A PROJECT REPORT

ON

Medic Hero 4

For the partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Abhishek Kumar (1619210015)

Dheeraj Singh (1619210084)

Nishant Rana (1619210161)

Yash Dwivedi (1619210307)

Under the Supervision of

Dr. Sanjeev Kumar Pippal

G.L. BAJAJ INSTITUTE OF TECHNOLOGY & MANAGEMENT, GREATER NOIDA

Affiliated to

DR. APJ ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW

2020

Declaration

We hereby declare that the project work presented in this report entitled **“ Medic Hero 4 ” ,** in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science & Engineering, submitted to A.P.J. Abdul Kalam Technical University, Lucknow, is based on our own work carried out at Department of Computer Science & Engineering at G.L. Bajaj Institute of Technology & Management, Greater Noida. The work contained in this report is original project work submitted by us for award of our Bachelor’s degree .

Signature:
Name: Abhishek Kumar

Roll No: 1619210015

Signature:
Name: Dheeraj Singh

Roll No: 1619210084

Signature:
Name: Nishant Rana

Roll No:1619210161

Signature:
Name: Yash Dwivedi

Roll No: 1619210307

Date: 22 Feb, 2020
Place: Greater Noida

Certificate

This is to certify that the Project report **entitled "Medic Hero 4" done by Abhishek kumar (1619210015), Dheeraj Singh (1619210084), Nishant Rana (1619210161) and Yash Dwivedi (1619210307)** is an original work carried out by them in Department of Computer Science & Engineering, G.L. Bajaj Institute of Technology & Management, Greater Noida under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Date: 22 Feb , 2020.

Dr. Sanjeev Kumar Pippal	**Dr. Sanjeev Kumar Pippal**
Signature of the Supervisor	**Head of the Department**

Acknowledgement

The merciful guidance bestowed to us by the almighty made us stick out this project to a successful end. We humbly pray with sincere heart for his guidance to continue forever.

We pay thanks to our project guide . **Dr. Sanjeev Kumar Pippal** who has given guidance and light to us during this project. His/her versatile knowledge has cased us in the critical times during the span of this project.

We pay special thanks to Dr. Sanjeev Kumar Pippal as Head of Department who has been always present as a support and help us in all possible way during this project.

We also take this opportunity to express our gratitude to all those people who have been directly and indirectly with us during the completion of the project.

We want to thanks our friends who have always encouraged us during this project.

At the last but not least thanks to all the faculty of CSE department who provided valuable suggestions during the period of project.

Abstract

The Medic Hero 4 is AI based web chatbot which helps people by providing assistance in medical cosultancy , searching nearby hospitals , doctors according to user’s concern and providing quick home remedies and predictions.

Our Medical Chatbot will have a great impact on the life of its user s. it provide them of having an virtual doctors on theirs fingertips with 24/7 service in accordance with real doctor’s advice.

TABLE OF CONTENT

LIST OF FIGURES

Chapter 1

Introduction

1.1 Problem Definition

Artificial Intelligence is based on how any device perceives its Environment and takes actions based on the perceived data to achieve the result successfully. It is the study of intelligent agents. The term "artificial intelligence" is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving. Artificial Intelligence gives the supreme power to mimic the human way of thinking and behaving to a computer. A chatbot (also known as a talkbot, chatterbot, Bot, IMbot, interactive agent, or Artificial Conversational Entity) is a computer program which conducts a conversation via auditory or textual methods. These programs are designed to provide a clone of how a human will chat and thereby it acts as a conversational partner rather than humans. For various practical purposes like customer service or information acquisition, chatbot is being used in the dialog system. Mostly chatbots uses natural language processing for interpreting the user input and generating the corresponding response but certain simpler systems searches for the keyword within the text and then provides a reply based on the matching keywords or certain pattern. Today, chatbots are part of virtual assistants such as Google Assistant, and are accessed via many organizations' apps, websites, and on instant messaging platforms. Non-assistant applications include chatbots used for entertainment purposes, for research, and social bots which promote a particular product, candidate, or issue. Chatbot’s are such kind of computer programs that interact with users using natural languages. For all kind of chatbots the flow is same, though each chatbot is specific in its own area knowledge that is one input from human is matched against the knowledge base of chatbot. Chatbot's work basically on Artificial intelligence, so using this capability we have decided to add some contribution to the Health Informatics. Various surveys in this area have proved that that chatbot can provide healthcare in low costs and improved treatment if the doctors and the patient keep in touch after their consultation. To answer the questions of the user chatbot is used. There is veryless number of chatbots in medical field.

1.2 Project Overview

The proposed system provides a text-to-text conversational agent that asks the user about their health issue. The user can chat as if chatting with a human. The bot then ask the user a series of questions about their symptoms to diagnose the disease. It gives suggestions about the different symptoms to clarify the disease. Based on the reply from the user the accurate disease is found and it suggests the doctor who needs to be consulted in case of major disease. The system remembers past responses and asks progressively more specific questions in order to obtain a good diagnosis. The three primary components of our system are (1) user validation and extraction of symptoms from the conversation with the user, (2) accurate mapping of extracted symptoms to documented symptoms and their corresponding codes in our database, and (3) developing a personalized diagnosis as well as referring the patient to an appropriate specialist if necessary. There are certain chatbots in the medical field that already exists they are Your.MD, Babylon, and Florence, but current implementations focus on quickly diagnosing patients by identifying symptoms based on pure system initiative questions like natural conversation. Our system focuses solely on the analysis of natural language to extract symptoms, which could make it easier for elderly, less technical users to communicate their symptoms as well as make it relatively straightforward to support spoken language by adding NLG components. In its current form, our bot's best application would be as a preliminary diagnosis tool that patients could use to assess their symptoms before going to the doctor, perhaps using the bot's specialist referral feature to choose the right care provider.

Chapter 2

Literature Survey

2.1 Introduction

The paper gives the information regarding products which is useful for consumers to obtain what they want exactly. Question Answering (QA) systems can be identified as information accessing systems which try to answer to natural language queries by giving answers suitable answers making a use of attribute available in natural language techniques. The system takes a plain text as input and answering all type of questions output by qualified user is the output . The purpose is to provide a generic solution to this problem. this paper helps in recognizing the reality in texts and giving the past content for developing a conversation which is used in middle-school CSCL scenarios. A smart chatbot for customer care by using Software as a Service which analyze message of each application server. It help the user to resolve the issue by providing a human way interactions using LUIS and cognitive services which is implemented on AWS public cloud. Admin feeds input to the machine so that machine can identify the sentences and taking a decision itself as a response to a question. The database used in the project is MySQL. The illustration and execution of SQL in the patternmatching operation is required. The conversation can be done so that it can add some knowledge to the database as it has not been modeled before. If in case the input sentences in the database did not match then it will be remodeled. The evaluation of sentence equivalence is completed with bigram that splits the input sentence in to two parts. The data of chatbot are deposited in the database. The database is appointed as information storage and predictor is used for storing the function and perform pattern matching. This application can be developed by using programming language of Pascal and Java. Paper uses artificial intelligence for predict the diseases based on the symptoms and give the list of available treatments. It can facilitate us to figure out the problem and to validate the solution. Author gives chatterbot which is based on AIML (Artificial Intelligent Markup Language) structure for training the model and uses Microsoft voice synthesizer for identification of the word spoken by the user. Natural language processing used for understanding and Microsoft speech recognition is used in speech recognition and speech synthesis for speech to text and text to speech so people get along with it easily.

2.2 Existing System

Many medical Chatbot designs have been proposed in the past few years which aim to provide the user with medicine recommendation after extracting the illness information from the user messages. A similar paper "Pharmabot: A Pediatric Generic Medicine Consultant Chatbot" proposed by Benilda Eleonor V. Comendador, Bien Michael B. Francisco, Jefferson S. Medenilla, Sharleen Mae T. Nacion, and Timothy Bryle E. Serac provides a design for a stand-alone medical Chatbot that is implemented using MS Access and Visual C#. For using the proposed design, the user has to navigate using the four options provided by the application. This design aims to work by converting the user input to SQL queries and execute it on MS Access to retrieve the solution to the illness. Also a research paper "MedChatBot: An UMLS based Chatbot for Medical Students" proposed by Hameedullah Kazi, B.S. Chowdhry and Zeesha Memon focuses on a design for an AIML based Medical Chatbot. This Chatbot design is implemented using a JAVA based AIML interpreter called Chatter bean. Once the illness is detected, the Chatbot provides the user about the necessary information about the problem. However the previous proposed designs in the past did not focus in understanding the intensity of the illness that the user is suffering through. Our proposed design aims to ask more questions to the user until it gets confident about the probable illness that the user is suffering through. Also our Chatbot design has the concept of threshold level that helps it to detect the intensity of the problem and connects the user directly to the doctor if it feels that the problem is too serious for the Chatbot to handle.

Chapter 3

Problem Formulation

3.1 Problem with current chatbots

Chatbots has emerged as a hot topic in the latest years, and it is used by numerous companies in various areas - help desk tools, automatic telephone answering systems, e-commerce and so on. Even though the technology has been around since the 60's (Atwell & Shawar, 2007). Why are we suddenly so interested in this technology now? This can likely be explained by the recent year's advancements in messaging applications and AI technology (Brandtzaeg & Følstad, 2017). In the article Chatbots: Are they really useful? Atwell and Shawar provide real-life examples of different chatbots in different contexts. One of the examples is Sophia, a robot that was developed to assist in mathematics at Harvard by answering students questions. This turned out to be applicable in many other contexts. Living in Norway you have probably noticed "Kommune Kari". A chatbot that many of the municipality have available on their web-pages. Kari is there to answer "easy" questions like "when will the garbage truck come?" and "where can I find available jobs?". Kari's goal and the job is to provide information so that you as a user don't have to navigate the "massive information flow" (Schibevaag, 2017). This way of using a chatbot is a part of the Question Answering (QA) field which is a combination between AI and information retrieval (Molla & Vicedo, 2007). QA can be defined as: "... the task whereby an automated machine (such as a computer) answers arbitrary questions formulated in natural language. QA systems are especially useful in situations in which a user needs to know a very specific piece of information and does not have the time—or just does not want—to read all the available documentation related to the search topic in order to solve the problem at hand". (Molla & Vicedo, 2007). Sophia and Kari are examples of chatbots that operate in “very specific” domains. This means that if you were to ask Kari about math and Sophia about when the garbage truck comes none of them would know the answer - because the question is outside of their domain. Chatbots have what is called a natural language user interface and therefore communicate with users via natural language — how a human would talk on a regular basis (Brandtzaeg & Følstad, 2017). Therefore they use what is called natural language processing (NLP) where the chatbot uses computational techniques to analyze text, where the goal is to produce a human-like answer based on a linguistic analysis.

3.2 Proposed Model

A Chatbot needs to be natural at responding to the user messages and therefore it needs to have a sustainable back-end logic to process user inputs and parameters to generate results.When a user starts interacting with the Chatbot, the Chatbot engine gets activated and captures every messages provided by the user.The Chatbot intends to use the AIML approach to reply to the user messages and to get the input that it can feed to the engine.The engine accepts initial symptoms and extracts keywords from the data. Using the keywords extracted from the symptom, our Chatbot engine shortlists some of the most likely illnesses that the user may be suffering through by matching the keywords with the disease tags. Once the engine has shortlisted diseases that the user may have, now it has to narrows down the selection to only one disease. For this, it sorts the list of selected diseases according to the most number of matches with the keywords and tags. Now the engine checks for top 3 symptoms from each of the shortlisted (sorted) selection until all the 3 symptoms matches with the user and it can know that the user has that particular disease. If any two or more disease qualifies for the same, the user is connected directly to the doctor. Now the engine identifies the disease, our Chatbot asks questions to the user related to the symptoms that commonly happens in that disease.The engine can measure the seriousness of the problem by assigning a predetermined threshold value against every disease. Every symptom also has a seriousness score against it. The engine maintains an Integer variable where it sums up the scores of the symptoms if it matches with the user input. If the score hits a value greater than or equal to the threshold level, the Chatbot would connect the user with a doctor and provide temporary tips and medication until the doctor is available to chat. The engine also maintains two String Arrays during the chat session named 'Medication' and 'Remedies'. For every symptom that matches with the user input, the corresponding solution gets stored in the respective arrays.When the Chatbot has finished checking for all the symptoms it would then provide the user with all the Medication and Remedies it has found during the session.

Chapter 4

System Analysis & Design

4.1 System Analysis

Most businesses and organisations are understanding the potential benefits of machine learning and artificial intelligence to have a positive change on how they perform business. Artificial intelligence has progressed to allow the development of more sophisticated chatbots. Organisations are focusing on specific areas of user engagement that take up a lot of time but can be replaced through the use of a chatbot. Chatbots can understand what the customer needs from a single text instead of the customer having to follow a process of multiple steps. Chatbots are used to automate customer service and reduce manual tedious tasks performed by employees so they can spend their time more productively on higher priority tasks. Establishments that regularly deal with its customers have discovered the potential of chatbots as a channel to distribute more efficient and immediate information to customers in comparison to a customer service representative regarding queries and issues . HDFC Bank has merged with Niki.ai an artificial intelligence company to develop a state of the art conversational banking chatbot. The chatbot is accessible through the banks Facebook Messenger allowing users to utilise e-commerce and banking transactions all within the chatbot. There is also a chatbot integrated within the HDFC login page to assist in online banking. The chatbot was developed as a concierge service, this is definitely one approach that can increase customer satisfaction within online banking enabling banks to develop a better relationship with its customers. Chatbots will renew and modernise the customer service industry and the main sectors outlined including the banking and healthcare sector (Newswire 2017). Industries such as; retail, healthcare, e-commerce and banking, are expected to achieve considerable savings from integrating a chatbot. Between

2017 and 2020 chatbots are estimated to make savings of $8 billion per annum for businesses. The majority of savings will be made through customer service as customers can now ask queries about banking through the chatbot rather than having to call the bank, allowing banks to save on call centre staff. The study predicted that the integration of chatbots within the banking industry would rise from 12% to 90% by 2022. Most online banking services would benefit from having a chatbot integrated into their services. The use of bots help internet banking service providers establish a better relationship with its customers. Customers can get answers to query's immediately, conduct e-commerce and banking operations all from within the one bot conversation. This is another benefit of using an online banking system with and integrated chatbot as "87% customers that bank online prefer to execute their personal finance operations within a single site". W.B. King, W.B. (2017), records a survey conducted by Personetics which identified "40% of its clients are planning to integrate a bot into its services within the next 2 to 3 years and 70% said AI was an exciting opportunity". This clearly identifies the demand and need for a chatbot within the financial services Industry.

4.2 System Design

In the proposed system the user dialogue is a linear design that proceeds from symptom extraction, to symptom mapping, where it identifies the corresponding symptom, then diagnosis the patient whether it's a major or minor disease and if it's a major one an appropriate doctor will be referred to the patient.

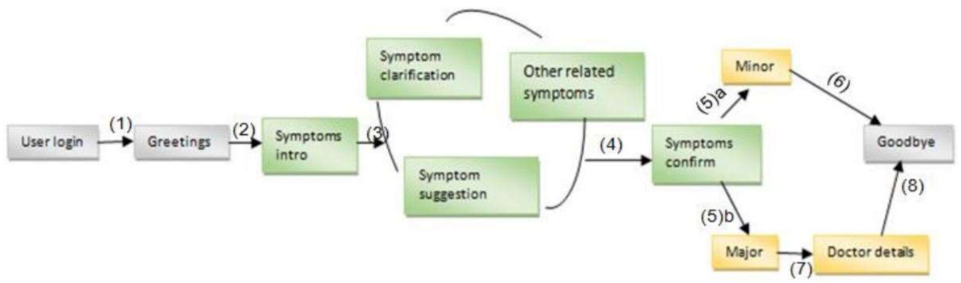
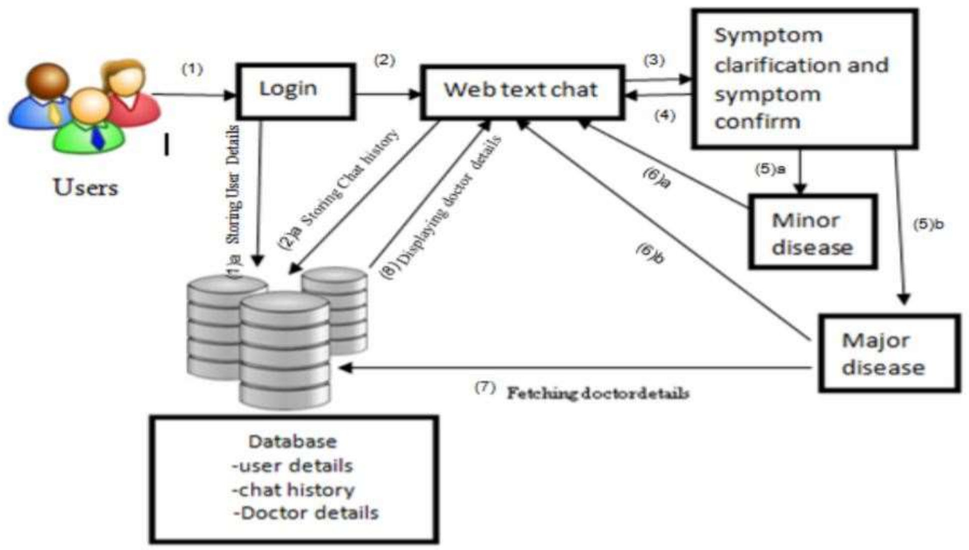


Fig 4.1 Chatbot's dialogue design

The doctor details will be extracted from the database, the user will be identified by the login details which is stored in the database. In fig 4.1, Chatbot's dialogue design is represented using finite state graph. In order to achieve an accurate diagnosis, the logic for state transitions are made, natural language generation templates were used, and system initiative to the user and get responses from the user. Besides its greetings and goodbye states, our agent has three main conversational phases: acquisition of basic information, symptom extraction, and diagnosis. Our bot starts off by asking about the user's email and password for login and then enters a loop of symptom extraction states until it acquires sufficient information for a diagnosis. Users have the option of entering the loop again to talk to the doctor about another set of symptoms after receiving their first diagnosis and the another option is that the user can view their history of chats about what they have discussed.



Users
(1)
Login
(2)
Web text chat
(3)
(4)
Symptom clarification and symptom confirm
(1)a Storing User Details
(2)a Storing Chat history
(8) Displaying doctor details
(6)a
(6)b
(5)a
(5)b
Minor disease
Major disease
(7) Fetching doctor details
Database
-user details
-chat history
-Doctor details

Fig 4.2 Architecture of Chatbot

The above Figure fig 4.2 proceeds with the users login where the users' details will be stored in the database. Then the user can start their conversation with the chatbot and it will be stored in the database for future reference. The chatbot will clarify the users symptoms with serious of questions and the symptom conformation will be done. The disease will be categorized as minor and major disease. Chatbot will reply whether it's a major or minor disease. If it's a major one user will be suggested with the doctor details for further treatment.

4.2.1 USER VALIDATION AND EXTRACTION OF SYMPTOMS

The validation of the user login details occurs here. Then Symptoms are extracted using String Searching Algorithm where substring representing the symptoms is identified in the natural language text input. When users give directly the symptom name such as(e.g. "I have a cough, fever, and nausea"), the system will easily identify it. But however, the system should also be able to handle input like, "When I read, I'm okay at first, but over time, my eyes seem to get tired, and I start to see double." In this case, the system should extract substrings like "eyes tired" and "see double" (and not substrings like "read" or "okay").

4.2.1 MAPPING EXTRACTED SYMPTOMS WITH TRAINED DATASETS

Given some extracted substring from the user's input, we generate a list of suggested closest symptoms. We then ask the user to confirm if they have any of the suggested symptoms. Based on their reply few diseases are being shortlisted. Then further symptom clarification and symptom suggestions are being done by asking the users a series of questions and the mapping of the symptoms to the exact disease is done.

4.2.3 SPECIFYING THE DISEASE AND REFERRING A DOCTOR

This process carries the list of diseases in the database and each symptom being entered is compared to the symptoms of the common diseases. Next symptom is checked until a matching one is found. The diseases are shortlisted based on the end users input on the question evaluation. The accurate disease is identified and specified to the end user by the chatbot. The chatbot checks whether the identified disease is a major issue or minor issue based on the conditions built in the chatbot. If it is a major issue the chatbot refers a specialist to the end user by sending the doctor details. And if it is a minor issue the chatbot specifies the disease and alerts the end user with a first aid or remedy and asks to visit a doctor shortly

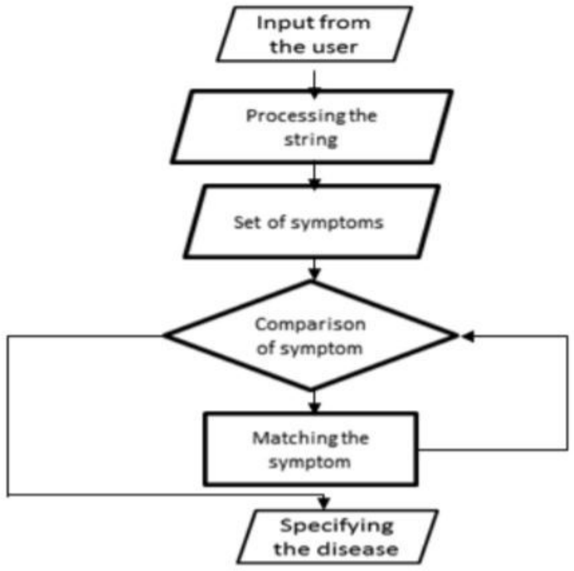


Fig 4.3 Flowchart of chatbot

Chapter 5

Implementation

5.1 NLP Introduction

Natural Language Processing (NLP) is the computerized approach to analyzing text that is based on both a set of theories and a set of technologies. And, being a very active area of research and development, there is not a single agreed-upon definition that would satisfy everyone, but there are some aspects, which would be part of any knowledgeable person's definition. The definition is Natural Language Processing is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications. Several elements of this definition can be further detailed. Firstly, the imprecise notion of 'range of computational techniques' is necessary because there are multiple methods or techniques from which to choose to accomplish a particular type of language analysis. 'Naturally occurring texts' can be of any language, mode, genre, etc. The texts can be oral or written. The only requirement is that they be in a language used by humans to communicate to one another. Also, the text being analyzed should not be specifically constructed for the purpose of the analysis, but rather that the text be gathered from actual usage. The notion of 'levels of linguistic analysis' refers to the fact that there are multiple types of language processing known to be at work when humans produce or comprehend language. It is thought that humans normally utilize all of these levels since each level conveys different types of meaning. But various NLP systems utilize different levels, or combinations of levels of linguistic analysis, and this is seen in the differences amongst various NLP applications. This also leads to much confusion on the part of non-specialists as to what NLP really is, because a system that uses any subset of these levels of analysis can be said to be an NLP-based system. The difference between them, therefore, may actually be whether the system uses ‘weak’ NLP or ‘strong’ NLP. ‘Human-like language processing’ reveals that NLP is considered a discipline within Artificial Intelligence (AI). And while the full lineage of NLP does depend on a number of other disciplines, since NLP strives for human-like performance, it is appropriate to consider it an AI discipline. ‘For a range of tasks or applications’ points out that NLP is not usually considered a goal in and of itself, except perhaps for AI researchers. For others, NLP is the means for 1 Liddy, E. D. In Encyclopedia of Library and Information Science, 2nd Ed. Marcel Decker, Inc. accomplishing a particular task. Therefore, you have Information Retrieval (IR) systems that utilize NLP, as well as Machine Translation (MT), Question-Answering, etc. Goal The goal of NLP as stated above is “to accomplish human-like language processing”. The choice of the word ‘processing’ is very deliberate, and should not be replaced with ‘understanding’. For although the field of NLP was originally referred to as Natural Language Understanding (NLU) in the early days of AI, it is well agreed today that while the goal of NLP is true NLU, that goal has not yet been accomplished. A full NLU System would be able to: 1. Paraphrase an input text 2. Translate the text into another language 3. Answer questions about the contents of the text 4. Draw inferences from the text While NLP has made serious inroads into accomplishing goals 1 to 3, the fact that NLP systems cannot, of themselves, draw inferences from text, NLU still remains the goal of NLP. There are more practical goals for NLP, many related to the particular application for which it is being utilized. For example, an NLP-based IR system has the goal of providing more precise, complete information in response to a user’s real information need. The goal of the NLP system here is to represent the true meaning and intent of the user’s query, which can be expressed as naturally in everyday language as if they were speaking to a reference librarian. Also, the contents of the documents that are being searched will be represented at all their levels of meaning so that a true match between need and response can be found, no matter how either are expressed in their surface form. Origins As most modern disciplines, the lineage of NLP is indeed mixed, and still today has strong emphases by different groups whose backgrounds are more influenced by one or another of the disciplines. Key among the contributors to the discipline and practice of NLP are: Linguistics - focuses on formal, structural models of language and the discovery of language universals - in fact the field of NLP was originally referred to as Computational Linguistics; Computer Science - is concerned with developing internal representations of data and efficient processing of these structures, and; Cognitive Psychology - looks at language usage as a window into human cognitive processes, and has the goal of modeling the use of language in a psychologically plausible way. Divisions While the entire field is referred to as Natural Language Processing, there are in fact two distinct focuses – language processing and language generation. The first of these refers to the analysis of language for the purpose of producing a meaningful representation, while the latter refers to the production of language from a representation. The task of Natural Language Processing is equivalent to the role of reader/listener, while the task of Natural Language Generation is that of the writer/speaker. While much of the theory and technology are shared by these two divisions, Natural Language Generation also requires a planning capability. That is, the generation system requires a plan or model of the goal of the interaction in order to decide what the system should generate at each point in an interaction. We will focus on the task of natural language analysis, as this is most relevant to Library and Information Science. Another distinction is traditionally made between language understanding and speech understanding. Speech understanding starts with, and speech generation ends with, oral language and therefore rely on the additional fields of acoustics and phonology. Speech understanding focuses on how the 'sounds' of language as picked up by the system in the form of acoustical waves are transcribed into recognizable morphemes and words. Once in this form, the same levels of processing which are utilized on written text are utilized. All of these levels, including the phonology level, will be covered in Section 2 however, the emphasis throughout will be on language in the written form. Levels of natural language processing research in natural language processing has been going on for several decades dating back to the late 1940s. Machine translation (MT) was the first computer-based application related to natural language. While Weaver and started one of the earliest MT projects in 1946 on computer translation based on expertise in breaking enemy codes during World War II, it was generally agreed that it was Weaver's memorandum of 1949 that brought the idea of MT to general notice and inspired many projects. He suggested using ideas from cryptography and information theory for language translation. Research began at various research institutions in the United States within a few years. Early work in MT took the simplistic view that the only differences between languages resided in their vocabularies and the permitted word orders. Systems developed from this perspective simply used dictionary-lookup for appropriate words for translation and reordered the words after translation to fit the word-order rules of the target language, without taking into account the lexical ambiguity inherent in natural language. This produced poor results. The apparent failure made researchers realize that the task was a lot harder than anticipated, and they needed a more adequate theory of language. However, it was not until 1957 when Chomsky published Syntactic Structures introducing the idea of generative grammar, did the field gain better insight into whether or how mainstream linguistics could help MT. During this period, other NLP application areas began to emerge, such as speech recognition. The language processing community and the speech community then was split into two camps with the language processing community dominated by the theoretical perspective of generative grammar and hostile to statistical methods, and the speech community dominated by statistical information theory and hostile to theoretical linguistics . Due to the developments of the syntactic theory of language and parsing algorithms, there was over-enthusiasm in the 1950s that people believed that fully automatic high quality translation systems would be able to produce results indistinguishable from those of human translators, and such systems should be in operation within a few years. It was not only unrealistic given the then-available linguistic knowledge and computer systems, but also impossible in principle. The inadequacies of then-existing systems, and perhaps accompanied by the overenthusiasm, led to the ALPAC (Automatic Language Processing Advisory Committee of the National Academy of Science - National Research Council) report of 1966. The report concluded that MT was not immediately achievable and recommended it not be funded. This had the effect of halting MT and most work in other applications of NLP at least within the United States. Although there was a substantial decrease in NLP work during the years after the ALPAC report, there were some significant developments, both in theoretical issues and in construction of prototype systems. Theoretical work in the late 1960's and early 1970's focused on the issue of how to represent meaning and developing computationally tractable solutions that the then-existing theories of grammar were not able to produce. In 1965, Chomsky introduced the transformational model of linguistic competence. However, the transformational generative grammars were too syntactically oriented to allow for semantic concerns. They also did not lend themselves easily to computational implementation. As a reaction to Chomsky's theories and the work of other transformational generativists, case grammar of Fillmore, semantic networks of Quillian, and conceptual dependency theory of Schank, were developed to explain syntactic anomalies, and provide semantic representations. Augmented transition networks of Woods, extended the power of phrase-structure grammar by incorporating mechanisms from programming languages such as LISP. Other representation formalisms included Wilks' preference semantics, and Kay's functional grammar. Alongside theoretical development, many prototype systems were developed to demonstrate the effectiveness of particular principles. Weizenbaum's ELIZA was built to replicate the conversation between a psychologist and a patient, simply by permuting or echoing the user input. Winograd's SHRDLU simulated a robot that manipulated blocks on a tabletop. Despite its limitations, it showed that natural language understanding was indeed possible for the computer. PARRY attempted to embody a theory of paranoia in a system. Instead of single keywords, it used groups of keywords, and used synonyms if keywords were not found. LUNAR was developed by Woods as an interface system to a database that consisted of information about lunar rock samples using augmented transition network and procedural semantics. In the late 1970's, attention shifted to semantic issues, discourse phenomena, and communicative goals and plans. Grosz analyzed task-oriented dialogues and proposed a theory to partition the discourse into units based on her findings about the relation between the structure of a task and the structure of the task-oriented dialogue. Mann and Thompson developed Rhetorical Structure Theory, attributing hierarchical structure to discourse. Other researchers have also made significant contributions, including Hobbs and Rosenschein, Polanyi and Scha, and Reichman. This period also saw considerable work on natural language generation. McKeown's discourse planner TEXT and McDonald's response generator MUMMBLE used rhetorical predicates to produce declarative descriptions in the form of short texts, usually paragraphs. TEXT's ability to generate coherent responses online was considered a major achievement. In the early 1980s, motivated by the availability of critical computational resources, the growing awareness within each community of the limitations of isolated solutions to NLP problems, and a general push toward applications that worked with language in a broad, real-world context, researchers started re-examining non-symbolic approaches that had lost popularity in early days. By the end of 1980s, symbolic approaches had been used to address many significant problems in NLP and statistical approaches were shown to be complementary in many respects to symbolic approaches. In the last ten years of the millennium, the field was growing rapidly. This can be attributed to a increased availability of large amounts of electronic text b availability of computers with increased speed and memory and c the advent of the Internet. Statistical approaches succeeded in dealing with many generic problems in computational linguistics such as part-of-speech identification, word sense disambiguation, etc., and have become standard throughout NLP. NLP researchers are now developing next generation NLP systems that deal reasonably well with general text and account for a good portion of the variability and ambiguity of language.

5.1.1 LEVELS OF NATURAL LANGUAGE PROCESSING

The most explanatory method for presenting what actually happens within a Natural Language Processing system is by means of the 'levels of language' approach. This is also referred to as the synchronic model of language and is distinguished from the earlier sequential model, which hypothesizes that the levels of human language processing follow one another in a strictly sequential manner. Psycholinguistic research suggests that language processing is much more dynamic, as the levels can interact in a variety of orders. Introspection reveals that we frequently use information we gain from what is typically thought of as a higher level of processing to assist in a lower level of analysis. For example, the pragmatic knowledge that the document you are reading is about biology will be used when a particular word that has several possible senses (or meanings) is encountered, and the word will be interpreted as having the biology sense. Of necessity, the following description of levels will be presented sequentially. The key point here is that meaning is conveyed by each and every level of language and that since humans have been shown to use all levels of language to gain understanding, the more capable an NLP system is, the more levels of language it will utilize. Synchronized Model of Language Processing Phonology This level deals with the interpretation of speech sounds within and across words. There are, in fact, three types of rules used in phonological analysis phonetic rules – for sounds within words phonemic rules for variations of pronunciation when words are spoken together, and prosodic rules for fluctuation in stress and intonation across a sentence. In an NLP system that accepts spoken input, the sound waves are analyzed and encoded into a digitized signal for interpretation by various rules or by comparison to the particular language model being utilized. Morphology This level deals with the componential nature of words, which are composed of morphemes – the smallest units of meaning. For example, the word preregistration can be morphologically analyzed into three separate morphemes: the prefix pre, the root registration, and the suffixation. Since the meaning of each morpheme remains the same across words, humans can break down an unknown word into its constituent morphemes in order to understand its meaning. Similarly, an NLP system can recognize the meaning conveyed by each morpheme in order to gain and represent meaning. For example, adding the suffix -ed to a verb, conveys that the action of the verb took place in the past. This is a key piece of meaning, and in fact, is frequently only evidenced in a text by the use of the -ed morpheme. Lexical At this level, humans, as well as NLP systems, interpret the meaning of individual words. Several types of processing contribute to word-level understanding – the first of these being assignment of a single part-of-speech tag to each word. In this processing, words that can function as more than one part-of-speech are assigned the most probable part-of speech tag based on the context in which they occur. Additionally, at the lexical level, those words that have only one possible sense or meaning can be replaced by a semantic representation of that meaning. The nature of the representation varies according to the semantic theory utilized in the NLP system. The following representation of the meaning of the word launch is in the form of logical predicates. As can be observed, a single lexical unit is decomposed into its more basic properties. Given that there is a set of semantic primitives used across all words, these simplified lexical representations make it possible to unify meaning across words and to produce complex interpretations, much the same as humans do. launch (a large boat used for carrying people on rivers, lakes harbors, etc.) The lexical level may require a lexicon, and the particular approach taken by an NLP system will determine whether a lexicon will be utilized, as well as the nature and extent of information that is encoded in the lexicon. Lexicons may be quite simple, with only the words and their part(s)-of-speech, or may be increasingly complex and contain information on the semantic class of the word, what arguments it takes, and the semantic limitations on these arguments, definitions of the sense(s) in the semantic representation utilized in the particular system, and even the semantic field in which each sense of a polysemous word is used. Syntactic This level focuses on analyzing the words in a sentence so as to uncover the grammatical structure of the sentence. This requires both a grammar and a parser. The output of this level of processing is a (possibly de-linearized) representation of the sentence that reveals the structural dependency relationships between the words. There are various grammars that can be utilized, and which will, in turn, impact the choice of a parser. Not all NLP applications require a full parse of sentences, therefore the remaining challenges in parsing of prepositional phrase attachment and conjunction scoping no longer stymie those applications for which phrasal and clausal dependencies are sufficient. Syntax conveys meaning in most languages because order and dependency contribute to meaning. For example the two sentences: ‘The dog chased the cat.’ and ‘The cat chased the dog.’ differ only in terms of syntax, yet convey quite different meanings. Semantic This is the level at which most people think meaning is determined, however, as we can see in the above defining of the levels, it is all the levels that contribute to meaning. Semantic processing determines the possible meanings of a sentence by focusing on the interactions among word-level meanings in the sentence. This level of processing can include the semantic disambiguation of words with multiple senses; in an analogous way to how syntactic disambiguation of words that can function as multiple parts-of-speech is accomplished at the syntactic level. Semantic disambiguation permits one and only one sense of polysemous words to be selected and included in the semantic representation of the sentence. For example, amongst other meanings, 'file' as a noun can mean either a folder for storing papers, or a tool to shape one's fingernails, or a line of individuals in a queue. If information from the rest of the sentence were required for the disambiguation, the semantic, not the lexical level, would do the disambiguation. A wide range of methods can be implemented to accomplish the disambiguation, some which require information as to the frequency with which each sense occurs in a particular corpus of interest, or in general usage, some which require consideration of the local context, and others which utilize pragmatic knowledge of the domain of the document. Discourse While syntax and semantics work with sentence-length units, the discourse level of NLP works with units of text longer than a sentence. That is, it does not interpret multisentence texts as just concatenated sentences, each of which can be interpreted singly. Rather, discourse focuses on the properties of the text as a whole that convey meaning by making connections between component sentences. Several types of discourse processing can occur at this level, two of the most common being anaphora resolution and discourse/text structure recognition. Anaphora resolution is the replacing of words such as pronouns, which are semantically vacant, with the appropriate entity to which they refer. Discourse/text structure recognition determines the functions of sentences in the text, which, in turn, adds to the meaningful representation of the text. For example, newspaper articles can be deconstructed into discourse components such as: Lead, Main Story, Previous Events, Evaluation, Attributed Quotes, and Expectation. Pragmatic This level is concerned with the purposeful use of language in situations and utilizes context over and above the contents of the text for understanding the goal is to explain how extra meaning is read into texts without actually being encoded in them. This requires much world knowledge, including the understanding of intentions, plans, and goals. Some NLP applications may utilize knowledge bases and inferencing modules. For example, the following two sentences require resolution of the anaphoric term 'they', but this resolution requires pragmatic or world knowledge. The city councilors refused the demonstrators a permit because they feared violence. The city councilors refused the demonstrators a permit because they advocated revolution. Summary of Levels Current NLP systems tend to implement modules to accomplish mainly the lower levels of processing. This is for several reasons. First, the application may not require interpretation at the higher levels. Secondly, the lower levels have been more thoroughly researched and implemented. Thirdly, the lower levels deal with smaller units of analysis, e.g. morphemes, words, and sentences, which are rule-governed, versus the higher levels of language processing which deal with texts and world knowledge, and which are only regularity-governed. As will be seen in the following section on Approaches, the statistical approaches have, to date, been validated on the lower levels of analysis, while the symbolic approaches have dealt with all levels, although there are still few working systems which incorporate the higher levels.

5.1.2 APPROACHES TO NATURAL LANGUAGE PROCESSING

Natural language processing approaches fall roughly into four categories: symbolic, statistical, connectionist, and hybrid. Symbolic and statistical approaches have coexisted since the early days of this field. Connectionist NLP work first appeared in the 1960's. For a long time, symbolic approaches dominated the field. In the 1980's, statistical approaches regained popularity as a result of the availability of critical computational resources and the need to deal with broad, real-world contexts. Connectionist approaches also recovered from earlier criticism by demonstrating the utility of neural networks in NLP. This section examines each of these approaches in terms of their foundations, typical techniques, differences in processing and system aspects, and their robustness, flexibility, and suitability for various tasks. Symbolic Approach Symbolic approaches perform deep analysis of linguistic phenomena and are based on explicit representation of facts about language through well-understood knowledge representation schemes and associated algorithms. In fact, the description of the levels of language analysis in the preceding section is given from a symbolic perspective. The primary source of evidence in symbolic systems comes from human-developed rules and lexicons. A good example of symbolic approaches is seen in logic or rule-based systems. In logicbased systems, the symbolic structure is usually in the form of logic propositions. Manipulations of such structures are defined by inference procedures that are generally truth preserving. Rule-based systems usually consist of a set of rules, an inference engine, and a workspace or working memory. Knowledge is represented as facts or rules in the rule-base. The inference engine repeatedly selects a rule whose condition is satisfied and executes the rule. Another example of symbolic approaches is semantic networks. First proposed by Quillian to model associative memory in psychology, semantic networks represent knowledge through a set of nodes that represent objects or concepts and the labeled links that represent relations between nodes. The pattern of connectivity reflects semantic organization, that is; highly associated concepts are directly linked whereas moderately or weakly related concepts are linked through intervening concepts. Semantic networks are widely used to represent structured knowledge and have the most connectionist flavor of the symbolic models. Symbolic approaches have been used for a few decades in a variety of research areas and applications such as information extraction, text categorization, ambiguity resolution, and lexical acquisition. Typical techniques include: explanation-based learning, rule-based learning, inductive logic programming, decision trees, conceptual clustering, and K nearest neighbor algorithms. Statistical Approach Statistical approaches employ various mathematical techniques and often use large text corpora to develop approximate generalized models of linguistic phenomena based on actual examples of these phenomena provided by the text corpora without adding significant linguistic or world knowledge. In contrast to symbolic approaches, statistical approaches use observable data as the primary source of evidence. A frequently used statistical model is the Hidden Markov Model (HMM) inherited from the speech community. HMM is a finite state automaton that has a set of states with probabilities attached to transitions between states. Although outputs are visible, states themselves are not directly observable, thus “hidden” from external observations. Each state produces one of the observable outputs with a certain probability. Statistical approaches have typically been used in tasks such as speech recognition, lexical acquisition, parsing, part-of-speech tagging, collocations, statistical machine translation, statistical grammar learning, and so on. Connectionist Approach Similar to the statistical approaches, connectionist approaches also develop generalized models from examples of linguistic phenomena. What separates connectionism from other statistical methods is that connectionist models combine statistical learning with various theories of representation - thus the connectionist representations allow transformation, inference, and manipulation of logic formulae. In addition, in connectionist systems, linguistic models are harder to observe due to the fact that connectionist architectures are less constrained than statistical ones. Generally speaking, a connectionist model is a network of interconnected simple processing units with knowledge stored in the weights of the connections between units. Local interactions among units can result in dynamic global behavior, which, in turn, leads to computation. Some connectionist models are called localist models, assuming that each unit represents a particular concept. For example, one unit might represent the concept “mammal” while another unit might represent the concept “whale”. Relations between concepts are encoded by the weights of connections between those concepts. Knowledge in such models is spread across the network, and the connectivity between units reflects their structural relationship. Localist models are quite similar to semantic networks, but the links between units are not usually labeled as they are in semantic nets. They perform well at tasks such as word-sense disambiguation, language generation, and limited inference. Other connectionist models are called distributed models. Unlike that in localist models, a concept in distributed models is represented as a function of simultaneous activation of multiple units. An individual unit only participates in a concept representation. These models are well suited for natural language processing tasks such as syntactic parsing, limited domain translation tasks, and associative retrieval. Comparison Among Approaches From the above section, we have seen that similarities and differences exist between approaches in terms of their assumptions, philosophical foundations, and source of evidence. In addition to that, the similarities and differences can also be reflected in the processes each approach follows, as well as in system aspects, robustness, flexibility, and suitable tasks. Process: Research using these different approaches follows a general set of steps, namely, data collection, data analysis/model building, rule/data construction, and application of rules/data in system. The data collection stage is critical to all three approaches although statistical and connectionist approaches typically require much more data than symbolic approaches. In the data analysis/model building stage, symbolic approaches rely on human analysis of the data in order to form a theory while statistical approaches manually define a statistical model that is an approximate generalization of the collected data. Connectionist approaches build a connectionist model from the data. In the rule / data construction stage, manual efforts are typical for symbolic approaches and the theory formed in the previous step may evolve when new cases are encountered. In contrast, statistical and connectionist approaches use the statistical or connectionist model as guidance and build rules or data items automatically, usually in relatively large quantity. After building rules or data items, all approaches then automatically apply them to specific tasks in the system. For instance, connectionist approaches may apply the rules to train the weights of links between units. System aspects: By system aspects, we mean source of data, theory or model formed from data analysis, rules, and basis for evaluation. - Data As mentioned earlier, symbolic approaches use human introspective data, which are usually not directly observable. Statistical and connectionist approaches are built on the basis of machine observable facets of data, usually from text corpora. - Theory or model based on data analysis: As the outcome of data analysis, a theory is formed for symbolic approaches whereas a parametric model is formed for statistical approaches and a connectionist model is formed for connectionist approaches. - Rules: For symbolic approaches, the rule construction stage usually results in rules with detailed criteria of rule application. For statistical approaches, the criteria of rule application are usually at the surface level or under-specified. For connectionist approaches, individual rules typically cannot be recognized. - Basis for Evaluation: Evaluation of symbolic systems is typically based on intuitive judgments of unaffiliated subjects and may use system-internal measures of growth such as the number of new rules. In contrast, the basis for evaluation of statistical and connectionist systems are usually in the form of scores computed from some evaluation function. However, if all approaches are utilized for the same task, then the results of the task can be evaluated both quantitatively and qualitatively and compared. Robustness: Symbolic systems may be fragile when presented with unusual, or noisy input. To deal with anomalies, they can anticipate them by making the grammar more general to accommodate them. Compared to symbolic systems, statistical systems may be more robust in the face of unexpected input provided that training data is sufficient, which may be difficult to be assured of. Connectionist systems may also be robust and fault tolerant because knowledge in such systems is stored across the network. When presented with noisy input, they degrade gradually. Flexibility: Since symbolic models are built by human analysis of well-formulated examples, symbolic systems may lack the flexibility to adapt dynamically to experience. In contrast, statistical systems allow broad coverage, and may be better able to deal with unrestricted text for more effective handling of the task at hand. Connectionist systems exhibit flexibility by dynamically acquiring appropriate behavior based on the given input. For example, the weights of a connectionist network can be adapted in real time to improve performance. However, such systems may have difficulty with the representation of structures needed to handle complex conceptual relationships, thus limiting their abilities to handle high-level NLP. Suitable tasks: Symbolic approaches seem to be suited for phenomena that exhibit identifiable linguistic behavior. They can be used to model phenomena at all the various linguistic levels described in earlier sections. Statistical approaches have proven to be effective in modeling language phenomena based on frequent use of language as reflected in text corpora. Linguistic phenomena that are not well understood or do not exhibit clear regularity are candidates for statistical approaches. Similar to statistical approaches, connectionist approaches can also deal with linguistic phenomena that are not well understood. They are useful for low-level NLP tasks that are usually subtasks in a larger problem. To summarize, symbolic, statistical, and connectionist approaches have exhibited different characteristics, thus some problems may be better tackled with one approach while other problems by another. In some cases, for some specific tasks, one approach may prove adequate, while in other cases, the tasks can get so complex that it might not be possible to choose a single best approach. In addition, as Klavans and Resnik pointed out, there is no such thing as a "purely statistical" method. Every use of statistics is based upon a symbolic model and statistics alone is not adequate for NLP. Toward this end, statistical approaches are not at odds with symbolic approaches. In fact, they are rather complementary. As a result, researchers have begun developing hybrid techniques that utilize the strengths of each approach in an attempt to address NLP problems more effectively and in a more flexible manner.

5.1.3 NATURAL LANGUAGE PROCESSING APPLICATIONS

Natural language processing provides both theory and implementations for a range of applications. In fact, any application that utilizes text is a candidate for NLP. The most frequent applications utilizing NLP include the following:

• Information Retrieval – given the significant presence of text in this application, it is surprising that so few implementations utilize NLP. Recently, statistical approaches for accomplishing NLP have seen more utilization, but few systems other than those by Liddy and Strzalkowski have developed significant systems based on NLP .

• Information Extraction (IE) – a more recent application area, IE focuses on the recognition, tagging, and extraction into a structured representation, certain key elements of information, e.g. persons, companies, locations, organizations, from large collections of text. These extractions can then be utilized for a range of applications including question-answering, visualization, and data mining.

• Question-Answering – in contrast to Information Retrieval, which provides a list of potentially relevant documents in response to a user’s query, question-answering provides the user with either just the text of the answer itself or answer-providing passages.

• Summarization – the higher levels of NLP, particularly the discourse level, can empower an implementation that reduces a larger text into a shorter, yet rich constituted abbreviated narrative representation of the original document.

• Machine Translation – perhaps the oldest of all NLP applications, various levels of NLP have been utilized in MT systems, ranging from the ‘word-based’ approach to applications that include higher levels of analysis.

• Dialogue Systems – perhaps the omnipresent application of the future, in the systems envisioned by large providers of end-user applications. Dialogue systems, which usually focus on a narrowly defined application (e.g. your refrigerator or home sound system), currently utilize the phonetic and lexical levels of language. It is believed that utilization of all the levels of language processing explained above offer the potential for truly habitable dialogue systems.

5.2 Implementation

5.2.1 Import and load the data file

First, make a file name as **train_chatbot.py**. We import the necessary packages for our chatbot and initialize the variables we will use in our Python project. The data file is in JSON format so we used the json package to parse the JSON file into Python.

```
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import json
import pickle

import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from keras.optimizers import SGD
import random

words=[]
```

```
classes = []
documents = []
ignore_words = ['?', '!']
data_file = open('intents.json').read()
intents = json.loads(data_file)
```

5.2.2 Preprocess data

When working with text data, we need to perform various pre-processing on the data before we make a machine learning or a deep learning model. Tokenizing is the most basic and first thing you can do on text data. Tokenizing is the process of breaking the whole text into small parts like words.

Here we iterate through the patterns and tokenize the sentence using **nltk.word_tokenize()** function and append each word in the words list. We also create a list of classes for our tags.

```
for intent in intents['intents']:
for pattern in intent['patterns']:

#tokenize each word
w = nltk.word_tokenize(pattern)
words.extend(w)
#add documents in the corpus
documents.append((w, intent['tag']))

# add to our classes list
if intent['tag'] not in classes:
classes.append(intent['tag'])
```

Now we will lemmatize each word and remove duplicate words from the list. Lemmatizing is the process of converting a word into its lemma form and then creating a pickle file to store the Python objects which we will use while predicting.

```
# lemmatize, lower each word and remove duplicates
words = [lemmatizer.lemmatize(w.lower()) for w in words if w
not in ignore_words]
words = sorted(list(set(words)))
# sort classes
classes = sorted(list(set(classes)))
# documents = combination between patterns and intents
print (len(documents), "documents")
```

```
# classes = intents
print (len(classes), "classes", classes)
# words = all words, vocabulary
print (len(words), "unique lemmatized words", words)

pickle.dump(words,open('words.pkl','wb'))
pickle.dump(classes,open('classes.pkl','wb'))
```

5.2.3 Create training and testing data

Now, we will create the training data in which we will provide the input and the output. Our input will be the pattern and output will be the class our input pattern belongs to. But the computer doesn't understand text so we will convert text into numbers.

```
# create our training data
training = []
# create an empty array for our output
output_empty = [0] * len(classes)
# training set, bag of words for each sentence
for doc in documents:
# initialize our bag of words
bag = []
# list of tokenized words for the pattern
pattern_words = doc[0]
# lemmatize each word - create base word, in attempt to represent related words
pattern_words = [lemmatizer.lemmatize(word.lower()) for word in pattern_words]
# create our bag of words array with 1, if word match found in current pattern
for w in words:
bag.append(1) if w in pattern_words else bag.append(0)

# output is a '0' for each tag and '1' for current tag (for each pattern)
output_row = list(output_empty)
output_row[classes.index(doc[1])] = 1

training.append([bag, output_row])
# shuffle our features and turn into np.array
random.shuffle(training)
training = np.array(training)
# create train and test lists. X - patterns, Y - intents
train_x = list(training[:,0])
train_y = list(training[:,1])
print("Training data created")
```

5.2.4 Build the model

We have our training data ready, now we will build a deep neural network that has 3 layers. We use the Keras sequential API for this. After training the model for 200 epochs, we achieved 100% accuracy on our model. Let us save the model as ‘chatbot_model.h5’.

```python
# Create model - 3 layers. First layer 128 neurons, second layer 64 neurons and 3rd output layer contains number of neurons
# equal to number of intents to predict output intent with softmax
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))

# Compile model. Stochastic gradient descent with Nesterov accelerated gradient gives good results for this model
sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])

#fitting and saving the model
hist = model.fit(np.array(train_x), np.array(train_y), epochs=200, batch_size=5, verbose=1)
model.save('chatbot_model.h5', hist)

print("model created")
```

5.2.5 Predict the response (Graphical User Interface)-testing

Now to predict the sentences and get a response from the user to let us create a new file ‘chatapp.py’.

We will load the trained model and then use a graphical user interface that will predict the response from the bot. The model will only tell us the class it belongs to, so we will implement some functions which will identify the class and then retrieve us a random response from the list of responses.

Again we import the necessary packages and load the ‘**words.pkl**’ and ‘**classes.pkl**’ pickle files which we have created when we trained our model

```python
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import pickle
import numpy as np

from keras.models import load_model
model = load_model('chatbot_model.h5')
import json
import random
intents = json.loads(open('intents.json').read())
words = pickle.load(open('words.pkl','rb'))
classes = pickle.load(open('classes.pkl','rb'))
```

To predict the class, we will need to provide input in the same way as we did while training. So we will create some functions that will perform text preprocessing and then predict the class.

```python
def clean_up_sentence(sentence):
# tokenize the pattern - split words into array
sentence_words = nltk.word_tokenize(sentence)
# stem each word - create short form for word
sentence_words = [lemmatizer.lemmatize(word.lower()) for word
in sentence_words]
return sentence_words
# return bag of words array: 0 or 1 for each word in the bag
that exists in the sentence

def bow(sentence, words, show_details=True):
# tokenize the pattern
sentence_words = clean_up_sentence(sentence)
# bag of words - matrix of N words, vocabulary matrix
bag = [0]*len(words)
for s in sentence_words:
for i,w in enumerate(words):
if w == s:
# assign 1 if current word is in the vocabulary position
bag[i] = 1
if show_details:
print ("found in bag: %s" % w)
```

```
return(np.array(bag))

def predict_class(sentence, model):
# filter out predictions below a threshold
p = bow(sentence, words,show_details=False)
res = model.predict(np.array([p]))[0]
ERROR_THRESHOLD = 0.25
results = [[i,r] for i,r in enumerate(res) if r>ERROR_THRESHOLD]
# sort by strength of probability
results.sort(key=lambda x: x[1], reverse=True)
return_list = []
for r in results:
return_list.append({"intent": classes[r[0]], "probability": str(r[1])})
return return_list
```

After predicting the class, we will get a random response from the list of intents.

```
def getResponse(ints, intents_json):
tag = ints[0]['intent']
list_of_intents = intents_json['intents']
for i in list_of_intents:
if(i['tag']== tag):
result = random.choice(i['responses'])
break
return result

def chatbot_response(text):
ints = predict_class(text, model)
res = getResponse(ints, intents)
return res
```

Now we will code a graphical user interface. For this, we use the Tkinter library which already comes in python. We will take the input message from the user and then use the helper functions we have created to get the response from the bot and display it on the GUI. Here is the full source code for the GUI.

```
#Creating GUI with tkinter
import tkinter
from tkinter import *


def send():
msg = EntryBox.get("1.0",'end-1c').strip()
EntryBox.delete("0.0",END)
```

```

if msg != '':
ChatLog.config(state=NORMAL)
ChatLog.insert(END, "You: " + msg + '\n\n')
ChatLog.config(foreground="#442265", font=("Verdana", 12 ))

res = chatbot_response(msg)
ChatLog.insert(END, "Bot: " + res + '\n\n')

ChatLog.config(state=DISABLED)
ChatLog.yview(END)

base = Tk()
base.title("Hello")
base.geometry("400x500")
base.resizable(width=FALSE, height=FALSE)

#Create Chat window
ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)

ChatLog.config(state=DISABLED)

#Bind scrollbar to Chat window
scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")
ChatLog['yscrollcommand'] = scrollbar.set

#Create Button to send message
SendButton = Button(base, font=("Verdana",12,'bold'), text="Send", width="12", height=5,
bd=0, bg="#32de97", activebackground="#3c9d9b",fg='#ffffff',
command= send )

#Create the box to enter message
EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")
#EntryBox.bind("<Return>", send)


#Place all components on the screen
scrollbar.place(x=376,y=6, height=386)
ChatLog.place(x=6,y=6, height=386, width=370)
EntryBox.place(x=128, y=401, height=90, width=265)
SendButton.place(x=6, y=401, height=90)

base.mainloop()
```

Chapter 6

Result & Discussion

6.1 Result from testing

The first participant enjoyed talking to the bot, but stressed the fact that you had to "talk like a dummy" for it to understand what you were asking. The participant pointed out that this really would have come in handy in his second week after the university exams , as he didn't always know how to ask - especially if he was in a hurry. He pointed out that the prototype needs to get more features like tell your Symptoms , body temperature , health problems. The second participant was a bit frustrated that the chatbot wasn't flexible enough. "I don't like having to guess what questions chronology to follow". He would like more instructions to know how to get more out of the chatbot. The third participant had also problems with understanding how the chatbot could help us . When given a hint for what the chatbot could do, the chatbot did not function much properly. Here we tried to restart the system and then the chatbot displayed it´s welcome message what it could do. Afterwards it was more clear what the participant could ask it, but the chatbot did not always give the response that the participant is diagnosed with .

6.2 Re-design of the prototype

This findings gave us a lot of insight in where the chatbot needed to be changed. e.g. adding a proper welcome message, defining the chatbots' limitations and presenting this to the user, segregating medical symptoms for various diseases and health problems. According to the paper published by Luger & Sellen (2016) argues that it's important to define goals and expectations so that your chatbot has a clear purpose. Knowing the capabilities and limitations of the system, before it crashes.

The test showed that it was hard to ask the ‘right’ questions, we therefore added more ‘AI ques’ to simplify the interaction. We also used the principles for designing conversational agents. When talking about User-centred design of AI there are three (tentative) design principles: learning, improve and fuelled by large data sets (Folstad, 2018). The principle of learning is how the system is designed for change. Setting the expectations right, with the system's ability to perform and its ever changing nature. The principle of improve is how the system should be designed with ambiguity. The system is more than likely to make mistakes, so learning from these are an important principle to improve the system. The principle fuelled by large data sets is how the system is reliant on getting access to enough data.

Chapter 7

Conclusion, Limitation & Future Scope

When testing the last prototype we got findings suggesting that the participants did not have a problem with getting information from a chatbot instead of a human. The information that they got was not seen as less trustworthy, this could be supported by the fact that the chatbot provided a source for the information it gave. It has been interesting to investigate how the participants interacted with the chatbot and how they reported on it afterwards. Our findings have some indicators leading towards that a chatbot could be a good alternative for acting as a helpful friend for freshmans at a new school. Still we have to stress the fact that the chatbot was not very intelligent and that the evaluators had to adjust their language to match the chatbots. Because of the scope of the project we did not have time to conduct as much user testing and re-design to the chatbot as we would have liked. This has an impact on the validity of our research. Through the project we have touched on some theory when making the chatbot. Even though the participants trusted the information given in this project we cannot say that people trusts a chatbot as much as they trust a human being. There are also biases in our project, one of them is that all the students that we included in the project already knew a lot of the answer the prototype could provide. Another bias is that the information the chatbot provides could be seen as “casual” and are not crucial and/or vitals regarding health diagnosis .This could have had an impact on the results regarding trustworthiness. With that being said we also think that some of our findings could give some insights into how a very small group of people think about using a chatbot to gain information in a medical context. Some of the characteristics of our chatbot was viewed as appropriate for the given context, like “casualness” and links to where the information was gathered. If the MEDIC HERO 4 chatbot is to be furthered developed, this could be something to draw upon.

References

1. Hung, V., Gonzalez, A., & DeMara, R. (2009, February). Towards a context-based dialog management layer for expert systems. In Information, Process, and Knowledge Management, 2009. eKNOW'09. International Conference on (pp. 60-65). IEEE.

2. Jung, M., Hinds, P., 2018. Robots in the Wild: A Time for More Robust Theories of Human-Robot Interaction. ACM Trans. Hum.-Robot Interact. 7, 2:1–2:5.

3. Lazar, J., Feng, J. H., & Hochheiser, H. (2017). Research methods in human-computer interaction. Morgan Kaufmann.

4. Lee, J. D., & See, K. A. (2004). Trust in automation: Designing for appropriate reliance. Human factors, 46(1), 50-80.

5. Lewicki, R. J., & Bunker, B. B. (1995). Trust in relationships. Administrative Science Quarterly, 5(1), 583-601.

6. Lindblom J., Andreasson R. (2016) Current Challenges for UX Evaluation of Human-Robot Interaction. In: Schlick C., Trzcieliński S. (eds) Advances in Ergonomics of Manufacturing: Managing the Enterprise of the Future. Advances in Intelligent Systems and Computing, vol 490. Springer, Cham

7. Luger, E., & Sellen, A. (2016, May). Like having a really bad PA: the gulf between user expectation and experience of conversational agents. In Proceedings of the 2016 CHI Conference on Human Factors in Computing

Systems (pp. 5286-5297). ACM.

8. Winograd, T. (1991). Thinking machines: Can there be? Are we (Vol. 200). University of California Press, Berkeley. (p.204-210).
9. Schank, R. C. (1987). What is AI, anyway?. AI Magazine, 8(4), 59.

10. Brandtzaeg, P. B., & Følstad, A. (2017). Why people use chatbots. In I. Kompatsiaris, J. Cave, A. Satsiou, G. Carle, A. Passani, E. Kontopoulos, S. Diplaris, & D. McMillan (Eds.), Internet Science: 4th International Conference, INSCI 2017 (pp. 377-392).

11. Minock, M. (2005): Where are the "'Killer Applications' of Restricted Domain Question Answering.

12. Molla, D. & Vicedo, J.L. (2006). Question Answering in Restricted Domains: An Overview.