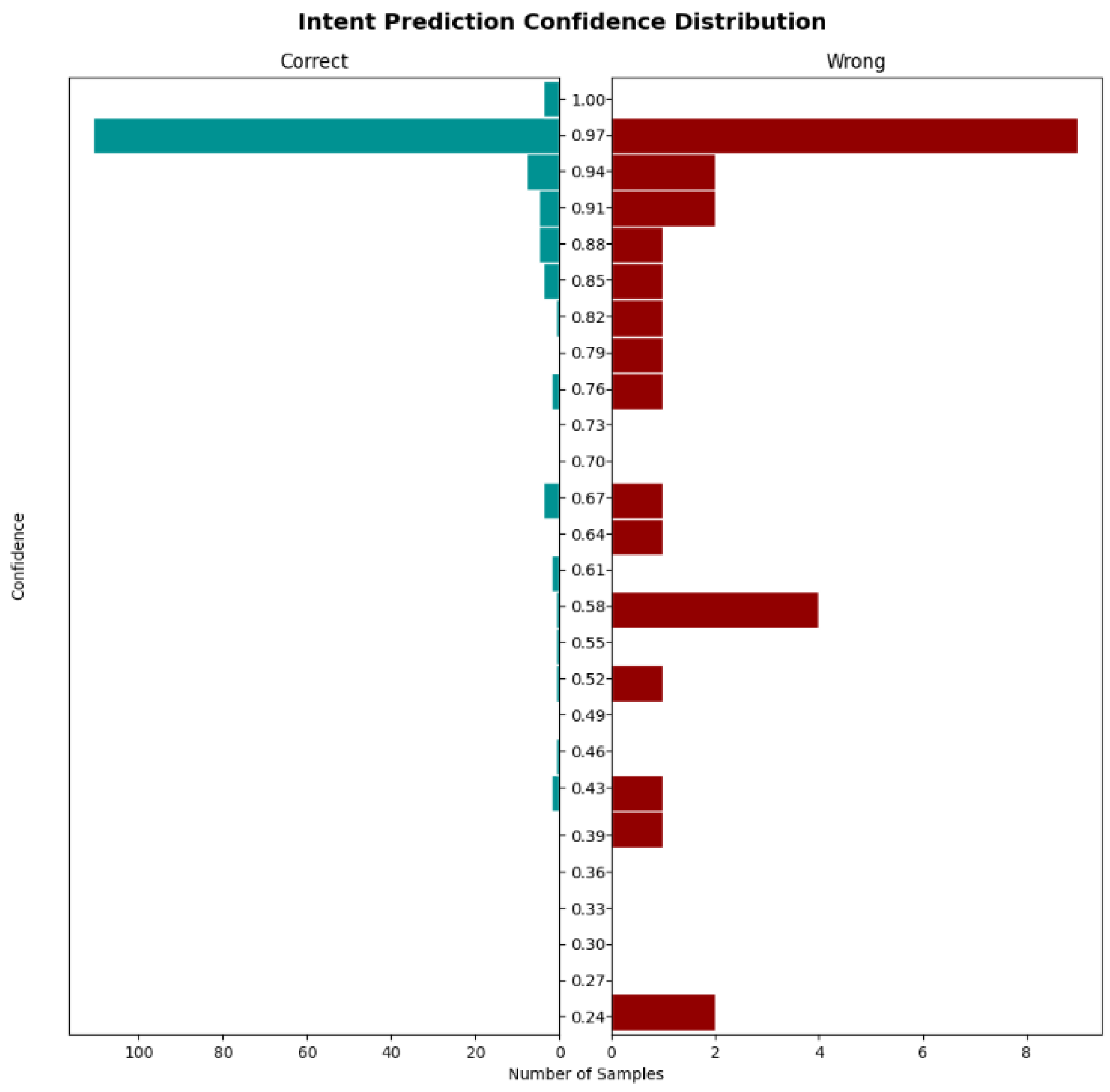
NLU model was performed using shuffle and split data stategy. The NLU training data was split with 80/20 split of train/test data.

Table 9: Intent classification

Metric	Score (weighted avg)
F1	0.8356757777299284
Precision	0.8343335238458118
Recall	0.8397790055248618
Accuracy	0.8397790055248618



True label \ Predicted label	bot_challenge	goodbye	greet	healthy_patient	inform_symptoms	out_of_scope	unhealthy_patient
bot_challenge	22	0	0	0	0	0	0
goodbye	1	50	3	3	0	0	0
greet	0	3	20	0	0	0	0
healthy_patient	0	5	0	16	0	1	3
inform_symptoms	0	0	0	0	11	0	0
out_of_scope	1	1	0	0	0	28	1
unhealthy_patient	0	1	0	5	1	0	5

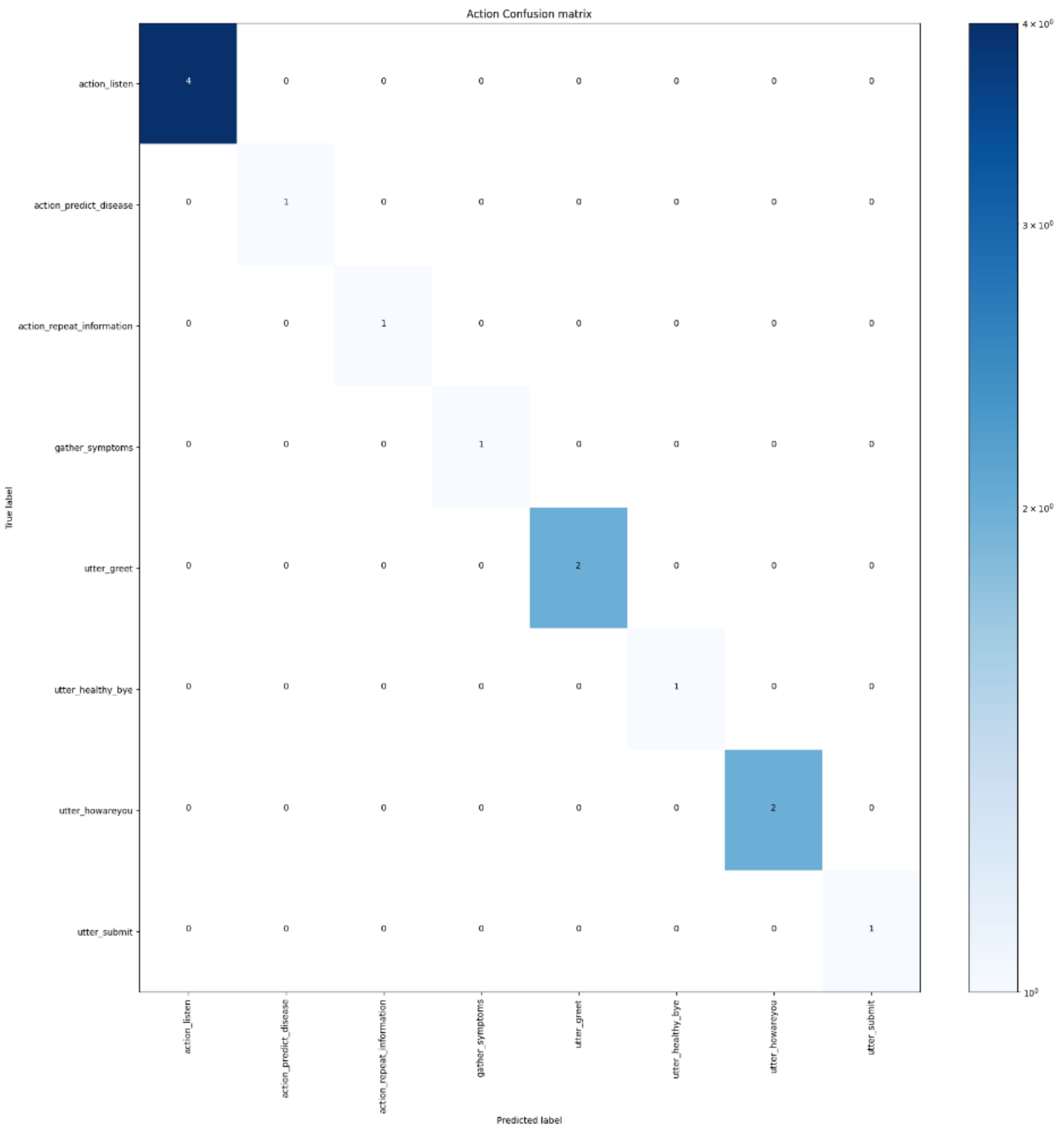


8.4 Evaluating Dialogue Model

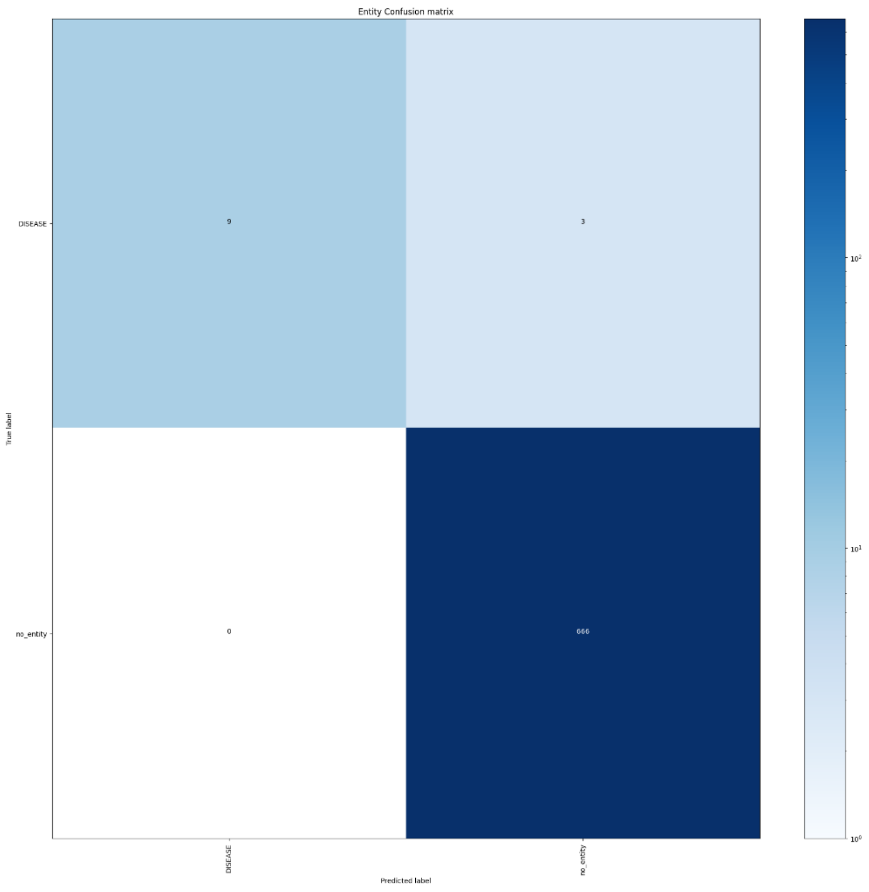
The diaologue management consists of stories , rules and the policies governing the next predicted action. The model is evaluldated based on how well it could predict the next action.

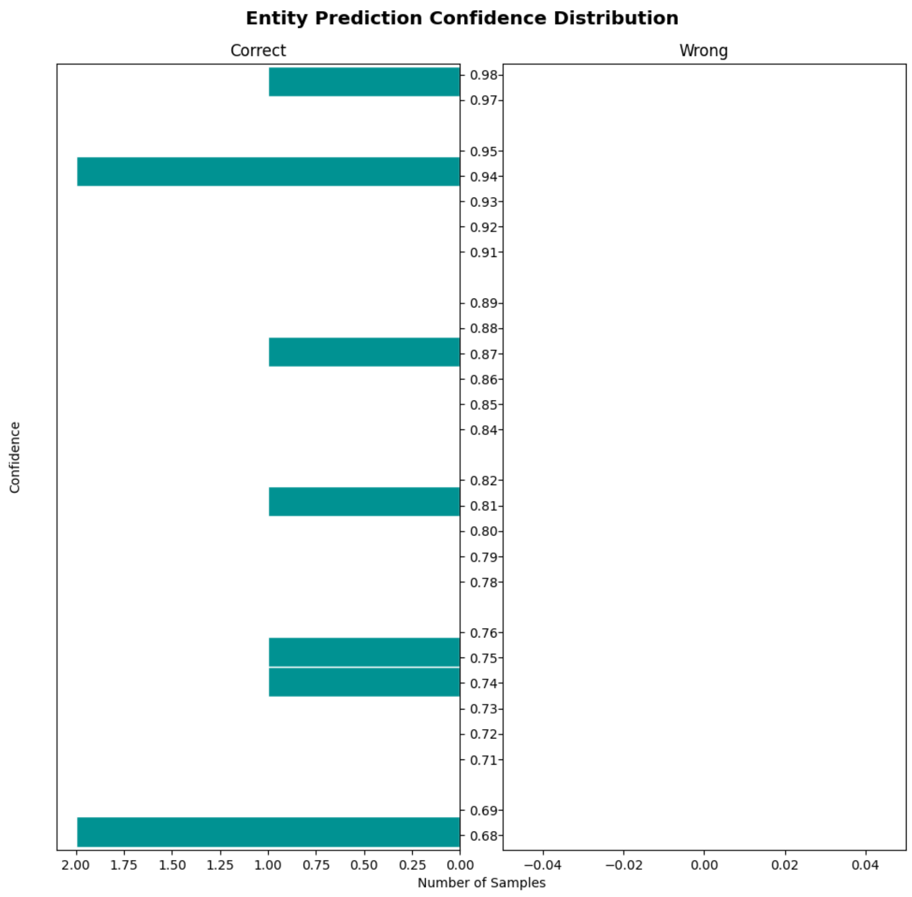
Table 10: Action prediction

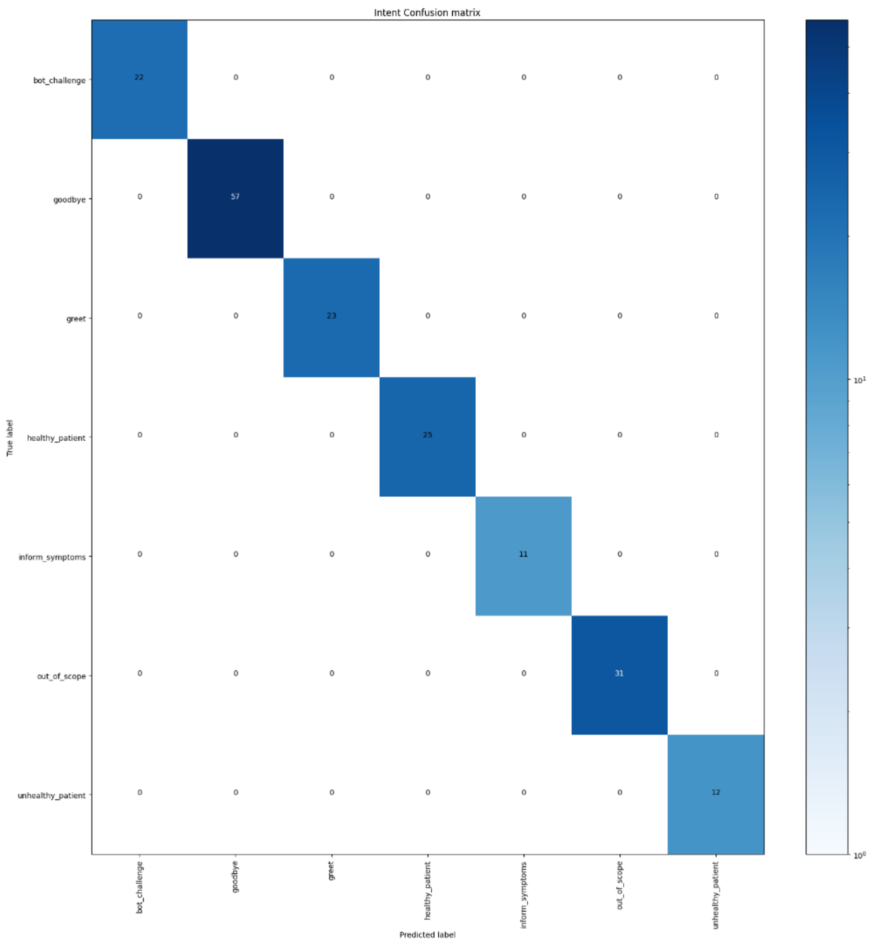
Metric	Score (weighted avg)
F1	
Precision	
Recall	
Accuracy	

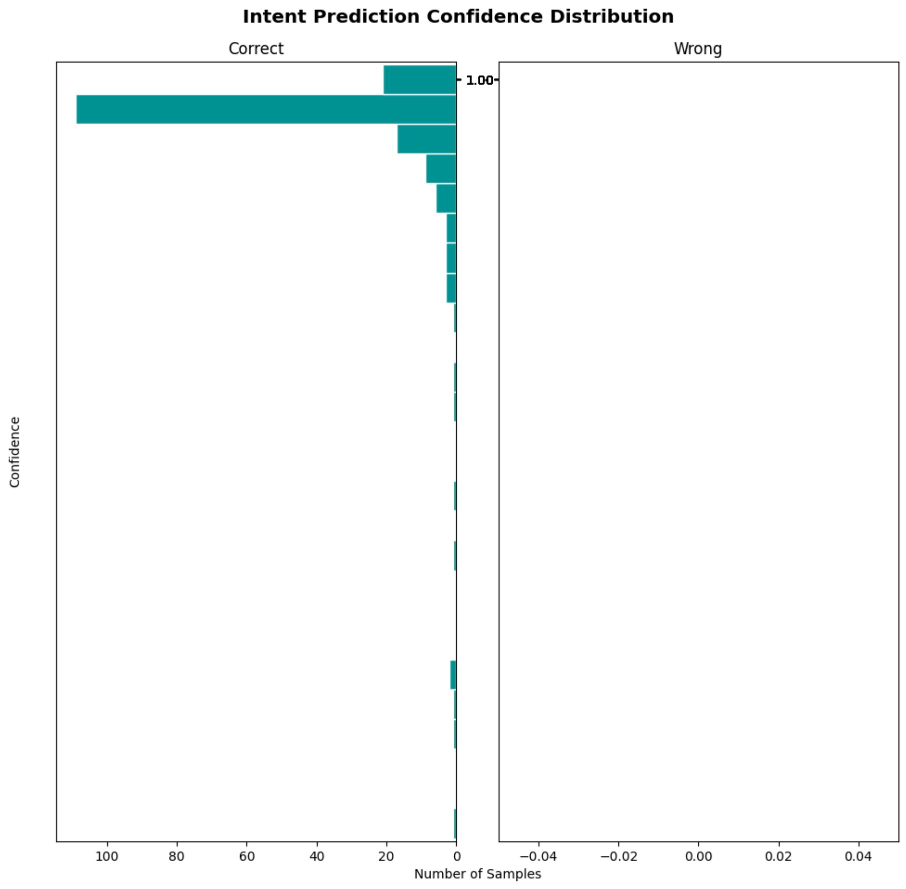


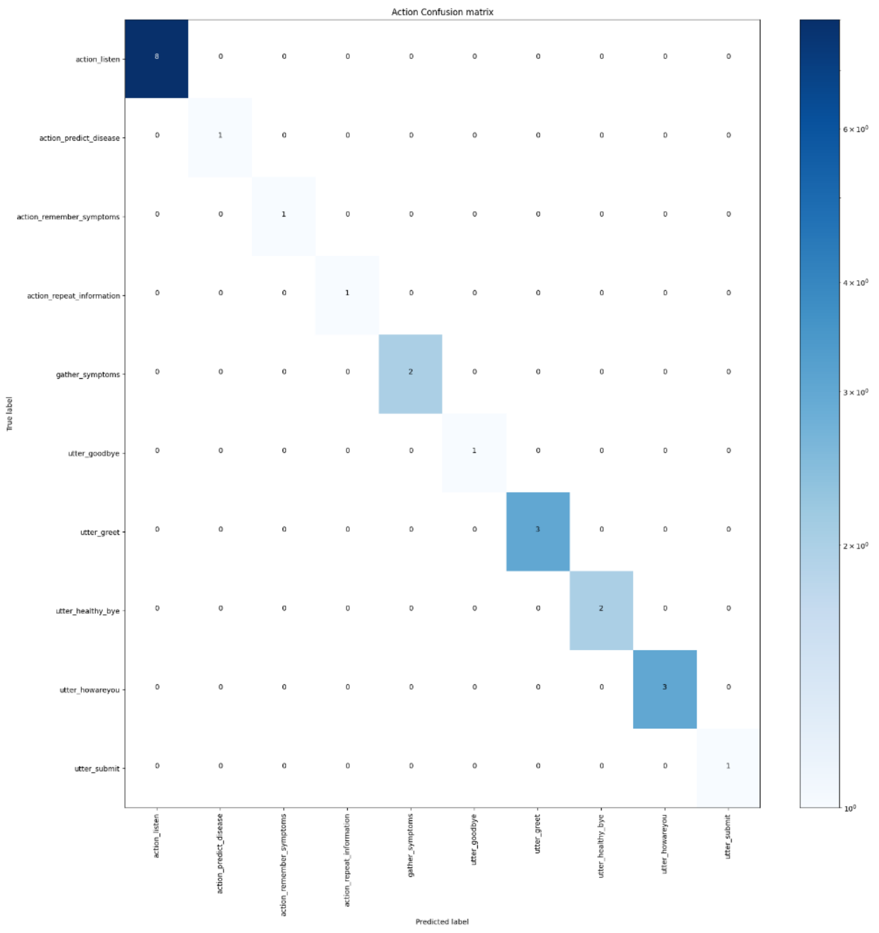
8.5 Conversation Flow tests


Entity Confusion matrix
True label
DISEASE
no_entity
9
3
0
666
Predicted label
DISEASE
no_entity
10^2
10^1
10^0


Entity Prediction Confidence Distribution
Correct
Wrong
Confidence
Number of Samples
0.98
0.97
0.95
0.94
0.93
0.92
0.91
0.89
0.88
0.87
0.86
0.85
0.84
0.82
0.81
0.80
0.79
0.78
0.76
0.75
0.74
0.73
0.72
0.71
0.69
0.68
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.04
−0.02
0.00
0.02
0.04


Intent Confusion matrix
True label
bot_challenge
goodbye
greet
healthy_patient
inform_symptoms
out_of_scope
unhealthy_patient
Predicted label
22
57
23
25
11
31
12
0
10^1
10^0


Intent Prediction Confidence Distribution
Correct
Wrong
Confidence
Number of Samples
1.00
100
80
60
40
20
0
−0.04
−0.02
0.00
0.02
0.04



8.6 Unit tests

The action server functionality was tested using Python unit tests and the test summary in attached in the appendix.

8.7 Interactive Mode validations

We validated the slot filling functionality using rasa's interactive command line interface to makesure slot filling is working well. The vlaidations are attached in the appendix.

8.8 Survey responses

A survey was conducted to get users' feedback about the naturalness of the chatbot conversation and the functionality of the system. The survey was hosted on the Qualtrics survey system provided by the University OF St Andrews and the survey link was shared with the respondents. The screenshots of two representative conversation flows were provided for the respondent's feedback. The survey was anonymous and voluntary and no personal information was collected during the survey.

Each of the questions can be answered with one of the six choices:

Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree, Prefer Not to Answer

Questions about the naturalness of the conversation with the chatbot:

1. The conversation with the chatbot sounded natural and smooth.
2. The chatbot could handle out-of-context interactions gracefully.
3. The chatbot kept the user informed about the progress and the conversation was engaging.
4. The chatbot politely acknowledged its shortcomings.
5. The conversation ended gracefully and sounded natural.

Questions about the functionality of the chatbot:

1. When asked, the chatbot introduced itself and the introduction was clear.
2. The chatbot clearly understood the user's intentions (healthy/unhealthy user, informing symptoms, asking about the chatbot).
3. The chatbot sought appropriate information from the user.
4. The chatbot recognized the symptoms entered by the user.
5. The chatbot appropriately responded to the user's question.
6. The displayed disease names were matching with the entered symptoms.
7. The disease names were displayed appropriately.

When to delete:

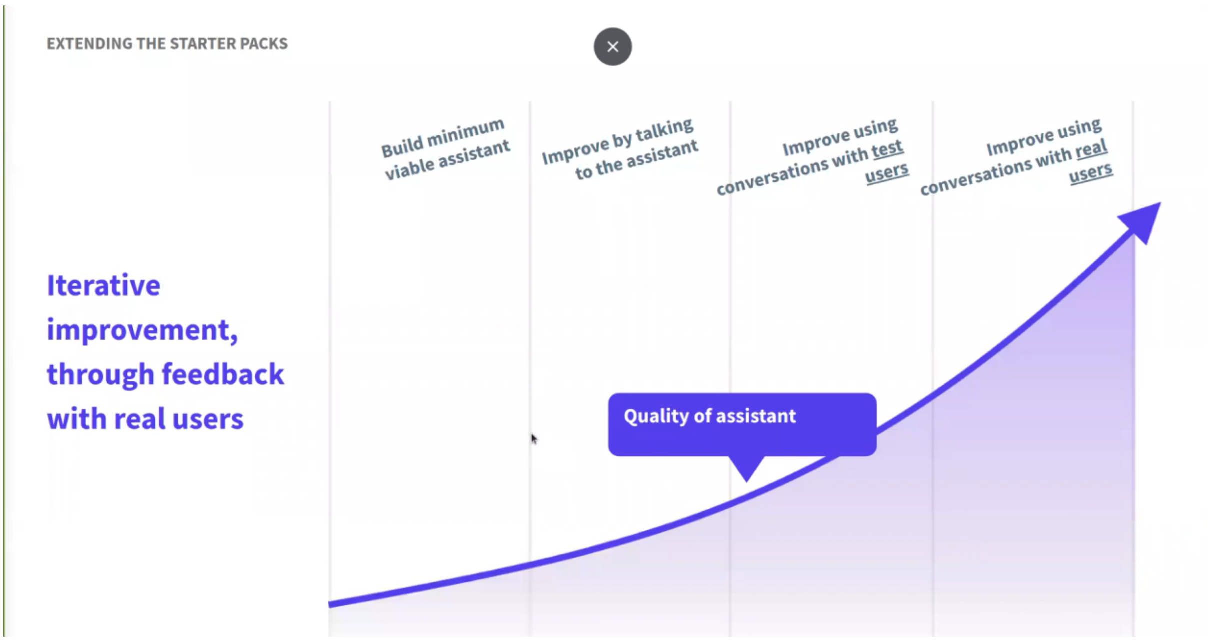
~~Testing the pipelines:~~

~~https://rasa.com/docs/rasa/testing-your-assistant/#comparing-nlu-pipelines~~

~~Bot maturity (productionization) process:~~

~~How we achieved pobj~~

~~\\~~


EXTENDING THE STARTER PACKS
Iterative improvement, through feedback with real users
Build minimum viable assistant
Improve by talking to the assistant
Improve using conversations with test users
Improve using conversations with real users
Quality of assistant

9 Future Work

The focus of the dissertation was developing a chatbot capable of understanding the user's health status and extracting the symptoms from the input text and maintaining the dialogue without forgetting the information collected in previous dialogue turns. During the development, it was observed that knowing users in detail can help the bot in delivering a better user experience. Bot system can maintain detailed user profiles in the backend and can during a conversation the system can extract key profile details and frame a contextualized and personalized conversation.

The NER model used for the disease was found to miss some of the words which are closely related to the actual symptom. It is believed that the model can be further customized to include those additional closely related word tokens as disease entities in the model

The disease and symptom mathcing system can be improved by replacing the text based mtching system with machine learning model that is capable of accepting symptoms (and some other extra feature maintained in user profile) as its inputs and predict possible disease with it's probability.

Knowledge graph for better data representation

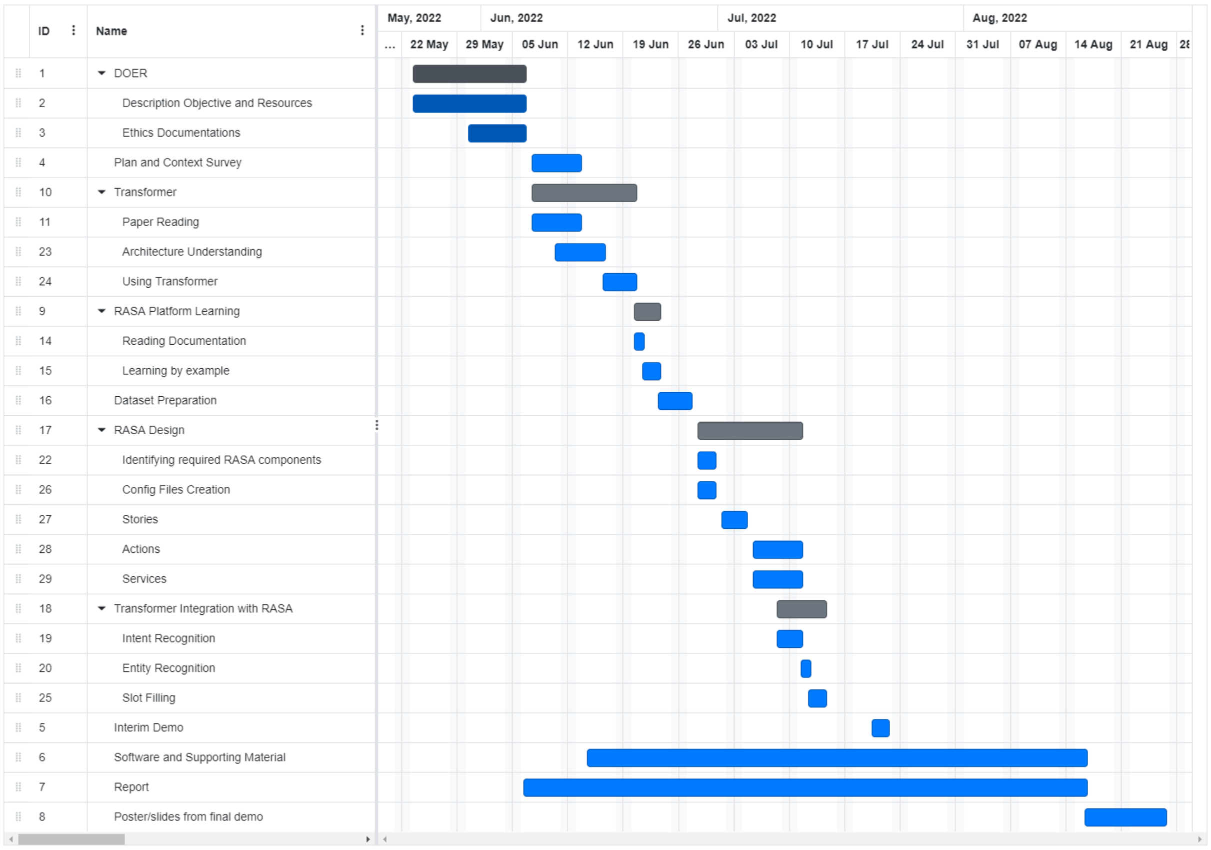
Bias in the pre-trained dataset

https://learning.oreilly.com/library/view/natural-language-processing/9781098136789/ch01.html#idm45146323434784 Main Challenges with Transformers

10 Conclusion

11 Project Timeline

Figure 7 Project Timeline



ID	Name
1	DOER
2	Description Objective and Resources
3	Ethics Documentations
4	Plan and Context Survey
10	Transformer
11	Paper Reading
23	Architecture Understanding
24	Using Transformer
9	RASA Platform Learning
14	Reading Documentation
15	Learning by example
16	Dataset Preparation
17	RASA Design
22	Identifying required RASA components
26	Config Files Creation
27	Stories
28	Actions
29	Services
18	Transformer Integration with RASA
19	Intent Recognition
20	Entity Recognition
25	Slot Filling
5	Interim Demo
6	Software and Supporting Material
7	Report
8	Poster/slides from final demo

12 Appendix B User Guide

- Clone the repository to a local drive.
- Change the directory to the root of the cloned repository.
- Make sure all the python dependencies are correctly installed.
- Follow the instructions mentioned below to access the chatbot:

Rasa server
To run the server, use the following command:

```
rasa run --cors "*"
```

The --cors "*" command is used to solve the cross-origin resource sharing (CORS) problem between the client and Rasa servers.

Run Action server

```
rasa run actions
```

This will start the action server service.

Web client-server
Run the following command:

```
python -m http.server
```

Accessing chatbot:

This will start an HTTP-based server in the local 8000 port.

Visit `http://localhost:8000` in a browser to access the chatbot.

13 Appendix C Ethics Documents

14 References

Amazon Web Services. (2019). *Amazon Lex – Build Conversation Bots*. Amazon Web Services, Inc. https://aws.amazon.com/lex/

Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*.

Botfront. (n.d.). *Better intent classification and entity extraction with Dietclassifier pipeline optimization*. Dialogue Technologies Inc. Retrieved July 10 from https://botfront.io/blog/better-intent-classification-and-entity-extraction-with-diet-classifier-pipeline-optimization

Brasoveanu, A. M. P., & Andonie, R. (2020). Visualizing Transformers for NLP: A Brief Survey.

Damani, S., Narahari, K. N., Chatterjee, A., Gupta, M., & Agrawal, P. (2020). Optimized Transformer Models for FAQ Answering. In (pp. 235-248). Springer International Publishing. https://doi.org/10.1007/978-3-030-47426-3_19

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dolan, S. (2022). *How mobile users spend their time on their smartphones in 2022*. Insider Intelligence. Retrieved June 29 from https://www.insiderintelligence.com/insights/mobile-users-smartphone-usage/#:~:text=Mobile%20usage%20statistics,digital%20media%20time%20per%20day

Freed, A. (2021). *Conversational AI*. Manning Publications Co.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.".

Gillioz, A., Casas, J., Mugellini, E., & Khaled, O. A. (2020). Overview of the Transformer-based Models for NLP Tasks.

Google. (2022). *Dialogflow*. Google Cloud. https://cloud.google.com/dialogflow

Hugging Face. (2022). *Hugging Face Hub documentation*. Hugging Face. https://huggingface.co/docs/hub/index

IBM. (2022). *IBM Watson Assistant - Virtual Agent*. IBM. https://www.ibm.com/uk-en/products/watson-assistant

Kandpal, P., Jasnani, K., Raut, R., & Bhorge, S. (2020). Contextual Chatbot for Healthcare Purposes (using Deep Learning).

Kong, X., Wang, G., & Nichol, A. (2021). *Conversational AI with Rasa: Build, test, and deploy AI-powered, enterprise-grade virtual assistants and chatbots*. Packt Publishing Ltd.

Magnus Revang, A. M., Bern Elliot. Magic Quadrant for Enterprise Conversational AI Platforms. https://www.gartner.com/document/4010683?ref=gfa

Microsoft. (2022). *Azure Bot Service – Conversational AI Application | Microsoft Azure*. Microsoft Retrieved June 1,2022 from https://azure.microsoft.com/en-us/services/bot-services/

Miller, G. A. (1995). WordNet. *Communications of the ACM*, *38*(11), 39-41. https://doi.org/10.1145/219717.219748

Mukhopadhyay, S. (2018). *Advanced Data Analytics Using Python With Machine Learning, Deep Learning and NLP Example*. Apress.

Nadarzynski, T., Miles, O., Cowie, A., & Ridge, D. (2019). Acceptability of artificial intelligence (AI)-led chatbot services in healthcare: A mixed-methods study. *DIGITAL HEALTH*, *5*, 205520761987180. https://doi.org/10.1177/2055207619871808

Neumann, M., King, D., Beltagy, I., & Ammar, W. (2019). ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. *ArXiv*, *abs/1902.07669*.

Palash Goyal, S. P., Karan Jain. (2018). *Deep Learning for Natural Language Processing: Creating Neural Networks with Python*. Apress.

Pandey, A., Mutreja, I., Brar, S., & Singh, P. (2020). Exploring Automated Q&A Support System for Maternal and Child Health in Rural India.

Perera, S., Sheth, A., Thirunarayan, K., Nair, S., & Shah, N. (2013). Challenges in understanding clinical notes: Why NLP engines fall short and where background knowledge can help. Proceedings of the 2013 international workshop on Data management & analytics for healthcare,

Rasa Technologies Inc. (2022). *Introduction to Rasa Open Source*. Rasa Technologies Inc. Retrieved June 5 from https://rasa.com/docs/rasa/

Rustamov, S., Bayramova, A., & Alasgarov, E. (2021). Development of Dialogue Management System for Banking Services. *Applied Sciences*, *11*(22), 10995. https://doi.org/10.3390/app112210995

Sheth, A., Yip, H. Y., & Shekarpour, S. (2019). Extending Patient-Chatbot Experience with Internet-of-Things and Background Knowledge: Case Studies with Healthcare Applications. *IEEE Intelligent Systems*, *34*(4), 24-30. https://doi.org/10.1109/mis.2019.2905748

Singh, P., Lin, T., Mueller, E. T., Lim, G., Perkins, T., & Li Zhu, W. (2002). Open Mind Common Sense: Knowledge Acquisition from the General Public. In (pp. 1223-1237). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-36124-3_77

Tunstall, L., Werra, L. v., & Wolf, T. (2022). *Natural Language Processing with Transformers* (Revised Edition ed.). O'Reilly Media, Incorporated.

Ur Rahman Khilji, A. F., Laskar, S. R., Pakray, P., Kadir, R. A., Lydia, M. S., & Bandyopadhyay, S. (2020). HealFavor: Dataset and A Prototype System for Healthcare ChatBot.

Vajjala, S., Majumder, B., Gupta, A., & Surana, H. (2020). *Practical Natural Language Processing*. O'Reilly Media.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Wikipedia. (2022). *Natural language processing - Wikipedia*. Wikipedia. Retrieved June from https://en.wikipedia.org/wiki/Natural_language_processing

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., & Funtowicz, M. (2020). Transformers: State-of-the-art natural language processing. Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations,

Yu, S., Chen, Y., & Zaidi, H. (2020). A Financial Service Chatbot based on Deep Bidirectional Transformers. *arXiv preprint arXiv:2003.04987*.