“Hello Chatbot”

Project Report Submitted by:

BHAVANA R [20BSR18008]

SHREE SHANGAAVI N [20BSR18030]

Y ARCHANA [20BSR18039]

JEEVAN S [20BSR18014]

R INDRA PAVAN [20BSR18027]

Under the guidance of Mr. KUNAL DEY

Towards the partial fulfilment for the degree of B.Sc. (Data Science and Analytics)

To

Department of Data Science and Analytics School of Sciences

JAIN (Deemed-to-be University) Bangalore

2023

Department of Data Science and Analytics, School of Sciences, JAIN (Deemed-to-Be University), Bangalore.

CERTIFICATE

This is to certify that the present Project titled **"Project Centric Learning**" has been the outcome of an original study carried out under the supervision of **Mr. Kunal Dey** towards the partial fulfillment of the requirements for the degree of B.Sc. Data Science and Analytics of the JAIN (Deemed-to-be University).

Is the approved record of project done by:
Bhavana R[20BSR18008]:

Shree Shangaavi N[20BSR18030]:

Y Archana[20BSR18039]:

Jeevan S[20BSR18014]:

R Indra Pavan[20BSR18027]:

TABLE OF CONTENTS

ABSTRACT

A Chatbot is a software which is developed for purpose of interaction between human and computers/systems. Which is implemented in a natural languages like human chats. Chatbot software takes the input by a client for example a question and gives output which is answer, it makes client to feel that they are chatting with a human while they are chatting with computer/system. Chatbot takes all kind of queries of its client and keep on answering. The use of chatbot is in every field and its use is constantly growing.

We're moving forward with the data science chatbot. The employment of natural language to promote student interaction with information systems has been shown to be advantageous in this paper. The goal of this project is to investigate the use of cognitive computing in blended learning settings. We present a modular cognitive agent architecture for instructional question answering that includes social interaction (small talk) and is tailored to a particular knowledge domain. This system serves as a personal assistant for students learning Data Science and Machine Learning skills. Its implementation entails the use of a human-like interface to train machine learning models and natural language comprehension algorithms. An experiment was conducted to verify the system's efficiency.

INTRODUCTION

A Chatbot is a software which is developed for purpose of interaction between human and computers/systems. Which is implemented in a natural languages like human chats. Chatbot software takes the input by a client for example a question and gives output which is answer, it makes client to feel that they are chatting with a human while they are chatting with computer/system. Chatbot takes all kind of queries of its client and keep on answering. The use of chatbot is in every field and its use is constantly growing. We're moving forward with the data science chatbot.

The employment of natural language to promote student interaction with information systems has been shown to be advantageous in this paper. The goal of this project is to investigate the use of cognitive computing in blended learning settings. We present a modular cognitive agent architecture for instructional question answering that includes social interaction (small talk) and is tailored to a particular knowledge domain. This system serves as a personal assistant for students learning Data Science and Machine Learning skills. Its implementation entails the use of a human-like interface to train machine learning models and natural language comprehension algorithms. An experiment was conducted to verify the system's efficiency.

The use of chatbots evolved rapidly in numerous fields in recent years, including Marketing, Supporting Systems, Education, Health Care, Cultural Heritage, and Entertainment. In this paper, we first present a historical overview of the evolution of the international community's interest in chatbots. There are numerous electronic organizations like Ebusiness, Entertainment, Virtual assistance and some more. Everything in this generation is getting related with the web. It's extremely efficient to utilize approach to manage benefit everything at your doorstep. The chatbots are sufficient to fool the users in believing that they're talking to a human being, they've a very limited knowledge base at runtime and have no means to keep track of all the conversations. Chatbots uses machine learning to reach AI for helping them to understand the user queries/doubts and provide the user with an appropriate response.

We looked at a number of research papers on chatbots and their applications in many fields. We learned about numerous chatbots and their many uses in various fields by doing a literature survey. We've learned about the history, technology, and applications of chatbots, which are conversational tools: chatbot

-Analysis of Intelligent Chatbots: Current State and Future Research Directions

-The constraints and opportunities of building chatbots from large-scale domain-specific knowledge bases

-An Overview of Chatbot Technology -The Impact of AI-based Chatbots in Customer Service on User Compliance

-Chat-Bot-Kit: A web-based tool for simulating text-based interactions between people and machines.

-An educational institute's research paper on chatbot development.

-Dialogue Learning in End-to-End Trainable Task-Oriented Dialogue Systems with Human Teaching and Feedback

-JAICOB: A Chatbot for Data Science

-Deep Learning ChatbotForCovid Vaccine.

The research publications to which we have referred are listed above. We've decided to go ahead with a Data Science chatbot. This chatbot will aid in the understanding of Data Science and aspects related. It will also perform some fundamental data science tasks.

LITERATURE SURVEY

Research paper- 01

Chatbots: History, technology, and applications

EleniAdamopoulou, LefterisMoussiades
Department of Computer Science, International Hellenic University, AgiosLoukas, 65404 Kavala, Greece

The use of chatbots evolved rapidly in numerous fields in recent years, including Marketing, Supporting Systems, Education, Health Care, Cultural Heritage, and Entertainment. In this paper, we first present a historical overview of the evolution of the international community's interest in chatbots. Next, we discuss the motivations that drive the use of chatbots, and we clarify chatbots' usefulness in a variety of areas. Moreover, we highlight the impact of social stereotypes on chatbots design. After clarifying necessary technological concepts, we move on to a chatbot classification based on various criteria, such as the area of knowledge they refer to, the need they serve and others. Furthermore, we present the general architecture of modern chatbots while also mentioning the main platforms for their creation.

Chatbots, also called chatterbots, is a form of artificial intelligence (AI) used in messaging apps. This tool helps add convenience for customers—they are automated programs that interact with customers like a human would and cost little to nothing to engage with. It uses Natural Language Processing (NLP) and sentiment analysis to communicate in human language by text or oral speech with humans or other chatbots Artificial conversation entities, interactive agents, smart bots, and digital assistants are also known as chatbots. Chatbots are more friendly and attractive to users than, for example, the static content search in frequently asked questions (FAQs) lists. They offer users comfortable and efficient assistance when communicating with them; they provide them with more engaging answers, directly responding to their problems human–chatbot communication has noticeable differences in the content and quality in comparison to the human–human discussion. The duration of a human–chatbot conversation is long.
ELIZA is an early natural language processing computer program created from 1964 to 1966Created to demonstrate the superficiality of communication between humans and machines, Eliza simulated conversation by using a "pattern matching" and substitution methodology that gave users an illusion of understanding on the part of the program, but had no built-in framework for contextualizing events. A chatbot can access a range of knowledge, which determines its Knowledge Domain. Chatbots that can answer any user question from whichever domain are called Generic chatbots.

There are two approaches in developing a chatbot depending on the algorithms and the techniques adopted: pattern matching and machine learning approaches.

Rule-based chatbots match the user input to a rule pattern and select a predefined answer from a set of responses with the use of Pattern Matching algorithms.

In most rule-based chatbots for single-turn communication, the answer is selected, taking into account only the last response. Chatbots use Natural Language Understanding to retrieve context from the unstructured user input in human language and respond based on the current user's intention. Artificial Neural Networks Retrieval and Generative-based chatbots use several kinds of Artificial Neural Networks. The system takes the user's input, computes its vector representations, feeds it as features to the neural network, and produces the response. General architecture an appropriate chatbot architectural design is useful to the study of chatbots and the aspiring chatbots creator. Developers of chatbots use Recurrent Neural Networks to take account of the previous context in a conversation. In RNN, the new user input and information of earlier data feed the neurons. In this way, a loop passes knowledge from one part of the network to the next. The operation of the chatbot begins when it receives the user's request through an application using text or speech input, such as a messenger application like Facebook, Slack, WhatsApp, WeChat, Viber, or Skype.

The chatbot retrieves the information needed to fulfil the user's intent from the Backend through external APIs calls or Database requests. Once the appropriate information is extracted, it is forwarded to the Dialog Management Module and then to the Response Generation Module. When Rule-based chatbots are used, there is a Knowledge Base. It includes a list of hand-written responses that correspond to the user's inputs. A Relational Database may be used so that the chatbot can recall past conversations, making in this way the communication more consistent and relevant. The Response Generation Component produces responses using one or more of three available models: Rule-based, Retrieval based, and Generative-based models. The Rule-based model selects the response from a set of rules without generating new text responses. The Dialog Management component passes the placeholder values that may be needed to fill the template for the response to the Response Generation Module. Developing the chatbot using a programming language or a chatbot development platform follows, and afterward, the chatbot functionalities are tested locally. When designing a chatbot, we must first determine the goals that it will serve. Based on the objectives, we will evaluate the primary character of the chatbot, for example, if we need a generic, cross domain, or closed-domain chatbot. In the first two cases, we will probably need to use NLP techniques. Before deciding, we should take into account whether the necessary data for chatbot training is available. Conversely, for closed-domain chatbots, the use of a scripting language may be preferable. In general, it is not easy to find suitable training data for specific purposes. Customers are quite familiar with communicating with companies using their phones, email, newsletters, or websites while they use messengers' applications mainly for their private communication.

Applications of chat bots are: Chatbots are a competitive advantage because they generate leads and respond to questions all the time. By interacting and providing answers in real-time.

- Education environments
- Customer service

- Health
- Robotics
- Industrial use cases

In recent years there has been a significant improvement in the development and use of chatbots with substantial benefits in many

domains. At customer service canter's, they work 24 h a day, 7 days a week, while managing many customers concurrently, improving payroll costs dramatically. Similarly, in education, they support an increased number of students to whom they provide educational content and personal assistance. In some cases, as in the case where they reduce language anxiety to foreign language students, they even surpass human-teachers. In the field of health care, they provide patients with various services. They also contribute to the development of natural language interfaces for the benefit of Robotics.

Research paper-02

A Tool of Conversation: Chatbot

By M. Dahiya

Dept. of Computer Science, Maharaja Surajmal Institute, Janakpuri, India.

The main purpose of paper is to address the issues related to the design and implementation of a Chatbot system for a text input.

In today's world computers play an important role in our society as they give information; entertain and help in lots of manners. A Chatbot is a program designed to counterfeit a smart communication on a text or spoken ground. But this paper is based on the text only Chatbot. Chatbot recognize the user input as well as by using pattern matching, access information to provide a predefined acknowledgment. When the input is available being in the database, a response from a predefined pattern is given to the user. A Chatbot is implemented using pattern comparing, in which the order of the sentence is recognized and a saved response pattern is acclimatized to the exclusive variables of the sentence. A Chatbot refers to a chatting robot. It is a communication simulating computer program. It is all about the conversation with the user. The conversation with a Chatbot is very simple. It answers to the questions asked by the user.

The following tools have been used to create Chatbot. Windows OS is used as it is user friendly. Eclipse Software is used for Programming as it contains basic workspace and is mostly used for Java applications. The Chatbot program is written in Java. The Chat is created using a pattern which is known to the user and the dialog box is created using Java applets. The pattern matching technique of Artificial Intelligence is used to design to Chatbot. The basic purpose of the Chatbot is to answer the questions asked by the user if the same is available in the Database. The Chatbot interacts with the user in Basic English language and it seems like talking to another person.

For the purpose of implementation of the Chatbot, Artificial Intelligence (AI) is used to mimic human conversation using Java Programming language. Java Applets are primarily used for creating the Dialog box required for the conversation between the user and the bot. The details of the implementation methodology are given in the following steps.

1.Creating the Dialog box. First all the required packages are imported. The size of the dialog box and text area inside the dialog box is clearly defined. Only the vertical scrollbar is used to scroll the screen as the conversation goes on. Since the size of the dialog box is fixed, Horizontal scrollbar is never used.

2.Creating a Database. A two dimensional array is used to create a Database. Rows in the array are used for questions of the user and the corresponding replies in the even and odd rows respectively. Columns in the array are used to save the different types of queries that the user may ask and the corresponding response. One default response is also stored for giving a response to a query not found in the Database.

3.The modules used in the implementation of the Chatbot are:

i.Chatbot () function is used to define all the variables used for creating the dialog box.

ii.Text () function is used to take the input from the user. Trim () function removes all the punctuation marks in the input. All the uppercase letters are converted into lower case. A variable called Response is used to store if the match for the input is found in the database or not. Random () function is used to choose the response saved in the Database.

iii.AddText () functions is used for adding input and output to the text area in the dialog box.

iv. InArray () function is used for pattern matching. A variable match is used to hold a Boolean value and it is set to false. If the match for the users input is found in the database, true is returned else false is returned as a result. This value is returned to keyPressed () function and the result is displayed in the dialog box.

The Chatbot is very simple and user friendly. Compared to other Chatbot where the working is very complicated, the working of this Chatbot is very simple and easy to understand. This Chatbot uses simple pattern matching to represent the input and output compared to the usage of

input rules, keyword patterns and output rules in other Chatbot. Based on the developer or the user, the required requests and responses can be stored in the database. Since own database can be created, it allows the user to understand how the response is generated. This Chatbot can be used for the entertainment purpose. Whenever a person is bored, he can chat with the bot for entertainment. It can also be used to provide information by modifying the program as needed by the user.

Chabot's are also referred to as virtual assistants which are rudimentary form of artificial intelligence software that can mimic human conversation. The Chatbot can be analysed and improved and can be used in various fields such as education, business, online chatting etc. It can be used in the field of education as a learning tool. In business field, it can be used to provide business solutions in an efficient way. When the solutions are efficient, the business can be improved and the growth of the organization will increase. This Chatbot can be used in online chatting for entertainment purpose. These bots can also be used to learn different kinds of language. They can also be used in the field of medicine to solve health related problems. Chatbot are only going to explode and can be really dominating in future. Chatbot can provide a new and flexible way for users. They are giving a tool to AI to enable it do something better. It is seen that Chatbot results in smart conversation and is advancing at an unprecedented rate with each new development. Chatbot usually store contextual data which can be used in the detection of geo location or a state.

To conclude a Chatbot is one of the simple ways to transport data from a computer without having to think for proper keywords to look up in a search or browse several web pages to collect information; users can easily type their query in natural language and retrieve information. Chatbot is a great tool for quick interaction with the user. They help us by providing entertainment, saving time and answering the questions that are hard to find. In this project, we looked into how Chatbot are developed and the applications of Chatbot in various fields. General purpose Chatbot must be simple, user friendly, must be easily understood and the knowledge base must be compact.

Research paper- 03

Survey on Intelligent Chatbots: State-of-the-Art and Future Research Directions

Ebtesam H. Almansor and FarookhKhadeerHussain Faculty of Engineering and Information Technology,University of Technology Sydney, Sydney, Australia.

Human-computer interaction (HCI) is a field of study that is important for understanding how humans and machines interact. Chatbots, voice control interfaces, and personal assistants are all

examples of HCI applications that have been designed to engage with users using natural language. Customers can use chatbots to find information that is relevant to their requirements. As a result, many businesses are turning to chatbots to automate their customer care. As a result, the demand for artificial intelligence has grown in response to the growing demand for automated services. However, creating intelligent bots that can reply in a human-like manner is difficult.

Human-computer interaction (HCI) is a field of study that is important for understanding how humans and machines interact. Chatbots, voice control interfaces, and personal assistants are all examples of HCI applications that have been designed to engage with users using natural language. Customers can use chatbots to find information that is relevant to their requirements. As a result, many businesses are turning to chatbots to automate their customer care. As a result, the demand for artificial intelligence has grown in response to the growing demand for automated services. However, creating intelligent bots that can reply in a human-like manner is difficult. In this study, we examine current chatbot systems from the standpoint of their capacity to generate suitable responses.

Alan Turing (1950) devised the "imitation" game or the Turing test, which inspired the concept of a chatbot. The goal of this game was to see if a computer could emulate human behavior. In 1966, ELIZA, the first chatbot, was invented. However, this bot was a basic system that lacked the ability to continue a conversation between humans and bots. It used keyword matching and minimal context identification. The ALICE (Artificial Linguistic Internet Computer Entity) chatbot was developed in the 1980s. Because of its use of the Artificial Intelligent Markup Language (AIML), this bot was deemed important. AIML was created with the goal of defining the pattern-matching rules that connect user-submitted words and sentences.

With the growth of AI technology, NLP researchers aim to build automatic conversation agents that respond to humans in a suitable time. There are two categories of conversation agents, namely task-oriented and non-task-oriented chatbots.

Task oriented dialogue system is a system that helps users achieve their goals or complete a specific task. For example, this system is widely used across in industry such as Facebook, Apple, and Microsoft. Under task-oriented there are two more topics which are supervised approach, unsupervised approach

•Supervised Approach: the supervised approach depends on feature extraction and annotated datasets are also which are limitations to its features incur high costs and have poor scalability.

•Unsupervised approach: The unsupervised approach learns features automatically from datasets. The most effective approach is a deep learning approach that trained end-end neural networks. The main advantage of this model is that the conversation can be developed without any need to use handcraft features to make assumptions.

A non-task-oriented dialogue system provides users with the means to participate in different domains such as a game, chitchat or entertainment, without providing 538 E. H. Almansor and F. K. Hussain the user with any help to complete any task in a specific job. An example of this system is the chatbot which chats with the user in a similar way to a human and provides reasonable and relevant responses [3,31,32]. ELIZA is a Chabot that uses a combination of rules and patterns. In 1981, another chatbot was developed using simple text parsing rules to construct the dialogue system, PARRY. Both systems did not use data for learning purposes. Therefore, a data-driven approach has been proposed to overcome some of the limitations of the non-task-oriented dialogue system. This approach enables chatbots to learn from the massive amount of available conversations on social media or the Web2.0, which enhances the communication between humans and chatbots.

The developed methods are either retrieval-based or generation-based. Retrievalbased models can obtain response candidates from a pre-built index, rank the candidates then chose the response from the top-ranked ones [3,32].On the other hand, generation-based methods use natural language generation (NLG) to selectthe response

Retrieval based chatbot : A retrieval-based chatbot uses a selection of responses from ranking data rather than generating a new response. A dialogue system with a knowledge base that contains a considerable number of question-answer pairs was developed based on a statistical language model to select suitable responses.

•Generation-Based Chatbot : A generation-based chatbot generates responses instead of selecting them from the underlying mode

In this paper, we briefly get to know the introduction of Chatbot. Then the classification of chatbots of existing approaches and types of chatbots were explained. As most of the research focuses on developing and challenges of improving the chatbots. There is a needfor standard measurements to measure the quality of chatbots. Deep learning is a promising approach that could be applied to develop more effective chatbots. We get to know from this research how the demand for chatbots across various organizations and various use of it and how it can be developed.

Research paper- 04

Building chatbots from large scale domain-specific knowledge bases: challenges and opportunities

WalidShalaby, Adriano Arantes, Teresa GonzalezDiaz, Chetan Gupta Hitachi America Ltd. Santa Clara, CA, USA

The main stress of this research paper is to demonstrate the Development of a Chatbot with a large volume of domain specific entity and show how it is better than the popular conversational forms available like Alexa Skills and Kit (ASK) and Google Actions (gActions). Virtual Assistants provide users with natural language interface through which they can ask questions or commands with their voice which is interpreted in the backend Artificial Intelligence (AI) services. While the Digital Assistant Technology is widely adopted in the consumer domain, the usage in the Industrial and enterprise scenarios is negligible. The Key challenge is the conversational Language Understanding (LU) which means to correctly interpret the intent and extract the elements of interest (entities) in the speech. Building high quality LU requires both the; lexical knowledge meaning access to general common sense and domain specific knowledge and ; Syntactic and semantic knowledge of all possible ways in which the users will utter their questions or commands.

The two popular LU tools which are Alexa Skills Kit (ASK) and Google Actions (gActions) come with Developer friendly web interface to allow the Developers to build the conversation flow. When these are used for general entity types like City names and airports very little integration is required. However the Developers have to provide 3 basic inputs i.e.

1)Provide Sample utterances as representing entity placeholders.

2)Provide a dictionary of entity values for each domain specific entity type

3)Customize the flow of responses and interactions.

The performance of LU engines when there are a large volume of domain specific entities which is unclear.

This paper describes the problem and challenges faced by focussing on understanding and responding to vehicle related problem which can be initiated either by the Driver or Technician. The three important questions that were put to test were; 1. How to facilitate acquisition of domain specific knowledge about entities? 2. How much knowledge off-the-shelf frameworks can digest effectively? 3. How accurately these entities built in LU engines can identify entities in user utterances? As there were limitations in scalability and accuracy in ASK and gActions, an alternative scalable pipeline was developed for; 1) extracting the knowledge about equipment components and their associated problems entities, and 2) learning to identify such entities in user utterances. It has been shown through evaluation on real dataset that the proposed

framework understanding accuracy scales better with large volume of domain-specific entities being up to 30% more accurate.

In the model user utterance is firstly transcribed into text using the Automatic Speech Recognition (ASR) module. Then, the LU module identifies the entities (component, problem) in the input. Afterwards, the parsed input is passed to the dialog manager which keeps track of the conversation state and decides the next response (e.g., recommended repair) which is finally uttered back to the user using the language generation module. The proposed framework automatically constructs a KB of equipment components and their problem entities with "component problem" relationships. The rules require less domain knowledge and should yield higher coverage. The constructed KB is expanded through two steps: 1) reorganizing the extracted vocabulary of components into a hierarchy using a simple traversal mechanism introducing relationships, and 2) aggregating all the problems associated with subtype components in the hierarchy and associating them with super type components introducing more relationships. Unsupervised curation and purification of extracted entities is another key differentiator of the proposed framework compared to prior work. The proposed framework utilizes a state-of-the-art deep learning for sequence tagging to annotate raw sentences with component(s) and problem(s). A Pipeline for KB Extraction Existing Chatbot development frameworks require knowledge about target entities which would appear in user's utterances. For each entity type (e.g., component, problem, etc.), an extensive vocabulary of possible values of such entities should be provided by the virtual assistant developer. These vocabularies are then used to train the underlying LU engine to identify entities in user utterance. Nevertheless, the design of the proposed framework is flexible and generic enough to be applied to several other maintenance scenarios of different equipment given a corpus with mentions of the same target entities. The proposed KB construction system is organized as a pipeline. It starts with a domain-specific corpus that contains target entities. The corpus is processed through five main stages including pre-processing, candidate generation using POS-based syntactic rules, embedding-based filtration and curation, and enrichment through training a sequence-to-sequence slot tagging model.

In the case of equipment troubleshooting, service and repair records and QA posts include complaint, diagnosis, and correction text which represent highly rich resources of components and problems that might arise with each of them. These records are typically written by technicians and operators who may not be good at language and also have shortage of time. Hence the text may have typographical and other language errors and also may have domain specific jargon and abbreviations. Hence pre-processing is required to reduce such inconsistencies and avoid inappropriate corrections. For Candidate generation a set of syntactic rules based on POS tags of complaint utterances after all the sentences have been extracted and parsed using a standard Library. The rules are defined considering only the most frequent POS patterns. In the next curation stage incorrect and noisy entities are removed which might have arisen due to noisy nature of the text. Different pruning techniques based on Statistical,

Linguistic, Embedded or Sentiment are used. In the slot tagging which is called S2Tagger a Machine Language model is built to identify chunks of tokens that correspond to target entities. S2STagger uses an encoder-decoder Recurrent Neural Network Architecture where during the encoding phase all sentences are processed sequentially and encoded as fixed length vectors capturing all the semantic and syntactic structures. During the decoding these Vectors produces a sequence of IOB tags and at this stage an attention mechanism is used to learn as to which input is more relevant for each output tag. At the final stage the Knowledge Base (KB) is enriched with new entities which have not been explicitly mentioned earlier. These primarily come from three sources.

1. S2STagger is used to tag all the newly unseen utterances. 2. In component backward traversal a hierarchy of super type components are created and a relationship is built between the subtype and super type component thus enriching the Knowledge base.

3. In problem aggregation a simple aggregation method is used to automatically associate between super type components and problems in their subtype. The entities from the Curation stage are consolidated along with the new entities discovered at each of the three steps to create a Knowledge Base (KB) of components and problems entities.

To evaluate the effectiveness of this model the data from two datasets available in the automotive sector are taken as reference. First is a dataset of Questions and Answers (QA) from the public Mechanics Stack Exchange QA forum and the second is another subset of questions related to cars maintenance from the Yahoo QA dataset. A general analysis of the Dataset reveals that the coverage of syntactic rules is noticeably low and the need for a learning-based entity extractor, such as the proposed S2STagger model to harness the knowledge from utterances not matching the pre-defined rules. A labelled Dataset for S2STagger has been created using the question utterances and tagging utterances with entities mentioned in the curated entity pool. The main pain point in creating voice-enabled agents is the definition of the vocabulary users will use to communicate with the system which has a large number of entities and are very costly and time consuming to define manually. The framework greatly facilitates this step through the construction of KBs with target entities which could be readily available to build voice-enabled agents. To test the effectiveness of the above model it is compared to two models using off-the-shelf NLU technologies including:

1) a skill (AlexaSkill) using Amazon Alexa Skills kit.

2) an agent (DiagFlow) using Google Dialog Flow.

The utterance structures defined in both the models equivalent to the syntactic rules structures. Same curated entities in the KB were fed to both the models as slot values and entities for AlexaSkill and DiagFlow respectively. The third model is S2STagger trained on all the tagged utterances in the QA dataset. The training utterances for S2Stagger were the same from which KB entities and utterance structures were extracted and fed to both AlexaSkill and DiagFlow.

The utterances were chosen such that three aspects of the model are assessed. Specifically, to quantitatively measure the model accuracy on utterances with:

1)Same syntactic structures and same entities as in the training utterances (119 in total).

2)Same syntactic structures but different entities from the training utterances (75 in total).

3)Different syntactic structures but same entities as in the training utterances (20 in total).

The accuracy in vehicle related components for the 119 same structures same entities type was 70% in AlexaSkill, 39% in DiagFlow while it was 93% in S2STagger. In case of the 75 same structures different entities the accuracy was 53% in AlexaSkill, 9% in DiagFlow while it was 89% in S2Stagger. This indicates that the proposed model is more lexical-agnostic and can generalize better than the other two models. AlexaSkill model comes second, while DiagFlow model can only tag few utterances correct, indicating its heavy dependence on exact entity matching and limited generalization ability. On the other hand, the three models seem more sensitive to variation in utterance syntactic structure than its lexical variation. Further detailed analysis of the results show that S2STagger can successfully tag new entities when they appear in similar to training structures; it fails to recognize same entities when they appear in slightly different structure demonstrating the ability of S2STagger to generalize to unseen entities better than unseen structures. It also demonstrates how the other two models seem to depend heavily to lexical matching causing them to fail to recognize mentions of new entities.

To conclude, the results demonstrate the superior performance of the proposed knowledge construction pipeline including S2STagger, the slot tagging model, over popular systems such as ASK and DiagFlow in understanding vehicle-related complaints. One important and must have additional feature to increase the effectiveness of S2STagger and off-the-shelf NLU systems would be to handle utterances which are different from training structures.

Research paper- 05

An Overview of Chatbot Technology

EleniAdamopoulou and LefterisMoussiades

Department of Computer Science, International Hellenic University, AgiosLoukas,

65404 Kavala, Greece.

Most chatbots are powered by artificial intelligence, or AI.AI chatbots tend to be more useful, purely because they are smarter and can learn over time. This is, of course, valuable to businesses. Artificial intelligence in chatbots comes in many forms. The most common are natural language processing (NLP) which powers the language side of the chatbot, to machine learning (ML) which powers data and algorithms. Chatbots can mimic human conversation and entertain users but they are not built only for this. They are useful in applications such as education, information retrieval, business, and e-commerce.

The Turing test was developed by Alan Turing (Computer scientist) in 1950. He proposed that the Turing test is used to determine whether or not a computer(machine) can think intelligently like humans. The Turing Test judges the conversational skills of a bot. According to the test, a computer program can think if its responses can fool a human into believing it, too, is human. Not everyone accepts the validity of the

Turing Test, but passing it remains a major challenge to developers of artificial intelligence. The Turing Test judges the conversational skills of a bot. According to the test, a computer program can think if its responses can fool a human into believing it, too, is human.

Not everyone accepts the validity of the Turing Test, but passing it remains a major challenge to developers of artificial intelligence. Chatbots seem to hold tremendous promise for providing users with quick and convenient support responding specifically to their questions. The most frequent motivation for chatbot users is considered to be productivity, while other motives are entertainment, social factors, and contact with novelty. However, to balance the motivations mentioned above, a chatbot should be built in a way that acts as a tool, a toy, and a friend at the same time. The reduction in customer service costs and the ability to handle many users at a time are some of the reasons why chatbots have become so popular in business groups. Chatbots are no longer seen as mere assistants, and their way of interacting brings them closer to users as friendly companions. According to a study, social media user requests on chatbots for customer service are emotional and informational, with the first category rate being more than 40% and with users not intending to take specific information. Machine learning is what gives the capability to customer service chatbots for sentiment detection and also the ability to relate to customers emotionally as human operators do.

Some fundamental concepts related to chatbot technology:

- Pattern Matching
- Artificial Intelligence Markup Language
- Latent Semantic Analysis
- Chat script
- Rive Script
- Natural Language Processing
- Natural Language Understanding

Chatbots can be classified using different parameters: the knowledge domain, the service provided, the goals, the input processing and response generation method, the human-aid, and the build method. Classification based on the knowledge domain considers the knowledge a chatbot can access or the amount of data it is trained upon.

Open domain chatbots can talk about general topics and respond appropriately, while closed domain chatbots are focused on a particular knowledge domain and might fail to respond to other questions. Classification based on the service provided considers the sentimental proximity of the chatbot to the user, the amount of intimate interaction that takes place, and it is also dependent upon the task the chatbot is performing. Interpersonal chatbots lie in the domain of communication and provide services such as Restaurant booking, Flight booking, and FAQbots. They are not companions of the user, but they get information and pass them on to the user. They can have a personality, can be friendly, and will probably remember information about the user, but they are not obliged or expected to do so. Intrapersonal chatbots exist within the personal domain of the user, such as chat apps like Messenger, Slack, and WhatsApp.

They are companions to the user and understand the user like a human does. Inter-agent chatbots become omnipresent while all chatbots will require some inter-chatbot communication possibilities. The need for protocols for inter-chatbot communication has already emerged.

Alexa-Cortana integration is an example of inter-agent communication.

Another classification for chatbots considers the amount of human-aid in their components. Human-aided chatbots utilize human computation in at least one element from the chatbot. Crowd workers, freelancers, or full-time employees can embody their intelligence in the chatbot logic to fill the gaps caused by limitations of fully automated chatbots. While human computation, compared to rule-based algorithms and machine learning, provides more flexibility and robustness, still, it cannot process a given piece of information as fast as a machine, which makes it hard to scale to more user requests

Open-source platforms provide the chatbot designer with the ability to intervene in most aspects of implementation. Closed platforms, typically act as black boxes, which may be a significant disadvantage depending on the project requirements. However, access to state-of-the-art technologies may be considered more immediate for large companies. Moreover, one may assume that chatbots developed based on large companies' platforms may be benefited by a large amount of data that these companies collect.

The design and development of a chatbot involve a variety of techniques. Understanding what the chatbot will offer and what category falls into helps developers pick the algorithms or platforms and tools to build it. At the same time, it also helps the end-users understand what to expect.

Once a chatbot reaches the best interpretation it can, it must determine how to proceed. It can act upon the new information directly, remember whatever it has understood and wait to see what happens next, require more context information or ask for clarification.

Minimal human interference in the use of devices is the goal of our world of technology. Chatbots can reach out to a broad audience on messaging apps and be more effective than humans are. At the same time, they may develop into a capable information-gathering tool. They provide significant savings in the operation of customer service departments. With further development of AI and machine learning, somebody may not be capable of understanding whether he talks to a chatbot or a real-life agent.

Further work of this research would be exploring in detail existing chatbot platforms and compare them. It would also be interesting to examine the degree of ingenuity and functionality of current chatbots. Some ethical issues relative to chatbots would be worth studying like abuse and deception, as people, on some occasions, believe they talk to real humans while they are talking to chatbots.

Research paper- 06

AI-based chatbots in customer service and their effects on user compliance.

Martin Adam & Michael Wessel & Alexander Benlian

Conversational software agents or chatbots, which are technologies meant to converse with human users using natural language and are often based on artificial intelligence, are frequently replacing human chat service operators nowadays. Despite the fact that cost- and time-saving benefits prompted broad adoption of AI-based chatbots, they typically fail to satisfy customer expectations, potentially making users less likely to comply with chatbot demands. We employ a randomised online experiment to test how verbal anthropomorphic design cues and the foot-in-the-door technique improve user request compliance, based on social response and commitment-consistency theory. Our findings show that both anthropomorphism and the requirement to be consistent improve the likelihood that users will comply with a chatbot's request for service feedback. Moreover, the results show that social presence mediates the effect of anthropomorphic design cues on user compliance.

Though rudimentary CAs have been there since the 1960s, the "second wave of artificial intelligence" has reignited interest in and increased devotion to the technology since it has prepared the way for systems that can interact with humans more naturally. We found that both verbal anthropomorphic design cues and the foot-in-the-door strategy increase user cooperation with a chatbot's request for service feedback in an online experiment with 153 participants. Second, we show that humans recognise CAs as a source of persuasive messages, and that the degree to which humans cooperate with artificial social agents is dependent on the communication tactics used between humans and chatbot.

Balance between service efficiency and service quality is a fundamental difficulty for customer service providers: Customer self-service provides a number of potential benefits, according to both studies and practitioners, including higher efficiency, lower costs, and a better customer experience. CAs, in the form of 24/7 electronic channels to help clients, offer to be quick, convenient, and cost-effective solutions. CAs, on the other hand, have the ability to actively influence service encounters and operate as surrogates for service professionals by accomplishing tasks that were previously completed by human service staff by emulating social actors. As a result, when compared to previous online service interactions, CAs can alleviate the loss of interpersonal engagement by creating feelings of social presence and personalization.

Individuals in compliance situations are under pressure to accurately comprehend, assess, and respond to a request in a short amount of time; as a result, they don't have enough time to form a fully developed rational judgement and instead rely on heuristics to evaluate the available options. People's natural desire to justify their initial acceptance to a simple request to themselves and others is used by the FITD approach. As a result, if people cooperate with an

initial request, they will form a bias that they must have found the request acceptable, making them more likely to consent to a similar or related future request.

When an individual's behaviour contradicts social standards, the danger of social exclusion often favours the latter, resulting in social bias and, as a result, non-rational decisions by the individual. Small initial requests are used by providers to exploit users' self-perceptions and elicit consistent behaviour when users are deciding whether or not to complete a larger, more accommodating request that they would not otherwise fulfil. The basic rationale is that users have an intrinsic desire to keep their past views and actions. Consistent behaviour after previous acts or statements is especially common when users' participation in the request is low and the individual cannot trace the first consent to external sources.

While people strive to be consistent to serve personal requirements while there are few anthropomorphic design cues, this may change once more social presence is sensed. Individuals may sense external, social reasons to cooperate and look consistent in their behaviour in the increased social presence condition. This is in line with previous research into electronic market moderating effects. As a result, we believe that when the user senses more social presence, the foot-in-the-door impact will be greater.

According to the notion of social response. The evolutionary skewed social orientation of humans is the identified psychological effect underlying the computers-as-social-actors concept. As a result, people apply the same social rules to computers as they do to people: Even a few anthropomorphic design signals in computer interactions can elicit social orientation and perceptions of social presence in an individual, and consequently reactions that are socially desirable. Nonverbal ADCs, like physical appearance or embodiment, try to strengthen social connection by adopting motoric and static human traits, whereas verbal ADCs, such the ability to talk, aim to establish the perception of intelligence in a non-human technological agent.

AI-based CAs are becoming more common in a variety of scenarios, and they have the potential to save time and money. We used an online experiment to demonstrate that both verbal anthropomorphic design cues and the foot-in-the-door strategy can boost user compliance with a chatbot's request for service feedback. In the context of electronic markets and customer service, our research is a first step toward better understanding how AI-based CAs might promote user compliance by harnessing the impacts of anthropomorphism and the requirement to remain constant.

Research paper- 07

Chat-Bot-Kit: A web-based tool to simulate text-based interactions between humans and with computers.

Kyoko Sugisaki, German department, University of Zurich Schonberggasse 9 ¨ Zurich, Switzerland 8001.

This paper it describes Chat-Bot-Kit which a web-based tool for text-based chats that we designed for research purposes in computer-mediated communication. Chat-Bot-Kit enables to carry out language studies on text-based real-time chats for the purpose of research.

The generated messages are structured with language performance data such as pause and speed of keyboard-handling and the movement of the mouse. The tool provides two modes of chat communications – quasi-synchronm and synchrom modes– and various typing indicators. The tool is also designed to be used in wizard-of-oz studies in Human-Computer Interaction and for the evaluation of chatbots in Natural Language Processing.

Apps like Facebook Messenger, Snap-Chat, WeChat, or WhatsApp belong to the most installed apps worldwide, these includes test messaging which is a very popular form of communication which has different usage patterns, usage scenarios and overall user experiences. major tech companies have also started to over chatbot platforms that allow businesses to automate conversations with consumers and reach them where they spend a lot of their time, on Facebook Messenger alone, more than 300.000 chatbots have been deployed by Mid-2018. The text-based chat communication with computers is needed to be better understood, more actively designed and more frequently tested to make them more usable and valuable for users, let alone feel more natural.

There are a number of tools with which one could simulate chats and WoZ-studies. One could use a tool like Skype, for example. There are also some commercial chat tools that provide a log file that is useful for the purpose of research.

The submitted messages can be exported into a structured data format for the purpose of further analysis on the chat communication. The language performance data is embedded into the structure of messages. The tool automatically measures the pause, speed, rhythm of keyboard stokes and the movement of the mouse, next to the time stamp of the message submission. The temporal feature of the chat interaction is of relevance in CMC. A user can have more than one name and role: this is considered to be used in HCI. In case of a wizard-of-oz simulation study the identity of chat constructors plays an important role for the credibility of wizards as machine. The submitted messages can be edited, rated and commented directly in the user interface and the data is integrated into the output file. The feature is considered for HCI and NLP studies in which the designer or developer of a chatbot wants to test the system and carry out the user

evaluation on the level of messages. The tool is scalable: the system is based on AngularJS components that are easy to be extended to a wizard assistant tool or a chatbot.

The paper which they presented has web-based chat tool designed for the research in Computer-mediated communication, Human-Computer Interaction and Natural Language Processing. In future work they plan to extend several wizard assistance methods for wizard-of-oz studies in HCI that include machine learning and allow to iteratively train models during a study.

Research paper- 08

Research Paper on Chatbot Development for Educational Institute.

School of Computer Science Engineering and Technology DrVishwanathKarad MIT World Peace University Pune, India – 411038.

Chatbot: It is the software used for interaction between human and computer through natural language.

Benton and Radziwill (2017) described a chatbot as the medium of interacting with humans online, where as they're actually interacting with a computer software, put to reality by natural language input. Others define it as a computer program which imitates conversation with users, applying Artificial Intelligence.

This technology has made more Artificial Intelligence more complex. The chatbots are sufficient to fool the users in believing that they're talking to a human being, they've a very limited knowledge base at runtime and have no means to keep track of all the conversations. Chatbots uses machine learning to reach AI for helping them to understand the user queries/doubts and provide the user with an appropriate response. Chatbot is also known as answering engines.

Through using Artificial Intelligence, we can develop different types of chatbots, in this paper they have developed a College Enquiry chatbot. It has variety of fields like Enquiry process, Fees structure, Course details, Eligibility criteria description and Admission. A College Enquiry Chatbot is developed using chatterbot algorithm that is a python library that makes it easy to generate automated responses to a user's input.

Users will put the questions through the chatbot that's used for chatting, questions can be related to the Enquiry process, course details, eligibility criteria description and Admission. The answers depend on the user queries. The users do not need to go to the college for enquiry always. The chatbot replies with the assistance of a decent GUI that suggests that as if a real person is rebuke

the user. This chatbot guides the students through the college enquiry process with just a click on the chatbot.

The paper has few applications of Chatbot on education institutions they are:

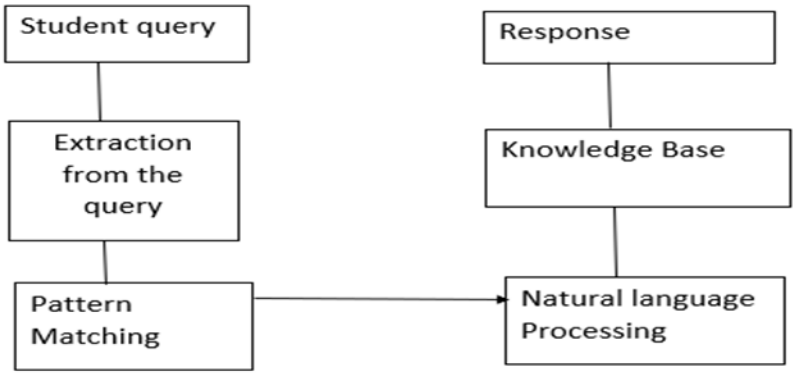
•College enquiry chatbot helps students to get the right source of information.

•Not only our chatbot but any chatbot will provide them with an instant as well as accurate response.

•AI based Chatbot system can be used by colleges and businesses.

In this paper they have a developed a chatbot which will interact with the users and provide all the college related information. The student/parent and the college admin are interacted through a chatbot. The questions which are not answered by the chatbot will be updated by the college admin.

The paper includes the future scopes of Chatbots in educational institution, in the future enhancement of our college enquiry chatbot they can make it more interactive in various different languages for users located in different regions they can include speech-based questions and responses for people who cannot read and type their quires. The future chatbot should not only provide the answer but also the solution to the problem of the student or parent. The College Enquiry Chatbot should not only provide the admission related information but also college achievements, placements related information and scholarship related information. The main objective of this chatbot was to develop an algorithm which will identify the user questions or queries and answer according. To develop a database were all the related data is stored and matched with the questions when question is raised. We successfully developed a chatbot in which the student or parents can ask a query related to the Enquiry process, course details, eligibility criteria description and Admission. The chatbot analyses the question and gives the response accordingly. This paper depicts a way in which we can deal with recognizing the most critical realities in writings depicting the life of an authentic figure for building a conversation operator that could be utilized as a part of centre school CSCL situations. CSCL is implemented online and in classroom learning environments, it takes place synchronously and asynchronously.



Research paper- 09

Dialogue Learning with Human Teaching and Feedback in End-to-End Trainable Task-Oriented Dialogue Systems.

Bing Liu, Gokhan , DilekHakkani, Pararth Shah, Larry Heck Carnegie Mellon University, Pittsburgh, PA, USA Google Research, Mountain View, CA,USA 3Samsung Research, Mountain View, CA, USA.

In this work, a hybrid learning method for training task-oriented dialogue systems through online user interactions is presented by Bing Liu, Gokhan ,DilekHakkani, Pararth Shah and Larry Heck.

Popular methods for learning task-oriented dialogues include applying reinforcement learning with user feedback on supervised pre-training models. Efficiency of such learning method may suffer from the mismatch of dialogue state distribution between offline training and online interactive learning stages. To address this challenge, they have proposed a hybrid imitation and reinforcement learning method, with which a dialogue agent can effectively learn from its interaction with users by learning from human teaching and feedback. A neural network has been designed based on task-oriented dialogue agent that can be optimized end-to-end with the proposed learning method. Experimental results show that this end-to-end dialogue agent can learn effectively from the mistake it makes via imitation learning from user teaching. Applying reinforcement learning with user feedback after the imitation learning stage further improves the agent's capability in successfully completing a task.

Task-oriented dialogue systems assist users to complete tasks in specific domains by understanding user's request and aggregate useful information from external resources within several dialogue turns.

Both supervised learning (SL) and deep reinforcement learning (RL) based systems are being studied to address the limitations with the conventional task-oriented dialogue systems. Recent efforts have been made in designing end-to-end learning solutions with neural network based models.

The main contribution done by this paper is:

•A neural network based, task oriented dialogue system is designed which can be optimized end-to-end for natural language understanding, dialogue state tracking and dialogue policy learning.

•A hybrid imitation and reinforcement learning method for end-to-end model training in addressing the challenge with dialogue state distribution mismatch between offline and interactive learning is poposed.

In the proposed method, a hierarchical LSTM neural network is used to encode a dialogue with a sequence of turns. A bidirectional LTSM is used to encode the user utterance to a continuous representation. This model maintains the dialogue state in a continuous form in the dialogue-level LSTM state which is updated after the model processes each dialogue turn by taking in the encoding of user utterance and the encoding of the previous turn system output. Then the dialogue state tracking outputs are used to form an API call command to retrieve information from a knowledge base. A dialogue policy selects the next system action in response to the user's input based on the current dialogue state. By connecting all the system components, an end-to-end model for task-oriented dialogue is created.

In this work, they have focused on training task-oriented dialogue systems through user interactions, where the agent improves through communicating with users and learning from the mistake it makes. A hybrid imitation and reinforcement learning method is presented, where they firstly train a dialogue agent in a supervised manner by learning from dialogue corpora, and continuously to improve it by learning from user teaching and feedback with imitation and reinforcement learning. The proposed learning method is evaluated with both offline evaluation on fixed dialogue corpora and interactive evaluation with users. Experimental results show that the proposed neural dialogue agent can effectively learn from user teaching and improve task success rate with imitation learning. Applying reinforcement learning with user feedback after imitation learning with user teaching improves the model performance further, not only on the dialogue policy but also on the dialogue state tracking in the end-to-end training framework.

Research paper- 10

JAICOB: A Data Science Chatbot.

DANIEL CARLANDER-REUTERFELT , ÁLVARO CARRERA , CARLOS A. IGLESIAS , ÓSCAR ARAQUE , JUAN FERNANDO SÁNCHEZ RADA , AND SERGIO MUÑOZ

This work was supported by the Educational Innovation Programme of Universidad Politécnica de Madrid under Grant IE1819.0908.

The application of natural language to improve student's interaction with information systems is demonstrated in this paper. In particular, advances in cognitive computing enable a new way of interaction that accelerates insight from existing information sources, thereby contributing to the process of learning. This paper aims at researching the application of cognitive computing in blended learning environments. We propose a modular cognitive agent architecture for pedagogical question answering, featuring social dialogue (small talk), improved for a specific knowledge domain. This system has been implemented as a personal agent to assist students in learning Data Science and Machine Learning techniques. Its implementation includes the training of machine learning models and natural language understanding algorithms in a humanlike interface.

This article presents a modular architecture chatbot named Jaicob, adapted to the learning of Data Science techniques that aims to take advantage of all the benefits for education previously described. It is designed in a modular way that allows its adaptation to other areas of knowledge. It includes a fexible conversation workflow and is easy to maintain. This contribution has been evaluated with real users for a specific use case in a Data Science class.

This paper mainly focuses on pedagogical solutions like:

•A definition of a concept is a consequence of the usual teaching style, which is deductive, starting from the main concepts and developing towards the applications. It is part of the process of learning, but cannot be the whole process.

•The learning of programming techniques can be enhanced by using examples of code using analogy and induction.

•Lastly, the human need for small-talk, such as joking and asking for the weather, must be satisfied to provide a more significant communication source.

The architecture for the JICOB bot includes- Knowledge base, Speech act classifier, Question answering module, Small talk module Graphical user interface.

The evaluation method for Jaicobchatbot is a Partial Least Squares (PLS) analysis.

This article identified the advantages of cognitive assistants in education and the corresponding challenges in implementation. A result is a tool for students with a comfortable and usabe interface and a human experience . It can provide insights and solve doubts about Data Science. The main contribution is the adaptation of students' real pedagogic needs to the design of the architecture and being flexible in maintaining a conversation

They can also outsource to Jaicob the answering of all the questions. The pedagogue is also an excellent asset to select the most valuable sources of information from which Jaicob feeds from, thus providing a curated source of information instead of a regular Google Search.

Research paper- 11

ChatbotForCovid Vaccine Using Deep Learning

ShaliniNaik, VishwaDesai ,Prashantsolanki ,RohitHirurkar , Sharmishtha Desai .

Student ,Mit-Wpu Survey No, 124, Paud Rd, Kothrud, Pune, Maharashtra 411038.

In this paper a chatbot is designed with the help of NLP and Deep learning to easily get information about covid and co – vaccine.

Covi-help chatbot is built with pytorch. This is very easy and user friendly chatbot. GUI feature is used to conveniently talk with the bot. This covi-help bot has enough information to know about vaccine and vaccination. With using NLP and Neural network it becames easy and helpful to the people who don’t know about vaccination. This chatbot gave information about vaccination, how much it is important, why it is needed and which types of vaccine available, etc.

They're utilising NLTK, a well-known NLP library. The NLP Pipeline that isutilised is as follows: we get the sentences first, then tokenize them. They also took off punctuation and computed the bag of words based on it. In read mode, they load the json file. After that, a empty list is created. Tokenization was imported from nltk. As a result, they have training data and must continue to create a bag of words. They constructed a pytorch dataset from x-train and ytrain. The model was then imported and saved. Deep learning model has two hidden layers and is fed by a forwarded neural network. As input, they have a bag of words, two completely connected hidden layers, and a number of distinct patterns as output.

CHATBOTS WITH DATASCIENCE

Cognitive computing has grown in the last few years, increasing the research and commercial interest in the topic. Conversational agents have evolved from simple pattern-based programs into rather complex systems, including Natural Language Understanding and Machine Learning Techniques, which have allowed them to be more flexible in maintaining a conversation. Every day more businesses include chatbots as a way to interact with consumers to answer requests and FAQs. Natural Language Interface (NLI) increases user satisfaction and can help to find the information needed in a more comfortable way than other less sophisticated and time-consuming search interfaces.

Like humans, cognitive systems can use their knowledge to deduce data meaning based on context. By having the advantage of computational power, a system like this can be even more successful than a human in this kind of task. Though they do not understand the meaning as humans do, the insights these systems provide can be beneficial. As they grow in time, it is expected that they gain abilities such as sensing and awareness.

Compared to other traditional e-learning training, chatbots generate a more positive response from the users. More-over, there are advantages in this type of learning, such as interaction, active learning, and sociability. Despite these reasons, these technologies have not been widely adopted yet in education, and the ones that have are usually very rule-based and, therefore, less practical and functional. This article presents a modular architecture chatbot named Jaicob, adapted to the learning of Data Science techniques that aims to take advantage of all the benefits for education previously described. It is designed in a modular way that allows its adaptation to other areas of knowledge. It includes a flexible conversation workflow and is easy to maintain. This contribution has been evaluated with real users for a specific use case in a Data Science class.

Data Science in machines is a very challenging discussion. It involves the creation of machines which can simulate intelligence. This paper discusses some of the current trends and practices of Chot Bot in Data Science and subsequently offers alternative theory for improvement in some of today's prominent and widely accepted postulates. For this, we have to focus on the structuring and functioning of a simple A.I. system - chatbots (or chatter bots) is made. The paper shows how current approach towards Data Science is not adequate and offers a new theory that discusses machine intelligence, throwing light to the future of intelligent systems.

The creation and analysis of intelligent agents (software and machines) is called Artificial Intelligence, or AI. It can be implemented in nearly each and every sphere of work. Intelligent machines can do many tasks from labour work to sophisticated operations. Prominent trends in this field are human brain simulation, natural-language processing and neural networking etc. One of the typical examples of an AI system is a "chatbot". The chatbot is a computer program which responds like an intelligent entity when conversed with. The basic foundation of chatbots is providing the best response of any query that it receives. The best response like answering the

sender questions, providing sender relevant information, ask follow-up questions and do the conversation in realistic way.

Theconversation may be through text or voice.Any chatbot program understands one or more human languages by Natural Language Processing. Due to this, the system interprets human language input using information fed to it. A chatbot may also perform some productive functions like calculations, setting-up remainders or alarms etc. The chatbot needs to be able to understand the intentions of the sender's message, determine what type of response message (a follow-up question, direct response, etc.) is required, and follow correct grammatical and lexical rules while forming the response. Some models may use additional meta information from data, such as speaker id, gender, emotion. Sometimes, sentiment analysis is used to allows the chatbot to 'understand' the mood of the user by analysing verbal and sentence structuring clues.

A popular example of chatbot is the ALICE Bot (Artificial Linguistic Internet Computer Entity), which utilizes AIML pattern matching techniques. "Turing test" is one of the most popular measures of intelligence of such systems. According to it, if a panel of human beings conversing with an unknown entity (via keyboard, for example) believes that that entity is human while the entity is actually a computer, the computer is said to have passed the Turing test. This test was proposed by British mathematician Alan Turing in 1950 paper "Computing Machinery and Intelligence" published in Mind. All these developments have been made but are we actually constantly progressing towards the higher goals of AI. These days, the evolution inAI has been limited around some key points like the fed-in information database, imitation of human abilities, targeting the Turing test and techniques of deception.

A newly initialized Chatterbot instance starts off with no knowledge of how to communicate. To allow it to properly respond to user inputs, the instance needs to be trained to understand how conversations flow. Since Chatterbot relies on machine learning at its backend, it can very easily be taught conversations by providing it with datasets of conversations.

Chatterbot's training process works by loading example conversations from provided datasets into its database. The Chatbot uses the information to build a knowledge graph of known input statements and their probable responses. This graph is constantly improved and upgraded as the chatbot is used.

In our article we like to identified the advantages of cognitive assistants in education and the corresponding challenges in implementation. A result is a tool for students with a comfortableand usable interface and a human experience. It can provideinsights and solve doubts about Data Science. The maincontribution is the adaptation of students' real pedagogicneeds to the design of the architecture and being flexible inmaintaining a conversation.

TYPES OF CHATBOTS

I. Menu/button-based chatbots

Menu/button-based chatbots are the most basic type of chatbots that are currently available on the market. Most chatbots are glorified decision tree hierarchies that are presented to the user in the form of buttons. These chatbots, like the automated phone menus we all deal with on a daily basis, require the user to make several choices in order to get to the ultimate answer.

While these chatbots are adequate for answering FAQs, which account for 80% of support queries, they fall short in more advanced scenarios where there are too many variables or too much knowledge at work to predict how users should arrive at specific answers with confidence. It is also worth noting that menu/button-based chatbots are the slowest to respond.

II. Linguistic Based (Rule-Based Chatbots)

If you can anticipate the types of questions your customers will ask, a linguistic chatbot could be the answer. Linguistic or rules-based chatbots use if/then logic to create conversational automation flows. You must first define the language conditions for your chatbots. Conditions can be created to evaluate the words, their order, synonyms, and other factors. If the incoming query matches the conditions defined by your chatbot, your customers will receive immediate assistance.

It is your responsibility, however, to ensure that each permutation and combination of each question is defined; otherwise, the chatbot will not understand your customer's input. This is why, while linguistic models are extremely common, they can take a long time to develop. These chatbots require precision and specificity.

III. Keyword recognition-based chatbots

Keyword recognition-based chatbots, as opposed to menu-based chatbots, can listen to what users type and respond appropriately. These chatbots use customizable keywords and an AI application called Natural Language Processing (NLP) to determine how to respond to the user.

When faced with a large number of similar questions, these chatbots fall short. When there are keyword redundancies between several related questions, the NLP chatbots will begin to falter.

It's common to see chatbot examples that combine keyword recognition and menu/button navigation. These chatbots give users the option of directly asking their questions or using the chatbot's menu buttons if the keyword recognition functionality is producing poor results or the user requires some assistance in locating information.

IV. Machine Learning chatbots

Have you ever wondered what a contextual chatbot is? A contextual chatbot is far more sophisticated than the previous three bots. These chatbots use Machine Learning (ML) and Artificial Intelligence (AI) to remember specific user conversations in order to learn and grow over time. Chatbots with contextual awareness, as opposed to keyword recognition-based bots, are intelligent enough to self-improve based on what users ask for and how they ask for it.

For instance, consider a contextual chatbot that allows users to order food; the chatbot will save data from each conversation and learn what the user prefers to order. As a result, when a user chats with this chatbot, it will eventually remember their most common order, delivery address, and payment information and simply ask if they want to repeat this order. Instead of answering multiple questions, the user only needs to say 'Yes,' and the food is ready!

V. The Hybrid model

Businesses appreciate the sophistication of AI-chatbots, but they do not always have the talent or large volumes of data to support them. As a result, they choose the hybrid model. The hybrid chatbot model combines the simplicity of rules-based chatbots with the complexity of AI-bots to provide the best of both worlds.

VI. Voice bots

Businesses are now beginning to use voice-based chatbots or voice bots to make conversational interfaces even more vernacular. Voice bots have been on the rise in recent years, with virtual assistants ranging from Apple's Siri to Amazon's Alexa, and why? Because of the benefits they provide. Speaking rather than typing is much more convenient for a customer. A voice-activated chatbot provides frictionless experiences to the end user.

CONSTRUCTION OF CHATBOT

It can be difficult to get started with chatbots. How to construct a chatbot involves a variety of factors, including strategy, conversational flow, technology, tools, procedure, reporting, and more.

Before you begin building a chatbot, you must first determine • What problem are you attempting to solve? This becomes your use case.

• How much time do you/your team currently devote to this issue? This will help you define your ROI later.

• Could you automate the entire process with a bot, or do you need human intervention? This assists you in determining whether you require the platform to have a chatbot to human handover functionality.

• Do you require the chatbot to push/pull data from a third-party system? This will help you narrow down platforms with ready integrations.

Acknowledging your strategic goals, the chatbot's objectives, and creating the chatbot conversation flow to handle input/output efficiently will aid you in your bot-building journey.

There are mainly three different types of bots that you can build, including –

- Rule-based chatbot:
  Rule-based bots follow a pre-determined conversation flow that allows the bot to respond logically to the user's inputs and choices. By clicking on buttons, options, carousels, and answering questions, users move through the conversation flow.

  Users will find it easier to navigate through rule-based bots because they are easier to create. Users cannot ask their own inquiries and can only provide information when the bot requests it (contact details, details pertaining to the use case and more).

- AI chatbot:
  Natural language processing is used by AI chatbots to recognise phrase structure and then process that information, all while getting better at answering the question at hand over time.

  Simply defined, instead of depending on a prepared output language produced by a human, AI chatbots first grasp the intent behind the customer's enquiry and then respond with an appropriate and contextual answer.

- Hybrid chatbot:

The hybrid chatbot, as its name suggests, combines the finest of rule-based and AI technology with live chat capability to create a greater client experience. You'll need to do the following to create a chatbot:

Determine the chatbot's exact tone and personality based on your specific business and use case.
To ensure comfortable and fluent discussions, provide a human factor to the chatbot.
Because the effectiveness of the conversation design is primarily dependent on the context and user intent, the scripting data you utilise should match your target audience.

Many frameworks, such as Watnot, Intercom, Drift chatbot, Landbot.ai, Live Person, Bold360, Octane Al, Flow XO, ManyChat, Botsify, Chatfuel, Pandorabots, Botscrew, and Aivo, facilitate us in developing a chatbot.

To develop our chatbot, we will use the Landbot.ai framework.Lanbot.io is an easy-to-use solution that allows you to create rule-based and AI-powered bots that effortlessly communicate with potential customers and generate high-quality dialogues. Human agents can also join the conversation in the middle and take control of the chatbot in real time with Landbot.
Features-

- Allows you to easily develop a chatbot using a drag-and-drop interface.
- Allows you to create dialogue flows, test and evaluate your chatbots, and integrate it with other web apps and tools without writing any code.
- Add brand aspects to your chatbots to make them more personalised.
  A free version is available. Simple to use.There are numerous integrations available. Make chatbots for a variety of platforms. Only the premium plan includes integrations. There is a limit to the amount of discussions you can have.

Step 1: Give chatbot a purpose

First, we will determine what we want to do with the chatbot.

What are we hoping to achieve with the chatbot? Is it better to automate customer service, improve customer experience, or generate leads?

• What are the most common client scenarios? Examine the questions and make a list of a few examples.

• What would be the most useful feature of a chatbot? Is it possible to answer questions on autopilot? Is it possible to send the questions to the customer service team? Is it better to save abandoned carts or qualify leads?

It will be a lot easier to identify the features and sorts of chatbots when we get solutions for the above questions.

Step 2: Decide where you want it to appear

What's our go-to method of communication? Do the majority of your customers contact you via social media or through a live chat widget on your website?

In any case, make sure that the chosen chatbot platform interfaces with the technologies we already use so that we can serve consumers wherever they are:

- Internet site. The majority of chatbot development platforms include interfaces with popular website platforms like WordPress, Magento, and Shopify.
- WhatsApp, Facebook Messenger, Instagram, and Telegram are examples of social media channels.
- You have other messaging systems and technologies in your stack (such as Slack).
- Alternatively, we should see if the integration can be configured via a code snippet or an open API.

Many chatbot creation platforms include various connectors, allowing you to use chatbots across multiple sites.

Step 3: Choose the chatbot platform

It's time to choose a provider now that we know the chatbot versions we want to produce and which channels we want to cover.

You have two options: the framework or the platform.

- Frameworks for artificial intelligence: Chatbot frameworks (like Google's Dialog Flow, IBM Watson, or Microsoft Bot) provide libraries for software developers who then use code to create chatbots.
- Platforms for chatbots: These platforms offer simple chatbot builders that allow you to design a chatbot using building components. They're becoming more popular because constructing bots with their assistance is considerably easier and takes less time while producing comparable outcomes. Not to mention that certain platforms, like Tidio, provide programs that are free for life. ‘We will use landbot.ai for creating the chatbot’.

Step 4: Design the chatbot conversation in a chatbot editor

By dragging and dropping construction bricks into a sequence, we may define the discussion flow.
Log in and head to the bot builder to get started. Start with the trigger, which is a situation that causes the chatbot to send a greeting message. Start with a Visitor visiting a specified page node for the chatbot to appear on a certain landing page.

Then add a decision node with quick responses and type in the message to be sent. Setting up separate messages for individuals who want a discount and those who don't.

Step 5: Testing chatbot

Now it's time to see if everything is working properly. We can do it with the 'Try it out option'. A window will popup that depicts how the chatbot will seem to the final user. We can always return to the editor and modify the flow thanks to the preview.

Step 6: Train your chatbots

We can add an NLP trigger to our chatbot to grasp the user's intent. This will "feed" the NLP engine, allowing the chatbot to detect similar inquiries in future discussions.

METHODOLOGY

"Hello"Chatbot

We're going to use deep learning to create a chatbot in this Python project with source code. The information, which includes categories (intents), patterns, and responses, will be used to train the chatbot. In order to determine which category the user's message falls under, we employ a specialised recurrent neural network (LSTM), after which we will select a random response from a list of options.

Let's use tools like NLTK, Keras, Python, etc. to build a retrieval-based chatbot.

We will use the dataset "intents.json". The patterns we need to identify and the responses we want to give the user are included in this JSON file.

```
{"intents": [
        {"tag": "greeting",
         "patterns": ["Hi there", "How are you", "Is anyone there?","Hey","Hola", "Hello", "Good day"],
         "responses": ["Hello, thanks for asking", "Good to see you again", "Hi there, how can I help?"],
         "context": [""]
        },
        {"tag": "goodbye",
         "patterns": ["Bye", "See you later", "Goodbye", "Nice chatting to you, bye", "Till next time"],
         "responses": ["See you!", "Have a nice day", "Bye! Come back again soon."],
         "context": [""]
        },
        {"tag": "thanks",
         "patterns": ["Thanks", "Thank you", "That's helpful", "Awesome, thanks", "Thanks for helping me"],
         "responses": ["Happy to help!", "Any time!", "My pleasure"],
         "context": [""]
        },
        {"tag": "noanswer",
         "patterns": [],
         "responses": ["Sorry, can't understand you", "Please give me more info", "Not sure I understand"],
         "context": [""]
        },
        {"tag": "options",
         "patterns": ["How you could help me?", "What you can do?", "What help you provide?", "How you can be helpful?", "What support is
offered"],
         "responses": ["I can guide you through Data Science"],
         "context": [""]
        },
        {"tag": "Data Science",
         "patterns": ["What is Data Science?"],
         "responses": ["Data science is a concept to unify statistics, data analysis, informatics, and their related methods in order to
understand and analyse actual phenomena with data Data science is an interdisciplinary field focused on extracting knowledge from typically
large data sets and applying the knowledge and insights from that data to solve problems in a wide range of application domains"],
         "context": [""]
        },
        {"tag": "Uses",
         "patterns": ["Uses of Data Science"],
         "responses": ["Data science may detect patterns in seemingly unstructured or unconnected data, allowing conclusions and
predictions to be made.Tech businesses that acquire user data can utilise strategies to transform that data into valuable or profitable
information. Data Science has also made inroads into the transportation industry, such as with driverless cars. It is simple to lower the
number of accidents with the use of driverless cars. For example, with driverless cars, training data is supplied to the algorithm, and the
data is examined using data Science approaches, such as the speed limit on the highway, busy streets, etc. Data Science applications
provide a better level of therapeutic customisation through genetics and genomics research."],
         "context": [""]
```

How to create a Chatbot?

We will now use Python to develop the chatbot, but first, let's look at the file organisation and the kinds of files we'll be making:

- **Intents.json** - The data file with predetermined patterns and replies is called Intents.json.
- **Train_chatbot.py** - We created a script to create the model and train our chatbot in this Python file.
- **Words.pkl –** We keep the words Python object, which includes a list of our vocabulary, in this pickle file.
- **Classes.pkl–** The categories list can be found in the classes pickle file.
- **Chatbot_model.h5 -** This is the trained model, which includes weights for the neurons as well as model information.
- **Chatgui.py –** This is the Python code we used to add a GUI to our chatbot. The bot is simple for users to communicate with.

Here are the steps to create a chatbot

1. **Import and load the data file :**

   We'll create a file with the name building a chatbot in python.ipynb first. We initialise the variables we'll need in our Python project and import the libraries our chatbot needs.

```
In [2]: import nltk
        from nltk.stem import WordNetLemmatizer
        lemmatizer = WordNetLemmatizer()
        import json
        import pickle
```

```
In [2]: import numpy as np
        from keras.models import Sequential
        from keras.layers import Dense, Activation, Dropout
        from keras.optimizers import gradient_descent_v2
        import random
        sgd = gradient_descent_v2.SGD(...)
        from tensorflow.keras.optimizers import SGD
```

```
In [3]: words=[]
        classes = []
        documents = []
        ignore_words = ['?', '!']
        data_file = open('intents(DS).json').read()
        intents = json.loads(data_file)
```

2. **Preprocess data**

   Before creating a machine learning or deep learning model when working with text data, there are a number of preprocessing steps that must be taken. We need to preprocess the data using a variety of processes depending on the needs.

The simplest and initial action you can take with text data is to tokenize it. The method of tokenizing involves dividing the entire text into smaller units, such as words.

Using the nltk.word_tokenize() function, we tokenize the sentence in this case and attach each word from the words list after iterating through the patterns. For our tags, we also make a list of classes.

```
In [4]: 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'])
```

We will now lemmatize each word and eliminate any repetitions from the list. Lemmatizing entails taking a word and changing it into a lemma. After that, a pickle file is made to house the Python objects that will be used for prediction.

3. Create training and testing data

```
In [5]: # 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'))

        31 documents
        13 classes ['Applications', 'Data Science', 'Data Scientist', 'Data science with Statistics', 'Differece', 'Example
        ', 'Uses', 'What Does a Data Scientist Do?', 'Who oversees data science process', 'goodbye', 'greeting', 'options',
        'thanks']
        61 unique lemmatized words ["'s", ',', 'a', 'and', 'anyone', 'application', 'are', 'artificial', 'awesome', 'be', '
        between', 'business', 'bye', 'can', 'chatting', 'could', 'data', 'day', 'difference', 'do', 'doe', 'example', 'for'
        , 'good', 'goodbye', 'hello', 'help', 'helpful', 'helping', 'hey', 'hi', 'hola', 'how', 'intelligence', 'is', 'it',
        'later', 'manager', 'me', 'next', 'nice', 'of', 'offered', 'provide', 'relationship', 'science', 'scientist', 'see'
        , 'statistic', 'support', 'thank', 'thanks', 'that', 'the', 'there', 'till', 'time', 'to', 'us', 'what', 'you']
```

Now that the training data has been created, the input and output will be included. The pattern will serve as our input, and the class to which the pattern belongs will serve as our output. However, since the computer cannot interpret text, we will translate it into numbers.

```
In [6]: # 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")

        Training data created

        C:\Users\fujitsu\AppData\Local\Temp/ipykernel_15660/2748295590.py:24: VisibleDeprecationWarning: Creating an ndarr
        ay from ragged nested sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths or
        shapes) is deprecated. If you meant to do this, you must specify 'dtype=object' when creating the ndarray.
          training = np.array(training)
```

4. Build the model

Now that we have our training data prepared, we will construct a three-layer deep neural network. For this, we employ the Keras sequential API. The model was trained for 200 epochs, and after that, we had 100% accuracy. Let's name the model "chatbot_model.h5" for saving.

In [7]:
```
# Create model - 3 layers. First layer 128 neurons, second layer 64 neurons and 3rd output layer contains number of
# 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")
```

```
Epoch 1/200
C:\Users\fujitsu\anaconda3\lib\site-packages\keras\optimizer_v2\gradient_descent.py:102: UserWarning: The `lr` a
rgument is deprecated, use `learning_rate` instead.
  super(SGD, self).__init__(name, **kwargs)
7/7 [==============================] - 1s 5ms/step - loss: 2.6464 - accuracy: 0.0645
Epoch 2/200
7/7 [==============================] - 0s 3ms/step - loss: 2.5285 - accuracy: 0.1613
Epoch 3/200
7/7 [==============================] - 0s 5ms/step - loss: 2.4571 - accuracy: 0.3226
Epoch 4/200
7/7 [==============================] - 0s 5ms/step - loss: 2.2680 - accuracy: 0.2581
Epoch 5/200
7/7 [==============================] - 0s 5ms/step - loss: 2.3894 - accuracy: 0.1935
Epoch 6/200
7/7 [==============================] - 0s 3ms/step - loss: 2.1201 - accuracy: 0.3226
Epoch 7/200
7/7 [==============================] - 0s 5ms/step - loss: 2.1585 - accuracy: 0.2903
Epoch 8/200
```

5. Predict the response (Graphical User Interface)

We need to build a new file called "chatapp.py" in order to anticipate the user's responses and predict the sentences.

After loading the trained model, a graphical user interface will be used to foretell the bot's answer. In order to extract a random response from the list of responses after identifying the class the model belongs to, we will construct several functions that first identify the class.

We once more import the required packages and load the pickle files we made when we trained our model, called "words.pkl" and "classes.pkl":

```
In [9]: 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(DS).json').read())
        words = pickle.load(open('words.pkl','rb'))
        classes = pickle.load(open('classes.pkl','rb'))
```

We must offer input in the same way as we did during training in order to predict the class. Therefore, we shall develop some methods that first preprocess text before determining the class.

```
In [10]: 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
```

We shall receive a random response from the list of intents after predicting the class.

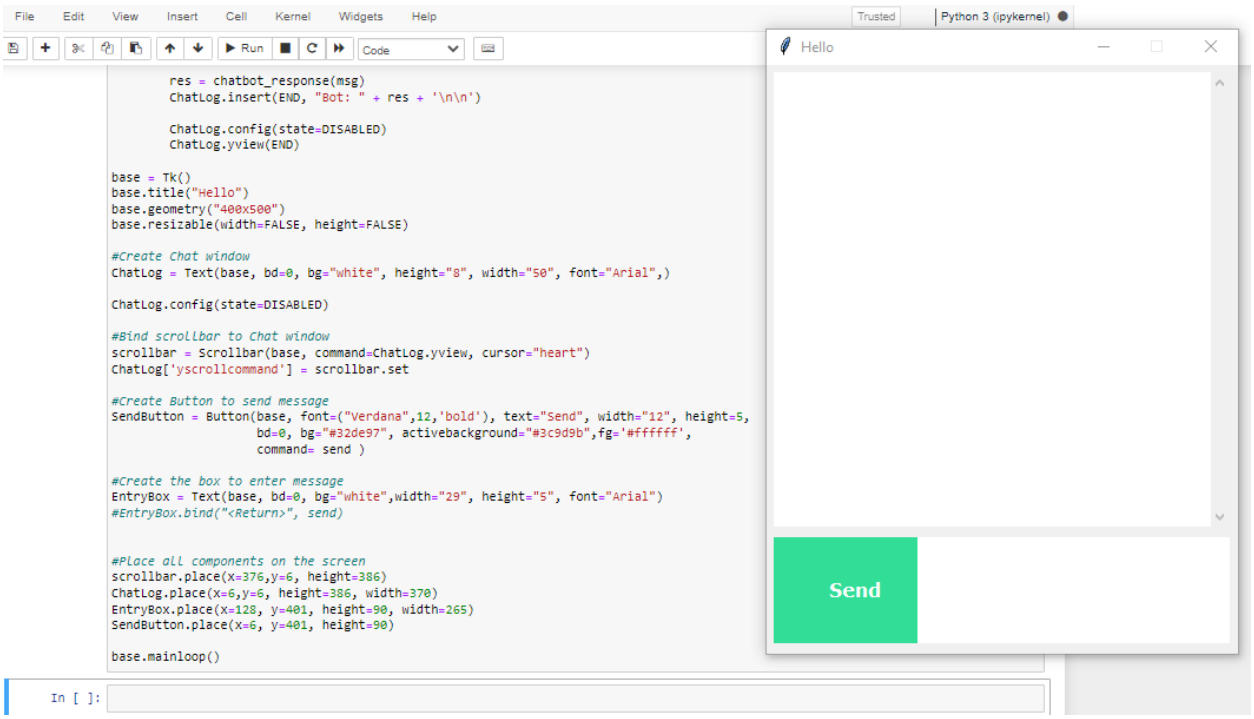
We'll start by creating a graphical user interface. Let's use the Tkinter library, which comes with a tonne of helpful GUI libraries. We'll take the user's input message and utilise the helper methods we've developed to retrieve the bot's answer and show it on the GUI. Here is the GUI's complete source code

In [11]:
```
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
```

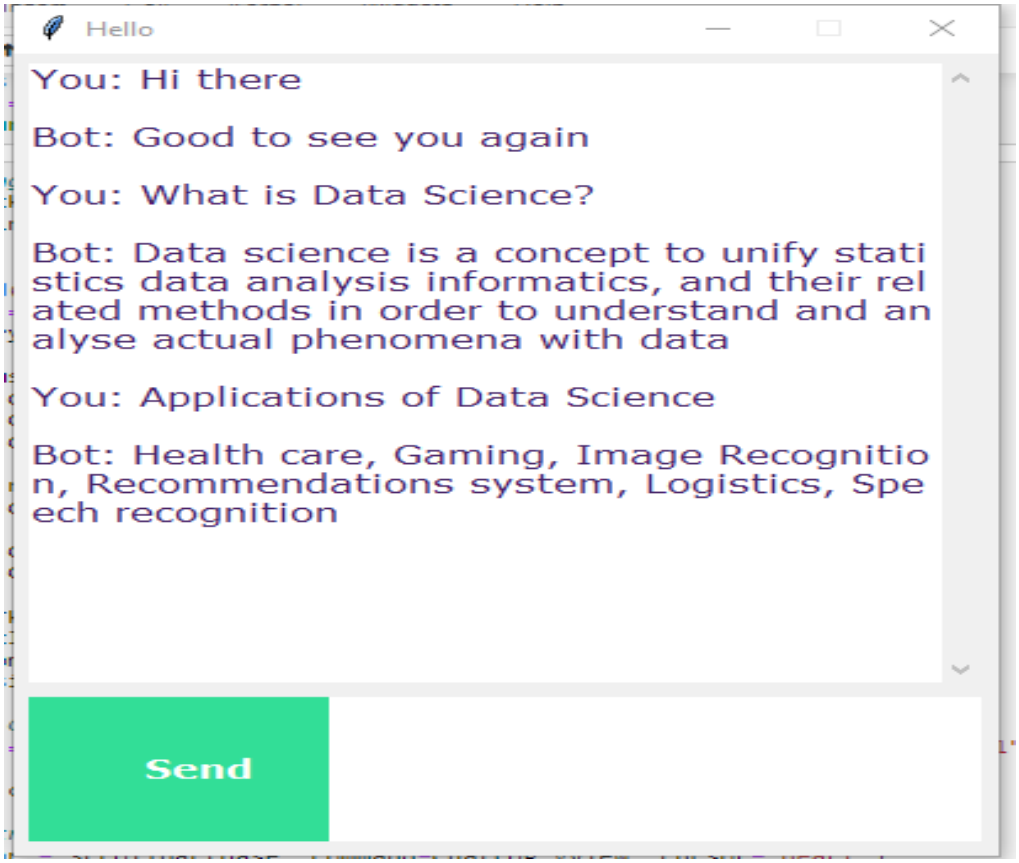
In [ ]:
```
#Creating GUI with tkinter
import tkinter
from tkinter import *


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()
```

6. Run the chatbot

Two primary files—train chatbot.py and chatapp.py—are required to execute the chatbot.



RESULT


Hello
You: Hi there
Bot: Good to see you again
You: What is Data Science?
Bot: Data science is a concept to unify statistics data analysis informatics, and their related methods in order to understand and analyse actual phenomena with data
You: Applications of Data Science
Bot: Health care, Gaming, Image Recognition, Recommendations system, Logistics, Speech recognition
Send

ADVANTAGES OF DATASCIENCE CHATBOT

1. Quick Data Collection and Analysis

   Chatbots could be an effective way to learn more about your target audience. They may interact with your target audience and acquire information like names, email addresses, and other personal information. You'll have rapid access to this information if you connect the chatbot to the CRM.
   A chatbot may quickly obtain and assess data while constantly interacting with customers or leads. Chatbots readily recall earlier conversations when an old customer returns to the site. AI chatbots can instantly learn all of these customers' likes and interests and engage them in a conversation until they are satisfied. AI chatbots make data collection and analysis straightforward in this way.

2. Effective Solutions

   To please the customer, you must provide the suitable solution. Chatbots, on the other hand, without AI can just provide replies, not the effective solutions that customers want. As a result, utilising AI chatbots simplifies and expedites customer service. They have the ability to filter through large amounts of data to find and offer the proper answer. AI examines prior responses in order to provide a fresh solution each time.

3. Customer Satisfaction in businesses

   People demand immediate satisfaction, and a chatbot may provide it for your commonly asked questions. When implemented effectively, a chatbot has the potential to boost customer satisfaction dramatically. Chatbots can help businesses reduce customer churn and improve customer satisfaction.
   Chatbots are artificial intelligence applications that, when they show a high level of comprehension, can provide superior customer service. Bots are critical for answering enquiries from customers at all hours of the day and night.
   AI chatbots ask inquiries, collect responses, and then direct the customer to whatever they're looking for. Installing a chatbot can help you figure out what a consumer has or hasn't found in order to intelligently guide them to their needs. Furthermore, AI chatbots personalise the questions they ask, resulting in higher client satisfaction during the transaction.

4. Availability

   Chatbots are available 24 hours a day, 7 days a week and may respond to your customers right away. They are available to assist a customer at any time, even if it is outside of

regular office hours. This means that if they message you for whatever reason, they'll get a response soon away.

Customers are more satisfied since they believe they can get help without having to wait for an email or voicemail to be returned. As a result, they'll be satisfied with your brand, and you'll be able to move them down the sales funnel.

5. Gaining a Better Understanding of Customer Behaviours

   Chatbots retain detailed and actionable records of your customers' most pressing problems. The information provided by the customer has an impact on the possibility of a sale, and chatbots can help speed up the data collection process. Businesses can use chatbots to learn more about their clients' needs. Based on the data provided by the chatbot-customer interaction, customer-specific objectives can be developed. Chatbots can provide feedback to the firm, and modifications can be made after the data is analysed.

6. Better Lead Generation

   Any company looking to expand its market should focus on lead creation. The finest thing they can do is ensure complete customer pleasure throughout their relationship with the company. With the client data that chatbots collect, they can send customised messages. A bot can ask all the relevant questions, persuade the user, and generate a lead. Chatbots ensure that the flow is in the right direction to enhance conversion rates.

   In addition to establishing new clients and updating sales employees, a chatbot may help you analyse incorrect leads using preset KPIs and prevent dealing with time-consuming leads.

7. Save Time and Money

   You may believe that implementing this technology in a business is an expensive task. The use of a chatbot may necessitate a certain level of investment. However, as compared to human labour and consumer communication, this cost is lower.

   The goal of chatbots is to make the customer care process easier for both the customer and the employees or firm. Chatbots can quickly handle many minor difficulties or questions that would otherwise necessitate a lengthy phone discussion. As a result, both parties save money and time.

8. Chatbots provide Longterm Financial Savings

Companies that use human customer service representatives must pay their staff a substantial salary. And if the company is big, the expenses will be big, too! A chatbot can help the organisation save money in the long run by reducing this expense. While it is true that establishing a chatbot takes a lot of time and money at first, the chatbot will eventually be able to answer all of the fundamental client questions that would take people a long time to respond. Furthermore, the chatbot may provide fast responses, which saves time.

CONCLUSION

In this data science project, we learned about chatbots and developed an accurate deep learning chatbot. The data can be modified in accordance with organisational needs, and the chatbot can be trained very precisely. The use of chatbots is widespread, and every organisation is eager to integrate them into its daily operations.

REFERENCE

I. **Customer service chatbots**

How to build Customer service Chatbots?

1. https://venturebeat.com/ai/10-steps-to-build-a-better-customer-service-chatbot/
2. https://www.netomi.com/campaigns/how-to-build-a-customer-service-chatbot

Top Customer Service Chatbots

1. HelloFresh Freddy

i. https://www.youtube.com/watch?v=WPOVsNdD7hM
ii. https://www.hellofresh.com/contact-us

II. **Healthcare chatbots**

How to build Healthcare Chatbots?

i. https://chatbotsjournal.com/how-to-build-your-own-healthcare-chatbot-528299fe351a
ii. https://sendpulse.com/blog/healthcare-chatbots

Top Healthcare Chatbots

1. *Haptik Develops COVID-19 Awareness WhatsApp Chatbot for Govt of India called "***MyGov Corona Helpdesk chatbot"**

i. *https://www.haptik.ai/blog/whatsapp-chatbot-for-government-of-india*

ii. https://www.youtube.com/watch?v=dBSLiEIqhnA

2. **Doctor.ai**

i. **https://neo4j.com/blog/doctor-ai-a-voice-chatbot-for-healthcare-powered-by-neo4j-and-aws/**

ii. **https://towardsdatascience.com/doctor-ai-an-ai-powered-virtual-voice-assistant-for-health-care-8c09af65aabb**

Lex was the natural language understanding engine behind Doctor.ai. To query the Neo4j database, the user dictated or typed the questions to Doctor.ai

3. **Baidu's Melody**

i. **https://www.the-digital-insurer.com/dia/baidus-melody-ai-powered-conversational-bot-for-doctors-and-patients-1/**

Proprietary deep learning and natural language processing (NLP) software developed by Baidu, a Chinese multinational technology company specializing in Internet-related services and products and artificial intelligence

III. **Therapist Chatbots**

How to build Therapist Chatbots?

i. **https://topflightapps.com/ideas/build-mental-health-chatbot/**

1. **Woebot**

https://woebothealth.com/

uses Proprietary AI and NLP tools

2. **Wysa**

https://www.wysa.io/

https://www.choosingtherapy.com/wysa-app-review/

3. **Tess**

https://www.x2ai.com/

https://www.techtimes.com/articles/264696/20210827/ai-chatbot-named-tess-helps-students-anxiety-depression-study.htm

Tess is designed using a combination of technologies, emotion algorithms, and machine learning techniques to support a variety of features. In collaboration with mental health professionals, Tess is trained to deliver pre-scripted interventions in order to replicate an empathic response that is appropriate to the inputted emotion or scenario. Specific interventions are triggered based on the individual's reported concern.

IV. **Conversion boosters & sales bots**

How to build Conversion boosters & sales bots?

i. https://www.smartinsights.com/customer-relationship-management/6-chatbot-best-practices-for-e-commerce-sites/
ii. https://chatfunnels.com/blog/the-data-doesnt-lie-increase-your-conversion-rates-by-50-with-chatbots/

1. **Landbot.io**

i. **https://landbot.io/**
ii. **https://www.producthunt.com/products/landbot-io**

2. **Domino's Pizza**

I. **https://pizzaonline.dominos.co.in/**
II. **https://www.youtube.com/watch?v=bhlzCgpb-MM**

3. **Kian**

i. **https://www.kia.com/us/en/chatbot**
ii. **https://www.youtube.com/watch?v=omC71UrFmcU**

4. **Trim**

i. **https://www.asktrim.com/**
ii. **https://techacute.com/trim-to-assist-in-optimizing-your-finances/**