ICSI499 Capstone Project Report

NLP Customer Service Chatbot for Synchronization Software in the Fitness Industry

Project Team 05

Toussaint Turnier (001311187)
Tianna Brown (001363109)
Hannah Kennedy (001325544)
Joseph Minozzi (000947482)
....
College of Engineering and Applied Sciences
University at Albany, SUNY

Project Sponsor

Jennifer Strout
MotivateU
Middletown, Delaware

10-12-2021

Acknowledgements

For our acknowledgements, we would like to thank our sponsor Jennifer Strout for the collaborative effort in the design of the Chatbot project. She gave incredibly detailed and helpful feedback that helped the project succeed, she pushed our team to always go past our expectations and aim high when it comes to setting and accomplishing goals.

Abstract

The development of a Natural Language Processing (NLP) Chatbot will be for the benefit of our sponsor MotivateU to achieve an efficient scalable form of customer service for their respective company. The following sections will demonstrate an overview of this particular NLP Chatbot's overall design. Categorizing each intended user that is being represented within the software and outlining the interactions that correlate by the illustration of a case diagram. Subsequently the paper will be discussing the utilized algorithms aided by the systems and data flow diagrams that will visualize the importance and significance of each decision that was made. Lastly the interpretations our team has envisioned for the interfaces of the NLP Chatbot will be showcased at the end of this report.

Contents

1. Introduction

The expansion of the business industry has grown at such an exponential rate that companies must adapt many of its practices to the modern age or else risk being left behind with the times. This growth has not only remodeled the way business is done today, but also what the modern customer demands. As the number of customers begin to scale to larger and larger quantities, it can be quite difficult to adequately match the amount of customer service that is necessary to meet the criteria for a consumer's experience to be that of a positive one. [8]

A customer is a business's most valuable asset, without them there is no business. That is why it is important to ensure customer satisfaction is an upmost priority for the company's values. However, this cannot be achieved purely by having the best product the market can offer.

To maximize a customer's satisfaction, businesses have to ensure that they exemplify an exceptional customer service system. Whether it be before or after a transaction has occurred, the goal for the company should be keeping the relationship between the two parties to be positive. This meeting place for a company and a consumer to interact with one another has been a constantly evolving endeavor as the world develops further alongside technology.

The existing solutions of today include in-person conversations, to consulting with the customers over the phone, email, and instant messaging. However, these all come with their own flaws. In person customer service is highly unrealistic as it does not scale with the business properly and is costly. Over the phone service can be troublesome as this is costly to employ numerous customer service representatives, likely could not be 24/7 without outsourcing and if it was consumers generally do like outsourced customer service. Coupled with usually long wait times, and it is time consuming to get through the call menus. Email can take a long time to get a reply back from not just the company but also the consumer, elongating the problem that is at hand. Instant messaging is troubled with the scale of the company, as the more the company and its consumer based grows the more customer service reps have to be employed, thus making the service costly.

Customer service must reach its next step in modernization to compete in today's corporate climate. That next step being the natural language processing chatbot, today's customer service's generational entry. It benefits similarly align with a regular instant messaging system but with the added benefit of reducing cost. A summarized list of what an NLP chatbot can bring to the industry are as follows:

- Resolves issues quickly
- Scalable with the size of the company.
- Automated instant messaging customer service tool.
- A guaranteed form of contact to the company.
- Instantaneous response. Accessible to those who have access to a browser with internet.
- 24/7 Availability

The architecture of our project has a Back-end is programmed in Python, utilizing Keras Tensorflow for our machine learning library. Our Front-end is based on JavaScript, CSS, and HTML. The tech stack is connected by Flask.

2. Background and Related Work

Before the internet had grown to become part of our daily lives in society, we had to conduct business in person. People had to travel to each other to sell, buy, and gather information. Nevertheless, as technology grew, so did the way they did business. Mailing systems reduced the need to travel as much. It allowed consumers to send a letter now if something went rye. This service, however, would cost the consumer money and resources. Not only that, but the problem itself would take a long time to be resolved due to the limitation of it being physical mail. Once the telephone became widespread, direct contact with companies was now possible with the dial of just a number. Soon, the holes in its usability began to show as the problem with customer service over the phone was its lack of scalability. The more customers the company had, the more callers called in, meaning the company would need more employees, equipment, and a call-center infrastructure. On the customer's end, inconveniences such as long queue times, inadequate speakers, and complicated menus would create a disdain for the company. The email helped off-load some work from the call centers, but that still required someone to read the email and respond accordingly. Email is inherently a prolonged method of communication; in scenarios where the first message the consumer sends is not understandable, it will take a while for the answer to be found amongst each other. The two must write emails back and forth explaining an unknown problem. Instant chat messaging was the next step; however, this still does not scale. The bigger the consumer base is, the more employees that are needed. It has come full circle at this point in customer service development. We have seen many solutions, but the same problem always comes up, which is scalability. [5]

Now, this brings us to the chatbot. Chatbot's in their current state have many flaws. These bots are designed only to recognize conversational rules set by the creator and only that. Suppose the input is outside the parameters of the database, then the chatbot will have a difficult time giving the necessary information to the consumer. These limitations make the chatbot feel robotic and incapable of relating to the user.

Our project's goal was to create an NLP (Natural language Processing) Chatbot. This chatbot version is strictly an improvement from a regular chatbot as an NLP Chatbot can adapt to conversational cues. By continuously improving itself to understand the end-user. The NLP Chatbot can understand typos, grammatical errors, and the user's intentions. These tools allow the NLP Chatbot to be an adequate stand-in for business agents. Thus, the consumer's time and the business money are both saved. Our project's NLP chatbot has a diverse use case developed for our sponsor's utilization. Because it is a company based in the fitness industry, we personalized our application for this role. [3] These curated responses will make customers feel comfortable and satisfied with the company. It is essential that customers' time is prioritized and that they are being treated well. Happy customers are essential for a company's success, so the investment MotivateU is making into creating an NLP Chatbot will undoubtedly contribute to their success.

Table 2.1: Solutions

Solutions	Strengths	Weaknesses
In Person	Most favorable form of service	Not scalable
Mail	Guaranteed form of contact to the company	Unreasonably long contact time
Telephone	Resolve issues quickly.	Does not Scale well
Email	Direct contact to company	Long wait times between contact
Instant messaging	Resolves issue quickly	Does not scale well
Chatbot	Scalable	Impersonal, Limited datasets
NLP Chatbot	Scalable, Specialized	Can make mistakes

3. Proposed System/Application

3.1 An Overview

Our customer service chatbot for MotivateU is a Natural Language Processing chatbot which utilizes Python's machine learning package TensorFlow Keras to process queries, HTML and CSS to create a pleasant user interface, Javascript to make the chatbot interactive, and Flask as a framework.

The chatbot has a simple use case. Users may query the chatbot to receive a response that is processed by the NLP technology. The proposed system is such that a user can begin a session, the chatbot will send a welcome message, and then the user can query the chatbot. The chatbot will then send this query to the Natural Language Processing which then will be handled by Data Services. When the chatbot gets the appropriate response from Data Services, it is sent back to the user. The process repeats with the client sending another query or ending the session.

In the use case diagram (Figure 3.1), the session begins, which includes a greeting from the system. Additionally, you can see the ability for users and the system to send messages and receive messages. Lastly, the user may close the window to end a session.

In the data flow diagram (Figure 3.2), data flows from the chat client, where it is sent to the chatbot and then processed by the Natural Language Processing. Next, the processed intent is responded to by the Data Service and the response is gathered to be sent to the chat client. It is sent through the chatbot interface and received by the client.

Some preliminaries are the frameworks and packages that we used. We also used several tutorials to shape our chatbot. Tianna and Joe contributed significantly to the design and implementation of the front and back end of the chatbot, whereas Toussaint and Hannah worked heavily on integration and the generation of the novel intents files.

Some further assumptions are that we will have a working server to host the chatbot on. Without a working server, the chatbot cannot operate. Additionally, we are assuming that the users will ask questions that are topically similar to the queries we wrote in our intents file.

We collaborated with our sponsor to come up with relevant queries for our chatbot. We also used MotivateU's software guidelines. We chose flask for our framework, which seemed most straight forward and succeeded in binding front end and back end elements together.

3.2 System Requirements

3.2.1 User Class

- Instructors - Individuals that teach or run classes. They use MotivateU's tracking software to train clients.
- Owners - The administrators of a given gym.

3.2.2 Functional Requirements

Instructors and Owners

- The user can begin a session with the MotivateU chatbot.
- The user can send queries to the chatbot.
- The user can receive responses from the chatbot.
- The user can close the chatbot window to end a session.

System

- The system can greet a user with a welcome message.
- The system can send a response to the user.
- The system can send a message indicating that the user will be escalated to customer support.

3.2.3 Non-Functional Requirements

Accessibility

- The chatbot's UI design must be reasonably large such that a user of varying technical ability can access it, send, and receive queries.

Reliability

- The chatbot's front end, back end and framework are modular, therefore any debugging can be easily handled without damaging other parts of the system.

Availability

- The chatbot must always be available if the host site is running.
- The system shall be available for all modern web browsers, both on desktops and mobile devices.

Scalability

- Due to the intents.json file, the amount of questions that can be added to the chatbot is extremely scalable, where each query is its own pattern and patterns can be added without much cost to the system.

Accuracy

- Model shall have a degree of accuracy of approximately 80 percent.

Performance

- There should be a response time of less than a second.
- The user must understand the chatbot's responses.

3.2.4 Operating Requirements

Operating Systems

- Supported Operating Systems: Windows, macOS, Linux, iOS, Android, and Chrome OS.
- Supported Browsers: Google Chrome, Mozilla Firefox, Apple Safari, and Microsoft Edge.

3.2.5 Design and Implementation Constraints

Interpretation

- Although machine learning can synthesize a human response, it can never replace it. Ultimately, a chatbot may be less helpful than personal service with a customer representative.

Server

- The chatbot can only operate as long as there is a connection to the server. Similarly, if the website is experiencing any issues, this can affect the performance of the chatbot.

3.3 Technical Design

3.3.1 Use Case Diagram

Our use case diagram represents the relationship between the user of the chatbot and the system. (Figure 3.1) The users and system work in together to fill the needs of the user. The user may enter a query. Then our chatbot will process the query and respond given the program. The response can be another query, or a request to escalate the user to a customer service representative. The purpose of these use cases are to assist the user with scalable, timely service.

Begin Session

- The user must open the chatbot, which triggers the chatbot to begin a session.

Greet User

- The chatbot must greet the user at the beginning of the session. It must also present the user with several queries the answer can enter to prompt them. This will lend to a more user-friendly, personable experience.

Send Query/Receive Query

- The user must be able to send a query to the chatbot.
- The chatbot system must receive the query and process it.

Send Response/Receive Response

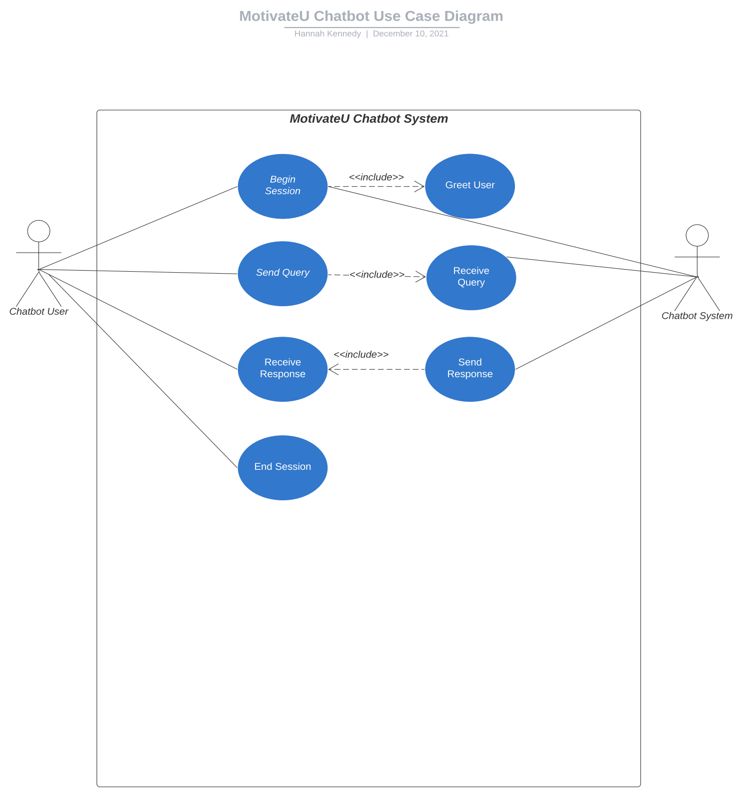

Figure 3.1: Use Case Diagram

- Upon receiving its query, the chatbot must respond to the query correctly. Natural Language Processing must process the user's query and give the user the appropriate response. This can be an answer to a question, a message notifying the user that their query will be escalated to customer service, or a message that the chat will be ended.
- The user must receive these possible responses.

End Session

- The user must be able to end the session. Once the user enters a query that confirms they want to end a session, the chatbot will respond by saying goodbye and requesting that the user close the pop up.

3.3.2 Users

Owners and Instructors

- The user can begin a session with the MotivateU chatbot.
- The user can send queries to the chatbot.
- The user can receive responses from the chatbot.
- The user can close the chatbot window to end a session.

System

- The system can greet a user with a welcome message.
- The system can send a response to the user.
- The system can send a message indicating that the user will be escalated to customer support.

3.3.3 System/Data Flow Diagram

The system and data flow diagram will show the main components of the system and the application, data flow, and user interaction points. (figure 3.2)

Chat Client

- The Chat Client is the main front user facing interface of the Chatbot and is split into two main sections. The first, top section, is where incoming and outgoing messages are arranged sequentially in the form of a conversation. The second, bottom section, is a text box where the user can input a phrase to send to the Chatbot.

User Input

- The Chat Client is to receive user input in the form of a phrase. This user input phrase is then sent along to our Chatbot for further processing.

Chatbot

- The Chatbot is the main point of contact for all the services that are performed and interfaced with. The user input from the Chat Client is sent to the Chatbot. The Chatbot then sends the phrase to our machine learning Natural Language Processing engine, TensorFlow Keras. When the Chatbot receives the intent from the analyzed NLP, it determines whether it can respond to the user directly or if there needs to be an additional service called, such as data services to provide an answer.

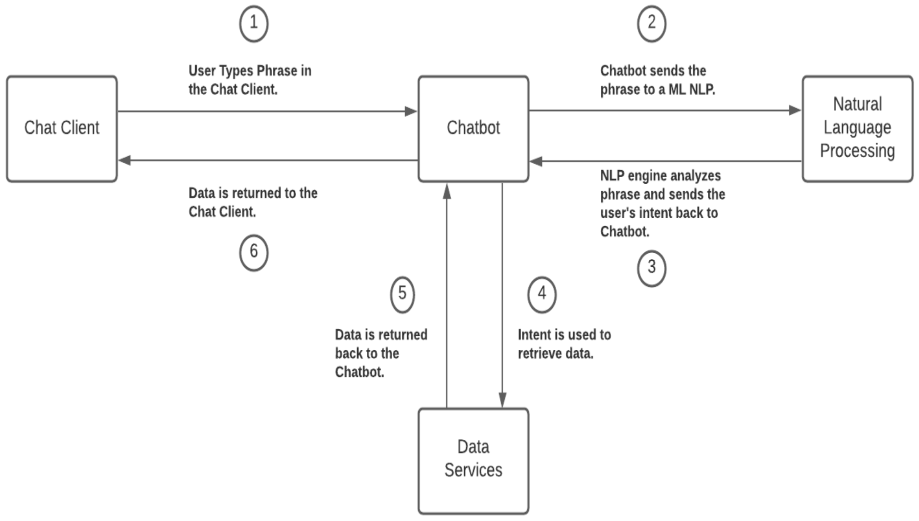
1
User Types Phrase in the Chat Client.
2
Chatbot sends the phrase to a ML NLP.
Chat Client
Chatbot
Natural Language Processing
NLP engine analyzes phrase and sends the user's intent back to Chatbot.
3
Data is returned to the Chat Client.
6
5
Data is returned back to the Chatbot.
4
Intent is used to retrieve data.
Data Services

Figure 3.2: System Data Flow Diagram

Natural Language Processing

- Natural Language Processing is a core function of the Chatbot system and application. Users will be able to input requests in natural language and expect a correct response back. The goal of NLP and this Chatbot is to process user queries and respond fast and adequately.

Data Services

- Data Services in this model are all services relating to the retrieval and storage of data. Information, such as raw data, needs to be in a format suitable for TensorFlow Keras and to work with our deep learning model. This processing will happen here.

3.4 System Implementation

To implement the NLP Chatbot, we used Python 3 as our main language, primarily with the TensorFlow 2.0 open-source library for machine learning and artificial intelligence. To handle the development of our deep learning model, we used the high-level Keras API that works on top of TensorFlow and allows us to use a sequential neural network model. JavaScript, HTML, and CSS are what we used for the graphical user interface that the user interacts with and displays the outgoing and incoming messages. To bring it all together, we use the Flask web framework, written in Python, that enables us to use REST web API request dispatching.

This allows us to send information from the frontend GUI to the backend to be processed. Then the processed response is sent back to the frontend GUI to be displayed to the user.

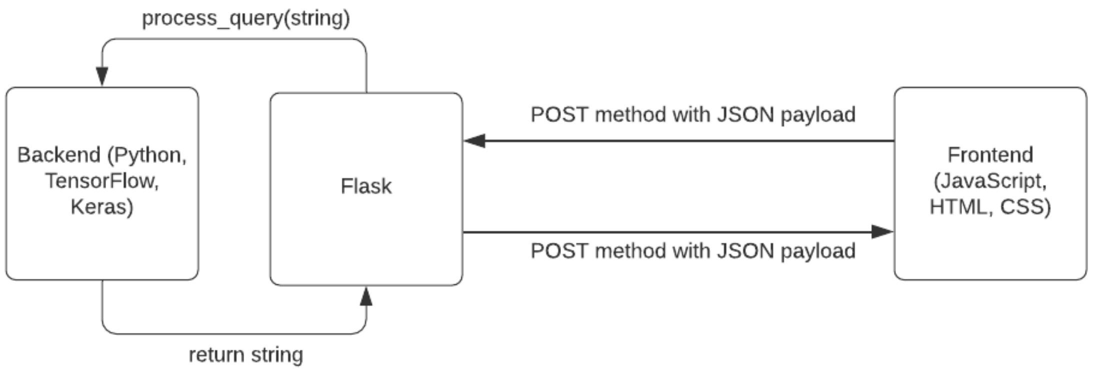
process_query(string)
Backend (Python, TensorFlow, Keras)
Flask
POST method with JSON payload
Frontend (JavaScript, HTML, CSS)
POST method with JSON payload
return string

Figure 3.3: System Implementation Flow

For our programming tools we used the PyCharm IDE with the Conda package manager provided by Anaconda. GitLab was used for version control and to share code with the underlying Git software.

3.5 Use of Computer Science Theory and Software Development Fundamentals

One instance of computer science theory that we implemented was the use of the Adam optimization algorithm for natural language processing. It is used to update neural network weights in deep learning. It is a relatively recent algorithm that was first published in 2014, but has roots from other, older, stochastic approximation method algorithms. They include the Robbins-Monro algorithm from 1951 and the Kiefer-Wolfowitz algorithm from 1952. [2] The benefits of the Adam optimization algorithm are that it is effective, especially considering how computationally efficient it is with low memory requirements, and that it is simple to implement given Keras's API. [7]

Another application of computer science theory that we used was tokenization. In order to process the language strings, we had to first tokenize them to convert them to numeric values for our model. Tokenization also helped accuracy and natural language comprehension by being the precursor to stemming. Stemming is a normalization technique that accounts for different inflection suffixes. For example, take the word "reconstruct". It could be inputted as "reconstructs", "reconstructed", "reconstructing", etc. We want to consider the overall meaning of the word, not specifically the tense. [6]

Another application of computer science theory that we used was the use of the array data structure and padding. Neural networks require the data to be the same size. [4] However, the string inputs that we are using of natural language will all be different sizes in word length. In order to get it into a format acceptable for processing in our model, we have to "pad" the data and put it into a 2D array data structure. In our model we set the max length of a sentence to be 20 words. If a sentence is greater than 20 words, it will drop words off the end. If a sentence is fewer than 20 words, we used Keras's pad_sequences() function to fill in the empty word spaces in our array with 0s.

One primary software development fundamental that we used to structure the development of our application was the use of separation of concerns. This is a design pattern and form of abstraction where we separated the development into a presentation layer, the user facing frontend, and the data access layer, the "logic" and "data" backend. This allowed the processing and logic of our code to be modular and reusable where it can be integrated into different applications with minimal effort. This also greatly allowed for the separation of tasks amongst the team. [9]

Another software development fundamental we utilized was the use of code documentation. Given the complex nature of machine learning, it was imperative we had well documented code. We were dealing with many machine learning libraries, which had their functionality abstracted in order to be used efficiently. Understanding all of the functions we imported from each library and being able to convey their usage was important in our development. Being descriptive and stating the purpose of every function created coherence in our code.

Another software development fundamental we used was the facade software design pattern. We did this by having different layers of our application interface with each other. For example, the portion of our application that the user interface interacts with imports functionality, such as our process_query(string) function, from a subclass. This allows for a simple interface to work with on the higher layer, while still providing the complex logic. [1]

4. Experimental Design and Testing

4.1 Experimental Setup

Experiment Objectives

In order to ensure that we were solving the stakeholder's problem we designed two experiments: One for accuracy and one for precision. An NLP chatbot that lacks the ability to respond correctly could lead to a number of issues for the company. These issues include: Consumer frustration due to unresolved queries, loss of customers, loss of revenue, reputation damage, or more detrimental setbacks. Consumer's want access to quick and accurate information, and a goal of ours was to provide that. In terms of precision, it's important to observe how an NLP chatbot responds if the same question was asked multiple times. Though we are testing for different things within these two experiments, precision and accuracy go hand and hand.

Experimental Setup

The experimental setup is on a laptop. The IDE being used is PyCharm. The coding language is Python 3.8, and which implements TensorFlow 2.0. A .json file contains all the intents. The queries and intents are written by group members.

Dataset

- The first dataset will be queries that are already in the intents file. The first dataset will be 29 queries.
- The second dataset will be queries that are not in the intents file but are similar to questions in the intents file. For example, the query "What day is it?" and "What day is today?" are functionally similar, but syntactically dissimilar. The second dataset will be 135 queries.
- The third dataset will be queries that are in the intents file, but with typos. A "typo" is spelling error. The typos will be defined by quantity of typos in the query and percentage error of typos, where percentage error is (typos in query $\div$ characters in query) x 100. For example, the query "Whot day ip it?" will be tested against the query "What day is it?". The total number of characters in the query is 14 and the quantity of typos is 2. Therefore, the percentage error is (2$\div$14) x 100. The total number of characters in a query includes spaces and does not include punctuation.The third dataset will be 29 queries.
- The fourth dataset is meant to test precision. It will be queries that are in datasets one, two and three. The fourth dataset will be a subset of three queries from each of the first three datasets, totaling nine queries.

The collection method for the datasets are as follows:

- A group member determined a list of intents for the chatbot by referring to the Sponsor and the Resources Page on the MotivateU website. It was written into an Excel Sheet file and distributed among group members.
  - Each group members reworded the each intent 4-5 times, without seeing other teammates' intents. This was to prevent corrupting the data by influencing the wording of the intents.
- Three teammates' lists of intents were added into the intents file.
- One teammate's intents were used as a list of queries.
- Datasets one and three are derived from the intents file as they are either an exact match and an exact match with typos, respectively.
- Dataset two is derived from the list of queries written by a group member.

- Dataset four is derived from dataset one, two and three, where dataset four has three queries from each of those datasets, totaling nine queries.

Experiments

Experiments 1-3 measure accuracy. We recorded the user query, the NLP Chatbot expected response, the NLP Chatbot's actual response, and whether the actual was correct or incorrect. For experiment 4, which measures precision, we recorded the user query and if the response was correct or incorrect.

Experiment 1:
First dataset: queries that are already in the intents file. The NLP chatbot is to process the queries from the dataset, assign the queries to an intent, and return a response defined by the intents file. The expected result is for the NLP Chatbot to respond to the query as defined in the intents file with 100% accuracy and 0% error.

Experiment 2:
Second dataset: queries that are similar (but not exact to) to queries in the intents file. The NLP chatbot is to process the queries from the dataset, assign the queries to an intent and return a response defined by the intents file. The expected result is for the NLP Chatbot to respond to the similar query as in the intents file with 100% accuracy and 0% error.

Experiment 3:
Third dataset: queries that are already in the intents file, but with typos. The NLP chatbot is to process the queries from the dataset, assign the queries to an intent, and return a response defined by the intents file. The expected result is for the NLP Chatbot to respond to the query (with typos) as defined in the intents file with 100% accuracy and 0% error.

Experiment 4:
Fourth dataset, which is a subset of nine queries from datasets one, two and three. The process to record precision will be different to the process to record accuracy. While the queries are still to be entered and processed, instead of entering the query once to compare the expected response to the actual response, each query will be entered ten times. For each trial, the actual response is to be be recorded. From those results, we can discern the precision. The expected result is for the NLP Chatbot to respond to the query with the same response for every trial. This would mean 100% precision and 0% error.

4.2 Results and Analysis

The tables provide information about each trial. Each one differs in the type of query submitted or the specific metric being measured.
In Table 4.1, we will be testing accuracy by testing queries with ones that exist verbatim in our training intents file.

Table 4.1: Experiment 1 (Accuracy)

Trial	Type of Query	User Query	Expected Response	Output
1	Exact Match	There is an emergency.	If there is an emergency...	
2	Exact Match	How do I name a class?	Click on the "Classes" tab...	
...	Exact Match	...	...	
29	Exact Match	My class won't save.	Click on the "Classes" tab	

In Table 4.2, we will be testing queries that are similar to our training intents file.

Table 4.2: Experiment 2 (Accuracy)

Trial	Type of Query	User Query	Expected Response	Output
1	Fuzzy Match	I may have an emergency.	If there is an emergency...	
2	Fuzzy Match	How do I title a class?	Click on the "Classes" tab...	
...	Fuzzy Match	...	...	
120	Fuzzy Match	My class isn't saving.	Click on the "Classes" tab	

In Table 4.3, we will be testing queries verbatim but with typos to our training intents file.

Table 4.3: Experiment 3 (Accuracy)

Trial	Type of Query	User Query	Expected Response	Output
1	Typing Error	There is amn emergency.	If there is an emergency...	
2	Typing Error	How do I vieqw my schedule?	Click on the "Schedule" tab...	
3	Typing Error	How do I name a clasj?	Click on the "Classes" tab...	
...	Typing Error	...	...	
29	Typing Error	My clpss won't save.	Click on the "Classes" tab	

In Figure 4.4, we will be testing queries from all datasets for precision. The data from these tables will allow for further analysis to guide in improving the chatbot.

Table 4.4: Experiment 4 (Precision)

Trial	Type of Query	User Query	Output
1	Exact Match	I may have an emergency.	If there is an emergency...
...	Exact Match	...	...
31	Fuzzy Match	My class isn't saving.	Click on the "Classes" tab...
...	Fuzzy Match	...	...
61	Typing Error	How do I see my schedule?	Click on the "Schedule" tab...
...	Typing Error	...	...
90	Typing Error	My class isn't saving.	Click on the "Classes" tab

Our NLP Chatbot application was designed with natural language analysis and typographical errors in mind. We used data from the actual usage of the application and our own experiments to substantiate our initial claims. This allowed us to improve our application and be competitive with competitors. Because our chatbot is being designed for a specific company, purpose, and industry, it is a bespoke solution and should be better than existing systems for its purpose.

What is expected from these results is a varying form of accuracy depending on the test conditions as well as near perfect precision. In the first experiment, we expect the responses to be completely accurate since they are exactly the phrases our Machine Learning model was generated from. For the second experiment, we expect fewer accurate responses. However, we expect most responses to be accurate because similar keywords will be shared. For the third experiment, we expect fewer accurate responses because of typographical errors on the main keywords. For our fourth experiment, we expect 100% precision regardless of the type of error in the query. This is because our chatbot is designed to return the same result if given the same query.

An unexpected finding from the experiments would be if the responses given are not relevant to the input query at all. For example, we have many entries that share similar keywords and phrases. We would expect a response that shares these similar keywords and phrases to be given to a query that may be vague about its intention. However, we do not expect a completely unrelated response.

4.3 Limitations

- A limitation of our NLP Chatbot is when a long intention is used as input. A long conditional statement has a lower accuracy rating; Short responses garner high accuracy ratings and high precision ratings.

- Another limitation is the NLP Chatbot's inability to decipher other languages. Our sponsor didn't ask for a multi-lingual chatbot however since the chatbot is for English native speakers, but it was interesting to see.

5. Ethics and Legal Practices

There are no legal concerns and unethical practices performed during the creation of the NLP Chatbot. There is no personal or private data from users used for the machine learning in the application, and users will not have to be asked for consent as there is no identifiable data. No boundaries are broken within the realm of intellectual property.

6. Effort Sharing

Our team worked vigorously to ensure a quality product for MotivateU. Here are each of our member's individual contributions (as shown in Table 6.1).

Toussaint Turnier:

- Worked on the functional Front-end using JavaScript
- Collaborated with the team on the intents.JSON file
- Integrated the Front-End and Back-end with Flask

Tianna Brown:

- Created GUI Sketches
- Worked on the Intents file
- Worked on the Front-End portion of the chatbot (HTML and CSS)

Hannah Kennedy:

- Wrote and compiled queries for the intents file
- Provided resources for front end integration
- Integrated front end so that it would work on a server

Joseph Minozzi:

- Worked on Machine Learning model in Python
- Worked on backend query processing and intent recognition
- Worked on the JavaScript and REST web API constraints for frontend and backend interfacing

Table 6.1: Effort sharing

Team size	Joint efforts	J. Minozzi	T. Turnier	T. Brown	H. Kennedy
4	J (30%)	I (17.5%)	I (17.5%)	I (17.5%)	I (17.5%)

J = description of tasks jointly performed

I = description of tasks individually performed

7. Conclusion and Future Work

We hope to have this chatbot integrated into MotivateU's server so that they can host it and receive messages from clients.

Going forward, there is much room for further analysis and improvement. One could add a loading throbber. One could also further expand the queries as more and more MotivateU clients use the chatbot, and ask questions we did not anticipate. Accuracy could be improved even further with additional testing and adjustments.

However, our application of machine learning to create a NLP chatbot was a robust solution to the issue of scalable customer service.

Bibliography

[1] Design patterns and refactoring.

[2] S. Andradottir. A new algorithm for stochastic optimization. In *1990 Winter Simulation Conference Proceedings*, pages 364–366, 1990.

[3] Botpress. Different types of chatbots: Rule-based vs. nlp, 2021. Last accessed 10 December 2021.

[4] Gerry Chng. Using the right dimensions for your neural network, Jul 2020.

[5] Sophie Flechas. How customer service has changed over the last 200 years, 2021. Last accessed 21 December 2021.

[6] Kerem Kargın. Nlp: Tokenization, stemming, lemmatization and part of speech tagging, Feb 2021.

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[8] Blake Morgan. The evolution of customer service, 2016. Last accessed 10 December 2021.

[9] Guy Rombaut. From monolith to separation of concerns: Software architecture teams, May 2018.