Linnæus University
Sweden

Bachelor Degree Project

Evaluating the Effectiveness of Open Source Chatbots for Customer Support

Author: Fabian Dacic, Fredric Eriksson Sepúlveda
Supervisor: Johan Hagelbäck
Examiner: Aris Alissandrakis
University: Linnaeus University

Term: Spring 2023
Subject: Computer Science

Abstract

Chatbots are becoming increasingly popular in various industries, and there are many options available for businesses and organisations. Several studies have investigated open-source chatbots and identified their core strengths, implementation, and integration capabilities however few have investigated open-source chatbot frameworks and libraries in a specific use case such as medicine. The project's objective was to evaluate a selection of chatbots or more specifically two frameworks: Botkit and Rasa, and two libraries: ChatterBot, and Natural which was utilised together with Botkit and a language model which is DialoGPT. The evaluation focuses specifically on accuracy, consistency, and response time. Frequently asked questions from the World Health Organization and COVID-19 related Dialogue Dataset from GitHub were utilised to test the chatbots' abilities in handling different queries and accuracy was measured through metrics like Jaccard similarity, bilingual evaluation understudy (BLEU), and recall oriented gisting evaluation (ROUGE) scores, consistency through Jaccard similarity between the generated responses and response time was taken to be the average time for a response in seconds. The analysis revealed unique strengths and limitations for each chatbot model. Rasa displayed robust performance in accuracy, consistency, and customisation capabilities if the chatbot works in a particular topic with acceptable response times. DialoGPT demonstrated strong conversational abilities and contextually relevant responses with trade-offs in consistency. ChatterBot showed consistency, though sometimes struggled with advanced queries, and Botkit with Natural stood out for its quick response times, albeit with limitations in accuracy and scalability. Despite implementation challenges, these open-source frameworks, libraries, and models offer promising solutions for organisations intending to harness conversational agents' technology. The study suggests encouraging further exploration and refinement in this rapidly evolving field.

Key words

chatbot, machine learning, software framework, software library, language model, artificial intelligence, customer support, medicine, COVID-19, ChatterBot, Botkit, Natural JavaScript library, DialoGPT, Rasa, natural language processing

Acknowledgments

We would like to express our deepest gratitude to our supervisor Johan Hagelbäck for his invaluable guidance and unwavering support throughout this journey which have been pivotal to our research and we sincerely appreciate the time and effort he has invested in our work. We also wish to convey our heartfelt appreciation to examiner Aris Alissandrakis, whose insightful comments and constructive criticism significantly enhanced the quality and depth of our thesis. In addition, we extend our sincere thanks to Daniel Toll, who provided us with a solid theoretical knowledge and guidance thereby laying the foundation for our research. Our gratitude also goes out to Tobias Andersson-Gidlund whose thoughtful suggestions and engaging discussions inspired us and helped refine our ideas. Furthermore we are deeply grateful to our families and friends who provided unending moral support and encouragement during the process. We would like to extend our heartfelt appreciation to the open-source community, specifically the developers and contributors of the frameworks, libraries, and language models utilised in our research. Finally, we wish to express our sincere gratitude to everyone else who has contributed in one way or another to our work. This thesis would have not been possible without the collective support and assistance we have received.

Author contributions

Author Fabian Dacic co-developed the research and refined the initial idea with author Fredric Eriksson Sepúlveda under the supervision of Johan Hagelbäck. Fabian Dacic investigated libraries and language models. Implemented a framework, utilised two libraries and fine-tuned a language model specifically for use in chatbots. Fabian Dacic is responsible for the three out of four chatbots used in the thesis, including their specifics. Wrote sections detailing the framework, libraries, and language model used for these specific chatbots as well gathered and analysed the results related to them. Fredric Eriksson Sepúlveda was responsible for implementing and training the Rasa framework and models, investigating other frameworks and proof-reading, also discussing the sections for Rasa in general. A repository was provided by both authors in which the details of all implementations can be found in. Both authors conducted discussions and conclusions related to the chatbots utilised in this thesis, providing insight into their impact and potential applications.

Table of Contents

Appendices

1 Introduction

This is a 15 HEC Bachelor thesis in Computer Science for Linnaeus University and as a team consisting of two students, through this thesis, the objective is to evaluate different frameworks, libraries and models that are utilised for chatbot building, outlining advantages and disadvantages as well as contributing to the already existing corpus of research on chatbot technology and its role in customer support. Through implementation, evaluation and comparison of popular chatbot frameworks, libraries and models, this thesis will primarily seek to illuminate how businesses and organisations can improve their customer interactions and provide a solid base for further exploration in this rapidly growing field.

1.1 Background

In today's society, chatbots have become an integral part of day-to-day life in one way or another. There are various applications of chatbots such as: automating tasks, providing instant support, and enhancing the general user experience [1]. To be more specific, chatbots are utilised in customer support in which they provide answers and solutions to commonly asked questions and common problems that the customers might have. Another example would be e-commerce where chatbots assist the customers by recommending products, order tracking, and more [2]. To meet this increasing demand, businesses and organisations are turning to chatbot technology to streamline their customer support and service operations [3]. There are numerous open-source chatbot frameworks and each offers a cost effective and customizable solution to develop an agent however selecting what can be considered as most suitable can be a daunting task. Selecting an appropriate open-source chatbot framework that can meet the unique requirements of each business is often a challenging task due to the wide variety of options available. Each framework offers a cost-effective and customizable solution to develop a chatbot. However, comparing and evaluating these frameworks necessitates reliable metrics that can measure their performance accurately. To gauge the effectiveness of chatbot responses, this study utilises several key metrics - the Jaccard similarity, bilingual evaluation understudy (BLEU), and variants of recall oriented understudy for gisting evaluation (ROUGE, ROUGE-1, ROUGE-2, and ROUGE-L). The Jaccard similarity, a measure of the similarity between two sets, is primarily used in this study to evaluate the accuracy, and the consistency of the chatbots. It compares the intersection and the union of words present in two text strings [4]. While this method does not account for semantic meanings, synonyms, or related words, it provides a straightforward way to compare the similarity between text strings

[4]. BLEU and ROUGE, on the other hand, are primarily used to evaluate the accuracy of the chatbot responses. BLEU measures N-gram overlap and aligns well with human judgments but may fall short in evaluating fluency and coherence, particularly in complex languages [5]. ROUGE variants each evaluate different aspects of text generation systems: ROUGE-1 focuses on unigram overlap, ROUGE-2 examines bigrams, and ROUGE-L measures the longest common subsequence, each providing insights into system performance concerning word similarity, sentence structure, and word order, respectively [6]. Though these metrics offer valuable insights into a system's performance, they only provide a partial view. Consideration of these metrics collectively, along with human judgement, could result in a more comprehensive evaluation. However, for the purpose of this project, and due to time constraints, the primary accuracy indicators will be BLEU and ROUGE, with Jaccard similarity indicating consistency. The interpretation and utilisation of these accurate answers will be discussed in more detail in the upcoming sections.

1.2 Motivation

Implications of the findings will be discussed to allow for organisations seeking to implement chatbots for customer support, and this will be achieved by highlighting the advantages as well as the disadvantages of each chatbot framework to guide the organisations in selecting the most suitable option. This thesis will contribute to the growing body of research on chatbot technology and its applications in customer support and by evaluating and comparing popular open-source chatbot frameworks light will be shed on revolutionising the way businesses interact with their customers and provide a solid foundation for further exploration and development in this rapidly growing field. This research offers substantial implications for the technology industry as well as the comparative evaluation of these open-source chatbot frameworks provides valuable insights to software developers, system architects, and technology decision-makers who are at the forefront of chatbot development and deployment. By pinpointing the key features, capabilities, trade-offs, and limitations of these chatbots, the study can guide technology professionals in choosing the right framework for their specific use cases. This thesis may also inspire the technology industry to explore the development of hybrid or composite chatbot frameworks that incorporate the strengths of multiple frameworks to provide better customer support solutions. The learnings from this thesis can aid in refining the tools and methods used in chatbot development, thereby contributing to the advancement of this significant sector within the technology industry.

1.3 Problem formulation

This thesis investigates key features and capabilities of four popular open-source frameworks, libraries and models utilised to build chatbots – DialoGPT, Rasa, ChatterBot, and Botkit with Natural - chosen for their distinct technologies, diverse approaches and popularity. Research question 1 (RQ1) asks:

> **RQ1.** What are the key features and capabilities of these specifically chosen open-source chatbots in customer support?

Understanding these features is essential for evaluating the chatbots. Performance metrics – accuracy, response time, and consistency will also be considered. Rather than seeking the 'best' chatbot, this thesis will identify strengths and weaknesses of each in these areas, without equal weightage. The thesis will also examine the trade-offs and limitations of each chatbot in customer support (RQ2a), and suggest ways to mitigate these challenges (RQ2b):

> **RQ2a**. What are the trade-offs and limitations of each chosen chatbot in customer support?
>
> **RQ2b.** How can businesses and organisations mitigate these challenges?

Overall, this research aims to provide a comprehensive understanding of open-source chatbot capabilities, offering insights to enhance customer support.

1.4 Related work

Pavel (2021) compared Rasa and Botkit, two frameworks for building chatbots, and concluded that Rasa is more suitable for development [7]. Although the potential of chatbots is recognized in e-commerce, other fields could benefit from their use, including healthcare. In this context, Pereira and Díaz conducted a mapping study titled "Using Health Chatbots for Behaviour Change," analysing the use of chatbots in the health domain [8]. Their study investigated the role of health chatbots in managing various illnesses and how these digital tools interact with different patient abilities. For instance, chatbots have been developed to provide guidance on nutritional and neurological disorders. The "human competences" pursued by these chatbots refer to the aspects of human behaviour and cognition that these chatbots are designed to influence for achieving healthier habits, and the context is a chatbot offering strategies for managing stress (affect) or improving memory (cognition). The study further underscores the value of personalization and easy-to-use interfaces in health chatbots. Despite the proliferation of studies on chatbots, there's a lack of research focusing on the performance and effectiveness of specific chatbot frameworks in customer support, and this thesis aims to address this gap by evaluating open-source chatbot frameworks, libraries and models using metrics which will be mentioned in the subsequent sections as well as a dataset within a healthcare customer support scenario. Given the time constraints, the study is focused on a specific customer support use case in healthcare, and this approach will reveal the strengths and weaknesses of each framework, guiding businesses in selecting the most suitable chatbot solution for their customer support needs.

1.5 Limitations

Scope of chatbots: The study is limited to open-source chatbot frameworks – DialoGPT, Rasa, ChatterBot, and Botkit with Natural. There are several other proprietary chatbot technologies and frameworks that are not explored due to accessibility or proprietary restrictions.

Dataset: The chatbots are trained and evaluated on a specific dataset (a healthcare dataset), which might limit the generalizability of the results. The chatbots might perform differently when trained and evaluated on other datasets or domains.

Limited metrics: The study employs a set of evaluation metrics such as accuracy, consistency, and response time, which does not fully encapsulate all the aspects of chatbot performance, points which will be discussed in more detail in the Discussion (section 6.4). Other factors like adaptability, scalability, ease of use, and cost might be equally important however are not considered due to time constraints.

Subjectivity: Different organisations or businesses might prioritise different features and capabilities based on their specific needs, which is hard to encapsulate in a universal evaluation.

Time constraints: The study is constrained by the project timeline, which could limit the depth of the evaluation or the number of chatbot frameworks that can be considered.

1.6 Target group

This thesis has different target groups: businesses and organisations, students within computer science, information technology and so forth, developers, and hobbyists of chatbots but primarily will serve as an evaluation of DialoGPT (Update 07/09/2022), ChatterBot (v1.0.8), Rasa (v3.3), Botkit (v4.15.0) and Natural (v6.5.0) for businesses looking to enhance their customer support by implementing these specifics frameworks, libraries and models which would improve customer satisfaction and reducing workload. The project offers insights and resources for businesses to explore and implement chatbot technology effectively, and additionally, developers and students interested in the field of chatbots can leverage this project to gain hands-on experience and explore the capabilities of chatbot development frameworks. The open-source nature of the project, coupled with its documentation and code samples, provides a valuable learning resource for those looking to delve into chatbot development.

1.7 Outline

The main goal of this thesis is to evaluate popular open-source chatbot frameworks, focusing on their application in customer support. The chatbots will be trained using an open-source healthcare dataset or more specifically a COVID-Dialogue Dataset to answer frequently asked questions (FAQs) and provide guidance based on symptoms. The performance of the chatbots will be analysed using metrics such as accuracy, consistency, and response time with the aid of Jaccard similarity, BLEU, and ROUGE scores. This study aims to provide businesses a guideline for choosing the right chatbot framework, detailing the strengths, weaknesses, and unique features of each framework. The methodology includes the implementation, training, and evaluation of chatbots, with the results providing insights into the capabilities of these chatbots in handling complex queries, providing timely responses, and maintaining consistency in conversation, among others. Thus the thesis will be a valuable resource in identifying the chatbot framework that best aligns with specific customer support needs.

2 Chatbots

This section will briefly describe the history of the chatbots, how they are categorised, their differences as well as a brief description of the chatbots that will be evaluated.

2.1 History

Chatbots were invented in order to facilitate human-computer interaction, primarily enabling devices such as computers to understand and respond to various natural language inputs thereby simulating a human-like conversation. In 1950, Alan Turing published an article named "*Computing Machinery and Intelligence*" in which he discussed the question of whether machines could think which followed the now famous Turing Test, which is a test aimed to determine whether a machine can exhibit intelligent behaviour equivalent to that of a human [9]. His work inspired many such as Joseph Weizenbaum, a German computer scientist and professor, who in 1966 developed the program ELIZA [10]. ELIZA is an early natural language processing (NLP) program and it was designed to mimic the responses of a Rogerian psychotherapist i.e non-directional evaluation in an initial psychiatric interview. This was done with the goal in mind to "trick" the user into giving them the illusion that ELIZA can understand and empathise with them. ELIZA worked by searching the input text for keywords, assigning values to the keywords in question and then transforming the input into an output based on a set of rules defined in its script (it is called DOCTOR) [10]. This program remains a significant milestone in the history of chatbots and artificial intelligence as a whole as it can be considered as the first attempt at creating human-machine interaction which aims to simulate a human-like conversation.

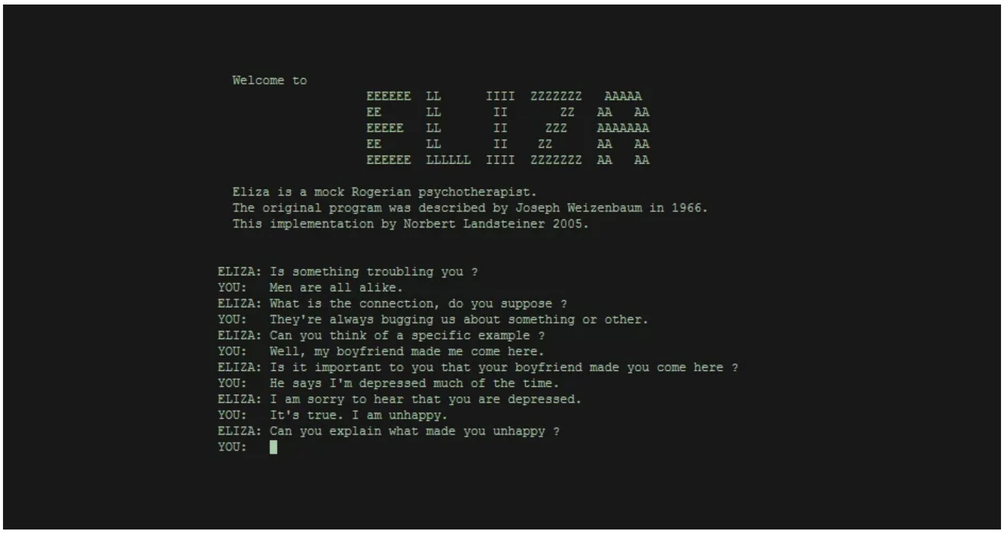

Figure 2.0: An interaction between a user and ELIZA [11].

ELIZA went on to inspire many other computer scientists around the world and the key contributions lead to development of new chatbots such as PARRY, ALICE and Jabberwacky [12]. These and other conversational agents expanded upon ELIZA’s principles and even introduced new techniques and methodologies for natural language understanding as well as generation [12].

2.2 Types of chatbots

Chatbots can be broadly categorised into several different categories depending on the underlying technology and strategy they employ for conversation, and understanding the differences between these types is crucial when evaluating chatbots for a specific context, such as customer support. For this particular study, chatbot technologies were selected to represent a diverse range of categories, in order to provide a well-rounded understanding of the current capabilities and limitations in the chatbot landscape however it is to be noted that the categorization in this thesis is not definitive in any way, as mentioned before, considering that there chatbots of various nature. Here are the chosen categories: Transformer and language models, machine learning based chatbots, and bot development frameworks.

2.2.1 **Transformer and language models**

Transformer models, introduced in a seminal thesis by Vaswani et al. in 2017 “*Attention is All You Need*” have revolutionised the field of natural language processing (NLP) which by itself is a subfield of artificial intelligence that focuses on how computers can understand and manipulate human language

[13]. NLP involves many tasks, such as machine translation, text summarization, and sentiment analysis. To briefly describe what each task does, machine translation is the task of automatically translating text from one language to another and text summarization aims to produce a shorter version of a piece of text while maintaining its key information. Sentiment analysis on the other hand, is the use of natural language processing to systematically identify, extract, quantify, and study affective states and subjective information. To focus back at the Transformer, its core is the self-attention mechanism, which allows the model to weigh the importance of words in an input sequence when generating an output sequence [12]. This mechanism has proven to be particularly effective for a range of tasks in NLP, including those mentioned above. How models such as DialoGPT take it further, will be discussed in more detail in section 2.3.3.

2.2.2 Machine learning based chatbots

Machine Learning-Based Chatbots leverage machine learning (ML) techniques and algorithms to generate responses to user inputs. Examples of such chatbots include Rasa and ChatterBot, which employ these techniques to provide a more engaging and dynamic user interaction. Rasa, an open-source machine learning framework, is widely recognized for its ability to build contextual AI assistants and chatbots. The framework combines several ML and NLP techniques, enabling the chatbot to understand and respond to user inputs contextually [14]. It supports entity recognition and uses intent classification to determine the user's purpose behind a message. On the other hand, ChatterBot employs a selection of ML algorithms to generate various types of responses [15]. When it comes to the ML algorithms that ChatterBot utilises, classification is one of them. ChatterBot takes the input statement and computes the most fitting response from its training data. This is essentially a classification task where the input statement is classified into the category of the most suitable response. One such algorithm is Naive Bayes, which is based on applying Bayes' theorem with strong (naive) independence assumptions between the features [15]. What similar libraries or frameworks for these sort of chatbots is that they learn from their training data and can construct new responses based on its training and the ChatterBot library utilises ML algorithms to provide responses [15], but its capacity to generate relevant responses to unfamiliar inputs is largely based on the breadth and quality of its training data. There are a variety of techniques that are utilised in chatbots through machine learning and one such tool is sentiment analysis.

2.2.3 Bot development frameworks

Bot development frameworks provide a platform for developers to build and deploy interactive chatbots quickly and they often offer an array of tools and libraries that assist developers in structuring conversational flows, defining user intents, entities, and managing dialogues [16]. They also often provide integrations with popular messaging platforms like Facebook Messenger, Slack, Microsoft Teams, and others, allowing the chatbots to be easily deployed on these platforms. Some chatbot development frameworks include Microsoft's Bot Framework, Google's Dialogflow, IBM's Watson Assistant, Botpress, Botkit and many more. These frameworks often vary in their approach to developing chatbots. Microsoft's Bot Framework, for example, enables developers to create chatbots that can communicate over multiple channels while maintaining the same conversation context. It supports languages such as Node.js and C# whereas Google's Dialogflow is known for its capabilities in natural language understanding (NLU), enabling the chatbot to understand the context of the user's queries, manage conversation flows, and extract entities. IBM's Watson Assistant is robust for enterprise-grade chatbot development and is known for its powerful NLP capabilities [16]. It also allows developers to implement machine learning for more advanced interactions. Botkit, which is another framework, sits slightly differently. Botkit is a comprehensive open-source tool for developing chatbots for major messaging platforms [17]. It's designed to handle everything from receiving and sending messages, integrating with various messaging APIs, maintaining conversation state, and more. Unlike some of the other more full-fledged chatbot development frameworks, Botkit places a strong emphasis on the core messaging infrastructure, making it a powerful tool for developers focusing on that aspect [17]. While these frameworks offer many features out of the box, it's worth noting that chatbots built with these frameworks often need to be combined with other tools or services to achieve a high level of intelligence or context understanding. For example, a Botkit chatbot might be combined with a model or NLP service to enhance its understanding of user input and this will be talked about in more detail in section 2.3.4. There are other chatbot development tools and frameworks of course, however these are some of the most known and supported [17].

2.3 Chosen chatbots for evaluation

This section will briefly describe each of the chosen frameworks, libraries and models used to build the chatbots that will be evaluated for this thesis.

2.3.1 ChatterBot

ChatterBot is an open-source Python library or more specifically a machine learning, conversational dialog engine for creating chatbots and it primarily relies on pattern matching to handle user queries [18]. Its primary focus is on creating chatbots that can engage in simple conversations, making it a suitable choice for evaluating the capabilities of chatbots in customer support applications. As mentioned in section 2.2.2, ChatterBot employs a combination of search algorithms, classification algorithms, and other machine learning techniques to provide responses. By utilising these techniques, ChatterBot can provide accurate responses to user inputs within a well-defined logic adapter while still maintaining a level of adaptability and flexibility. In the context of this study, implementing ChatterBot as a library used to construct a chatbot for customer support applications is relatively easy due to its user-friendly design and straightforward setup process. The key benefits of using ChatterBot include its simple installation, flexible storage adapters, and customizable logic adapters. Moreover, its list trainer functionality allows for quick and efficient training of the chatbot using existing conversation data. Overall, ChatterBot's ease of implementation and adaptability make it a viable option for evaluating chatbot frameworks in the context of customer support.

2.3.2 **Rasa**

Rasa is a machine learning framework that is open-source in which conversational agents can be developed such as chatbots using and natural language processing [14]. Rasa offers a comprehensive solution for creating sophisticated chatbots capable of handling complex customer queries. It employs machine learning algorithms for intent recognition, entity extraction, and dialogue management, which classifies it as an AI-based chatbot [14]. Rasa's robust feature set and flexibility make it an ideal candidate for evaluating AI-based chatbots' potential in customer support.

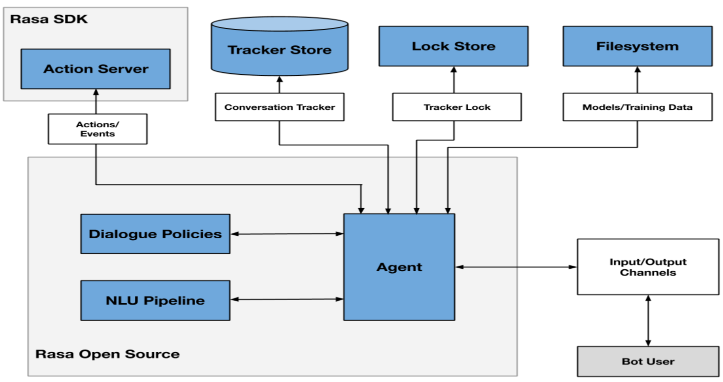

Figure 2.2: The fundamental structure of Rasa is underpinned by two key components: Natural Language Understanding (NLU) and Dialogue Management [19]. The role of the NLU pipeline is to interpret the user's inputs or expressions,

whereas the Dialogue Policies utilise the context to establish the subsequent steps in the conversation [19].

2.3.3 DialoGPT based chatbot

Like mentioned in section 2.2.1, transformer models such as OpenAI's GPT (Generative Pretrained Transformer) ones and DialoGPT takes this a step further by pre-training a Transformer on a large corpus of text data [20, 21]. Pre-training allows the model to learn the statistical patterns of the language, which can then be fine-tuned for a specific task. DialoGPT [21], specifically, was trained on a dataset extracted from Reddit comments, enabling it to generate human-like text that is contextually relevant to a given prompt. As a language model, DialoGPT can generate diverse responses, making it suitable for open-ended tasks such as chatbot conversations. DialoGPT, proposed in DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation by Zhang et.al, is an extension of Generative Pre-trained Transformer (GPT)-2 developed by OpenAI [22]. It leverages the power of a large-scale language model to generate human-like text in conversational response generation, and is trained on long comment chains retrieved from Reddit that span from 2005 through 2017 [21]. DialoGPT extends the PyTorch transformer (HuggingFace) to attain a close performance to humans in single-turn dialogue environments. Conversational agents leveraging DialoGPT proved that they were able to generate more contextually aware text than strong baseline systems [21]. For training or fine-tuning DialoGPT, causal language modelling training can be used, and the model follows the OpenAI GPT-2 approach, modelling a multi-turn dialogue session as a long text and framing the generation task as language modelling. The method concatenates all dialog turns within a dialogue session into a long text, ended by the end-of-text token. As mentioned previously, GPT-2 itself is developed by OpenAI, and it leverages the power of a large-scale language model to generate human-like text. This transformer model, chatbots capable of generating more human-like responses and handling a wide range of queries can be built.

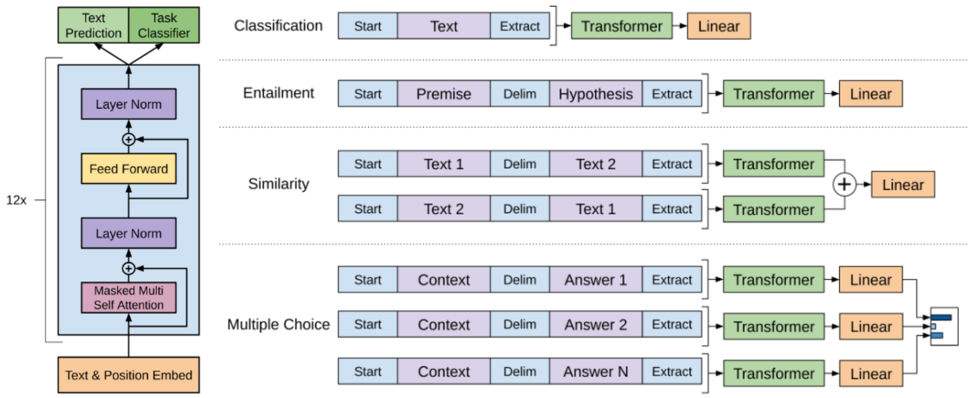

Figure 2.3: (On the left side) The Transformer architecture and training objectives employed in the study by Radford et al. (On the right side) Input transformations for fine-tuning the model on various tasks. This figure is present in the work of Radford et al. in "Improving Language Understanding by Generative Pre-Training".

2.3.4 **Botkit and Natural**

As mentioned in section 2.2.3, Botkit is a relatively easy to use chatbot development framework and its versatility and thus making it a suitable candidate for exploring the potential of chatbot development frameworks in customer support applications. It is worth mentioning that Botkit is also a part of Microsoft Bot Framework [17]. As mentioned in 2.2.3, Botkit can be combined with other third party services or libraries and in this specific case, Natural was chosen. Natural is a library in Javascript that is described as a general natural language facility for Node.js applications and it offers a broad range of natural language processing as well [23]. One of the core techniques used in Natural is tokenization and tokenization as mentioned in the previous sections, is the process of splitting text into individual words or sentences that makes them manageable pieces of data. Stemming is another crucial technique that Natural utilises where words are reduced to their root or base form and this allows for different forms and variations of the same word. Natural also utilises powerful classification tools enabling developers to categorise text data on predefined categories or criteria and is particularly useful for sentiment analysis, topic modelling, document classification, and so forth. Natural also utilises string similarity algorithms, phonetic matching, N-grams and offers a wide range of techniques and tools that enable developers to create language-based applications and conversational agent experiences. Natural will be used in combination with Botkit [23].

3 Methodology

This section will discuss the aspects of the project such as the methodology itself, the test cases used for the chatbots, the data used, potential risks, ethical considerations as well as other significant elements that contribute to a comprehensive evaluation.

3.1 Setup

The selected context for this project is healthcare, viewing medical chatbots as a unique form of customer support. These are chatbots with various functionalities such as: provide users with information, guidance, and assistance, enhancing accessibility and efficiency in healthcare delivery, providing personalised assistance, aiding in triage and appointment scheduling, and acting as reliable information sources, and at times these functionalities are even combined. The study will involve implementing three distinct chatbots (DialoGPT-based, ChatterBot, and Botkit with Natural) and one Rasa-based chatbot, training them on a healthcare dialogue dataset or more specifically COVID-19 Dialogue Dataset, and assessing their performance using multiple metrics. The performance evaluation for each chatbot will be based on its nature and expected capabilities. The DialoGPT-based, ChatterBot, and Botkit with Natural chatbots will be evaluated on their ability to handle complex queries, considering their broad language understanding and generation capabilities. The Rasa chatbot, known for its intent-based capabilities, will be assessed by checking if the chatbot can choose the correct pre-programmed answer based on the user's query intent because RASA itself lacks the properties of generating an answer. It is important to note that 'performance' in this context has multiple dimensions, depending on the chatbot's intended function and design. Metrics will be chosen and applied judiciously to respect these differences. The setup for the evaluation draws upon the work done with the COVID-Dialogue dataset (Ju et al., 2020) and the development of fine-tuned models such as GPT-2, BART, and DialoGPT. We also acknowledge the contributions of Azmarie's GPT-2 fine-tuning project and Rushi Chitre's DialoGPT-Finetune project, which employ the UCSD AI4H's COVID-19 related dialogue dataset [24, 25, 26]. Our research builds upon these efforts, adapting and expanding their methodologies to a comparative evaluation of several open-source chatbot frameworks in a healthcare context.

3.2 Equipment

A laptop utilising the AMD Ryzen™ 5 5500U with a base clock speed of 2.10 gigahertz (GHz), Nvidia GeForce® GTX 1650 Ti and 8 gigabytes (GB) will be used for the measurements and training of ChatterBot, Botkit and Natural whereas Google Colaboratory will be used for training of the DialoGPT model. The operating system of this laptop is Microsoft Windows 10 (Version 21H2). For Rasa the following equipment will be used: Intel(R) Core(™) i5-9300H CPU with a clock speed of 2.40GHz, 8GB of RAM, Windows 11 x64 processor.

3.3 Metrics

This section will describe each of the metrics and what they entail in the specific context of utilising chatbots for customer support in medicine.

3.3.1 **Accuracy**

Accuracy is a crucial metric for determining how well a chatbot can understand user queries and provide correct and relevant information and in the medical domain providing accurate information is of utmost importance to ensure users receive reliable guidance on health-related issues. The accuracy will be measured, if possible, in a comparison between the chatbot's responses to a set of predefined or correct answers or in other cases what would be called a ground truth which is available in the appendix and sourced from credible medical sites. The similarity will be measured in a number of ways depending on the chatbot and for individual answers, and as mentioned in section 1.1, the Jaccard similarity (character-level similarity), bilingual language evaluation (BLEU) and recall oriented gist evaluation (ROUGE – ROUGE-1, ROUGE-2, ROUGE-L) scores will be used. To meaningfully compare these metrics, a weighted scoring system is proposed that takes into account the relative importance of each metric. For instance, in a context where sentence-level precision is vital, such as medical advice, more weight might be given to ROUGE scores, which assess the overlap of unigrams (ROUGE-1), bigrams (ROUGE-2) and longest common subsequences (ROUGE-L), providing insights into sentence structure and word order. In contrast, if the focus is on assessing how many correct key terms or phrases the chatbot is using, the Jaccard similarity might be prioritised which means that it considers the intersection of unique characters in the words between the chatbot's response and the ground truth or in other words the expected response present in the training and testing dataset as well as answers from medical sources. BLEU scores, which measure the overlap of n-grams, provide a balance between assessing the presence of correct terms and the structure of the response. By adjusting the weightage according to the specific requirements of the use case, a combined score can be derived that can be used to compare the overall performance of the different chatbot frameworks. It's important to note, however, that no evaluation metric is perfect, and different metrics might be more or less suitable depending on the specific goals and context of the chatbot. As such, human judgement and interpretation will play an essential role in understanding and leveraging these scores effectively. The scores range from 0 to 1 where the closer the score to 1, the better. To be more specific, for Jaccard similarity, the closer the score is to 1, it means that the characters the two texts share are very similar to one another. BLEU focuses more on precision therefore a score closer to 1 would represent an ideal match whereas a 0 would indicate a mismatch between the generated and reference text. Similar to BLEU, the scores for ROUGE range from 0 to 1, with 1 indicating perfect overlap with the reference text and 0 indicating no overlap. ROUGE is recall-oriented, focusing on how much of the reference text was captured in the generated text, as opposed to precision (which BLEU focuses on). The proposed approach allows us to harness the strengths of each metric, providing a nuanced understanding of chatbot performance that can inform the choice of the most suitable chatbot framework for customer support in the medical field and in order to test Rasa's accuracy, a comparison will be drawn if the chatbot can pick the correct pre-programmed answer which that will be based on the ground truth as mentioned above.

3.3.2 **Consistency**

To assess consistency, the chatbot's ability to provide similar or identical answers to equivalent user queries must be measured. The consistency for can be checked in the following way: by submitting a set of queries, rephrasing them and then submitting them again. If the chatbot itself offers what could be considered as accurate responses by meeting a certain threshold for the rephrased queries then it can be said that it is consistent indeed. Consistency, although similar to accuracy, is not exactly the same as in this case, the focus will be more on the ability of the chatbot to handle complex and similar queries. Three distinct queries were selected from the testing dataset that were related to COVID-19 [Appendix 2] and rephrased each of them in three different ways [Appendix 3], resulting in a total of 9 queries. Each query variation aimed to convey the same underlying question but with slightly different wording and sentence structure. Using the responses that were generated for each of the queries, and then the pairwise Jaccard similarity was computed between the responses for each set of three rephrased queries. The Jaccard similarity (at character-level) was recycled from the accuracy testing because it's a simple and effective way to measure the similarity between text responses, taking into account both the presence and absence of words in the responses although there will be human evaluation included as well. Higher Jaccard similarity scores indicate that the generated responses are more consistent, while lower scores suggest that the responses are less consistent. The chosen queries and the medical source answers for them can be found in Appendix 2. RASA will follow the same format of taking three different queries and rewriting them. Afterwards, a check will be done if the chatbot can attribute the correct intent and pick the correct pre-programmed answer. Instead of generating a response to handle the query, the focus was on whether the Rasa-based chatbot could attribute the correct intent and pick the correct pre-programmed answer.

3.3.3 **Response time**

Although response time is heavily dependent on hardware, we can still glean valuable insights from our testing. Specifically, two key pieces of information can be ascertained:

- ➢ Determine the time it takes for the chatbots to provide an answer, ensuring that the response time is reasonable and does not render the program ineffective.
- ➢ If the programs are executed on similar systems, their response times should be consistent with one another.

Considering that the two programming languages utilised are Javascript and Python, for Javascript, console timers do not provide adequate accuracy therefore **Performance** timers will be utilised and in Python, the **timeit** module will be utilised and both functions will provide accurate measurements of response time. For ChatterBot, DialoGPT and Botkit combined with Natural chatbots, the response time was measured simultaneously with the accuracy results. For Rasa, a server was set up, and through a Python file multiple queries were inputted 10 times with the aim to obtain the average response time for each query.

3.4 Training

In this section, the training process for each of the selected chatbot frameworks: Rasa, DialoGPT, ChatterBot, and Botkit with Natural will be briefly described. The primary dataset used for training these chatbots is the COVID-Dialogue dataset [26], which comprises medical conversations between patients and doctors pertaining primarily to COVID-19, pneumonia, however there is a small number of conversations related to other illnesses as well. Additionally, Frequently Asked Questions (FAQ) regarding COVID-19 from the World Health Organization (WHO) [28] have also been incorporated in the training process for ChatterBot and Botkit with Natural specifically. While the training process varies across the different chatbot frameworks, each is designed to leverage the COVID-Dialogue dataset and additional information to maximise their capability in handling medical inquiries accurately. The training for DialoGPT was based on previous works utilising the same dataset, one of which is a project on fine-tuning GPT-2 [29] and the other for Dialo-GPT [29]. The training for Botkit and Natural or more specifically Natural, was based on the official documentation and the examples provided [31], and this case was the same for ChatterBot and Rasa as well [32, 33]. These are high-level details and for simplicity's sake, all the cited documentations and resources as well as the GitHub repository provided contain step-by-step guidelines for the training of each respective chatbot framework.

3.4.1 **DialoGPT**

The training process involves fine-tuning the model using the COVID-Dialogue dataset to ensure they can generate meaningful and relevant responses in the context of medical conversations. The model is fine-tuned using the PyTorch machine learning library and by leveraging this dataset, the chatbot gains an enhanced understanding of medical terminology, common questions, and appropriate responses related to COVID-19 and other pneumonia-related topics.

3.4.2 **Rasa**

The training of Rasa with the COVID-Dialogue dataset follows two primary methods. The first method involves the manual construction of sentences based on the data, which is a labour-intensive process requiring thorough understanding and expertise. The second method introduces key terms into the raw sentences, offering a more straightforward but slower process. More intricate details of these training methodologies can be found in the Rasa's documentation and in Appendix 1.

3.4.3 **ChatterBot**

For ChatterBot, the training process involves not only using the COVID-Dialogue dataset but also potentially web scraping additional information from reputable sources, such as COVID FAQs from the World Health Organization (WHO) website. This additional information helps to enrich the chatbots' knowledge base and ensure that it can provide up-to-date and accurate information to users.

3.4.4 **Botkit and Natural**

Botkit, by its nature, does not support native training. It instead relies on manual data provisioning from relevant sources or APIs to populate its knowledge base. To ensure comparability with the other chatbots, Botkit is supplemented with the Natural NLP library, which will be trained on the

COVID-19 dataset. Details on how Botkit and Natural can be combined for chatbot development can be found in their respective documentations.

3.5 Implementation of chatbots

In this section, important pieces of information about the implementation of each of the chatbots will be briefly described; however the more technical details will be found in Appendix 1.

3.5.1 **Rasa**

The most important details required to implement a RASA chatbot are to understand the concepts of Intent and Action. **Intents** are the essence of what a sentence means. For example the query `Hello` has the intent of greeting. **Actions** are as simple as a preprogrammed answer or as complex as a python script. For instance `action_utter_greet = "Hello I am covid bot"` Both concepts are used to train the chatbot, and one can do it by pairing an intent and an action, `[intent greet, action utter_greet]` for instance.

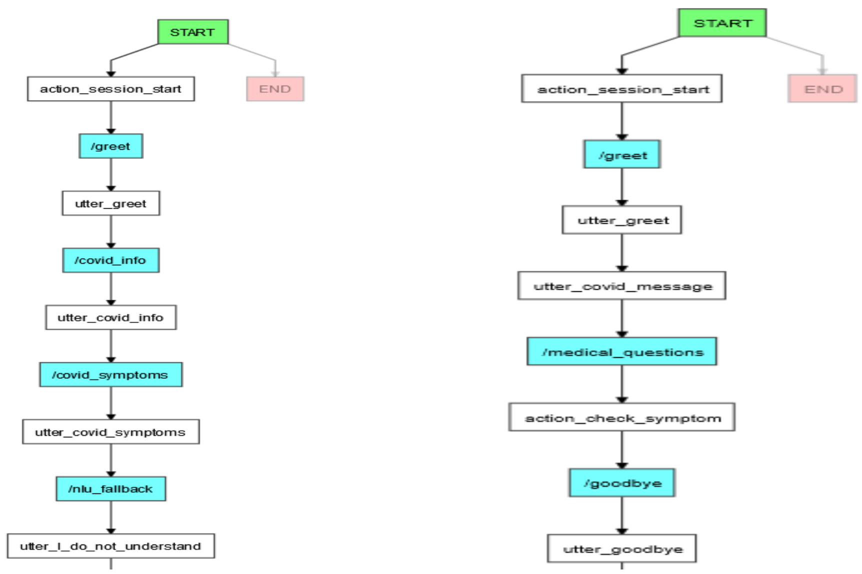

Figure 3.0: Intent-action table showing how the chatbot processes queries.

In Figure 3.0 these are two general dialogue flows of each conversation, they are very similar in practice but the difference is that the rightmost one utilises more scalable techniques, as can be seen leftmost has: utter_covid_info, utter_covid_symptoms and nlu_fallback actions, while the rightmost one just utilises action_check_symptom which in practice streamlines both actions listed before and more. The main difference is that this method takes longer to set up and also requires making the training data for the intent “medical_question” larger so it can recognize more queries.

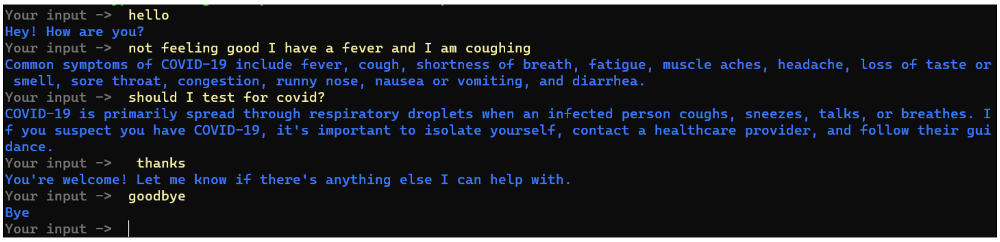

Figure 3.1: Rasa chatbot in operation.

3.5.2 **ChatterBot**

Implementing a chatbot with ChatterBot involves a series of steps. First, the ChatterBot library needs to be installed in a Python environment. The chatbot is then instantiated with various configurations, which include storage and logic adapters. Briefly described, storage adapters manage the conversation data storage and retrieval, with options including SQL and MongoDB. The logic adapters determine how the chatbot processes input and generates responses. For this thesis, we have used the BestMatch logic adapter, and the BestMatch helps select the closest matching response from the chatbot's training data. The training process for ChatterBot relies on the ListTrainer function, which takes in a list of conversation data and improves the chatbot's responses through iterative training. The data used for this project includes a list of COVID-19 FAQs and a JSON file containing COVID-Dialogue conversation data. To generate responses, ChatterBot uses the BestMatch logic adapter in combination with a strategy for selecting the most frequent response. This strategy uses the Levenshtein distance as a measure of similarity between the user's input and known statements in the chatbot's database. Finally, to enable user interaction, a chat loop is implemented and doing so allows the chatbot to listen for user input and respond based on its trained logic, continuing this cycle until an exit condition is entered. To customise ChatterBot further, developers can create their own logic adapters or integrate popular machine learning libraries like TensorFlow, PyTorch, spaCy, and more. After ensuring that the chatbot functions as intended, code for testing accuracy (Jaccard similarity, BLEU, and ROUGE scores) is also implemented. For more technical details, please refer to Appendix 1. As a note, the current ChatterBot application is built upon a Flask application template provided by chamkank [34]. As far as it goes for the application as a whole, it is based upon chamkank’s template for a Flask application of ChatterBot work [34].

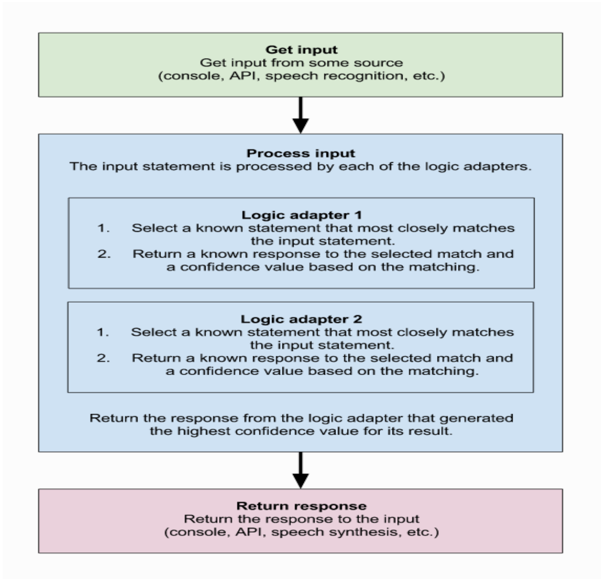

Figure 3.14: Process flow diagram for ChatterBot [35].

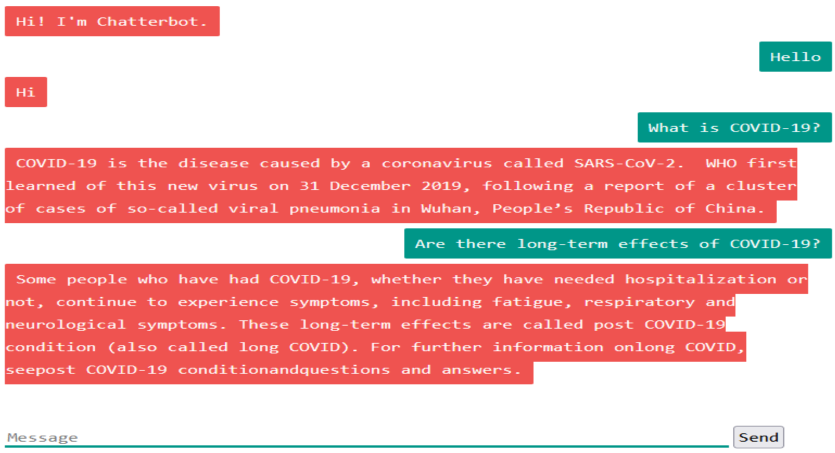

Figure 3.2: An example application of ChatterBot using Flask. Based on the chamkank's application [34].

3.5.3 **DialoGPT-based chatbot**

The DialoGPT chatbot's construction begins by leveraging the HuggingFace Transformers library, and more specifically DialoGPT-large is used, which is a pre-trained language model that serves as the base for our chatbot. The training process for DialoGPT, powered by PyTorch, includes defining specific classes and functions. This includes a custom dataset class (DatasetFromJSON) for data preparation and a main function (train_dialo_gpt) to manage the model's fine-tuning. DatasetFromJSON handles the tokenization and encoding of the conversation data, while the train_dialo_gpt function is responsible for initiating the training process. To fine-tune DialoGPT, conversation data extracted from the COVID-Dialogue dataset is fed to the pre-trained language model. Post-training, the model is capable of generating COVID-19 related responses. Following the training phase, Rushi's user-friendly chatbot interface was constructed with the help of Streamlit and the model was swapped for the one that was trained for the purpose of this thesis. Rushi used DialoGPT-small, and in this thesis DialoGPT-large is used instead. This application uses the fine-tuned DialoGPT model to answer COVID-19-related queries, creating a useful tool for information retrieval. After ensuring that the chatbot functions as intended, code for testing accuracy (Jaccard similarity, BLEU, and ROUGE scores) is also implemented, and for more detailed technical instructions and code snippets, please refer to Appendix 1.

Figure 3.3: An example of the implementation of DialoGPT-based chatbot in reference to Rushi's work.

3.5.4 **BotKit and Natural**

Botkit, an open-source framework for developing chatbots, serves as the foundation of the last chatbot, and it is written in JavaScript and running on an Express server with socket.io. This chatbot primarily handles user interactions in real time, responding to events such as option selection or illness naming. However, Botkit does not inherently support machine learning capabilities or training as it is simply a chatbot development framework, which limits the complexity and adaptability of the chatbot's conversational abilities. To overcome this limitation, the Natural Javascript library was integrated, a powerful tool for text processing and classification in Node.js. The Natural library extends the Botkit chatbot's capabilities by enabling it to understand and respond to more complex queries. The data fed to Natural is parsed from JSON files, which are then used with Natural's classifiers. More specifically, we leverage the Naive Bayes classifier - a popular, simple, and effective probabilistic classifier commonly used in text classification tasks, and even in medicine diagnosis. The function used to generate responses relies on the Naive Bayes classifier, and is then used to predict the best matching conversation index for a given user input. It effectively selects the most appropriate response to a user's query, enriching the chatbot's conversational abilities. After ensuring that the chatbot functions as intended, code for testing is then implemented. For more detailed technical instructions, code snippets and testing methodology, please refer to [Appendix 1](#).

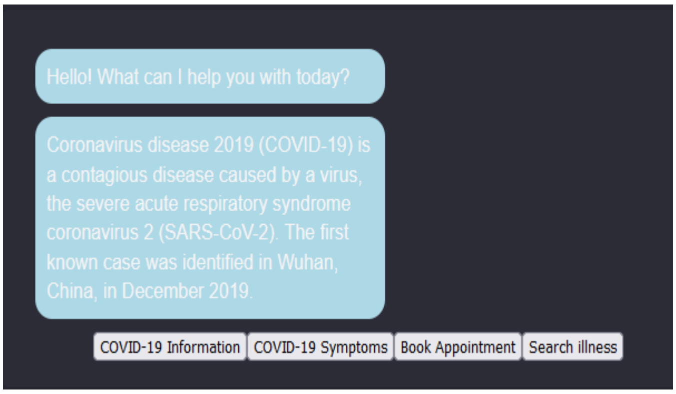

Figure 3.4: An example application of Botkit without Natural.

Figure 3.5: An example of another application of Botkit without Natural.

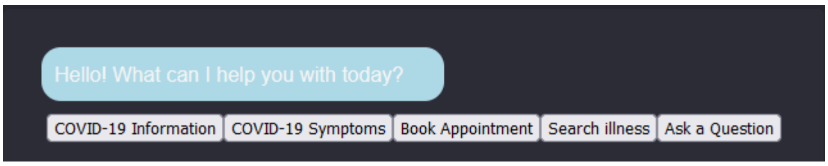

Figure 3.6: An example of another application of Botkit with Natural.

3.6 Ethical considerations

The reliance on open-source frameworks, tools, and data sets for the development and evaluation of chatbots in this thesis underlines the importance of practising ethical responsibility in research and development. Given that the data was not directly collected but instead utilised publicly available resources, the onus is on the students conducting the research for this thesis to ensure these resources are handled responsibly, and that any output aligns with ethical guidelines.

- ➢ **Data:** The conversation data used to train the chatbots, particularly with the DialoGPT model and Botkit framework and Natural Javascript library, as well as the ChatterBot library, to protect individuals' privacy, and to the best of the authors' knowledge, all significant personal identifiers were removed from the data used by the datasets' creators.
- ➢ **Use of open source work:** While open-source work offers significant advantages in terms of accessibility and knowledge sharing, it's essential to respect the original contributors' intentions and terms of use. Strict adherence to the terms of use for all open-source tools, libraries, frameworks, and data sets utilised in this study.
- ➢ **Transparency and accountability:** There is commitment to transparency about the methodology, data sources, and potential limitations of the chatbots developed in this thesis. In the spirit of open and ethical research, feedback is encouraged and scrutiny as well from the wider community.

In summary, adopting open-source materials provides a valuable opportunity to contribute to the research community while also presenting challenges that require mindful navigation, and the commitment to ethical responsibility serves as a guide to ensure that the chatbots developed contribute positively and responsibly to the research community.

4 Results

This section will provide the results or more specifically plots that will portray each of the chatbots

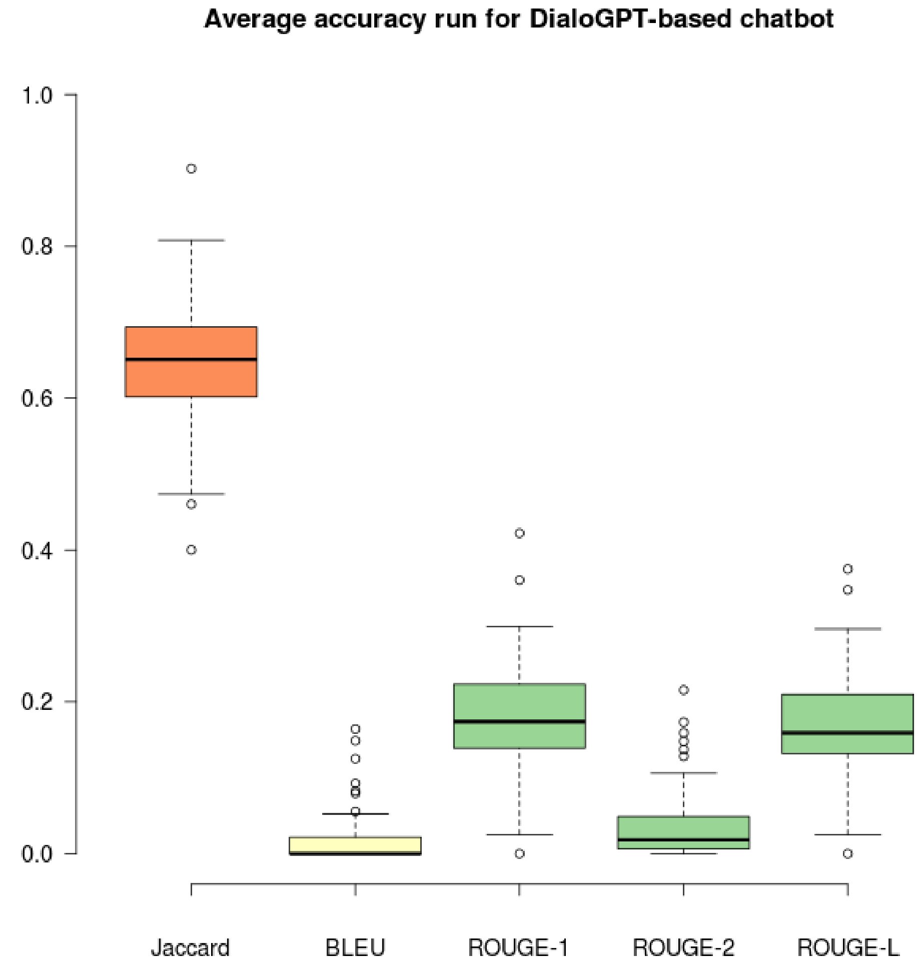

Figure 4.0 – Boxplot for average accuracy results for DialoGPT based chatbot.

In the figure above, DialoGPT showed the highest degree of overlap between the generated and target responses, as indicated by the Jaccard Similarity Mean of 0.644, the best of the three. While its BLEU Score Mean remained low at 0.018, it was still significantly higher than the other two systems (the figures below). The ROUGE scores were also the highest among the three systems, with ROUGE-1 Mean at 0.180, ROUGE-2 Mean at 0.037, and ROUGE-L Mean at 0.169. This indicates that DialoGPT was marginally better in generating responses that shared unigrams, bigrams, and longest common subsequences with the target responses compared to the other two chatbots below.

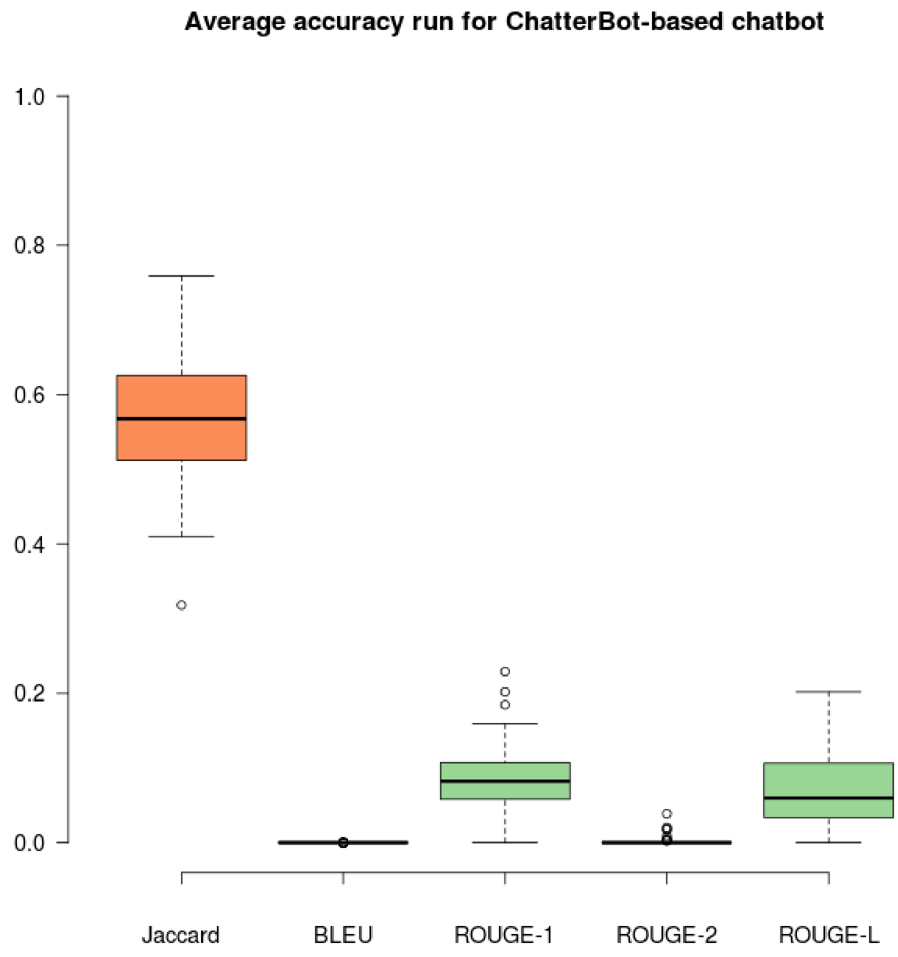

Figure 4.1 – Boxplot for average accuracy results for ChatterBot based chatbot.

In the figure above, accuracy results reveal that ChatterBot exhibited high overlap between generated and target responses, as reflected by the Jaccard Similarity Mean (0.566), lowest of the three. However, it had similar issues as Botkit with Natural, more specifically with the BLEU score Mean remaining extremely low (near 0). The Mean ROUGE-1 score was 0.085, ROUGE-2 was 0.0019, and ROUGE-L was 0.0703. While this indicated a slight improvement from Botkit with Natural, especially in matching unigrams (ROUGE-1), it still suggested that the responses generated by ChatterBot often didn't share long sequences with the target responses.

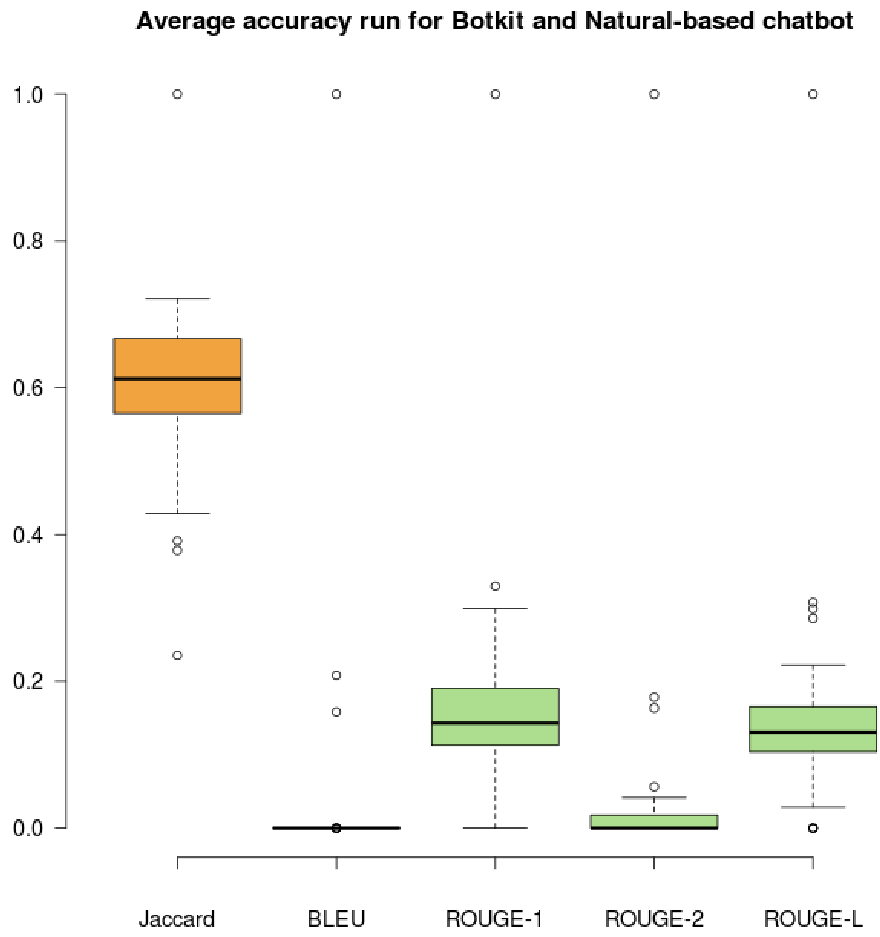

Figure 4.2 – Average accuracy results of Botkit and Natural based chatbot.

In the figure above, it can be seen that Botkit with Natural exhibited a moderate degree of similarity between generated and target responses, as indicated by the mean Jaccard Similarity of 0.610, second best in this aspect. The chatbot's performance in constructing sentences similar to the reference responses, however, was notably poor, as evidenced by the near-zero mean BLEU score and this points towards a deficiency in linguistic fluency and precision. The evaluation of sequence overlap via ROUGE metrics further highlighted the limitations of Botkit and Natural. The mean ROUGE-1 score, considering unigrams, was only 0.14, indicating a lack of substantial overlap with single words in the reference sentences. The situation was even more stark for the mean ROUGE-2 score (evaluating bigrams), which came to be near zero (0.02), demonstrating a considerable deficit in capturing phrase-level or two-word combinations in the output. The mean ROUGE-L score, which considers the longest common subsequence, was slightly higher at 0.13. Nonetheless, it still underscores a deficiency in producing longer coherent sequences of text similar to the target. There are 5 outliers present in the data.

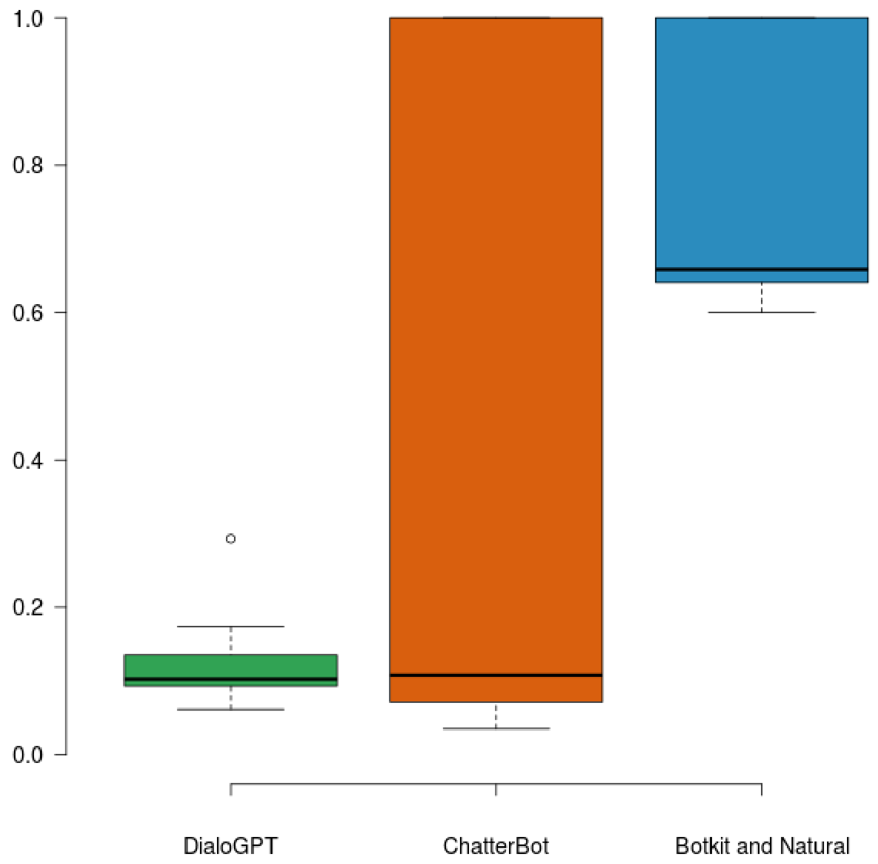

Figure 4.3 – Consistency results for DialoGPT, ChatterBot and Botkit with Natural-based chatbots.

For DialoGPT, the consistency of responses to differently phrased but equivalent queries was relatively low. This is indicated by a Jaccard Similarity Mean of 0.10. The boxplot results further show a spread of values from 0.06 to 0.17, indicating that the performance varied depending on the specific set of queries. ChatterBot, on the other hand, showed a higher level of inconsistency. While the upper whisker and 3rd quartile were at a perfect score of 1, the median dropped to 0.11, indicating that half of the tested queries resulted in significantly dissimilar responses. This inconsistency is further emphasised by the lower whisker at 0.04, suggesting some responses had very low overlap. Botkit with Natural, conversely, exhibited a relatively high degree of consistency, the highest of the group. The Jaccard Similarity Mean was at 0.66, and the interquartile range from 0.64 to 1 indicates that the majority of differently phrased but equivalent queries resulted in highly similar responses. There is an outlier present in the data for DialoGPT-based chatbot.

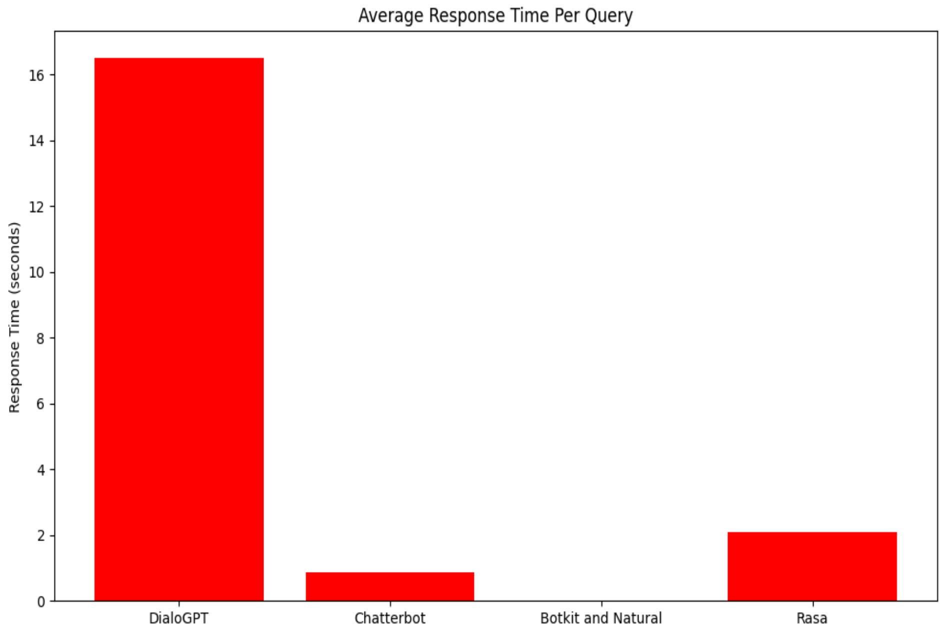

Figure 4.4 – Average response time per query results for all of the chatbots.

The figure above displays that DialoGPT chatbot was the slowest of all with an average response time per query of approximately 16.4 seconds whereas Botkit and Natural the fastest with a response time of 0.01 seconds. Rasa was excluded from the previous graphs because as mentioned in the implementation section Rasa does not generate or retrieve answers, instead what Rasa does is choose a pre-programmed answer, making it irrelevant to compare an already defined answer with the real answer, so instead the tests will be about Rasa's identification of users intent from a query and then check if the chatbot can pick the correct pre-programmed answer, which are actions. For those interested in a deeper and more detailed analysis, the appendices of this degree project contain comprehensive results. All the raw data, individual scores, and response timings, are meticulously presented in the GitHub repository under the results folder provided in Appendix 1 accessible via the provided link. Information located there will provide more context and further understanding of the testing beyond this section.

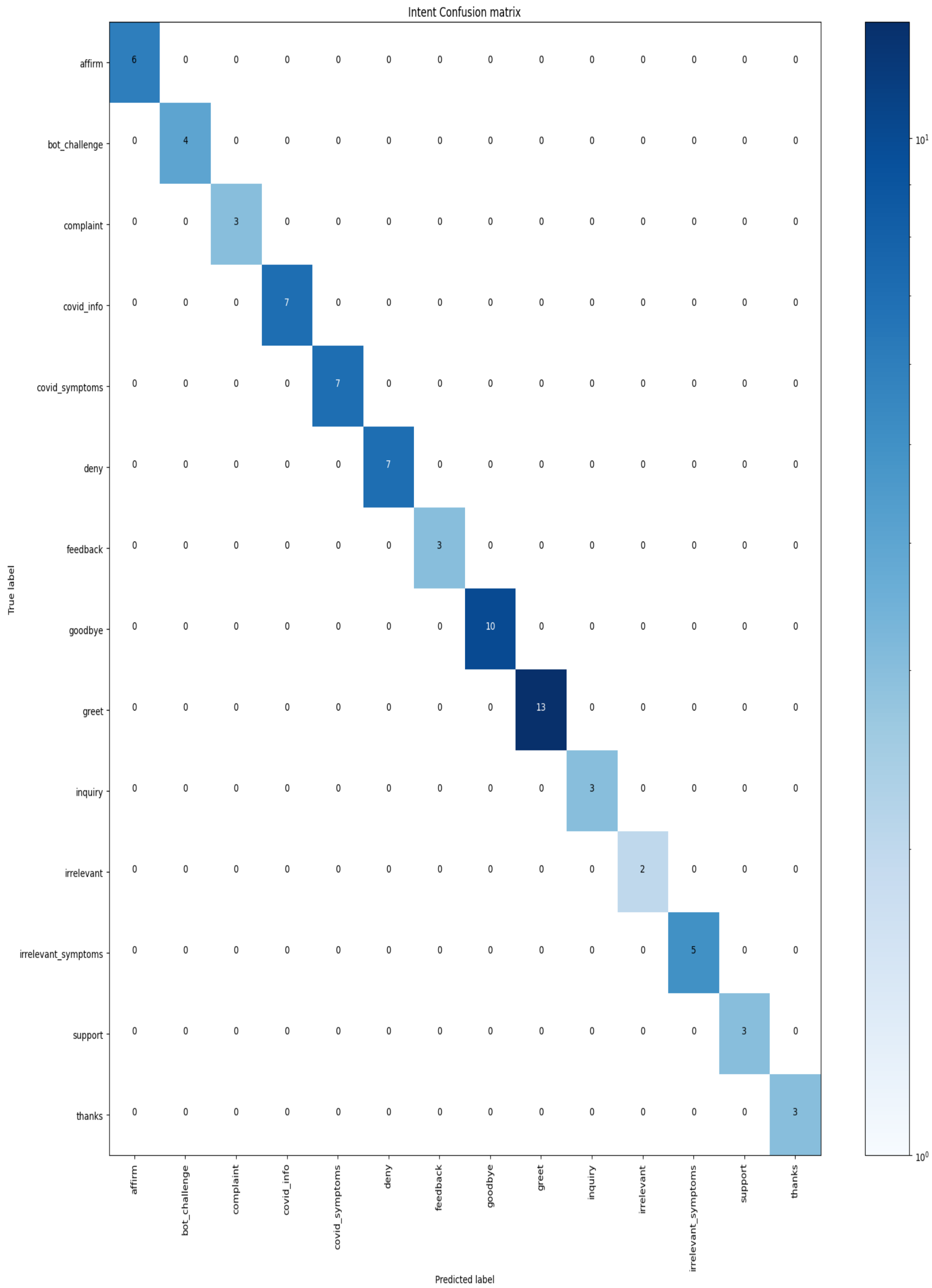

Figure 4.5: Confusion matrix of predicted intents and actual intents.

```
- Evaluation Results on ACTION level:
-  Correct:            22 / 24
-  F1-Score:           0.943
-  Precision:          0.963
-  Accuracy:           0.944
```

Figure 4.6: Action results in numbers.

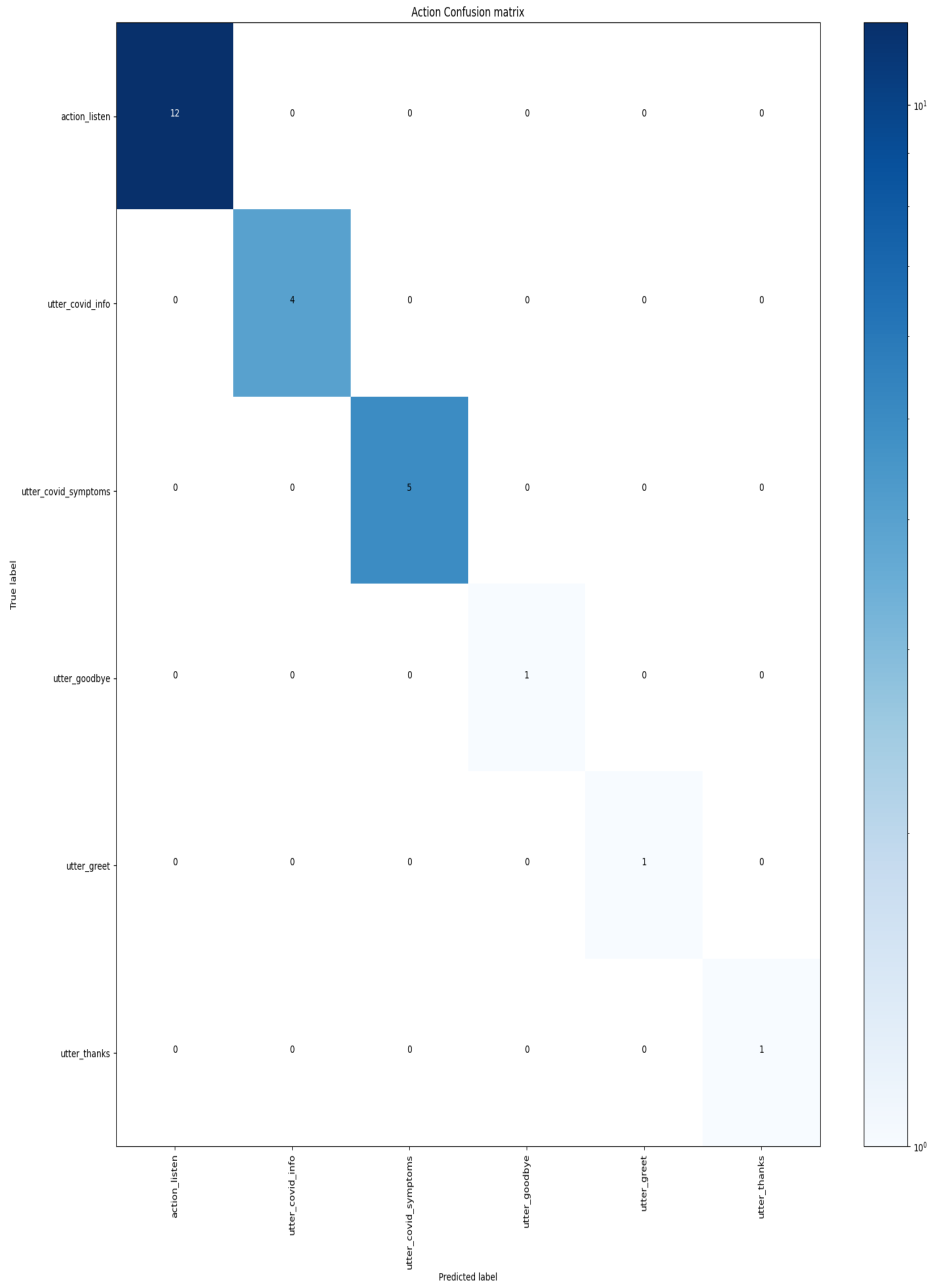

Figure 4.7: Confusion matrix of predicted actions versus actual actions.

Figure 4.5 displays the intent confusion matrix, it tests if the chatbot can predict the correct intent from a query, the chatbot must pick an intent so in this case it shows that it managed to pick all correct intents for each query. Figure 4.6 and Figure 4.7 are about actions, Figure 4.6 presents the actions that were accurate and those who are not, 22/24 are the raw results, 22 of the 24 queries were identified with the correct intent and action. As can be seen there is a F1-score, Precision and Accuracy, Accuracy: is how often a machine learning model is correct overall, Precision: precision is the ratio `tp/(tp+fp)` where `tp` is the number of true positives and `fp` the number of false positives, F1-score is the harmonic mean between precision and recall, for all the closer to 1 the better. All three present favourable scores of Accuracy: 0.944, Precision 0.963 and F1-score 0.943. For the actions that are not presented in the table are for actions where the chatbot does not know how to proceed and so it does nothing, Figure 4.7 illustrates the results. The chatbot was tested through a story with given answers.

```
- story: symptoms and medical treatment
  steps:
  - user: |
      hello
      intent: greet
  - action: utter_greet
  - user: |
      I have had a runny nose and have been coughing and have had on and
off headaches. I went to the doctor the first time, he gave me an
antibiotic and I got worse. Then I went back and he gave me a different
one, but it seems I still have these symptoms?
      intent: covid_symptoms
  - action: utter_covid_symptoms
  - user: |
      thank you
      intent: thanks
  - action: utter_thanks
  - user: |
      good bye
      intent: goodbye
  - action: utter_goodbye
```

Code 4.0: Excerpt from test_stories.yml file.

Code 4.0 shows how the intent, the action and the queries are already set up, when the chatbot tests this it tries to predict the intent/action through the query and then proceeds to compare the predicted to the real one.

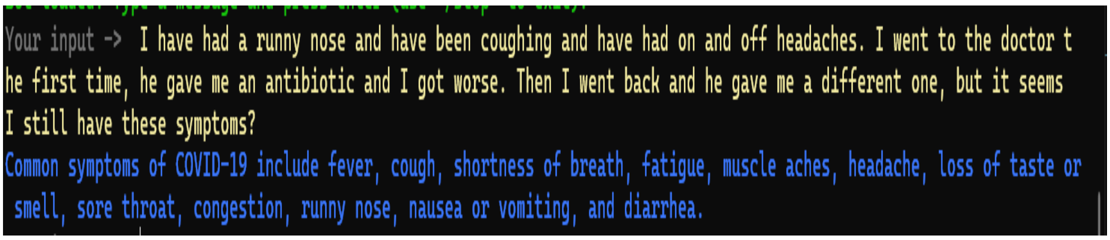

Figure 4.8: Example of the prompt in action.

Consistency testing

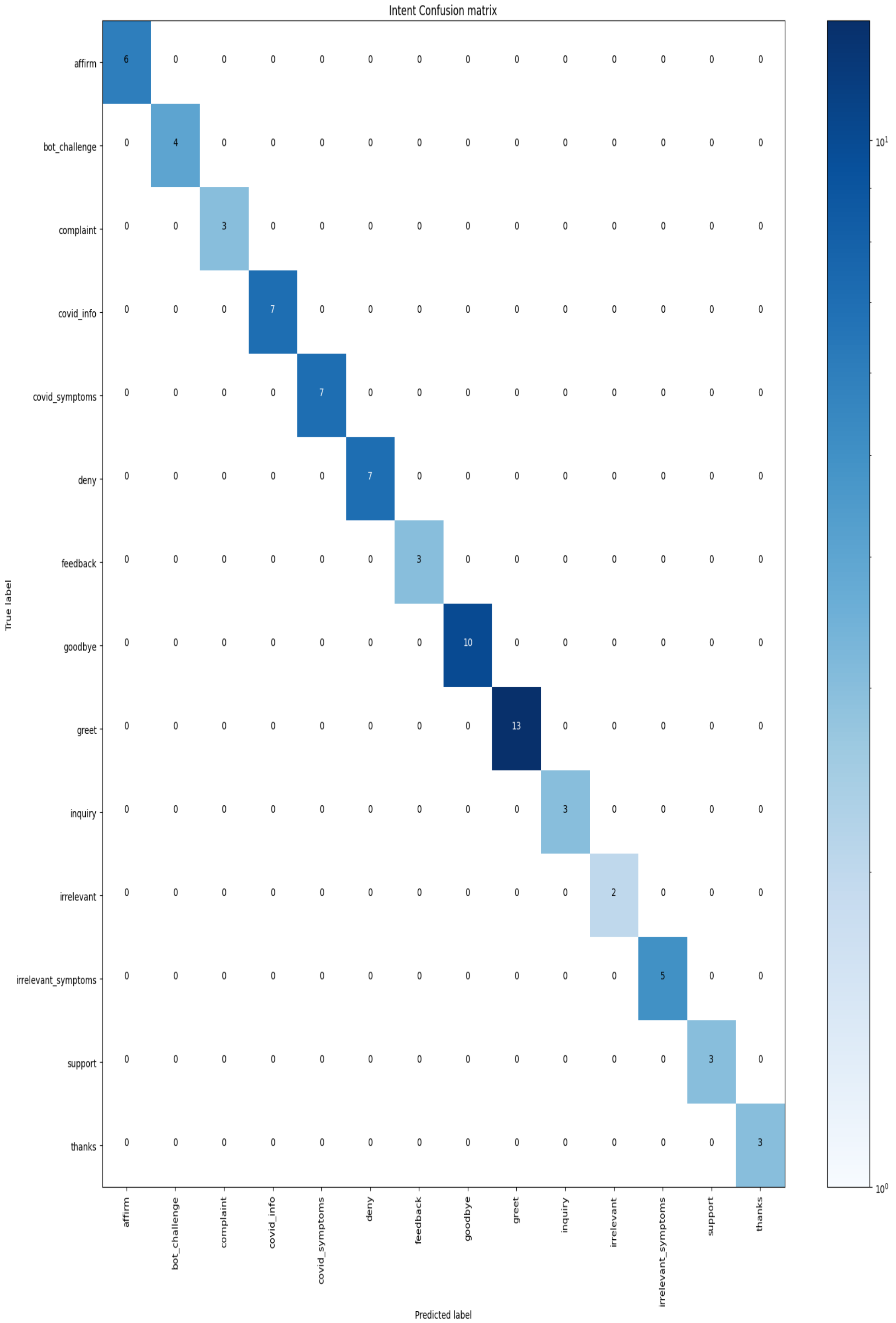

Figure 4.9: Consistency results intent recognition.

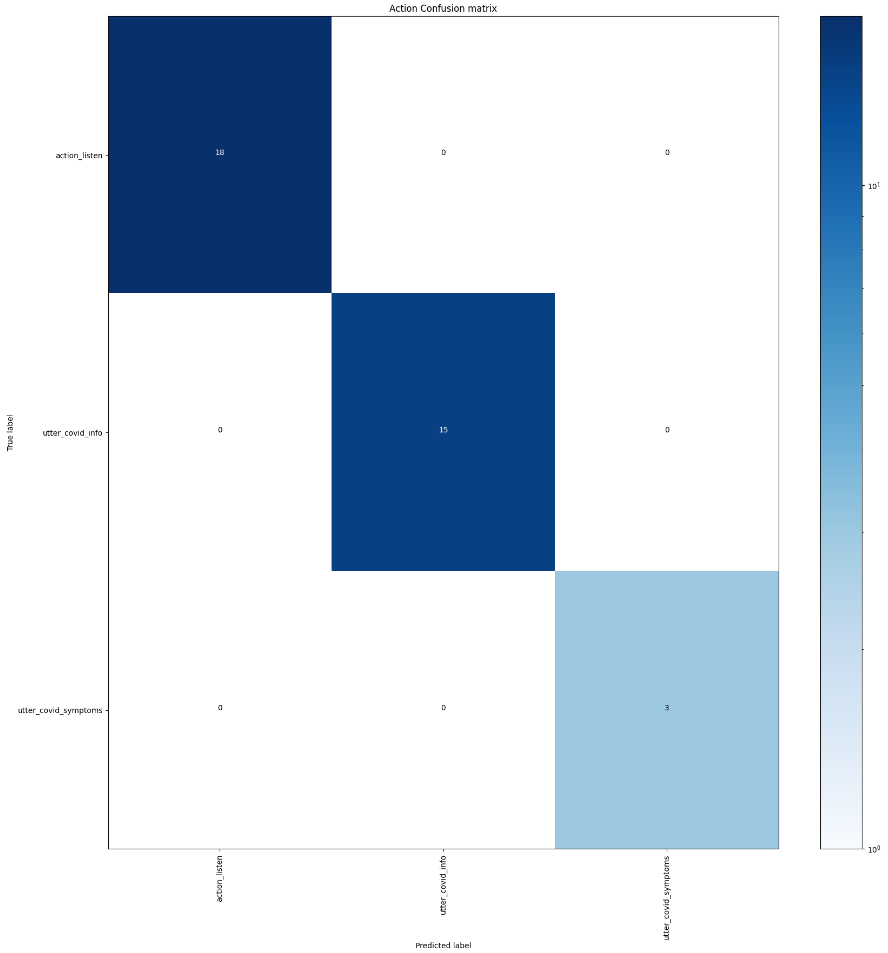

Figure 4.10: Action-Consistency Confusion matrix.

```
2023-06-04 22:59:27 INFO     rasa.core.test  - Evaluation Results on ACTION level:
2023-06-04 22:59:27 INFO     rasa.core.test  -  Correct:          28 / 36
2023-06-04 22:59:27 INFO     rasa.core.test  -  F1-Score:         0.829
2023-06-04 22:59:27 INFO     rasa.core.test  -  Precision:        0.883
2023-06-04 22:59:27 INFO     rasa.core.test  -  Accuracy:         0.796
2023-06-04 22:59:27 INFO     rasa.core.test  -  In-data fraction: 0
2023-06-04 22:59:27 INFO     rasa.utils.plotting  - Confusion matrix, without normalization:
```

Figure 4.11: Action-Consistency Values.

Figure 4.9 similar as the previous the chatbot was providing solid intent recognition from the queries Figure 4.10 and 4.11 shows that the chatbot had the correct intent and action 28/36 of the time with F1-score of 0.829, Precision of 0.883 and Accuracy of 0.796 and the reason for this is that the chatbot instead of picking a wrong pre-programmed answer the chatbot just decided to pick nothing and just skip the query.

5 Analysis

This section will analyse the results for each of the chatbots and provide insight to their key features, capabilities as well as trade-offs and limitations.

5.1 DialoGPT-based chatbot

While DialoGPT-based chatbot performed well in terms of accuracy with an average Jaccard similarity score of 0.644, it struggled with consistency as there was a degree of variance. For example, in the first run for consistency, there were inconsistencies in DialoGPT chatbot's responses and these inconsistencies manifested as different advice or recommendations for similar situations. This could be potentially confusing or misleading to users, particularly in the chosen context of health-related advice. To be more exact, there is the query of a patient asking whether he should get tested since his relative living in the same household tested positive for COVID-19, and when comparing these generated responses to the actual medical source answer, it is clear that the generated responses are contradictory in the instance of suggesting that one should get tested if COVID-19 are observed while in another response it suggests that a test might not be necessary. There are also outliers in which DialoGPT chatbot scored very high, close to 0.9 in terms of Jaccard similarity which will be discussed in the Discussion (section 5). Additionally, despite its accuracy, DialoGPT's response time was quite high, averaging around 16.4 seconds per query, which might not be ideal for applications that require rapid response times. The lack of consistency is further proved by Figure 4.3 in the Results (section 4) shows a median score of approximately 0.130. The outlier present in Figure 4.3 is related to the queries 1a, 1b and 1c in Appendix 3 in which a patient is experiencing lung discomfort and sporadic symptoms like sneezing and coughing once daily. They are inquiring whether these symptoms indicate COVID-19 and if they should seek testing. Although the responses seem similar in the context of advising the inquirer about COVID-19, they differ significantly in the specific advice given where one recommends getting tested while the other does not, except suggesting to report symptoms to the general physician. There are differences in the exact phrasing and word usage even in similar parts of the response therefore, a Jaccard similarity of 0.3 at character-level indicates that there is considerable dissimilarity in the characters used in the responses, which is plausible given the differences observed in the responses. Thus, it can be considered as an outlier in terms of consistency between the responses. Given that the 1st quartile and 3rd quartile range also spans from 0.090 to 0.160, it implies the Jaccard similarity scores are generally quite low for this specific chatbot. From the implementation and results, the key features and capabilities of the DialoGPT-based chatbot can be deduced.

While the DialoGPT chatbot failed to provide entirely accurate information, it still provided relevant answers which highlighted contextual understanding as its capability. As mentioned in section 2.3.3, being based on the GPT architecture and examining the responses therefore allowing DialoGPT is more capable of generating human-like text which can enhance the user experience. When it comes to the defining features and capabilities, considering its GPT architecture, DialoGPT can handle a wide variety of topics making it versatile in handling diverse customer queries. There is also the matter of training the model with the COVID-19 Dialogue Dataset through the Hugging Face Transformer library as well as using the Streamlit interface for easy integration and development. The trade-offs as seen in the results include the response time being high, and in most cases DialoGPT chatbot struggles to maintain consistent replies on identical prompts and this could lead to confusion of the users. Another thing to note is DialoGPT is highly dependent on the training data, the quality and diversity of it as well. If the training data is accurate, comprehensive, and well-structured, DialoGPT will learn to generate more accurate and relevant responses and conversely, if the training data is riddled with inaccuracies, the model may learn and reproduce these inaccuracies, leading to incorrect or nonsensical responses.

5.2 ChatterBot-based chatbot

Despite its response time averaging 0.785 seconds per query, ChatterBot struggled with accuracy, attaining an average Jaccard similarity score of 0.566. There are outliers when it comes to accuracy, with one being near 0.2 which will be further examined in the Discussion (section 6.1.2). Additionally, there was a significant divergence in sentence structure, vocabulary, and phrasing, as indicated by the BLEU and ROUGE scores that were close to zero. These results point to the need for improvements in ChatterBot's response generation capabilities. In the consistency test, the chatbot had notably low Jaccard scores, revealing high variability in responses to the same prompt. This suggests a high degree of variability and inconsistency in ChatterBot's responses. Looking at the various queries that the ChatterBot-based chatbot answered [Appendix 1], it appears that ChatterBot is plucking relevant responses from the training dataset based on the Levenshtein distance method and the BestMatch adapter. However, the variation between the responses seems to indicate that ChatterBot struggles to provide consistent advice or information when queried with similar prompts. This characteristic, same way as for the DialoGPT-based chatbot, could also potentially lead to confusion for users. For instance, in the first run for consistency, when posed with a query about a patient potentially having COVID-19 symptoms, ChatterBot's responses varied significantly.

The chatbot, at one point, recommended getting tested and in another response, suggested waiting a few days. This contradictory advice to identical queries clearly illustrates the chatbot's inconsistency. ChatterBot's key features and capabilities include its quick response times, which would be beneficial for real-time customer interactions, and its ability to locate responses from a variety of pre-loaded or custom conversation corpora. However, its trade-off is the compromise between speed and accuracy and consistency. Its responses may often not align with the expected outcomes, and there can be significant variation in responses to identical or similar prompts. The limitations also include its dependency on preset conversation datasets which restrict its ability to learn and adapt unique answers, unlike language models such as DialoGPT. To mitigate these challenges, the chatbot's training data should be enriched, updated regularly, and monitored for its performance consistently for necessary improvements. The ChatterBot-based chatbot shows room for improvement in generating relevant and structured responses.

5.3 Botkit and Natural-based chatbot

Botkit with Natural demonstrated an impressive level of efficiency, offering the fastest response time averaging 0.01 seconds per query. The mean Jaccard Similarity score of 0.610, along with low BLEU and ROUGE scores, shows a noticeable divergence from the desired responses. However, there were outliers nearing 1.0, indicating a few instances where the chatbot produced highly similar responses to the targets. Interestingly, when evaluated for consistency, Botkit with Natural demonstrated a relatively high degree of coherence based on the score. With a mean Jaccard similarity of 0.66, the chatbot generated similar responses to differently-phrased but essentially equivalent queries. However it is worth noting that there were outliers nearing 1.0, and these will be discussed further in the Discussion (section 6.1.3). This points to a more robust response generation for this chatbot in terms of maintaining context relevance, even though the content of the responses might not always align perfectly with the target. A concern, however, remains about the variation in the chatbot's responses. To be more specific, in the run for consistency, the chatbot was inconsistent in advising a patient expressing potential COVID-19 symptoms. In one response, it recommended immediate testing and self-quarantining, while in another response, it suggested staying at home and taking medication. The responses seem to be derived directly from the training dataset based on symptom similarity in the queries, which can lead to conflicting advice. Considering the disparity in the responses, this also poses a threat at confusing users, indicating a need for improvements in the chatbot's response generation approach. The Botkit with Natural chatbot's primary strength lies in its ultra-fast response times, making it suitable for applications where rapid response is vital. However, its major trade-off is what could be considered a heavy reliance on keyword-based matching. Since the Naive Bayes Classifier was used in this specific implementation of Natural in Botkit, it treats each feature (in this case, each word in the query) independently, it can overemphasise certain keywords and fail to consider the broader context. For example, in a query where a patient was inquiring whether they should do a test since they were in direct contact with another person however were asymptomatic for the time being. The classifier most likely attached significant importance to the word "asymptomatic" and failed to give due importance to the user's direct exposure to a COVID-19 positive individual, thereby resulting in an inappropriate response in which it retrieved an answer which stated that no test was required [Appendix 1, first run of Botkit and Natural]. Its responses are derived directly from training dataset thus limiting its capability to retrieve unique answers and to counter these challenges, businesses could focus on improving the quality of training data, incorporating a broader range of data, and regularly updating the training data to reflect the most current information. Another mitigation is to utilise this combination for information retrieval entirely. The results confirm that the Botkit and Natural chatbot, although quick, struggles to generate relevant and structured responses such as in the case aforementioned above, and improvements in its response generation are warranted.

5.4 Rasa-based chatbot

Rasa exhibits proficiency in comprehending user queries and accurately categorising them based on their underlying intentions. In terms of action alignment, Rasa demonstrates a satisfactory ability to associate intents with predefined responses. It effectively avoids any misinterpretation between user intentions and system actions. A notable pattern seen is that RASA rarely picks the wrong action, instead it just does not pick an action instead implying that it can be addressed through more training and generally it's less likely to make mistakes. The primary limitation of Rasa lies in its implementation process. It necessitates the manual creation of "generic" sentences derived from the training data, which heavily relies on the individual responsible for this task. Additionally, when custom actions are not employed, Rasa's scalability is significantly compromised, rendering it highly effective in addressing a single topic but lacking in managing multiple topics simultaneously. Furthermore, the setup of these custom actions proves to be time-consuming. Nonetheless, as evidenced in the implementation section, Rasa has the capability to consolidate three distinct actions into a single action in addition to having an average response time of 2.1 seconds it displays a reasonable amount of waiting time.

6 Discussion

This section will discuss the results, various aspects in which the project could have been improved, what was learned and what should be kept in mind for similar projects.

6.1 Discussion of results

This section will delve more into the results and discuss them in more detail. Details such as the outliers and how the findings relate to the research questions of the study. The comparative evaluation of the three chatbots, as in the Analysis section, reveals distinctive strengths and weaknesses of each solution.

6.1.1 **DialoGPT-based chatbot**

DialoGPT displayed good performance in generating contextually relevant responses, achieving an average Jaccard similarity score of 0.644. There were actually two cases in which the DialoGPT chatbot actually achieved an approximately 0.9 Jaccard similarity score. The first case is related to a query of a user asking "*Can animal byproducts spread coronavirus? Do I have to worry about milk, eggs, or fruits?*" [Appendix 1, DialoGPT run 1]. The generated response is correct in regards that there is no evidence that animals play a significant role in spreading the virus that causes COVID-19 although there are repeated sentences, more specifically "*Would you like to video or text chat with me?*" which frankly would lead to misleading the users into believing that video or voice chats are possible. The generated response is somewhat on par with the expected response as well, which was this: "*No, we have no evidence of any concern with such spread…*". When it comes to Jaccard similarity, the score represented measures the similarity between two sets, and a higher score means a greater overlap of characters in words between the two sets. Therefore, if a generated response contains a significant number of characters in the words that are also in the expected response, it will have a high Jaccard similarity score. The example above as well as the other instance where DialoGPT got approximately 0.93 score in Jaccard similarity [Appendix 1, DialoGPT run 3] illustrates this principle as the generated and expected responses share common phrases i.e "*Would you like to video or text chat with me?*" and the initial sentence aligns closely with the expected output. These instances also were the outliers in BLEU and ROUGE-2 scores as well. The reasons as to why it repeated this sentence multiple times in the same response itself, could also be related to a number of factors. It is quite possible that during training, a lot of sentences offering video or voice chats were present and therefore the model incorporates this in a wide array of scenarios. There is also the matter of different sampling parameters when it comes to generating the responses such as temperature, top-k and top-p which allows for repetition of similar but not identical sentences. During testing, there was a function that was not utilised and it basically removes unfinished sentences from the generated response which affected the results. Overall, the answers were somewhat relevant and this is a testament to the chatbot's ability to understand the context and provide pertinent answers. It also supports the information in section 2.3.3 that DialoGPT, based on the transformer-based GPT architecture, can deliver human-like text responses. However, a considerable challenge lies in its inconsistency, as evidenced by discrepancies in advice provided for similar situations as evident in the consistency results [Appendix 2, DialoGPT consistency run]. This inconsistency is further underscored by the wide interquartile range of Jaccard similarity scores, indicating variability in the quality of responses. Another trade-off includes the high response time, which averaged around 16.4 seconds per query. Therefore, while DialoGPT demonstrates proficiency in generating relevant and unique responses, there is a clear need for improvements in response time mostly. It is better suited for conversations that do not have a strict or defined flow and allow for more flexibility.

6.1.2 ChatterBot-based chatbot

ChatterBot, on the other hand, demonstrated fast response times, averaging around 0.785 seconds per query. This attribute makes ChatterBot suitable for applications requiring real-time interaction. However, the chatbot's responses displayed significant variation in sentence structure, vocabulary, and phrasing, as highlighted by the low BLEU and ROUGE scores. Furthermore, the average Jaccard similarity score of 0.566 points towards a compromise in accuracy. This is evident in all of the runs for accuracy for the ChatterBot-based chatbot in which for the most queries, it failed to provide an accurate response and instead utilised the fallback response: "I am not sure how to answer that since I am still learning, could you please try again?" which was set in the parameters in case the chatbot fails to locate an appropriate response. For example, for the same query as mentioned in the DialoGPT-based chatbot section above, it gave the fallback response to it however the expected response addresses the user's concern. The Jaccard similarity between the two responses was 0.7 although the answer provided by the ChatterBot-based chatbot provided no actual answer. The reason why it provided the fallback response in the first place is due to a number of factors such as lack of variety in the training data, the BestMatch logic adapter with a Levenshtein distance comparison function and through this, a calculates the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into the other is done and the results with the closest distance is chosen. If a user's question closely resembles one in the WHO FAQ dataset, the ChatterBot can respond appropriately, and this could contribute to higher Jaccard similarity scores in some instances. However, this method may struggle with more complex or novel queries, leading to the use of the fallback response. It is worth noting that the ChatterBot's `maximum_similarity_threshold` parameter was set to 0.5, meaning the chatbot would opt for the fallback response if it could not find a match with a similarity score higher than 0.5. This could also be a factor in the high occurrence of the fallback response. Now, the reason as to why there was such a high Jaccard similarity score even though the answer was irrelevant, it probably is related to the fact that the way the Jaccard similarity is calculated is on character-level instead of word-level which is a limitation that will be discussed later. To briefly explain what happened, the inflated Jaccard score for the fallback response compared to the expected response is perhaps due to the reason that they share a significant number of individual characters. This is also the reason as to why it scored very high Jaccard similarity scores in consistency as well since the fallback response was quite prominent. ChatterBot is better suited for conversations with already defined or strict flows.

6.1.3 Botkit and Natural-based chatbot

Lastly, Botkit with Natural displayed remarkable efficiency with the fastest average response time of 0.01 seconds per query. Despite this impressive feat, the chatbot's accuracy and consistency were moderate with an average Jaccard similarity score of 0.610 and almost very low BLEU and ROUGE scores. Similar to the other chatbots, Botkit with Natural also provided inconsistent advice for identical situations, potentially creating confusion for users. Although compared to ChatterBot, Botkit and Natural provided identical results for all accuracy runs and proved to be consistent in that regard. The answers provided were repeated in most cases for different queries and at times, they were relevant to the context, for example, Botkit and Natural responded to the query mentioned in the DialoGPT-based chatbot analysis (section 5.1) in which a user was concerned that they were also infected with COVID-19 due to a recent contact with their sibling who showed signs of infection. Botkit and Natural managed to retrieve a relevant response: "*Isolate. No symptoms, no test. You really do not need testing if there are no symptoms, but act like you are contagious. avoid contact with people for 14 days and only consult your GP if you get symptoms.*" This answer is derived from the training dataset, more specifically a query in which a user is seeking advice on the appropriate actions for an asymptomatic individual who has been in contact with a confirmed COVID-19 case, and it emphasises the individual's concern for not waiting until symptoms manifest to take action. Just like with ChatterBot, Botkit and Natural (or more specifically Natural with the classifier utilised) has a high dependency on the training dataset, and despite its ultra-fast response times, the trade-off lies in the quality and consistency of its responses and therefore, although Botkit with Natural is well-suited for tasks requiring quick responses, conversations with a very strict or already defined flow (options-based even), significant improvements in accuracy and consistency are required. Botkit and Natural also scored very high in consistency although the reason for that is due to a high repetition of the same answer, and this is due to the fact that Natural was not set up extensively and the chatbot's inability to accurately differentiate between distinct queries and provide relevant responses shows the limitations of relying heavily on keyword matching or simple similarity metrics for response generation in chatbots.

6.1.4 Rasa-based chatbot

The dataset creation process involved examining the dialogues within the training data and deriving a generalised understanding from them. This procedure is inherently subjective and challenging to replicate consistently. Despite yielding favourable outcomes, achieving such results necessitates the expertise of diligent developers. As the dataset size increases, the difficulty of accomplishing this task escalates.Regarding the obtained results, they were accurate to the best of my abilities. The chatbot demonstrated proficiency in correctly identifying the intended user queries and aligning them with appropriate pre-programmed responses. However, this approach did not yield highly personalised answers for the users. To address this limitation, a potential solution involves integrating Rasa with other models or frameworks discussed. For instance, combining Rasa with DialoGPT can entail a process wherein Rasa comprehends and accurately classifies a query, extracting relevant keywords. Subsequently, a Python script can be utilised to execute a DialoGPT model primarily trained on the identified keywords to generate and extract an appropriate answer, which is then presented to the user. It is important to note that this combined approach may introduce a delay of approximately 2.1 + 14 seconds, which may not be optimal in terms of response time. Furthermore, it is worth noting that limited information was presented regarding the more scalable version of the system. Achieving scalability predominantly necessitates a substantial amount of setup, including the creation of individual Python scripts, each containing pre programmed responses, as well as establishing accurate entity mapping. This extended setup process is considerably more involved and because of that it was decided against using the basic implementation and the less scalable version was chosen instead since that demonstrates the more realistic outcomes. Rasa proved to be competent at understanding what a query means and classify it correctly so it can align the correct pre programmed response/action, seeing at the fact that the chatbot rarely makes mistakes instead it just refuses to pick an action if its not sure, which means that it requires more training and fine tuning.

6.2 Findings and research questions

To answer research question 1 (RQ1), the evaluation of the chatbots illustrates a clear distinction in their key features and capabilities. DialoGPT, due to its GPT architecture, demonstrated its strength in providing human-like text generation, reflecting a high level of contextual understanding. It is versatile in handling diverse customer queries due to its broad knowledge base and therefore best suited for such scenarios. ChatterBot, on the other hand, showed quick response times and the ability to generate responses based on pre-loaded or custom conversation corpora, making it useful for information retrieval and customer interactions for simple queries and for redirection to human agents for more complex queries. Botkit with Natural showcased ultra-fast response times, demonstrating its efficiency and suitability for applications requiring rapid responses and is best suited for applications in which the conversation flow is very defined (option-based is optimal) lastly Rasa showed strength at detecting intent from user queries and consistency of rarely picking a wrong action and just doing nothing instead which is preferable than being wrong also with a decent response time of 2.1 seconds. DialoGPT's strength lies in its capacity to handle a wide variety of topics and generate human-like text. This, as mentioned before, makes it an excellent option for handling diverse customer queries particularly where nuanced responses may enhance the customer experience. ChatterBot's defining feature is its speed, coupled with its ability to pull responses from a variety of conversation corpora. This makes it suitable for information retrieval, real-time customer interactions where the time of response is a critical factor. The defining feature of Botkit with Natural is its unparalleled response speed, making it highly suitable for scenarios where the swiftness of response is crucial and the conversation flow is defined strictly and RASA's defining feature is precision, the definition of not making mistakes, RASA rarely showed outright picking a wrong answer. While the strengths and ideal scenarios have been identified, this allows for further exploration into weaknesses (trade-offs and limitations) and mitigation strategies therefore, research questions 2a (RQ2a) and 2b (RQ2b) aim to address these concerns. To answer RQ2a, DialoGPT, while proficient in generating contextual responses, struggled with maintaining consistency across identical prompts and exhibited longer response times. Its quality of responses is also highly dependent on the quality and diversity of its training data. ChatterBot, despite its fast response times, demonstrated limitations in accuracy, consistency, and relevance of responses. Its dependence on preset conversation datasets may limit its ability to learn and generate unique responses. The primary trade-off for Botkit with Natural is its low accuracy and consistency. It is heavily reliant on a preset training dataset, limiting its ability to provide unique responses or handle out-of-distribution queries, lastly RASAs trade-offs is that its difficult to set up, there are two contexts and both are complicated, one that the aim is a single topic requires the developers to go through the training data and dilute general queries from them, and if the chatbot were to work with more than one topic it would require the usage of custom actions which entail python scripts, which require some development time. Moving to RQ2b, to mitigate these challenges, businesses and organisations must determine what kind of queries they are expecting to handle and how the conversation flow looks. For DialoGPT, the key would be improving response times by utilising better-performing hardware and ensuring consistency of responses which could potentially be achieved by tuning hyperparameters, refining training methods, or incorporating additional training data to cover a more comprehensive set of scenarios. In the case of ChatterBot, improving the quality, diversity, and regularity of training data updates could boost accuracy. Developing custom logic adapters and conversation corpora that are specifically targeted at the anticipated user base or problem domain could improve response relevance and consistency massively. For Botkit with Natural, businesses could improve its performance by focusing on the quality and diversity of training data. They could also consider incorporating a broader range of data and updating the training data regularly to reflect the most current information. Finally, combining Botkit with Natural with other chatbot technologies (other models or third-party services) could potentially compensate for some of its limitations, provided the integration is executed effectively. Finally, a proactive approach to planning is crucial for the successful implementation of RASA. This planning phase should address important considerations such as determining the specific topics that the chatbot will handle and assessing the available training data. By gaining clarity on these aspects, potential conflicts arising from overlapping information can be effectively managed. As demonstrated in the implementation section, the utilisation of custom actions aimed at replacing multiple intents highlights the importance of thorough planning. Without careful consideration and coordination, conflicting custom actions may inadvertently collide with one another, leading to undesirable outcomes. As mentioned before, the choice of chatbot must align with the specific requirements and constraints of the organisation, including the nature of the customer queries, the desired customer experience, the available resources for chatbot development, and the pace at which responses are expected.

6.3 Improvements

There are a couple of aspects to ask how this thesis can be improved upon if there is interest to extend the work presented. Enriching the training dataset with paraphrased questions and answers or additional COVID-19 related information can help the model generalise better and generate more diverse, accurate responses regardless of the nature of the chatbot itself. Another dataset that can be utilised is the COVID-19 Open Research Dataset (CORD-19) which is a corpus of academic thesis about COVID-19 and related coronavirus research. It would also be interesting to also see other use cases such as finance more specifically see how chatbots would perform in providing information and acting as assistants so to speak. An example of this would be to see how chatbots can be utilised as assistants for start-ups and train it on the Swedish Companies Registration Offices' (Bolagsverket) general information data about forms, various processes, and so forth [36]. Another use case which would have been interesting to investigate would be to see how chatbots perform in order and inventory management, and invoicing. There are many potential use cases and these are just some of them. For the metrics themselves, it would have been interesting to examine user satisfaction and perhaps conduct a survey in which people can rate answers from the chatbot, keeping in mind regulations and privacy of course. Another investigation would be to see a custom logic adapter for sentiment analysis and how it would perform when combined with ChatterBot. For Botkit only, considering it is a chatbot framework and by itself it can only suit rule-based or more specifically a chatbot in which it interacts with the user through options, this by itself is a great way to display information quickly and neatly to the user however in the case of handling more complex queries, the Natural library was utilised together with Botkit. There is room for improvement with Botkit and Natural as well such as: expanding the natural language understanding capabilities of Natural and a specific example would be to add named entity recognition (NER) to extract entities such as dates, locations, times and so forth. This would be of great help when doing invoices for example. Another improvement would be to have context-aware conversations by maintaining the context of the conversation by storing the conversation history and use it to inform the chatbot's response generation. There are other aspects in which it can be improved such as communication with other external APIs for weather, news, booking, handling various languages, sentiment analysis, and so forth. Botkit and Natural are quite flexible. Integrating advanced natural language processing techniques like transformer-based models such as BERT and working with similar frameworks like Botkit can provide enhanced context understanding and response generation. An increased number of frameworks to include would be incredibly useful as only four as in this case is not exactly the most inclusive. Another area of exploration could be the fusion of approaches in a single chatbot system combining the strengths of both methods to create a more robust and efficient solution. Furthermore, investigating the impact of real-time reinforcement learning on chatbot performance could lead to dynamic adaptations and improvements based on user interactions. Enhancements to RASA can be achieved through the implementation of the advanced method, which was previously introduced solely within the implementation section. Additionally, the analysis of results reveals that further refinement of the training data is necessary to address issues related to aligning actions with user queries. Moreover, conducting additional training sessions would have been beneficial. Specifically, conducting more comprehensive tests for the intent "*covid_symptoms*" would have been ideal. However, due to the inherent limitations in sentence permutations, it becomes challenging to generate additional variations of the sentence *"I have X symptom, do I have Covid*?" in order to expand the training data. Notably, the chatbot's ability to detect the presence of relevant keywords such as "*symptom*," a recognized symptom, and "*Covid*" was sufficient, leaving limited room for further sentence modifications. There are other aspects in which this project can be improved upon however it is also important to note that evaluating the effectiveness of chatbots in various industries such as healthcare, legal, and customer support can help identify specific use cases where chatbots excel or require further development, and by pursuing these innovative ideas and pushing the limits of existing chatbot technologies, we can create a future where virtual assistants play an even more vital and transformative role in our lives.

6.4 Limitations

Metrics like Jaccard similarity at the character level, ROUGE-1, ROUGE-2, ROUGE-L, and BLEU are quantitative methods to compare the similarity of responses. They are useful, but they have inherent limitations, for example, Jaccard similarity does not take into account the order of the characters, meaning it could lead to falsely high scores for unstructured responses as evident in the case for Botkit and Natural as well as ChatterBot. Had the measurements been taken at word-level, the scores would have been most likely lower. On the other hand, ROUGE and BLEU scores assess the overlap of specific n-grams between the generated response and the reference, and while this allows these scores to take the order of words into account, they may fail to capture the semantic similarity in responses. The chatbots' performance is contingent on the quality, diversity, and amount of training data. In this study, it is possible that the chatbots were trained on relatively limited or unvaried data (COVID-19 related queries from various healthcare centres), which could impact their ability to provide relevant and accurate responses in a real-world setting. The consistency tests performed in this study might not cover all possible scenarios and variations of the queries. In real-world customer support settings, customers may phrase their queries in numerous different ways that might not be adequately captured in this study. This study only evaluated four types of open-source chatbots, and there are various other open-source and proprietary chatbot platforms and tools available in public, each with their own strengths and weaknesses, which were not evaluated in this study. These limitations could be addressed in future research by expanding the training data, including more diverse types of chatbots, utilising more comprehensive and diverse metrics, being some of the many potential improvements in section 6.3.

7 Conclusion

The focus of the project was to examine several chatbot platforms, implement them, and assess their performance on COVID-19 related queries using datasets derived from frequently asked questions from the WHO website as well as dialogues about COVID-19 found on GitHub. The methodology centred on evaluating the performance of selected chatbot models, including DialoGPT, Rasa, ChatterBot, and Botkit with Natural, through the lens of accuracy, response time, and consistency. Each framework, library and model had distinct features and capabilities: Rasa demonstrated robust performance in terms of accuracy, integration capabilities, and consistency, supplemented by the ability to customise responses. DialoGPT showed strong conversational abilities, offering contextually relevant responses, albeit with trade-offs in accuracy and consistency. ChatterBot, with its BestMatch logic adapter provided consistent responses however the adapter itself struggled with more complex queries. Botkit with Natural offered the fastest response times and platform integration however might face limitations in terms of pin-pointing a correct answer and scalability. That being said, the chatbots did provide what could be considered correct answers occasionally however trade-offs and limitations were an inherent part of each of the chatbots. Rasa demanded careful implementation of custom actions and resource allocation for optimal performance, DialoGPT's dependency on pre-training and fine-tuning processes could be computationally intensive, ChatterBot's logic adapters and training method, though providing consistent responses, could struggle with complex queries, while Botkit and Natural’s simplicity might restrict scalability were just some to name few. Businesses and organisations can mitigate these challenges through strategic investment in expertise, careful dataset curation, and thoughtful resource allocation, and by focusing on continuous innovation, the integration of diverse approaches, and addressing ethical concerns, chatbots can be refined to serve users more effectively, contributing to better user experiences across various domains. In conclusion, this project has enriched our understanding of chatbot technology, providing insights into the available open-source options and their potential applications. Despite some limitations in implementation, the ease of working with these intuitive frameworks was evident. As chatbots continue to evolve, exploration of novel techniques and improvements to existing platforms remain critical. Open-source chatbot frameworks, libraries and language models, as demonstrated, offer compelling options for businesses and organisations seeking to leverage conversational agents technology for customer support and many potential other use cases that require further investigation.

8 References

[1] C. BasuMallick, "What Is a Chatbot? Meaning, Working, Types, and Examples," in Spiceworks, June. 17, 2022. [Online]. Available: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-chatbot/. [Accessed: May 23, 2023].

[2] V. Bhargava, "Ecommerce Chatbots: Everything You Need to Know About Using Chatbots to Grow Your Online Business," in Shopify, May 12, 2022. [Online]. Available: https://www.shopify.com/in/blog/everything-you-need-to-know-about-ecommerce-chatbots/. [Accessed: May 23, 2023].

[3] A. Payani, "Council Post: Embracing The Future: How AI Is Revolutionizing Marketing And Sales," in Forbes. [Online]. Available: https://www.forbes.com/sites/forbesbusinesscouncil/2023/03/08/embracing-the-future-how-ai-is-revolutionizing-marketing-and-sales/. [Accessed: May 23, 2023].

[4] F. Karabiber, “Jaccard Similarity,” learndatasci.com. [Online]. Available: https://www.learndatasci.com/glossary/jaccard-similarity/ [Accessed: June 10, 2023].

[5] B. Marie, “BLEU: A Misunderstood Metric from Another Age,” Medium, Nov. 05, 2022. [Online]. Available: https://towardsdatascience.com/bleu-a-misunderstood-metric-from-another-age-d44e18f1b37 [Accessed May 23, 2023].

[6] K. Ganesan, “An intro to ROUGE, and how to use it to evaluate summaries,” freeCodeCamp.org, Jan. 26, 2017. https://www.freecodecamp.org/news/what-is-rouge-and-how-it-works-for-evaluation-of-summaries-e059fb8ac840/ [Accessed June 10, 2023]

[7] I. Pavel, "Comparing Chatbot Frameworks: A Study of Rasa and Botkit," 2021. [Online]. Available: https://trepo.tuni.fi/bitstream/handle/10024/132928/PavelImran.pdf. [Accessed: May 23, 2023].

[8] J. Pereira and Ó. Díaz, "Using Health Chatbots for Behavior Change: A Mapping Study," J. Med. Syst., vol. 43, p. 135, 2019.

[9] A. M. Turing, "Computing Machinery and Intelligence," Mind, vol. LIX, no. 236, pp. 433-460, Oct. 1950.

[10] J. Weizenbaum, Computer Power and Human Reason: From Judgment to Calculation. San Francisco, CA: W.H. Freeman, 1976, pp. 2-8.

[11] Botwiki. (n.d.). Eliza. Retrieved June 11, 2023, from https://botwiki.org/bot/eliza/

[12] "From ELIZA to ChatGPT: The evolution of chatbots technology," technologymagazine.com, Dec. 07, 2022. [Online]. Available: https://technologymagazine.com/articles/from-eliza-to-chatgpt-the-evolution-of-chatbots-technology. [Accessed: May 23, 2023].

[13] A. Vaswani et al., "Attention Is All You Need," 2017.

[14]"Rasa Open Source," GitHub, May 11, 2022. [Online]. Available: https://github.com/RasaHQ/rasa [Accessed: May 23, 2023].

[15] "About ChatterBot — ChatterBot 1.0.8 documentation," chatterbot.readthedocs.io. [Online]. Available: https://chatterbot.readthedocs.io/en/stable/# [Accessed: May 23, 2023].

[16] A. Rehan, "10 Best Chatbot Development Frameworks to Build Powerful Bots," Geekflare, Oct. 03, 2018. [Online]. Available: https://geekflare.com/chatbot-development-frameworks/. [Accessed: June 10, 2023]

[17] "howdyai/botkit," GitHub, May 23, 2023. [Online]. Available: https://github.com/howdyai/botkit. [Accessed: May 23, 2023].

[18] G. Cox, "gunthercox/ChatterBot," GitHub, Dec. 2019. [Online]. Available: https://github.com/gunthercox/ChatterBot. [Accessed: June 10, 2023]

[19]Rasa, "Rasa Architecture Overview," rasa.com. [Online]. Available: https://rasa.com/docs/rasa/arch-overview/. [Accessed: June 10, 2023]

[20] OpenAI, "GPT-4," openai.com, 2023. [Online]. Available: https://openai.com/product/gpt-4 [Accessed: May 23, 2023].

[21] "A State-of-the-Art Large-scale Pretrained Response Generation Model (DialoGPT)," GitHub, Mar. 14, 2022. [Online]. Available: https://github.com/microsoft/DialoGPT [Accessed: May 23, 2023].

[22] Y. Zhang et al., "DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation," www.microsoft.com, Nov. 01, 2019. [Online]. Available: https://www.microsoft.com/en-us/research/publication/dialogpt-large-scale-generative-pre-training-for-conversational-response-generation/. [Accessed: May 23, 2023].

[23] "NaturalNode/natural," GitHub, Apr. 24, 2020. [Online]. Available: https://github.com/NaturalNode/natural. [Accessed: June 10, 2023]

[24] A. Wang, "Finetune GPT2 on Covid-19 English Dataset," GitHub, Jan 12, 2021. [Online]. Available: https://github.com/Azmarie/GPT2-finetune. [Accessed: May 23, 2023].

[25] R. Chaudhari, "An attempt to finetune DialoGPT," GitHub, May 25, 2022. [Online]. Available: https://github.com/rushic24/DialoGPT-Finetune. [Accessed: May 23, 2023].

[26] Ju, Zeqian, et.al, "UCSD-AI4H/COVID-Dialogue," GitHub, May 31, 2021. [Online]. Available: https://github.com/UCSD-AI4H/COVID-Dialogue. [Accessed: May 23, 2023].

[27] Rasa, "Model Configuration," rasa.com, May 23, 2023. [Online]. Available: https://rasa.com/docs/rasa/model-configuration/ [Accessed: May 23, 2023].

[28] WHO, "Question and answers hub," www.who.int. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/question-and-answers-hub [Accessed: June 10, 2023]

[29] A. Wang, "GPT2-finetune," [Online]. Available: https://github.com/Azmarie/GPT2-finetune [Accessed: June 10, 2023]

[30] R. Chaudhari, "DialoGPT-Finetune," [Online]. Available: https://github.com/rushic24/DialoGPT-Finetune [Accessed: June 10, 2023]

[31] R. Chaudhari, "DialoGPT-Covid-Help-Doctor," Hugging Face Model, [Online]. Available: https://huggingface.co/spaces/rushic24/DialoGPT-Covid-Help-Doctor/tree/main [Accessed: June 10, 2023]

[32] C. Umbel, K. Koch, and R. Ellis, "Natural," [Online]. Available: https://github.com/NaturalNode/natural [Accessed: June 10, 2023]

[33] G. Cox, "ChatterBot Training Documentation," 2021. [Online]. Available: https://chatterbot.readthedocs.io/en/stable/training.html [Accessed: June 10, 2023]

[34] chamkank, "Flask-ChatterBot," [Online]. Available: https://github.com/chamkank/flask-chatterbot [Accessed: June 10, 2023]

[35] G. Cox, "Examples — ChatterBot 1.0.8 documentation," 2021. chatterbot.readthedocs.io. [Online]. https://chatterbot.readthedocs.io/en/stable/examples.html [Accessed: May 23, 2023].

[36] Bolagsverket, "Bolagsverket," bolagsverket.se. [Online] Available: https://bolagsverket.se/ [Accessed: June 10, 2023]

[37] Centres for Disease Control and Prevention (CDC), "Coronavirus Disease 2019 (COVID-19) – Symptoms," Centers for Disease Control and Prevention, Mar. 22, 2022. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html [Accessed: June 10, 2023]

[38] Centres for Disease Control and Prevention (CDC), "Travellers' Health | CDC," wwwnc.cdc.gov. [Online]. Available: https://www.cdc.gov/travel/diseases/covid19 [Accessed May 23, 2023].

[39] A. Macpherson, "Do I have COVID, a common cold, or something else? How you can tell the difference between symptoms," Portsmouth CCG, Apr. 27, 2022. [Online]. Available: https://www.portsmouthccg.nhs.uk/do-i-have-covid-a-common-cold-or-something-else-how-you-can-tell-the-difference-between-symptoms [Accessed: May 23, 2023].

Appendix 1

The repository provided below provides the main and supplementary (i.e testing) code for the different frameworks, libraries and models as well as the files obtained from the results:

https://github.com/OGFaeralks/eval-cbot-dp

Appendix 2

Table 3.0: The chosen queries for testing the consistency of the following chatbots: ChatterBot based chatbot, DialoGPT-based chatbot, and a chatbot constructed with Botkit and Natural.

Query	Medical source answer
I have pain and discomfort in my lungs. I don't experience simultaneous on both lungs and it not always at the same position. I don't have high temperature. I sneeze and cough maybe once a day. Do I have corona, should I get tested?	COVID-19 symptoms, which can range from mild to severe, may manifest 2-14 days after virus exposure and include fever, cough, breathing difficulty, fatigue, body aches, loss of taste or smell, and digestive issues. If you exhibit these symptoms, consider getting tested for COVID-19 and follow CDC's isolation guidance. In case of emergency signs like trouble breathing, persistent chest pain, confusion, extreme drowsiness, or discoloration of skin, lips, or nail beds, seek immediate medical attention. [37]
Can I go for Corona virus testing if my nose is blocked and I have traveled from a high risk country?	Consider getting a COVID-19 test if you: Develop COVID-19 symptoms before, during, or after travel, will be traveling to visit someone who is at higher risk of getting very sick from COVID-19, were in a situation with a greater risk of exposure during travel (e.g., in an indoor, crowded space like an airport terminal while not wearing a mask). If you traveled and feel sick, particularly if you have a fever, seek medical attention. [38]
I have a dry cough and sore throat. It's been a week now and the cough seems to be getting worse... No runny nose or fever, sometimes a headache, and no shortness of breath. Should I get tested for COVID-19?	Sore throats, often resulting from viruses or smoking, can be managed at home with remedies like gargling salt water, over-the-counter medications, and maintaining hydration. Conversely, COVID-19, characterised by varied symptoms, demands home care such as rest and hydration, but requires testing or immediate medical attention for breathing difficulties. [39]

Appendix 3

The 9 queries that were used for testing consistency of the chatbots evaluated.

1a) "I'm experiencing lung pain and discomfort, but it's not constant or in the same place. I don't have a fever, and I only cough or sneeze once a day. Should I get tested for COVID-19?"
1b) "My lungs feel uncomfortable and painful, but the location keeps changing and it's not in both lungs at the same time. I have no fever, and I sneeze and cough infrequently. Do I need a coronavirus test?"
1c) "I have some lung discomfort and pain that moves around and isn't simultaneous in both lungs. I don't have a high temperature, and I only cough and sneeze occasionally. Is it necessary for me to get tested for COVID-19?"

2a) "Should I get tested for COVID-19 if I have a blocked nose and recently returned from a high-risk country?"
2b) "I just came back from a country with a high COVID-19 risk, and my nose is congested. Do I need a coronavirus test?"
2c) "If I have nasal congestion and have traveled from a high-risk area, should I be tested for COVID-19?"

3a) "I've had a dry cough and a sore throat for a week, which seems to be worsening. I don't have a runny nose or fever but sometimes experience headaches. Do I need a COVID-19 test?"
3b) "My dry cough and sore throat have been persisting for a week and are getting worse. There's no fever or runny nose, but I occasionally have headaches. Should I get tested for COVID-19?"
3c) "I've been dealing with a worsening dry cough and sore throat for the past week. I don't have a fever or runny nose, but I do get headaches from time to time. Is it necessary to test for COVID-19?"