Machine Learning from Casual Conversation

2019

Awrad Mohammed Ali
University of Central Florida

MACHINE LEARNING FROM CASUAL CONVERSATION

by

AWRAD MOHAMMED ALI
Msc. in Computer Engineering, University of Central Florida, 2014

A Dissertation submitted in partial fulfilment of the requirements
for the degree of Philosophy in Computer Sceince
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida

Spring Term
2019

Major Professor: Avelino J. Gonzalez

ABSTRACT

Human social learning is an effective process that has inspired many existing machine learning techniques, such as learning from observation and learning by demonstration. In this dissertation, we introduce another form of social learning, Learning from a Casual Conversation (LCC). LCC is an open-ended machine learning system in which an artificially intelligent agent learns from an extended dialog with a human. Our system enables the agent to incorporate changes into its knowledge base, based on the human's conversational text input. This system emulates how humans learn from each other through a dialog. LCC closes the gap in the current research that is focused on teaching specific tasks to computer agents. Furthermore, LCC aims to provide an easy way to enhance the knowledge of the system without requiring the involvement of a programmer. This system does not require the user to enter specific information; instead, the user can chat naturally with the agent. LCC identifies the inputs that contain information relevant to its knowledge base in the learning process. LCC's architecture consists of multiple sub-systems combined to perform the task. Its learning component can add new knowledge to existing information in the knowledge base, confirm existing information, and/or update existing information found to be related to the user input. The LCC system functionality was assessed using different evaluation methods. This includes tests performed by the developer, as well as by 130 human test subjects. Thirty of those test subjects interacted directly with the system and completed a survey of 13 questions/statements that asked the user about his/her experience using LCC. A second group of 100 human test subjects evaluated the dialogue logs of a subset of the first group of human testers. The collected results were all found to be acceptable and within the range of our expectations.

This dissertation is dedicated to my beloved parents and family for their love, endless support, encouragement and sacrifices.

ACKNOWLEDGMENTS

I would like to express the deepest appreciation and thanks to my adviser, Professor Avelino Gonzalez who supported me at every step working on this dissertation. This work would not be possible without his guidance. I would also like to thank my committee member and Master's advisor Dr. Gita Sukthankar for supporting me and introducing me to Dr. Gonzalez. I would like to thank Drs. Annie Wu, Arther Weeks and Fei Liu for serving on my dissertation committee. I would like to thank my husband Zaid Alchalabi and my kids Dema and Yazen for supporting me in every step of this journey. Also, I want to thank my parents Emad Mohammed Ali, Fatin Jaber and my siblings Ahmad, Islam, and Baraa for their love and support. I want to express my warm thanks to my friends and labmates Josiah Wong, James Hollister, Ramya Pradhan, Andreas Marpaung and Vera Kazakova for their support. Finally, I would like to thank my friends inside and outside the university, especially Mina Alomari, Talib Alsharook and Omar Nakhila for their support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION AND BACKGROUND

Machine learning plays an important role in many applications in various different disciplines. In general, traditional machine learning algorithms can be grossly divided into two general branches; 1) learning classification and/or patterns from (large) datasets. These include supervised learning and unsupervised learning; and 2) behavioral learning that involves reinforcement learning and imitation learning. Supervised learning is about learning a mapping function among labeled input data and their corresponding outputs to predict the correct output for new provided inputs. Classification algorithms generally employ a type of supervised learning [56]. Unsupervised learning involves learning the grouping of unlabeled input data. Clustering is a type of unsupervised learning [33].

Behavioral learning is a natural learning process used by humans and animals because both live in societies where the major type of learning process is by interacting with one other. To emulate this type of learning, behavioral learning has been applied to artificial intelligence agents. This discipline has not been as extensively investigated as supervised and unsupervised learning, but several interesting and potentially transformational approaches have emerged in the last few decades.

In this research, we explore a type of behavioral learning that involves interaction with humans to learn new knowledge. Our aim is to make an improvement in the agent's knowledge by deploying a learning capability from a conversation with humans or other agents, as humans do.

One of the earliest forms of behavioral learning is reinforcement learning (RL) [122], a type of machine learning algorithm that allows a computer agent/robot to learn new behaviors while interacting with the environment and receiving rewards that reflect how the agent is preforming. The idea is to learn actions that maximize the cumulative reward that can be earned. RL can be costly in some domains such as robotics because it involves significant trial and error to teach the robot

the desired actions. Moreover, building a policy for search methods requires the robot to execute actions in the real world to receive rewards, which is not practical in many cases considering that designing the reward function is non-trivial [15]. Such limitations make reinforcement learning not an ideal candidate for training robots in practical applications, given that successful real-world computer agents and robots require robust control algorithms.

An alternative to reinforcement learning is learning from direct interaction with a human trainer. This approach opens new avenues to improve behavioral learning. In robotics and in virtual life-like agents, learning behaviors and/or actions is an important task. As an example, autonomous automobiles must somehow be taught to perform the actions necessary and appropriate to safely and efficiently conduct the vehicle to its destination on a network of roads while interacting with pedestrians and other traffic.

The focus of this dissertation is on learning from direct human interaction. More specifically we explore how a computer agent can learn through a natural conversation with a human. We explore how an agent can reason about the knowledge provided by the human to update its own knowledge base (KB) accordingly.

The best known learning approaches related to direct interaction are *learning from observation* (LfO), *learning from demonstration* (LfD), *behavioral cloning*, and *learning from instructions* (LfI). All these learning approaches seek to mimic human behavior to perform a given task. Learning from human interaction is beneficial when there is insufficient access to training data or when the behavior that needs to be learned is complex and requires multiple steps and high levels of accuracy to be able to attain. Learning from interaction with humans uses an available and reliable resource to train the computer agent directly - the human.

We next discuss some of the recent related research to provide the reader with a discussion of the background and related works in this field.

1.1 Learning from Observation

Learning from Observation (LfO) has been inspired by human social learning customs. A child starts learning new movements and how to speak mostly by observation of her/his caretakers [8]. LfO is the first learning process that a child develops watching the behavior of others.

By applying this same concept, an agent can imitate human behavior by observing how a human performs a task, and then repeating it. Traditional methods for training an agent to learn a task, such as reinforcement learning, usually require knowledge modeling and multiple stages of programming. LfO on the other hand, can observe the human behavior once and compose a policy that can be used to repeat the task, as well as possibly generalize the learned behavior. Additionally, LfO can apply to problems that do not have exact solutions. Thus, it is suitable for learning complex behaviors, such as driving a car [104], playing video games [100], helicopter control [34], robotic soccer [53], simulated soccer [46], flying an aircraft [126] and imitating a human's body motion [60]. Learning from observation has four conceptual levels of difficulty in learning [92]:

- Level 1 is strict imitation, where the agent needs to imitate the exact task observed in a precisely repeatable manner. This process requires only memorization and duplication, with no generalization or planning necessary. An example is a robot in a factory painting a particular piece in an assembly line, where the same piece will always be in the exact same position every time.

- Level 2 is reactive behavior, where the agent learns to react to events in the environment. The agent needs to learn generalization, but no planning is necessary to perform the task. It requires the agent to learn how to map the current state to actions. One example is learning to play older video games such as Pong or Pac-Man.

- Level 3 reflects tactical behavior in known environment. The agent needs to learn both

generalization and planning behavior. The agent also needs to remember things from prior states whose indications may no longer persist, but their implications may remain valid. For example, to play a strategy game, the player needs to keep track of previous opponent moves to determine the types of the remaining pieces of the opponent.

- Level 4 is tactical behavior in unknown environment. The agent needs not only to learn the task as in level 3 but also the environment, as it cannot make any inherent assumptions about it.

Most of the prior work in LfO have been at Level 1 and 2, with some newer work addressing level 3 and 4.

1.1.1 State of Art Review in LfO

The earliest work in learning from observation dates back to 1979 when Bauer [21] used a set of instructions to learn how to perform abstract computations. Following his work, many other researchers have used LfO to learn different tasks. For example, in 1995 Wang presented a method to learn planning operations using LfO by employing limited amount of interactions with humans [130]. Sidani and Gonzalez [116] presented a system named *Situational Awareness Module* (SAM) to learn implicit behaviors in automobile driving by observing a human expert drive a car simulator. The knowledge was observed and learned using recurrent neural networks. The system was successful in learning basic skills and it was able to generalize the learned skills to more complex scenarios. Nakaoka et al. [96] used LfO to teach a humanoid robot how to perform some dance movements. This framework allows a biped humanoid robot to use its legs to dynamically balance its body while learning the dance motions. The results demonstrate the system was able to imitate the human moves and maintain stability during dynamic-style steps. Fernlund et al. [41] argue that

leaning from observation is suitable for acquiring tactical knowledge for selecting the best action in a given situation. They developed the Genetic Context Learning (GenCL) system that uses contextual reasoning and genetic programming to learn how to drive a car. Their results indicate that an agent can generalize the learned behavior by performing the correct action on some previously unobserved situations.

Grollman and Jenkins [53] proposed a robotic soccer player that learns to play by observing a human player. Their approach was successful in learning actions such as moving to the ball and kicking, but it has difficulties learning when to transition between actions.

Recently, Floyd et al. [47] used deep learning to remove the need by the programmer to provide full information related to the environment by using virtual representation provided by the system. Their work uses the raw visual representation of the environment, which means the agent only has access to partial information about the environment. Their results indicate that although the system has access to partial observations, the agent can still learn the desired behavior given a full access to the set of possible actions and by observing the behavior of an existing RoboCup player. The agent observes the spacial configurations of the objects that the RoboCup player needs to be aware of, and then uses case-based reasoning to determine the correct action given similar configurations.

Oftentimes the learning process depends on the system's design. For example, Sammut and Michie [112] used inductive learning to teach a machine a reactive strategy for learning a complex motor skills by observing a human subject. The authors concluded that learning from a human that occasionally makes course corrections is more beneficial to the agent than learning from an expert, because the former helps the machine to explore different situations when things do not go as planned. In another work, Sammut et al. [112] used LfO to learn how to fly a simulated airplane. In their work, induction trees were built through observation. The trees decide the actions that need to be taken based on the sensor inputs. If the task has changed, the induction trees need to be

re-learned from new observations [127].

Some researchers combined LfO with other types of learning. For example Bentivegna et al. [23] used learning from observation as an initial learning mechanism and later used experiential learning to complete the learning process by using the observed data to further learn through practice. Stein and Gonzalez [117] built an effective learning algorithm named PIGEON to combine learning from observation and experiential learning using NeuroEvolution of Augmenting Topologies (NEAT) and particle swarm optimization (PSO). PIGEON uses learning from observation to learn how to perform a task and then uses learning from experience to improve its performance. The results reflect that using both learning techniques was more successful in learning a task than uaing each one individually.

Floyd et al. [45] presented an interesting approach inspired by how the brains of animals and humans respond to changes in sensors. Their work employs case-based learning by observing a human expert performing the task. Their system permits adding new sensors that are programmed by observation only. According to the authors, their system was successful in learning a new task (avoiding obstacles) by creating a new case from observation only. While their approach removes the need to reprogram the agent each time a new sensor is added, it still requires that the sensor itself be initially programmed in order to connect it to the existing agent.

Even though LfO has shown success in reproducing an actor's behavior in some domains, there are drawbacks with this learning approach. For example, it is not efficient where the environment is noisy and the agent cannot unambiguously capture the human's actions. Additionaly, there is a risk that the expert has a low level of competence, which results in limiting the agent's ability to learn the task correctly [137].

1.2 Learning from Demonstration

Learning from Demonstration (LfD) also known as *programming by demonstration* and *teaching from showing* [17] is very similar to LfO in concept. Ontanon et al. [101] argue that learning from demonstration is a special case of LfO because the former requires the human to purposely teach the agent how to expressly perform the same task by demonstrating it, possibly several times. One can also view the process of learning from demonstration as mapping between the world's states and actions from demonstrations and examples provided by the teacher. LfO, on the other hand, does not assume that the actor is teaching or even knows that he/she is being observed. Thus, LfO is always unobtrusive and does not allow any interaction between actor, agent, and human intermediary other than the arm's length observation. Another difference between LfD and LfO is that learning from demonstration usually involves physical interaction with the robot, for example when teaching the robot to close a box, the teacher may use different shapes and sizes to demonstrate that to the robot. This requires multiple interactions to make the robot learn how to do it. On the other hand, LfO allows the robot to observe how to perform the task without guiding the robot/agent through the process [44]. For this reason, LfO generally does not involve multiple demonstrations by the actor.

LfD applications usually share two common features [15]: 1) a teacher to demonstrate how to perform a task, and 2) a learner that needs a set of demonstration to derive a policy to reproduce the same behavior. Therefore, LfD requires two phases; collecting the demonstration examples, and deriving a policy. Gathering these examples usually requires multiple demonstration from the user, which can sometimes be impractical.

LfD has been applied mostly in robotics. The earliest form of LfD was reported by Brady when he studied motion planning and control aspect in robotics [25]. Another example is reported by Bentivegna and Atkeson [22] to teach a robot how to play air hockey. Atkeson and Schaal [17]

used LfD to teach a robot a reward function and a task model to drive a policy. The authors were interested in investigating learning strategies to teach a robot how to generalize a learned task to a slightly different task.

In an attempt to reduce the number of demonstrations required for learning a task, Alexandrova et al. [9] reported an investigation that requires only one demonstration from the user, and later used a visualization of the action to extract explicit information. Thus, generalizing a learned task involves teaching the agent all the possible actions and changes in the environment because the learnt action is associated with specific states. Forbes et al. [48] used crowdsourcing to gather actions from the crowd in an attempt to reduce the number of demonstration needed. Therefore, the teacher only needs to provide a single demonstration; the robot uses crowdsourcing to update the action for application to different scenarios.

Some research has been done to combine LfD with other types of learning, For example, Brys et al. [28] used reinforcement learning and LfD to quciken the learning process by using fewer demonstrations. The process, called *reward shaping*, uses previous knowledge as part of the learning process. This process results in having many intermediate rewards upon completing each given task.

Learning from demonstration has been applied recently in different applications, including but not limited to mobile applications. For example, Li et al. [80] proposed EPIDOSITE that uses LfD to program Internet of Things (IoT) devices.

A common drawback of the reported research in LfD and LfO is that most of them have been tested in only one domain; therefore, they involve solving specific problems. This makes a direct comparison between different approaches within LfD and LfO impossible [15].

1.3 Learning from Instruction

Learning from Instruction (LfI) depends on a lesson-based learning protocol instead of using an example-based learning protocol [50]. LfI allows the user to issue instructions in natural language to direct a computer agent about how to perform a task. The instructions needs to be clear and direct. Therefore, the user has limited ability to modify the instruction or use different words. Furthermore, the task needs to be defined a priori and there is an ultimate goal that the agent needs to attain, e.g., making coffee, delivering a paper, etc. Because LfI is highly relevant to this dissertation, we leave an extensive discussion of the work conducted in this field for Chapter 2 of this dissertation.

1.4 Learning from Conversation

One of the natural ways in which humans learn is through conversation with other humans. This is similar to traditional formal education, where a teacher teaches a class mostly by speaking to them. This approach also holds much promise for teaching machines. Yet, the literature contains few works that take advantage of this concept. Our work centers on capturing knowledge from a human interlocutor by a machine in the course of a casual conversation between them. We refer to our approach as *learning from casual conversation* (LCC).

LCC works similarly to how humans learn from each other through a casual conversation. In the course of a conversation, humans are able to learn new concepts and update existing information. Furthermore, if we have pre-existing knowledge about conflicting information, we process the information by either trying to convince the speaker of a mistake in the spoken information, or we adopt the new information if we become convinced that it is correct. Following the same process, LCC will update existing knowledge and add new information when appropriate. This process

enhances the knowledge of an agent with the new learned information.

We propose LCC to be a part of the overall area of human interaction learning. We can summarize the differences between LCC and other existing learning techniques as follows:

- Existing learning from interaction models, such as LfO, LfD and LfI are applied only to goal-oriented tasks, where the agent has to learn how to accomplish a procedural task through specific and limited number of interactions. In contrast, LCC continuously learns declarative knowledge during the course of a natural conversation, with no restriction to the number of interactions with the human.

- Because LCC learns from conversation, the system needs to interpret the human's speech in detail to be able to relate the knowledge acquired from the human to the system's existing knowledge base.

- To ensure that LCC only considers true information to possibly effect its knowledge base, trust evaluation is used to determine the trustworthiness of the user, as it is very important to not allow an unqualified person to change existing information in the learning agent's knowledge base (KB).

In the next chapter, we discuss the published works relevant to this approach. However, we first precede that discussion with one about conversational agents, as those will be required in order to learn from conversation.

CHAPTER 2: CONVERSATIONAL AGENTS AND RESEARCH CLOSELY RELATED TO LCC

Conversational systems have rapidly increased in popularity because of their ability to provide assistance to users in many applications, such as telephone interactive systems that use spoken natural language to provide assistance to callers to banks, airlines, restaurants, etc. As a result, researchers have sought to improve these systems to better satisfy the user's expectations, and convince the user to not ask for a human agent, which can be costly. Yet, many of the existing automated agents are designed for fixed tasks with fixed capabilities. Additionally, any software update requires significant effort from the programmers.

In general, the existing conversational systems have been classified into two broad categories: 1) chat-oriented systems used for entertainment purposes [19] [118]; and 2) task-oriented systems designed to help the user accomplish specific tasks, such as making flight and/or restaurant reservations, or provide information about specific topics [65] [134]. Chat-oriented systems have been considered more challenging than task-oriented systems because the former require a broad range of utterances to cover user inputs. Task-oriented systems on the other hand, only need a limited number of utterances because the tasks are typically limited to narrow topics.

2.1 Conversational Agents and Chatbots

There is a long history of studying conversational agents and chatbots. The earliest known chatbot was ELIZA [132], which was designed to emulate a Rogerian therapist. To provide its responses, ELIZA found keywords in the user input, and utilized them with scripts selected through pattern-matching to turn the user input around. This required little or no contribution from the program.

Hence, to generate a response, ELIZA is only concerned with identifying these keywords to transform the user input appropriately into a response. It also considers how to generate a response in case of absence of keywords in the user's input. Additionally, it makes sure not to reach the limit of its capacity for the scripts. Yet, ELIZA does not understand the meaning of each word; in fact, it maintains a minimal understanding of the user scripts. For example if the user employs the keyword "mother", it responds with "tell me more about your family". This rule is based on the fact that mother and family are related and could be part of the patient's psychological problems. As a result, ELIZA often sounds intelligent at first, but it invariably produces repetitive and/or incoherent responses as the conversation continues. An example of a conversation with ELIZA from [114] is as follows:

User: Men are all alike.

ELIZA: In what way?

User: They're always bugging us about something or other.

ELIZA: Can you think of a specific example?

User: Well, my boyfriend made me come here.

ELIZA: Your boyfriend made you come here?

User: He says I'm depressed much of the time.

ELIZA: I am sorry to hear you are depressed.

PARRY is another early chatbot that was designed to simulate the critical thinking of a paranoid patient [87]. It uses more advanced tools than ELIZA, such as initiating new topics, asking questions and admitting that it doesn't know the answer when there is no suitable response in its knowledge base. Both PARRY and ELIZA apply tricks, such as directing the conversation to another direction by asking questions or by using the user input to be part of its responses to convince the user that it does understand his/her input [114].

An early attempt to integrate learning within the conversational system was seen in Jabberwacky

[87]. This system saves all the conversation and makes new conversations by using the learned input when the context of the conversation matches previous contexts from where the user input was learned. Jabberwacky is unable to generate its own responses as it has no knowledge of the grammatical rules of any language. Therefore, the program mirrors the user input and uses it in its future conversations. The program won the Loebner prize [85] for two consecutive years in 2005 and 2006. An example dialog with this system is shown below [87]:

Jabberwacky: Are you a participant or a judge in the great competition?

User: Hello.

Jabberwacky: Hello kitty!

User: How did you know that I was a cat?

Jabberwacky: You told me and you told me that your name was chaos jill.

User: No, my name is Ichi. I'm a Japanese Maine Coon cat.

Jabberwacky: I am a Japanese movie monster too.

User: Which one?

Jabberwacky: Which one? I don't know which one! Which one of what?

User: Are you Gamera or Godzilla?

Jabberwacky: Yes.

User: Hi Gamera! Doesn't it get hot inside your shell?

Jabberwacky: Not if you're with someone you connect with.

From this dialogue, we can infer that the system is good in introducing new topics and being able to lead the conversation. However, its answers lack consistency and are sometimes unrelated to the user's previous utterances.

In 2005, ALICE was introduced to the AI community [129]. ALICE stands for an Artificial Linguistic Internet Computer Entity, and it used Artificial Intelligence Mark-up Language (AMIL) to

store its knowledge base. AIML consists of data objects that are known as AIML objects [114] that are formed from units called *topics* and *categories*; the topic has a name and a set of categories are associated with that topic. Categories consist of patterns that are used to match the user input to a template. ALICE's answers can be inconsistent because it does not keep track of previously said information. An example of a dialog with ALICE from [114] is shown below:

Judge: Hi Alice.

ALICE: Hi there!

Judge: How's it going?

ALICE: I'm doing good thanks how are you?

Judge: Pretty good.

ALICE: Oh no, not that well. I have a long way to go yet.

Judge: Where are you going?

ALICE: I am going into the future.

Judge: Aren't we all?

ALICE: We are all very similar in many ways.

Judge: True.

ALICE: Yeah that's right.

Judge: Do you like being a computer?

ALICE: Being a program is the only existence I know.

Judge: Being a human is the only existence I know.

ALICE: I know you are.

Judge: Would you like to be a cat?

ALICE: No thanks I am happy with who I am.

Judge: I'd like to be a cat for a day.

ALICE: Interesting. I would like to be more intelligent.

ALICE answers were more relevant to the user utterances than the previously discussed systems [87] [132]. This reflects that ALICE can show some understanding to the user input. Yet, ALICE answers are often inconsistent because ALICE lacks some important factors, such as using memory to remember previous responses; therefore, it could answer the same question with two different answers. Nevertheless, the given example above shows some improvement in producing better answers when compared to Jabberwacky and ELIZA.

Sgt. Blackwell [78], Sergeant Star [16] and Hassan [124] are basic pre-programmed question answering systems designed to answer specific questions. Sgt. Blackwell is a biographical system that can answer questions related to the system itself, its training and questions about what it is capable of saying. Sergeant Star and Hassan are both successors of Sgt. Blackwell, where Hassan plays the role of an Iraqi role-player that helps train soldiers on how to interact with Iraqi villagers, and provides feedback on their performance. Hassan uses a simple classifier to choose its answer, and a dialogue manager to avoid repeated answers. It also uses emotions that affect how it answer future questions. Even though using emotion is a new component to the previous architecture of Sgt. Blackwell, it was one of the reasons that Hassan produced irrelevant responses. Test results show that users' satisfaction of the system's performance was 3.4 on a 7 points Likert scale.

Sergeant Star acts as an army officer that can answer general questions related to the army. It has limited number of responses, which are all pre-generated. It also has the ability to open new conversations by going off topic.

IBM Watson developed DeepQA [131] that was presented as a question-answering system to play the Jeopardy TV show. DeepQA is a massively parallel probabilistic evidence-based architecture that involved more than 30 researchers for three years. The key idea behind the success of DeepQA is to use a massively parallel architecture and a dedicated high-performance computing infrastructure. DeepQA won a Jeopardy competition by defeating two former champions.

In 2016, Amazon launches Alexa Prize competition that involves choosing 16 teams from different universities to create a social bot that can communicate efficiently and effectively with humans on different topics for 20 minutes. The competition prize is US $2,500,000. Amazon provides the teams with real data that can be used to develop their system in addition to resources, technologies, and personnel for Alexa that can be used as baselines for their projects. The competition does not focus on the Turing test [125]; the human knows that he/she is conversing with a computer. Yet, the goal is to determine which conversation is more natural, interesting, coherent, and can keep the human engaged in the conversation. Alexa users are the evaluators for this competition. Alexa users can contribute in the testing of the social bots by saying "Alexa let's chat" or "Alexa let's chat about <topic>". Then Alexa will randomly choose a chatbot and instruct the user to how he/she can end the conversation. At the end of the conversation, Alexa asks the user to rate the system using a Likert scale between 1 - 5, where 1 means dissatisfaction and 5 means satisfaction. Furthermore, it asks the user if he/she wanted to leave feedback for the team.

Conversational systems can be further divided according to the common features they share. Hence, we next give an overview of recent research conducted in this area.

2.2 Multifunctional Systems

Researchers have introduced multifunctional systems to build effective applications that can assist users in more than one task using the same interface. Planells et al. [103] introduce a multi-domain dialog system that combines three pre-existing dialog systems: personal calendar, sporting event booking and weather services. These sub-systems are independent from each other. Thus, the dialog manager communicates with them through a fake user. For example, to close the current domain and activate another one, the dialog manager sends a fake message to the active domain implying that the conversation is finished. The system's importance comes from its ability to gen-

eralize by including more sub-systems without needing to change its underlying structure because each system thinks it is the only one in the architecture. Test results indicated that the system's accuracy decreased as more tasks became involved in the dialog, from 94.3% for one task to 76% for all the three tasks.

Banchs et al. [18] present AIDA (Artificial Intelligent Dialogue Agent) that has six different dialogue engines, each of which is responsible for answering questions related to a different topic. For example, restaurant reservation, flight reservation, time checking, and answering questions related to a specific institute related to the project. The appropriate task and engine are selected by a user intention interface model, which is a component responsible for making decisions and selections among the six available engines.

DaHaro et al. [39] introduced a multifunctional conversational system called CLARA that uses a natural language search mechanism to combine two applications: one that provides information about a conference and another that is a tourist guide agent. A server communicates with the search modules in the system architecture to provide information to the user based on different resources, including databases, dictionaries and models. In tests with human subjects, the authors reported that the system failed to answer about half of the users' queries because most of the questions were out-of-domain.

The common element in AIDA [103] and CLARA [39] is that at each dialogue turn, there is only one active sub-system. However, activating one system at a time could have both positive and negative effects on system performance. Positive because handling one topic is generally easier than handling more topics at the same time. Yet, this can result in producing wrong answers when the user asks questions that fit more than one system. It also becomes a problem when a misclassification occurred and the user input was assigned to the wrong system. In this case, the system is guaranteed to produce a wrong answer.

Additionally, all the previously discussed models have simple dialogue managers because the systems are task-oriented, which limit the user to a small number of utterances. Therefore, these systems do not require large databases.

2.3 Multi-Models Systems

In this section, we describe works that use more than one method to generate the system's responses. One recent example is a chatbot reported by Nio et al. [99]. The authors compare and contrast the performance of a chatting system that uses a statistical machine translation model (SMT) with the chatting system performance using a combination of two example-based dialogue manager (EBDM) methods: 1) syntactic-semantic similarity retrieval, and 2) TF-IDF[1] based cosine similarity retrieval. EBDM provides proper responses using dialogue examples that are semantically interpreted to a database while SMT can generate related responses to the user input, even if it has not been trained on similar responses [99]. The authors report improvement of TF-IDF cosine similarity metrics after applying a semantic similarity filter. Using this filter not only improves the system performance but also minimizes the time required to provide responses because it reduces the examples in the training set [99]. Subjective evaluation of the naturalness of the generated responses did not indicate any improvement after combining both models. In fact, using syntactic-semantic similarity retrieval was rated to have slightly higher values than combining both models. Objective evaluation achieved better results when both systems were combined.

SARA was introduced by Niculescu et al.[97] as a multi-modal dialogue system that provides tourist information on a mobile application. SARA's dialogue manager consists of two different strategies to determine system responses: a rule-based approach and an example-based approach.

[1]TF-IDF stands for Term Frequency-Inverse Document Frequency, a numerical measure that reflects the importance of a word in a document or corpus.

The most similar response to the user input is selected based on the cosine similarities, and by using TF-IDF weights [97]. The system responses were evaluated by human subjects in terms of usability, reliability and functionality. The system's responses were tested in scenarios that involved either general or specific questions. Answers related to general information achieved 60% of objective completion rate, while asking questions related to specific questions scored 33%.

Shibata et al. [115] present a chat-like conversational system that has multiple reply generating modules. Each module has a specialized action, such as selecting a sentence from a Web news source, question-answering, finding a definition in Wiki pages, etc. Each module generates a response for each input from the user. It works by selecting the appropriate module instead of learning the best system response.

Morbini et al. [93] created a flexible multi-initiative dialogue manager known as FLoReS (Forward Looking, Reward Seeking) that has a set of operators that are responsible for controlling the dialogue flow. Each contains a sub-dialogue structure represented by a tree consisting of system/user action and the resulting state after taking that particular action. For each state in the sub-dialogue, there is a reward to reach that state. Based on the received reward, the system decides the appropriate operator. There were no results reported for the FLoReS system. However, Morbini et al. [93] mentioned that users' feedback indicates their satisfaction with the system's performance.

Generally, multiple sub-systems can help generate relevant responses to the user input because each sub-system responds to its relevant topic. Moreover, in all the proposed models, each sub-system generates a response, but only the one that has the highest score is chosen to produce the system feedback. However, this could be a drawbacks considering a situation when more than one system can generate a reasonable response. Therefore, it is more practical to create a system that has multiple models that work together to generate one effective response.

2.4 Systems Designed to Avoid Out-of-Domain Responses

Out-of-domain responses have been a problem reported with many conversational systems. Therefore, in this section we discuss several approaches used to address this issue.

Yoshino et al. [136] introduce a spoken dialogue system that also uses data filtering to extract information from the Web to generate its responses. The extracted information is represented by a predicate-argument (P-A) structure. However, to avoid producing irrelevant responses and to ensure that the dataset contains useful information, the authors use TF-IDF and a Naïve Bayes model to measure the importance of a word in a given domain or topic. Results indicate that the system was able to match the answers between the Web and the user input for only 30% of the cases, and it did not provide any answers for 68%. These results are even worse when the input to the system used a speech recognition system, as the correct answers dropped to 19.4% and the system did not provide responses for 79% of the cases. However, using back-off models to produce partial matching improved the correct responses to 66.2%.

Another way to identify out-of-domain responses is to relate the system responses to the topic extracted from previous dialogue turns. Based on this idea, Sugiyama et al. [118] proposed an open domain conversational system that uses template filling of the most relevant words from the user utterances with related words extracted from Twitter using web-scale dependency structures to generate its answers. However, the authors realized that using a template-filling approach did not generate appropriate responses in some cases. Therefore, they used another utterance generation model based on Dialogue Acts (DA) [2]. The system performance was measured by 10 test subjects using a 7-point Likert scale that evaluates whether the responses were valid and whether the users had the desire to use the system again. The overall average score was 4.05 compared to two

[2]Dialogue act (DA): is an expression that denoted user intention, such as greetings, confirmation, Wh-questions, etc.

retrieval-based approaches of Ritter et al. [109], which averaged 2.93 and 3.5 scores on the same set of questions. This reflects that using a template-based approach could improve the ability of an agent to produce more meaningful responses considering that this approach uses the most relevant words form the user utterances to match while producing an answer.

Higashinaka et al. [57] presented an open domain conversational system that is fully based on natural language processing (NLP) modules. This system considers intention, topic and content of the sentence when generating the system responses. The system performance was compared to that of a rule-based model and a retrieval-based model. The system performed similarly to the rule-based model, as the average score given for the system was 3.3 based on a 7-point Likert scale, while the rule-based model scored 3.7. Retrieval had the lowest score at 2.7.

Both systems by Sugiyama et al. [118] and Higashinaka et al. [57] used topic extraction from user utterances to ensure having coherent responses; however, each system used a different part of speech to extract the potential topic. Sugiyama et al. [118] extracted the topic from proper nouns, common nouns or predicates while the system in Higashinaka et al. [57] used only noun phrases to determine the topic. Higashinaka et al. [57] used several modules to generate its answers, which makes it less likely to have the problem reported in [118] of not being able to generate responses in many cases.

Out-of-domain responses can arise when the system misunderstands the user utterance. This is common when using an Automatic Speech Recognizer (ASR) during communication with the users because most ASR systems have high error rates. One solution was reported by Hung and Gonzalez [59], where the authors introduced CONCUR, a conversational system that provides responses based on understanding the context (speaker intent) of the conversation instead of trying to understand the exact syntax of the user utterances. Doing this eliminates some of the risk associated with trying to understand the complete utterances to provide the system responses. CONCUR

achieved a high usefulness score of 60.5% when the average ASR word error rate (WER) was high 58.5%. Usefulness score measured the system's ability to be considered as a useful tool to provide information and being able to accomplish tasks in a productive manner. These scores are compared to the scores of another system known as the Virtual Kyoto agent, which scored 61.4% with WER of 29.4%, making the CONCUR usefulness score nearly similar to that of Virtual Kyoto, which had a word error rate less than half of that seen by CONCUR.

2.5 Different Uses of Conversational Systems

This section presents an overview of some notable applications of conversational systems. Kim et al. [65] designed an electronic programming guide for TV. The system automatically builds its database, as it has online access to periodically update the data through a web-mining module [65]. The gathered data from the Web and the saved dialogue examples in the database are used to build the language model. The results imply user satisfaction from the proposed model because their score was 0.73 using a scale from 0 to 1, where 1 indicated full satisfaction.

Another application for conversational systems is in human-robot interaction (HRI) [68]. This work applied a conversational system to two scenarios: imitation games for arm movement and quiz game using a robot. The system was able to work as intended but it was not sufficiently robust for untrained users.

Another example of HRI systems was reported by Woo and Kubota [134]. This work involves agents acting as social companions to exercise for elderly people who live alone. This project creates conversations between a robot partner and a human using multimodal perception [134]. This system proposed a new idea, which is the ability to generate utterances based on the robot observations. The robot's ability to communicate improved as more utterances were added to the

database from chatting with the human. Yet, this system has no ability to distinguish between users because the robot does not use syntax analysis [134]. Additionally, because the robot memorizes human utterances and repeats them, the robot can use sentences that could be very personal.

2.6 Memory Systems

Learning cannot be accomplished without some form of memory to remember previous events and utterances. Therefore, employing memory in the system is important in the learning process and also in producing coherent responses that improve the system's general performance. Therefore, in this section we provide a brief overview of some conversational systems that use memory as part of their architectures.

The earliest and simplest form of memory was seen in the ELIZA chatbot [132]. ELIZA saves the user's previous utterances and used them in its future responses. This was followed by many other systems that used memory as an integral part of their architectures. As a recent example, Laird and Rosenbloom [74] proposed SOAR, one of the earliest cognitive architectures with the objective of creating a general computational system with human-like cognitive capabilities. Its architecture includes semantic memory composed of declarative and episodic memory.

Banchs et al. [19] introduced IRIS (Informal Response Interactive System), a chat-oriented dialogue system that learns new concepts from users and semantically relates them to its previous knowledge. The system saves profiles of previous users to recall prior conversations, and uses them to chat with the user. IRIS considers user feedback to improve its future responses by allowing the user to rate the system's responses. The main objective of IRIS was to generate relevant responses to the user utterances. However, it has no ability to revise existing knowledge. No results were reported about IRIS performance other than some dialogue examples that reflect where IRIS

performs well and when it fails to produce reasonable answers.

Kim et al.[66] presented a spoken dialog system that uses long-term memory to save the user's previous utterances and use them later as part of the system responses. This is done by collecting the user facts in term of triples (arg_1, relation, arg_2), and saving them in memory. Hence, the system uses natural language processing tools, such as part-of-speech taggers (POS), dialogue act classifiers, and knowledge extractors to process the system input to extract the triples. The system was tested by measuring the ratio of reasonable responses. Results improved significantly from the baseline (i.e., similarity of input and output) score of 57% to 75% when a relevance score measurement was used.

Bang et al.[20] introduced a chat-oriented dialogue system that combines EBDM with the personalized long-term memory proposed in [66]. Additionally, this system also uses three features: 1) POS-tag to match the sentences; 2) named entity (NE) types and values to search for the appropriate response; 3) back-off model to provide responses to unmatched user's sentences with the examples in the database.

The work was evaluated by using different combination of its components, i.e., baseline that uses simple lexical similarity to find similar examples; a system that uses POS; a system that uses NE; and a system that uses both POS and NE. The results were also compared against those of ALICE [129]. Eight users were asked to interact with all the variations of the system and thereafter rate the system performance using a scale from 1 to 5. The results reflect that the system that uses both NE and POS has slightly better rating (3.7) than ALICE (3.4).

Even though using memory showed some improvement in producing relevant responses, it did not prevent the systems by Kim et al. [66] and IRIS [19] from producing meaningless results. This is because of noisy examples in the database. Thus, filtering the data is important to solve this problem. Moreover, IRIS [19] suffers from another problem of not being able to maintain

consistency with its previous answers. IRIS [19], however, has an advantage over the systems by Kim et al. [66] and Bang et al. [20] by having a back-off model to provide answers when no matches are found with the user input.

Like humans, a memory system must choose to remember what might be important and forget the rest of the conversation. This idea was applied by Elvir et al. [40] in an algorithm that can extract important words from the sentence and remember them as episodic memory. This process involves three issues: what to remember, how to store the information to be remembered and what to retrieve during a conversation. Their results indicate partial success in several aspects of applying memory to embodied conversational agents.

2.7 Machine Learning in Conversational Systems

In this section, we highlight recent works in conversational systems where machine learning plays an important role in their architectures. Machine learning has been used widely for estimating the next user/system action and the transition state. This is mostly applicable to task-oriented models, as the transitions and actions are considered deterministically. One example is a work presented by Lison [81] that uses model-based Bayesian reinforcement learning to estimate transition models for dialogue management. The system uses Partially Observable Markov Decision Process (POMDP) to teach the system which action to take by interacting repeatedly with the user. This work was evaluated using a human-robot interaction scenario. The experiment was applied to a robot that was asked to perform some tasks, both verbally and virtually. The user utterances are limited to 16 predefined dialogue acts and the robot has a limited number of actions, including physical and conversational actions. The evaluation was performed on both multinomial distribution and on probabilistic models. Both models show improvements in estimating the transition during interaction with the simulated user. However, the probabilistic distribution model converged faster than

the model with multinomial distribution because it was better able to capture the domain structure with limited number of parameters.

Machine learning has other applications, such as predicting the dialogue acts for future responses. Thus, Yoshimura [135] proposed a conversational system that generates casual responses using large-scale data. In order to generate the system responses, the system needs to understand and analyze the aim and the context of the user utterances. This system generates the utterances using data extracted from the web, which are in the form of nouns and their corresponding predicates (e.g., predicate (eat), noun (bread)). The nouns and their predicates are extracted based on common human knowledge. It is difficult to say how well this system performs because no results are reported [135].

Machine learning in the form of reinforcement learning has been applied by Shibata et al. [115] (discussed previously in Multi Models Systems section) to learn the best strategy that the system can follow for future responses.

Through our discussion of the works presented by Lison [81], Yoshimura [135], and Shibata et al. [115], we have seen that machine learning algorithms have been applied to predict future actions, tags, topic, and transitions. The ability to predict future events can help improve the system performance because the prediction is based on previous events.

2.8 Cognitive Architecture

Cognitive architectures deals with the structure of the human mind. The goal of developing cognitive architectures in a computer agent is to create an architecture similar to that of a human. The goal of having such architectures is to provide a baseline that is necessary for intelligent computer agents to learn a wide range of tasks and procedures. Additionally, these seek to learn different

types of knowledge, similar to the ability of humans.

The State Operator and Results (SOAR) cognitive architecture was conceived by Laird at el. [72]. It can be considered the earliest form of a cognitive architecture, although it has evolved to add more components over the years. SOAR had been designed to be able to 1) work on a wide range of tasks - from highly routine to difficult open-ended problems; 2) provide a full range of problem-solving methods and representation needed for the desired task; and 3) learn about all the aspects related to the task [72]. The earlier versions of SOAR consisted of pure symbolic processing with long-term knowledge being presented as production rules [72].

Figure 2.1 shows a block diagram of the earliest version of SOAR from [72]. The main components of this architecture are the working memory and the processing structure. The working memory holds SOAR's processing state for problem-solving. This memory has three components: 1) a context stack that specifies the hierarchy of problem spaces, states, operators, and active goals; 2) objects that holds goals and states; and 3) preferences that encode the procedural knowledge. The processing structure consists of two components: 1) a production memory that consists of a set of productions that can test any part of the working memory; and 2) a decision procedure that examines the context stack, changes the context stack and examines the preferences.

In addition to those components, there is a working-memory manager that manages the working memory by deleting elements, and a chunking mechanism that is responsible for adding new productions. A more detailed description of each of those components can be found in [72]. The most recent version of SOAR [71] adds reinforcement learning, semantic memory, episodic memory, mental imagery, and a model of mental emotions as shown in Figure 2.2 from [71].

The SOAR architecture has been reported to be used mainly in four areas: cognitive models [77], autonomous agents [84], expert systems [82] and human behavior models [58].

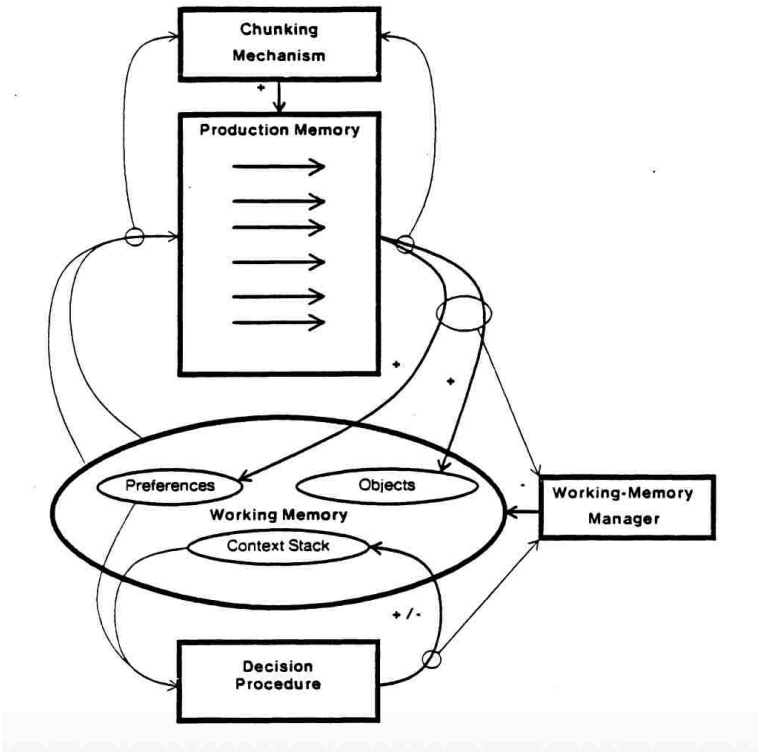


Figure 2.1: SOAR Original Architecture (reproduced without permission from [72])

Using SOAR in Cognitive models: SOAR was first introduced as a problem-solving platform that can be used in a different application, but because of its characteristics, it has also been used to model human behavior. Some models have been designed to understand human behavior, such as the work by Chong and Wray [62] that used SOAR to understand human performance in air-traffic control.

Using SOAR in Autonomous agents: SOAR has provided a good environment for research about agents that include reasoning and planning; therefore, it has been used for autonomous agents that interact with an external environment [71], such as using it in simulated soccer team [84].

Human Behavior models: those models observe the knowledge that humans use to perform a task.

Because SOAR uses functional constraints inspired by psychology and can perform tasks in real time, it has been combined with human behavior models [71], such as using SOAR in TacAir-SOAR (a combat pilot) [73].

Expert Systems: computer programs that use knowledge or data collected from human experts and apply it in the same domain to solve problems or answer questions, such as diagnosing a disease. LG-SOAR is an example of expert systems that have been used in information-extraction applications, such as in formulating queries for assessing clinical trial eligibility [83].

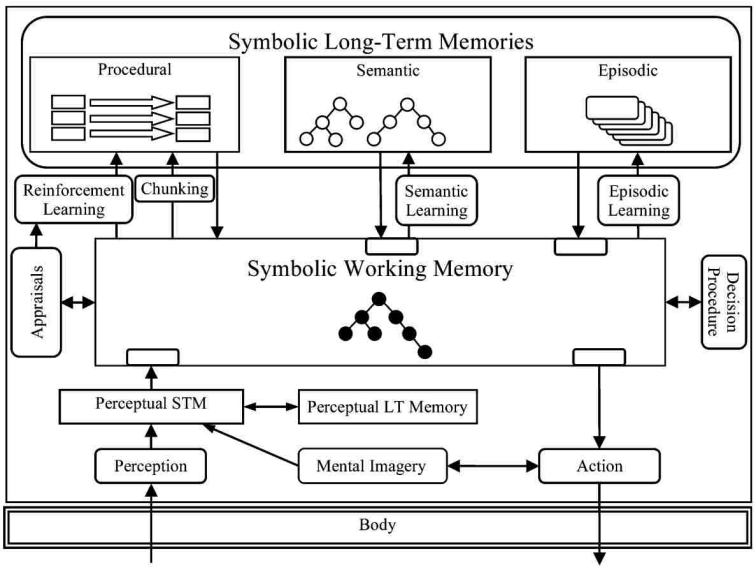


Figure 2.2: SOAR Most Recent Architecture (reproduced without permission from [71])

The adaptive Control of Thought—Rational (ACT-R) system is another cognitive architecture that was developed by Anderson [11] to be a model of high-level cognition. Figure 2.3 shows ACT-R architecture from [30]. Its architecture contains two main modules: a procedural model that deals with the interface, and a memory module that has two types of memory: 1) procedural, which contains information about how to perform a task, and 2) declarative, which includes information

about facts, such as “Three plus four is seven.” ACT-R declarative knowledge is represented in term of chunks that use a schema-like structure that uses pointers to represent the category and the content encoding [13]. An example of the chunk encoding from [13] is shown in Figure 2.4. Production rules in ACT-R are used to apply declarative knowledge to solve problems. Those roles are in the form of if-then statements that apply declarative knowledge to solve sub-problems. The goals unit keeps track of the current goal of the model. The aural and visual units accept the input from the environment and encode it as chunks of declarative knowledge. The oral and manual unit contain mechanisms that enable the cognitive model to act on the environment.

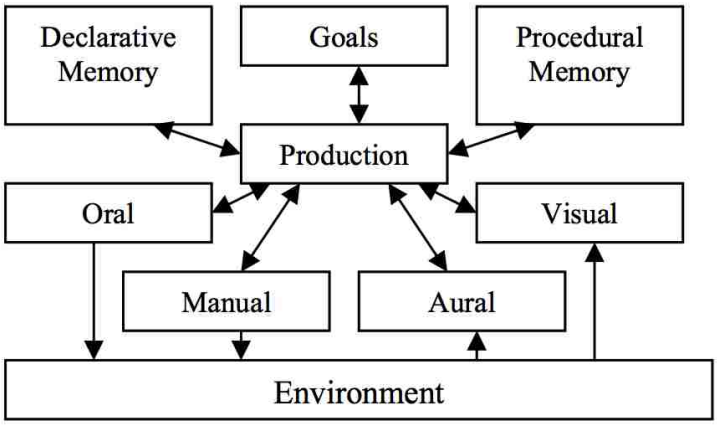


Figure 2.3: ACT-R Architecture (reproduced without permission from [30])

There are many applications that had been conducted using the ACT-R architecture, such as, language learning [108] [26], graphical user interfaces [55], games [12] and intelligent tutoring assistants [67].

Executive Process-Interactive Control (EPIC) is another cognitive architecture that focused on modeling human cognition and exploring the limitation in human performance that helps to explore the effects of a specific interface design [64]. EPIC’s contributions have centered in modeling

the interaction among cognition, perception and action [70].

Connectionist Learning with Adaptive Rule Induction On-line (CLARION) is another cognitive architecture, focused on modeling different interactions that can occur with human cognition, including cognitive-metacognitive interaction, implicit-explicit interaction, and cognitive-motivational interaction [121]. CLARION is designed to capture all the important cognitive processes within the cognition of an agent.

There are many other cognitive architectures reported in the literature, but they are beyond the scope of this discussion. The main point of the discussion about cognitive architecture is to show the reader that human behavior has been studied from many perspectives, such as the research in cognitive architecture that focuses on the cognitive architecture of the human brain. Yet, LCC studies human behavior in term from learning while interacting with other users.

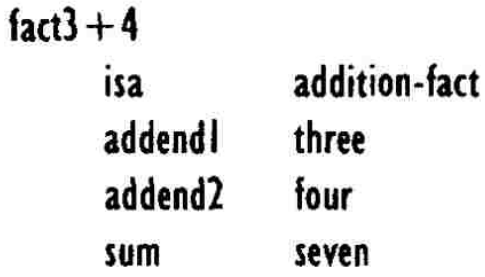


Figure 2.4: An Example of ACT-R Chunk Encoding (reproduced without permission from [13])

2.9 Learning from Instruction

Now that we have seen the state of the art in conversational agents, we next discuss how such technologies have been used as a vehicle for machine learning of behaviors and actions. The main

area in this is called Learning from Instruction or LfI.

The earliest form of LfI provided feedback to a system as an additional resource for learning. For instance, Kuhlmann et al.[69] used instructions to provide feedback in plain English about the system's performance while using reinforcement learning in RoboCup soccer. Torrey et al. [123] used reinforcement learning to learn how to move in a KeepAway game, and later used verbal advice to transfer the learned task to a new environment. Transfer advice involves determining which actions are best in a given situation. Those actions should be related, i.e., they can be transferred to a second task that is very similar to the first one, with only minor modifications. For example, in the KeepAway game [123], the task was to pass the ball to the nearest teammate rather than give it to an opponent, while the transferred advice was used in a BreakAway game, which involves passing to the nearest teammate if a defender is near.

LfI requires identifying the instructions explicitly, including their post conditions, actions and goals. One example is a work proposed by Rybski et al. [111], which is interactive task training for a mobile robot. In this system, the learning process is a mixture of learning from demonstration and learning from instruction. The user needs to verbally specify when the robot needs to start learning. For such reasons, learning from instruction requires syntactical constrains on the user utterances. An example from [111] is shown below:

> When I say deliver message: if person1 is present, give message to person1; otherwise, report message delivery failure; go to home.

Therefore, learning from instructions depends on the efficiency of the communicated knowledge, which could result in providing wrong instructions if the communication environment is noisy. Improving this process is not a trivial task [50].

To alleviate this problem, Mericli et al.[88] presented an approach to help improve the communi-

cated information by using a keyword-based filter to search for specific words in the user's speech to execute the commands. However, these keywords need to be in the correct order and the system does not accept synonyms for the same words, which is impractical. For example, "Go forward 10 meters" and "It is 10 meters away" are not considered to be the same instruction. The system operates by accepting the user input and parsing the instructions to convert them to a graph-based representation. Later, it passes the generated representation to the robot to execute the task [88]. The authors used learning from instruction to teach the agent to modify existing instruction by requiring the user to specify that explicitly. Such an approach can be useful when the goal is to teach the agent a specific task and the user is who decides what needs to be changed. However, we believe that learning should occur in a more natural way, i.e., the system should be able to convert its knowledge when it believes that the provided information is more accurate than what it already "knows". Our approach, therefore, involves studying how to make a computer learn from the user without giving the user the ability to decide what needs to be learned.

Learning from incomplete, noisy instructions in an undefined environment has interested many researchers. For example, the research of Grizou et al. [52] involved proving that learning from noisy and unknown instructions is possible while learning a new task. The unknown instructions correspond to "correct" or "wrong". The robot maps the unknown words to a binary feedback signal that is previously provided by the teacher. The system also examines how knowledge from previous experiments helps predict the meaning of the unknown instructions. The system selects the best hypothesis that describes the correct meaning. published results reflect that the system successfully inferred the meaning of the given instructions in few iterations.

The work of Misra et al. [90] handles long instruction sequences and missing natural language information. For example, in the instruction of "heat the pot", it is missing that the pot needs to be in the stove first. Such information can be inferred from the environment by considering previous similar instructions in the training dataset. The model outperforms two other existing models,

Instruction-Tree model [24] and Predefined Templates [54] with an accuracy of 61.8%.

Cantrell et al. [29] presented a robot that learns unknown actions from natural language constructions by allowing a planner to combine the new information with the existing knowledge to understand the meaning of the coming instructions. The system deploys two techniques to handle new plans by either removing the existing plan and starting from scratch, or updating the plan by removing, adding or updating an action. Hence, the system was not able to handle input from users who did not know the system's capability.

Other researchers have looked at how to generalize the learning process. For example, Volkova et al. [128] presented a procedural dialog system that learns from task-oriented textual resources using light, non-expert supervision. The system is quite flexible because it allows the users to change their goals slightly before completing the original task. Following a similar direction, Mohan and Laird [91] describe a system that learns to generalize its performance to include different variations of the task by interacting with humans using natural instructions. Mohan and Laird system uses SOAR architecture as the basis for their work. This work demonstrates that the learned knowledge could be transferred by applying it to different tasks that have similar structures.

Grizou et al. [51] proposed an interactive learning system that can learn a task from unlabeled instructions. The system requires a user's feedback to acknowledge whether the movement taken is correct or not. However, the system has limited knowledge about what the goal should be and does not know a priori what "correct" and "incorrect" words mean. Therefore, it needs to determine through trial and error what the goal is and what are the right movements to attain it.

Recently, Khandelwal et al. [63] introduce a human-robot interaction system that is called BWI-Bots. The robot is part of a project at the University of Texas called Building-Wide Intelligence (BWI). The long-term goal of the BWI project is to create an autonomous robot that can better serve visitors inside a building, able to perform complex actions and a long sequence of commands. The

BWIBots was implemented to identify shopping request tasks. The robot has the ability to ask clarification questions and update its belief about the human's request. This represents the learning process in this system. After the robot identifies the object that needs to be delivered, it will be ready to deliver that object. An example from [63] is shown below:

ROBOT: How can I help?

HUMAN: please bring the item in slot 5 to Dave Daniel

ROBOT: What should I bring to Dave Daniel?

HUMAN: a calendar

ROBOT: I'm sorry, but I couldn't pinpoint what you mean by that.

ROBOT: What should I bring to Dave Daniel?

HUMAN: a day planner

ROBOT: I'm sorry, but I couldn't pinpoint what you mean by that.

ROBOT: What should I bring to Dave Daniel?

HUMAN: calendar

ROBOT: You want me to bring calendar to Dave Daniel?

HUMAN: yes

ROBOT: I thought so

ROBOT Happy to help

In this example, the robot had not been previously trained that "day planner" and "calendar" refer to the same thing. Yet, During the conversation, the robot was able to learn that both terms are synonymous.

The systems above present interesting approaches in this relatively new area of research. Nevertheless, there are several major difference between our proposed LCC system and learning from instruction. We summarize these differences as follows:

- Learning from instruction is targeted toward task-oriented systems, in which there is a specific task that the agent/robot needs to accomplish. LCC on the other hand is designed to be an open-ended learner that learns to aquire new declarative knowledge and/or modify its existing knowledge through a casual conversation with a human. This casual conversation has no defined objective.

- In LfI, the new task can be changed only slightly from the original task that the agent already "knows". This is because the agent's capabilities and actions are fixed. On the other hand, because LCC is not so limited, to preform, it provides more freedom to extend and modify its knowledge according to user input.

- LCC is designed to make decisions about what should be learned and what should be modified or neglected. This process means that LCC does not always accept the modification suggested by the human conversant. This property is important because it lends a more realistic behavior to our system and protects the system from acquiring incorrect or inappropriate knowledge.

We next discuss the research that uses dialogue to improve the performance of conversational systems.

2.10 Dialogue-Based Learning

Dialogue-based learning has been proposed recently for using human feedback in natural language to improve a computer agent's dialogue skills during a conversation. Weston [133] developed an agent that can learn through dialogue by receiving feedback from the teacher as a form of natural supervision. Weston studied the effect of using different modes on the agent's ability to learn. These modes are:

1. Imitating an expert: the computer agent tries to imitate the expert while it answers the teacher's questions. Therefore, the agent does not take part in the conversation.

2. Positive and negative feedback: the learner receives feedback from the teacher to reflect whether its answer is correct or not.

3. Answers supplied by the teacher: the teacher provides correction to the answer through feedback.

4. Hints provided by the teacher: hints are provided to the agent instead of giving the right answer directly.

5. Supporting facts provided by the teacher: the teacher provides supporting facts to show that the agent's answers are wrong.

6. Partial feedback: when feedback is given 50% of the time only.

7. No feedback: no feedback is given to the agent.

8. Mixture of imitation and feedback that combines both mode 1 and 2.

9. Asking for help/corrections: the learner asks the teacher to provide the correct info.

The learning model for Weston's system [133] uses end-to-end memory network (MemN2N). This memory architecture takes the last user utterance with a set of memories, which are called *contexts*. Later, it converts that input to a vector. These set of contexts are used to generate a suitable output based on the user input. The main task of the memory network is to generate an output based on the user input that best matches the information from memory. There are four training strategies that can be used by the system [133]:

1. Imitation learning that involves imitating the user's speech. The memory model here is trained using stochastic gradient descent and it does not have restrictions on what to imitate.

2. Reward-based imitation: here, the agent only imitates the action that had received a positive reward, which can only be received if the action is correct. This process eliminates poor choices from being part of the learning process.

3. Forward prediction: This process predicts the correct next answer for the teacher by using the answers provided by the agent to the speaker question/input.

4. Reward-based imitation, plus forward prediction that combines strategies 2 and 3.

Test results indicated that in general, forward prediction performed better than the other strategies.

The work of Li et al. [79] can be considered an extension to Weston's work [133]. It asks questions to improve system performance. They investigated the importance of the learning agent asking questions by examining how the chatbot can benefit from asking questions in two settings; 1) offline supervised setting, and 2) online reinforcement learning setting that also includes knowing when to ask. The authors focused on three tasks:

- Question clarification, when the agent has difficulty understanding the user and/or his/her question.

- Knowledge operation, when the agent asks a question to clarify existing information in its KB.

- Knowledge acquisition, when the agent's knowledge is incomplete [79].

Question clarifications are needed when the user input has typos or spelling mistakes. Therefore, the authors must make sure that spelling mistakes were not present in the testing or training data.

To solve this problem, the authors suggested two methods; 1) question paraphrase such as asking "what do you mean", and 2) question verification when the agent relates the misspelled word to another question without spelling mistakes.

Expecting that the behavior of asking questions may reflect that the agent/learner is not engaged with the conversation, the system limits the agent's tendency to ask questions by setting a cost associated with this action. This is accomplished by allowing the agent to answer the questions directly if it knows their answers; or if the system thinks that the question is so difficult that no clarification can help. In the latter case, the system responds by indicating that it has no answer for such question. The system also depends on user feedback (both negative and positive) to evaluate the system performance. Evaluation was performed on both offline and reinforcement learning. Observations of the results concludes that asking questions at both settings was helpful, because it provides additional information for the learner to answer correctly.

Zhang et al. [139] presented a system that enables the agent to learn a natural language by interacting with a teacher and considering his/her feedback. The focus of their work is on engaging the agent in the learning process. This process is influenced by the idea that imitating the speaker is not enough to drive a successful conversation. The learning process involves two components:

- Imitation: learning a language model by observing the teacher's behavior during a conversation. The training data depend only on the teacher's utterances, while the training process depends on predicting future words and sentences.
- Reinforcement: learning involves trial and error using the teacher's feedback.

Interaction with the agent involves asking questions and the learner answers them, describing an object and the learner repeats the description. The agent describes an object and receives a reward in form of positive feedback if the description is correct. Positive feedback can also be received

when the learner answers correctly.

Test results reflect that combining imitation with reinforcement learning achieved better results than using each one separately. An evaluation was also performed to test the ability of the system to generalize to unseen test cases. In this case, imitation learning was more successful than reinforcement learning in producing correct answers.

Recent research by Lee [75] [76] studies the possibility of applying continual learning to conversational agents. His approach involves applying a continual learning to a neural network that is used to allow a conversational agent to accumulate skills. His system allows a neural network to accumulate knowledge because neural networks tend to forget what they had learned previously when it continues to be trained on new tasks. Lee's research involves introducing Adaptive Elastic Weight Consolidation (AEWC) algorithm that intends to learn new tasks without forgetting previously learned tasks. The model was tested by training it first using general greeting statements, and later on 1) human-computer password reset dialogue, 2) human-computer password reset dialogue plus greeting dialogue, or 3) human-human password reset conversation. The resulting model using the AEWC algorithm was shown to be able to handle both dialogues without forgetting the first one (general greeting).

Lee's approach is similar to our LCC system in the sense that it seeks to improve the conversation with the computer agent by adding new information; however, his approach is very different from ours in that it depends on accumulating the knowledge using a neural network. Our LCC system uses similarity measures between the user input and the existing information in the knowledge base not only to add new knowledge but also to change or confirm existing information.

Schloss et al. [113] introduced the term "conversational learning" to refer to a virtual teacher that teaches a human learner new information.

In all three previously discussed works, we can see a common trend of using conversation as a way to improve the agent's ability to communicate better with the human. Our LCC system shares with these systems the idea of using conversation to improve the system's performance, but our objective is different because we focus on how to learn declarative knowledge rather than actions.

As a pre-cursor to LCC, the work of Mohammed Ali and Gonzalez [10] involves updating a computer agent's knowledge base through a conversation with a human. The computer agent classifies information as either being important to its knowledge base or not. It also requires the computer agent to decide whether the information represents existing knowledge, contradicted knowledge or new knowledge. Their research is the baseline for this dissertation. However, it has several limitations, for example, the system can only handle short statements. Furthermore, it can add the same information to the knowledge base as a new information when the human uses different words than what exists in the system's knowledge base. Moreover, any human could have access to the knowledge base, which is unwise from the standpoint of knowledge base security. . We cover this work in more detail in Chapter 4.

2.11 Summary

In this chapter, we first discussed some well-known chatbots to serve as a background for the reader. Those bots have several limitations such as lack of consistency with previous answers and limited ability to understand user input. Later, we discussed current research and the techniques that have been used for such chatbots, as well as their existing limitations. We found that many issues that was faced in early research have been successfully addressed, such as handling out-of-domain queries, improving the system's answers by using machine learning and natural language processing. We then shifted the discussion to include research conducted in cognitive architectures and learning from instructions. LfI uses natural language instructions from a human to teach the

agent a task. Although this technique is powerful, it has limited ability to generalize to learn different tasks. Moreover, the instructions needs to be clear and sometimes contain specific words to execute the action. Then we discussed the recent research on using human feedback in natural language to improve a computer agent's dialogue skills during a conversation. Later, we discussed two recent research studies by Schloss et al. [113] and by Lee [75] [76] that introduce the concept of conversational learning. Both studies introduced and defined the term conversational learning differently, where Schloss et al. used this term to refer to a virtual teacher that can teach human learners new knowledge, while Lee uses continual conversational learning to accumulate the learned knowledge to previously learned information using a neural network. Lastly, we discussed our baseline system and described its current limitation.

In the next chapter, we discuss the gap in the current research and state our hypothesis to validate our work.

CHAPTER 3: PROBLEM DEFINITION

3.1 General Problem

Machine learning algorithms are well-suited to different applications and are used to solve many problems, including but not limited to classifying data [61] [102], and predicting data labels or clusters [86][43]. Many of the existing algorithms have poor ability to transfer the learning approach to new applications; therefore, any slight change would require major modifications to the algorithm. As a result, computer scientists have investigated the area of behavioral learning to learn directly from human users, as humans can be considered an available and reliable resource for learning. The current behavioral learning techniques focus on learning specific tasks.

3.2 Specific Problem

There are many reasons for creating an artificially intelligent agent that can learn from interacting with humans. Many existing machine learning algorithms require labeled data to learn. Hence, this is not applicable to many tasks for which it is difficult to acquire and collect the necessary labeled data. It is also difficult to operate a robot or agent in a changeable environment because even slight changes in the environment would require a significant modification in the agent/robot program. Therefore, one alternative is to use human performance to assist in training the agent by interacting with humans.

In this dissertation, we propose learning from casual conversation, inspired by human social learning techniques. Our aim is to have a system that can selectively update its knowledge through a user's utterances. Our objective is to make the learning process occur in a natural way, to convey

knowledge to the agent and to reduce the need to provide explicit feedback by the human trainer that can often be costly. Therefore, the learning decisions are made only by the system. i.e., the user would discuss something but cannot force the system to learn it or update its information accordingly. The system is responsible for making that decision. This is important in some domains where there are multiple users sharing the same system at different times, or when there is a possibility of unauthorized access to the system knowledge base.

3.3 Hypothesis

A learning mechanism can be devised that allows a computer agent to learn declarative knowledge through casual conversation with humans.

3.4 Contribution and Novelty of Research

Through our discussion in the previous chapters, we saw that learning from human interactions has been applied to different applications using mainly three learning approaches; learning from observation, learning from demonstration and learning from instruction. Although, there were several successful applications that benefit from those approaches, there are several limitations associated with them, including that the learner could require multiple demonstrations to be able to perform the task. This can be costly and impractical in many domains. Also, the learner's ability to learn is limited by the teacher performance. This means that the learner cannot outperform the teacher even if the agent has more inherent capabilities. Additionally, all those types of learning are directed toward learning how to execute a specific task. Because all the previously discussed systems are designed to accomplish specific tasks with a defined goal, it is not easy to apply them to completely different problems. Hence, our LCC system does not address all those limitation but

rather, seeks to find an innovative way to look at the problem from a different perspective. Our contribution to the current research literature can be summarized as follows:

- A Learnng from Conversation algorithm (LCC) that can learn from information obtained through a conversation with a human. It can build an internal network to represent the knowledge base. LCC is different from other types of learning, such as LfO, LfD and LfI because all those types of learning are designed to learn specific tasks or actions, while LCC is an open ended learner that can learn continuous declarative knowledge.
- LCC system architecture that contains several new contributions that have not been used before in the literature. Those contributions are in the learning process, the classification algorithm, and the knowledge base representation and data retrieval. We discuss later in Chapter 5.
- A working prototype that combines the algorithm and the architecture described above.
- A set of tests and test results that measure the performance of the prototype.

Therefore, creating the LCC is a challenging task considering the unlimited patterns that the human user can be used in the conversation with the system. Therefore, there is no single classifier or machine learning algorithm that is able to to help LCC decide whether the given knowledge is important or not, or whether or not a given piece of knowledge exists in the knowledge base. For example, humans often use different words that have the same meaning to refer to the same thing. We propose to use multiple machine learning algorithms and natural language processing tools to enable us to attain our goal. We discuss our approach later in more detail in the following two chapters.

CHAPTER 4: BASELINE RESEARCH

In this chapter we discuss our early work in the LCC system [10], which we refer to it as LCCv0, as it is the basis of the research described in this dissertation.

4.1 LCCv0 Architecture

The ultimate vision of the LCC architecture consists of several components as illustrated in Figure 4.1. Because the problem is very complex and requires many components to make the system function as desired, the focus of LCCv0 was on the learning process, which assumes that all information from the user is reliable.

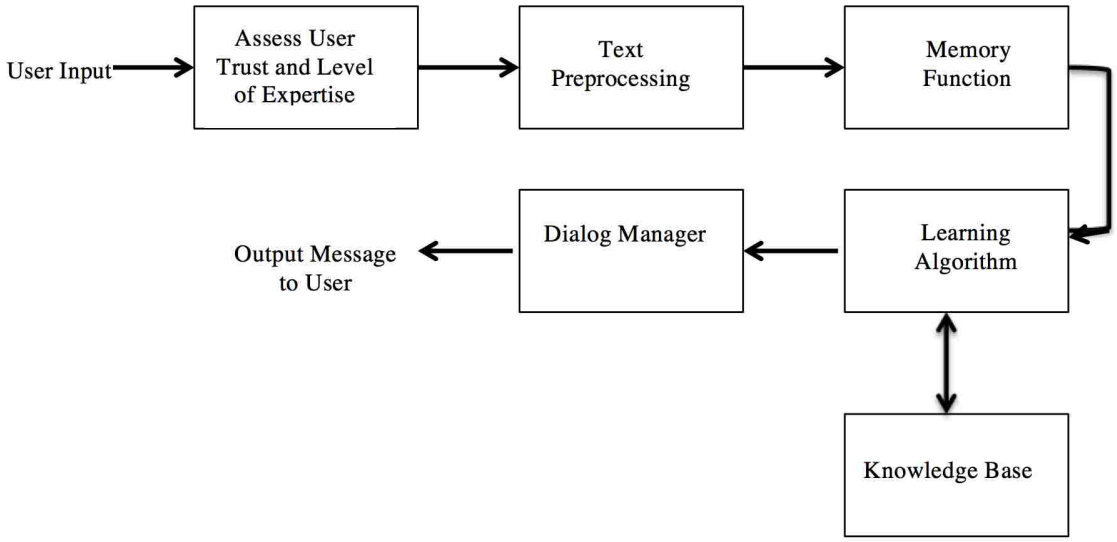


Figure 4.1: LCCv0 Architecture

Interaction with the LCCv0 system is done through text rather than speech to avoid any misunder-

standings that can occur because of speech recognition errors. As shown in Figure 4.1, the human sends a text input to the system. The system accepts the input and assesses the user's trust. The trust evaluation is done only once when the user enters his/her name. Later, LCCv0 applies text preprocessing such as changing the user input to lower-case letters and removing punctuations, which makes it easier to correlate the provided text with the system's knowledge base. Later, the system passes the processed text to the memory function, which is responsible for classifying the human input text as either important information that the system might decide to learn, or information that should not be learned. If the system decides that the information is important, it will continue the process; otherwise, the system discards it. The classification process is performed using a Naïve Bayes classifier. After the system has determined in the previous phase what is important to learn, the learning algorithm is used to decide whether the current information is new. If new, the system saves it in the knowledge base as new information. If the information contradicts what is in the knowledge base, then the system modifies the current information in its knowledge base. Finally, if the information is already in the knowledge base, the system indicates that to the user but does not add it to the knowledge base.

4.1.1 Implementation

The core challenge in the LCCv0 system was how to reason about and understand the given information provided by the human, and then determine whether there is related information in the knowledge base. Hence, our ultimate goal was to make LCCv0 able to infer whether or not the

given piece of information has an effect on existing knowledge.

```
1  while user input != termination statement do
2      NaïveBayesClassifier(input);
3      if input not chitchat then
4          POS(input) ;
5          if input in KB or matchScore >= 95 then
6              Output "similar info exist"
7          else if POS(nouns) in KB and fuzz.ratio ≥ 70 or fuzz.ratio > 85 then
8              Exchange information in KB;
9              Output a statement reflecting the change occurred
10         end
11         else
12             Add new information to KB;
13         end
14     else
15         Chat with the user;
16         Introduce a topic related to the KB information;
17     end
18 end
```

Algorithm 1: Updating Existing Knowledge

The LCCv0 algorithm is shown in Algorithm 1. The system starts by training the Naïve Bayes classifier using labeled examples of sentences that are either labeled as chitchat or not. The classifier's purpose is to determine whether the human input contains important information that the system needs to consider manipulating further or only a chatting statement that has no important information related to the system. The LCCv0 system is not designed for chatting purposes. Thus,

we have limited its chatting ability to merely greeting the user and giving him/her a brief introduction of what this system is about. If the user continues chatting with the system, LCCv0 introduces a topic related to the information in the knowledge base by asking the user his/her opinion about a given statement. If the user declares that the given statement is wrong, the system asks the user to correct this information. Based on the user response, the system updates its knowledge base, if necessary.

After LCCv0 has classified the user input as information to be potentially learned, it uses part-of-speech tagging to analyze the tags of the user input and compares the extracted nouns with the existing nouns in each entry of the knowledge base.

Because we cannot always predict the pattern of the given input sentence from the user, it is practical to use a similarity measure to decide the similarity between the text given by the user and the existing information in the system's knowledge base. Additionally, the high error rate of the available POS taggers can cause false positive matching between the user input and the system's knowledge base. Therefore, we applied fuzzy string matching using the FuzzyWuzzy library [35]. The FuzzyWuzzy library uses Levenshtein Distance to calculate the difference between two sequences. The Levenshtein Distance [31] is the minimum number of edit operations (insertion, deletion or substation) of a single character that need to be applied to transfer one string to the other. The most common way to calculate the Levenshtein Distance is by using dynamic programming. Mathematically, the Levenshtein Distance is calculated as follows:

$$D_{s_1,s_2}(i,j) = min \begin{cases} D_{s_1,s_2}(i-1,j)+1 & \text{deletion} \\ D_{s_1,s_2}(i,j-1)+1 & \text{insertion} \\ D_{s_1,s_2}(i-1,j-1)+1_{s_{1i}\neq s_{2_j}} & \text{substitution} \end{cases} \tag{4.1}$$

Where $D_{s_1,s_2}(i,j)$ is the Levenshtein Distance between the first i characters of s_1 and j characters

of s_2.

LCCv0 uses three types of string matching as follows:

1. **Token sort ratio** (s_1, s_2): considers exact match between s_1 and s_2, where s_1 and s_2 are the two strings to be compared. In the token sort ratio, the order of words is not important because the strings are tokenized and sorted in alphabetical order to determine the similarity score. The similarity score is given in term of percentage. For example "Paris is a city" and "A city is Paris" is a 100% match.

2. **Simple ratio** (s_1, s_2): the two strings need to be identical in length. the similarity score (SS) is calculated as follows:

   $SS = 200*M/N$, where M is the number of matches, and N is the total number of elements in both strings [95]. SS is 100 if both strings identical and 0 if nothing in common.

3. **Partial ratio** (s_1, s_2): if strings are of different length, say s_1 is of length m and s_2 of length n (the shorter string). Then partial ratio is used to find the most similar substring of length n in both strings and use simple ratio method to calculate the similarity score [95].

The similarity measurer provides an indication of whether elements of comparison are similar or not. Therefore we used three similarity thresholds in the learning process. 1) Perfect match; when the information exactly matches that in the KB and is thus not added to the KB; 2) Partial match; where LCCv0 modifies the existing information in the KB to be similar to the user input (assuming the user is trustworthy); 3) No match; here LCCv0 considers this as new information and adds it to the knowledge base, again assuming that the human is deemed trustworthy. The thresholds scores were determined experimentally as discussed later.

In order to determine whether or not the current input is exactly similar to existing information

in the KB, LCCv0 used a fuzzy token-sort approach. We considered sentences that achieve a similarity score of 95% and above as perfect matches. This score was chosen instead of 100% to ensure that small mismatches do not affect the process. The system does not update the knowledge base and only reports to the user that this information already exists.

For partial matching, we compare the nouns extracted using POS tagging in the user input with those of each entry in the knowledge base. There are two ways for the information to be partially matched: 1) when all the nouns in the user input are matched with an entry in the knowledge base and the fuzzy string similarity score is 70% or above; 2) Fuzzy string similarity score 85% or above, in this case, the nouns are not part of the comparison. The second approach is used to allow matching to ignore the nouns as they are already part of how the fuzzy string matching counts the similarity between the two compared sentences. If no exact match or partial match is encountered, LCC adds the user input as new information.

There is a tradeoff between adding new information and modifying existing information, depending on the similarity score. However, we believe that modifying the wrong information is worse than adding new information. For example, by decreasing the similarity score from 70% to 20%, the system considers "apples are rich in vitamins and minerals" and "apples are rich in dietary fibers" to be partially similar because it matches "apples" and "rich" from both sentences. However, they in fact contain different information. Therefore, the system should not modify its existing knowledge and add the second statement as a new entry to the knowledge base.

In all cases, the user receives a message indicating the status of the update that occurs in the knowledge base. The process LCC continues until the user terminates the discussion by typing the word "exit" as his/her input.

4.2 Experiments and Results

To focus on how the system expands its knowledge base, we used WORDij [37] to create a semantic network that visualizes our dataset before and after interacting with a human user. In Figure 4.2, LCCv0 had only 12 facts related to the topic of food. In Figure 4.3, after 30 interactions with the user, the system's knowledge base had grown in size significantly.

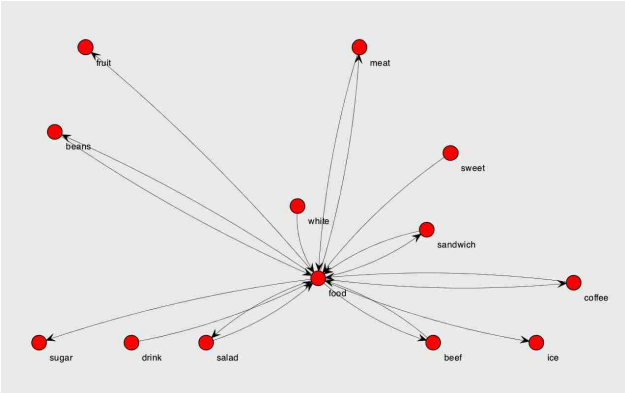


Figure 4.2: Semantic Network that Represents the Knowledge Base before Interacting with the user

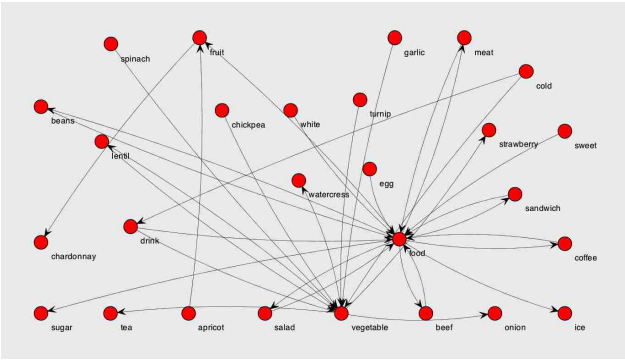


Figure 4.3: Semantic network that represents the knowledge base after 30 interactions with the user.

Table 4.1 displays some examples of the system's responses regarding the three phases of informa-

tion update: perfect matching (no knowledge added), updating existing information, and adding new information respectively. In this table, we report the similarity score between the user input and the chosen entry from the knowledge base.

Table 4.1: Information Handling Example

Speaker	Turn	Score
User	Are coffee beans fruit pits?	
LCCv0	Yes, I have the exact information in my KB coffee beans are fruit pits	100%
User	banana has 90 calories	
LCCv0	You have changed banana has 105 calories to banana has 90 calories	91%
User	Apple is rich in dietary fiber	below
LCCv0	LCCv0 added this information to its KB	70%

Table 4.2 shows an example of how the program starts and how it responds to chatting statements. As shown, in turn 5, the user tried to continue chatting with the system but the system changed the subject and asked the user a question from its knowledge base to force him/her to discuss relevant information to its knowledge base.

4.2.1 Results

LCCv0 was evaluated by creating 100 entries that varied between chatting statements and statements that are related to the system's knowledge base. For this task, we used a knowledge base that contained 3000 statements of general facts. Before testing our system with these 100 entries, we estimated the system's responses related to each entry, i.e., ignore, when it is a chatting statement; confirm, when it already exists in the knowledge base; update, when contradict existing knowledge base entry, or add when new knowledge is encountered.

Table 4.2: Chatting Example

No.	Speaker	Turn
1	LCCv0	Hi, enter your name please.
2	User	Awrad
3	LCCv0	hi Awrad! how are you today?
4	User	doing alright
5	LCCv0	Amazing, What is your input?
6	User	Hi again
7	LCCv0	Hi, I hope you are doing well. I am an AI agent that needs your help to verify and update my information related to food. Do you think that strawberries have more vitamin C than the oranges?
8	User	I don't think so
9	LCCv0	Then, could you correct my information please?

Table 4.3: Evaluation Results

KB	Test Entries	Successes	Misses
3000	100	88	12

We marked how many times the system matched our expectation and how many times it did not. As shown in Table 4.3, out of the 100 queries, the system had 88 correct actions and 12 misses compared to our expectations of how the system should respond. Four of those misses were chatting statements that the Naïve Bayes classifier incorrectly classified as relevant information; therefore, LCCv0 system added them to its knowledge base. In the rest of the misses, LCCv0 added the information as new when our expectations were that it should have replaced existing knowledge. We expected these cases to occur because we set the similarity score to be ≥ 70, which is high enough to eliminate any risk of overwriting correct information in the knowledge base with incorrect information.

4.3 Current Research Limitations

The current research of LCCv0 has many limitations that are addressed in the final version of LCC. LCCv0 uses basic Naïve Bayes classier to decide whether the given information is important or not. In the final version of LCC, we used multiple classifiers to decide the label of a given input (related to the knowledge base or chatting statements). LCCv0 does not use an efficient data representation for the knowledge base as it only uses plain text.

We plan to enhance this representation by utilizing a semantic network representation for the knowledge base. This will help ease the process of deciding whether or not the information is relevant to the current KB, and to help build some inferences. LCCv0 lacks the use of memory; therefore, we plan to include a memory model to improve the system's performance in general, and to remember previous users and their previously said information.

The learning process of LCCv0 depends only on fuzzy string matching to find the best match in the knowledge base with the user input. Yet, this is not enough to measure the similarity, as the user can use a different variation of a word to refer to the same thing. Therefore, we plan to use more than one similarity measure that uses different approaches to measure the similarity between texts and that use semantic similarity to consider the meaning of the words in the matching process.

Lastly, LCCv0 was not tested extensively. Therefore, the final version of LCC will incoreporate more robost testing. The results of these tests are discussed later in this dissertation

4.4 Summary

In this chapter, we presented LCCv0, a learning from conversation system that updates and modifies its existing information based on human user input during a conversation. The system differs

from existing approaches by its ability to learn declarative knowledge by discarding information when LCCv0 determines it to be unimportant, and modifying or adding knowledge when LCCv0 believes that the user has better knowledge about that information. The preliminary results indicate that the system was able to learn new knowledge and update its knowledge base mostly according to our expectation.

There are several factors that affect LCCv0 performance, including spelling mistakes in the user input, misclassifying the user input (i.e., chatting statement or not), and incorrect speech tagging. We propose to upgrade the system to overcome these limitations and to provide more functionality to interact more efficiently with the human. Next, we discuss our approach improvement ideas in the next chapter.

CHAPTER 5: APPROACH

Our baseline work *LCCv0* discussed in Chapter 4 introduced the concept of learning from a conversation and put it into practice with a limited application. While the test results were positive, a more robust system that is more reliable and more effective in accomplishing the designed task in a more complex application was deemed essential to make a meaningful contribution. This chapter discusses the components and algorithms used to build our final LCC system that meets these criteria.

The chapter starts by describing the overall architecture of the final LCC system, and later we describe the individual components of this new architecture.

5.1 Description of Overall LCC Architecture

Our final LCC system consists of multiple components as shown in Figure 5.1. These components are combined to accomplish the desired task of learning declarative knowledge while making the conversation sound natural and interesting to the user. LCC consists of eight components (besides the user interface unit). The main components in this architecture are the classification unit, the knowledge base unit and the learning unit. Each of those units consists of multiple subcomponents that operate using different algorithms, as described later in this Chapter. LCC components can be summarized as follows:

- **Input unit** that accepts the user input in the form of text.
- **Trust determination** that determines the user trustworthy level to allow access to certain component (left for future research).

- **Preprocessing unit** that applies text preprocessing procedures to the user's input.
- **Classification unit** that determines the next unit to which the user's input will be directed.

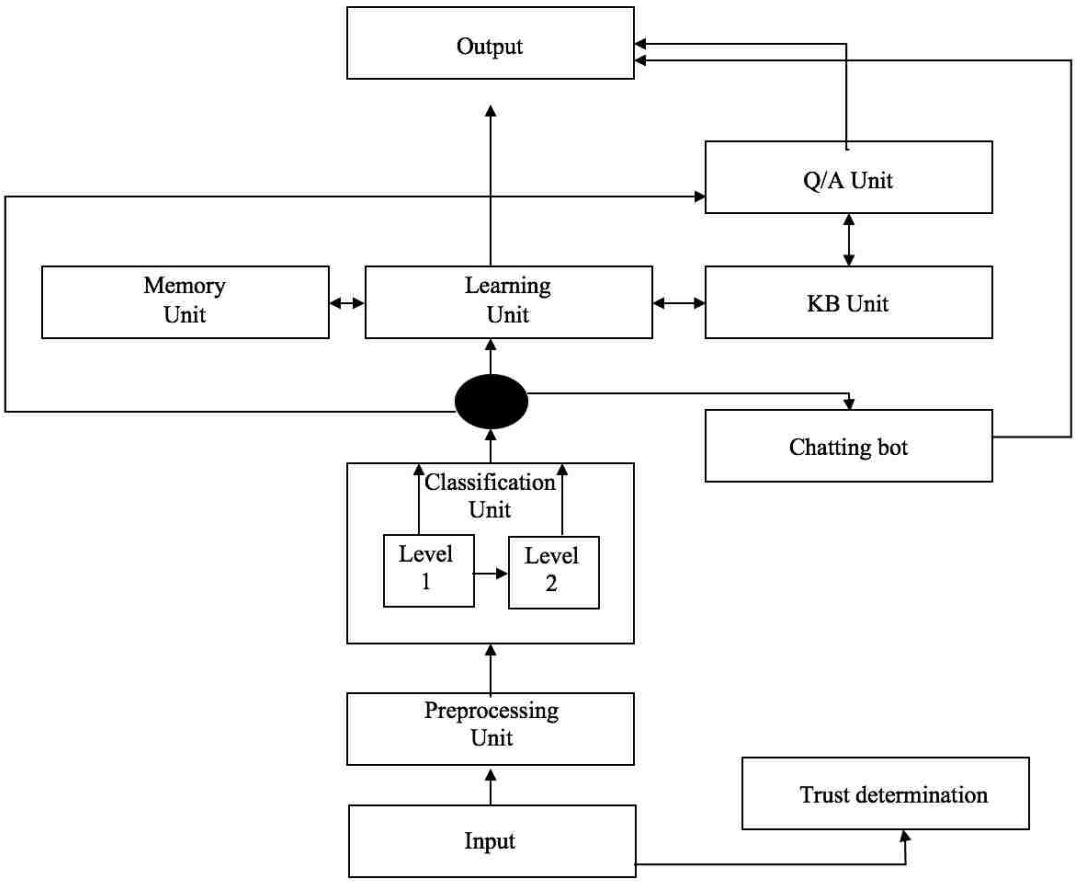


Figure 5.1: LCC Architecture

- **Knowledge base unit** that contains the knowledge of LCC.
- **Learning unit** that is responsible for learning the knowledge that needs to be learned.
- **Memory unit** consisting of text files that record a users' previous interactions and previous information of the LCC knowledge base.

- **Question/Answering unit** that is responsible for answering questions related to the knowledge base of LCC.
- **Chatbot** that is responsible for responding to a users' chatting statements.
- **Output unit** to provide the output of LCC to the user.

The input interface unit accepts user input in the form of unrestricted natural language text. This input is directed to a pre-processing unit to facilitate its classification (as a chatting, content-related statement, or question) and then matching the user text input to the information that exists in the knowledge base. The user trust level is evaluated in the trust determination unit that only allows users considered trustworthy to access the learning unit. This unit was left for future work, as the focus of this research was on developing the learning process, building the classification unit, and using a meaningful representation for the system's knowledge base.

One of the most important units in the LCC system is the classification unit. It decides the label of the provided input (content-related knowledge, chatting statement, or content-related question). The classification unit is essential for determining which unit of LCC will next be dealing with the user input. The classification unit consists of three classifiers: Naïve Bayes classifier, Decision tree classifier, and Max Entropy classifier. Those classifiers were trained on content related statement and chatting statements to makes them capable of classifying the user input and directing it to the appropriate unit as we will discuss later.

The learning unit is the core of the LCC system as it decides what needs to be learned and then modifies the system's knowledge base accordingly. The chatbot unit deals with the input that is classified as chatting statements, while the question/answering unit produces a response to the input that contains a question related to the knowledge base. The knowledge base (KB) unit holds the knowledge of the LCC system and can only be accessed from the Q/A unit and the learning

unit, but can only be modified by the learning unit. The learning unit is also connected to a memory unit that holds various information related to previous interactions with the LCC system. The last unit is the output unit that reports the system's response to the user in natural language. We next describe each of these units in more details.

5.2 Trust Determination

As shown in Figure 5.1, the first stage to interact with LCC is to determine the trustworthy level of the user. LCC allows humans labeled as *trustworthy* to have full access to the learning process (add, change information). This is done to protect the contents of the knowledge base from unauthorized users. Humans labeled as *untrustworthy* users can only access the system's question-answering componet and the chatbot. If unauthorized users try to change information, the system will ignore their request and instead either change the subject or chat with the user. LCC is built such that the user trust level will be provided by a component that uses authority evaluation and security measures to provide that label for the user. However, the trust evaluation component is left for future work, as it is not deemed to be an essential part of the scope of this research. The focus of this research was to implement the core architecture of the LCC system that consists of the learning unit, a classification unit, and the knowledge base representation. In its place, LCC merely asks the user to enter "1" if he/she is labeled as trustworthy and "2" if not. If the user enters "1", he/she will have a full access to modify information in the KB chat with the system, and ask questions. However, if he/she enters a "2", he/she can only ask questions and chat with the system.

5.3 Input and Text-Preprocessing

LCC accepts user input in typed natural language text. We preferred texting to communicate with LCC rather than using voice to avoid the high error rates of translation and voice recognition. LCC accepts the user input in English without putting restrictions on how the user input should be formatted. LCC applies text pre-processing, as shown in Figure 5.1, to normalize the user input to be more compatible with the information in its knowledge base and the training data for the classifiers. This process will facilitate classification of the user's input and match it with the existing information in the knowledge base. The pre-processing step involves changing upper-case letters to lower-case to keep the format consistent in the knowledge base. We used a spell checker from the autocorrect library in python to correct misspelled words by the user. A spell checker works by comparing the words in the user input to lists of thousands of correctly spelled words to determine the correct spelling of a given word. The auto correcter uses a Levenshtein distance (discussed earlier in Chapter 4) to measure the distance between the current word in the user's input and the words in its lists. The word with the shortest distance is the word that will be considered most likely to be the correct one.

5.4 Classification

An important step in the LCC system process is to classify the user input correctly in order to determine whether the user input text reflects declarative knowledge that is to be learned, a question that needs an answer, or a chitchat statement that should be directed to the chatbot. Therefore, it is essential to improve the system's ability to find the correct label of the provided input text because a wrong label can result in producing an incorrect response from the system.

As shown in Figure 5.1, LCC uses two levels of classificartion that apply Ensemble learning to

decide the label of a given input. Ensemble Learning is the process of using different classifiers, then apply majority vote on the obtained labels to decide the most likely labels for the given data. The benefit of using multiple classifiers is to improve the overall decision-making process and to avoid making wrong selections [38]. In our approach, the Ensemble Learning considers the majority vote among Naïve Bayes classifier, Decision tree, and Max Entropy classifiers.

```
1  while input != termination-statement do
2      preprocessing(input);
3      level1-classifier(input);
4      if majority-vote-count < 3 then
5          POS(input);
6          if POS(input) is noun then
7              level2-classifier(nouns in input);
8              if lable == content-related && majority-vote == 3 then
9                  input = reconstruct(input);
10     if input == chatting then
11         output chatbot(response);
12     end
13     else if input == question && label == content-related then
14         QA(input);
15         output QA(response);
16     end
17     else if label == content-related then
18         learning(input);
19         output learning(response);
20     end
21     if input == termination-statement then
22         Exit();
23     end
24 end
```

Algorithm 2: Directing the User Input to the Right Engine

An overall diagram that shows the two levels and how they interact with each other is shown in Figure 5.2. Each level uses a majority vote among those classifiers to determine the label of the user's input that can be either 1) declarative content-related knowledge, 2) content-related question or 3) a chatting statement. Those classifiers were selected because they have been successfully used

in the literature for text classification: Naïve Bayes classifier in [36] and [4]; Decision tree in [14]; and Max Entropy in [98] and [32].

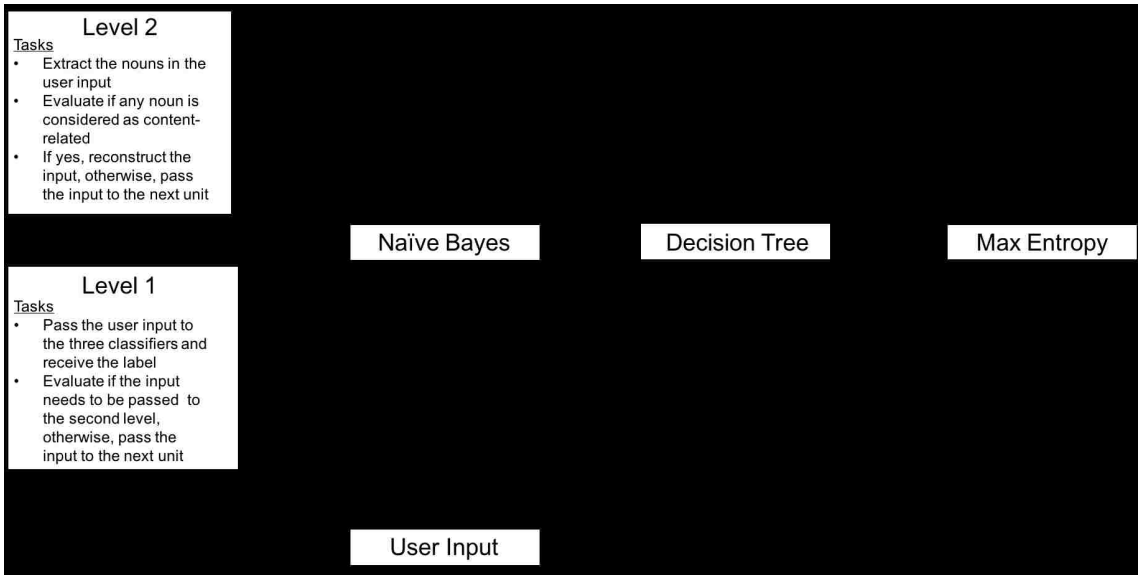


Figure 5.2: A Block Diagram to Show Level 1 and Level 2 Classification Process

Another reason to consider those classifiers is the mechanism they use to find the label of the given input. Each of these operates in a different manner to determine the label as we discuss later. This is essential, as using different approaches will improve the overall decision of the majority vote because each classifier will follow a specific algorithm to determine the label of the user's input. This will improve the overall decision, as the classifiers are not following the same path to decide on the label.

5.4.1 Classifiers Training and Testing

To train the classifers, we collected 650 sentences from various resources. Sentences that are labeled as *non-chatting* were gathered from an online article about random facts [5] and from data

extracted from reviews on various topics as part of a project at the University of Illinois [49]. The chatting statements were collected using statements that humans use in everyday conversation. The classifiers were trained on 80% of the collected data and tested on the remeaning 20%, i.e., the three classifiers were trained on 522 sentences that were labeled manually and then tested on the remainig set of the 128 sentences.

The classifiers training and testing occur after launching the LCC system. The training is offline, i.e., the training set and the testing set are fixed. Appendix C contains the training data that had been used to train the three used classifiers, and Appendix D contains the testing data for those classifiers.

5.4.2 The Two Levels of LCC Classifiers

The overall algorithm is shown in Algorithm 2. The first level of the classification process (line 3 in Algorithm 2) classifies the input text at the sentence level, where all words in the user input contribute to the classification process and the decision will be made based on the sentence level. When all the classifiers have the same decision regarding the user input, i.e., they are said to have total agreement to be either a chatting statement or content related to the knowledge base.

If the decision of the classifiers in level 1 are not unanimous, the second level is activated. The second level constructs the part of the input that include related information when at least two classifiers in level 2 classify all or some of the nouns in the input as related information (lines 4 - 9 in Algorithm 2).

The second level uses the same classifiers (as shown in Figure 5.2) in the first level but rather than passing the whole input of the user, LCC only constructs the nouns of the user input and passes it to the classifiers to determine whether they consider as chatting expressions or content-related

expressions. The second level uses part of speech tagging to extract the nouns in the input text. These nouns are fed to the same three classifiers to determine their labels. If the three classifiers agree that at least one noun is a non-chatting expression, the sentence is reconstructed to include everything after that noun, and neglects the words coming before the non-chatting noun of the sentence. This will only happen if the sentence is a complete sentence and not a fragment. To determine that, LCC checks whether the resulting text contains a noun and verb and the sentence consists of at least three words. This approach is used to exclude fragments that can be generated after removing the parts of the sentence that consider chatting. The second level works only when the chatting expression is at the beginning of the sentence. For example, "hi, do you know if apple is a fruit?" The first part "hi do you know if" is considered a chatting expression while the rest "apple is a fruit" is considered content related information. FOur future research will apply a similar concept to other variation of sentences structure that will consider removing chatting-related information in different positions in the sentence. This will require rebuilding the sentence and making sure its meaning remains accurate and grammatically correct. An overview of the classifiers that contribute in the majority vote is described next.

5.4.3 Naïve Bayes Classifier

The Naïve Bayes classifier has been reported widely for text classification in [36], as it is fast and easy to be trained in a small database. Because the Naïve Bayes classifier has been used in the baseline version of LCC (as described in Chapter 4) and it was largely successful, it has been kept in this final version.

The Naïve Bayes algorithm operates by first using the Bayes rule to express $P(label|features)$ in

terms of $P(label)$ and $P(features|label)$ as follows:

$$P(label|features) = \frac{P(label) * P(features|label)}{P(features)} \tag{5.1}$$

The Naïve Bayes classifier uses a bag of words to extract the feature vectors by segmenting each word in the text file and counting the number of occurrences of those words to measure their frequency.

The algorithm then makes the naïve assumption that all features are independent given the label:

$$P(label|features) = \frac{(P(label) * P(f_1|label) * \cdots * P(f_n|label))}{P(features)} \tag{5.2}$$

Where f_1, $f_2 \ldots f_n$ are the individual features in which each feature f_i is conditionally independent of every other feature f_j for $j \neq i$.

Instead of computing $P(features)$, the algorithm only calculates the numerator for each label and then normalizes them to sum to 1. The naïve assumption of independency is obviously not accurate with respect to texts. For example, a fruit can be considered a peach, if it is round in shape, within 10 cm diameter, and has orange color. The naïve assumption considers each of those features (round, diameter of 10 cm, color orange) to be independent to the probability that this fruit is a peach. This assumption is impractical, as each of those features are dependent on each other as all contribute to the fact that the fruit is a peach. However, Naïve Bayes classifier has been used widely in text classification and it yields acceptable results, such as it did in our described baseline LCCv0 (discussed in section 4.1.1).

5.4.4 Decision Tree Classifier

The decision tree is a well-known algorithm that has also been used widely for text classification. It can easily construct a tree and assign a label for a given data. The branches of the tree correspond to conditions in feature values, while leaves correspond to label assignments. A decision tree is used to classify an instance by starting at the root node and moving through the tree branches until reaching a leaf node, which will represent the label assigned. Yet, the limitation of this classifier is that it can be easily overfitted, as it works best when the input is close to the training data; this requires some pruning to test the classifier.

The decision tree used here is from the NLTK library [3] that follows CART (Classification And Regression Trees) model representation to construct the decision tree [27]. The representation of a CART model is a binary tree. An example of how decision tree is constructed is shown in Figure 5.3 where the task is to decide whether or not a specific weather condition is good to play tennis or not. Each internal node within a box (humidity and wind) tests for a feature, each branch (sunny, overcast, rain, high, normal, strong and weak) corresponds to a feature value, and each leaf node (yes or no) assigns a label. For example, when the outlook is sunny and the humidity is high, that indicates it is not a good day to play tennis. While, when the outlook is sunny and the humidity is normal, this indicates a good day to play. Yet, constructing a tree for text follows the same procedure except it is a very high dimensional as there is a feature for each word.

Therefore, the most difficult part in a decision tree is to construct a tree for the whole training set. As a result, a *decision stump* is constructed first. A decision stump is a decision tree that has one node representing one feature that are connected directly to the leave nodes. Therefore, a decision stump makes a decision based on one feature only. For example, consider the feature “humidity > 70%” if yes then high, otherwise the humidity is good.

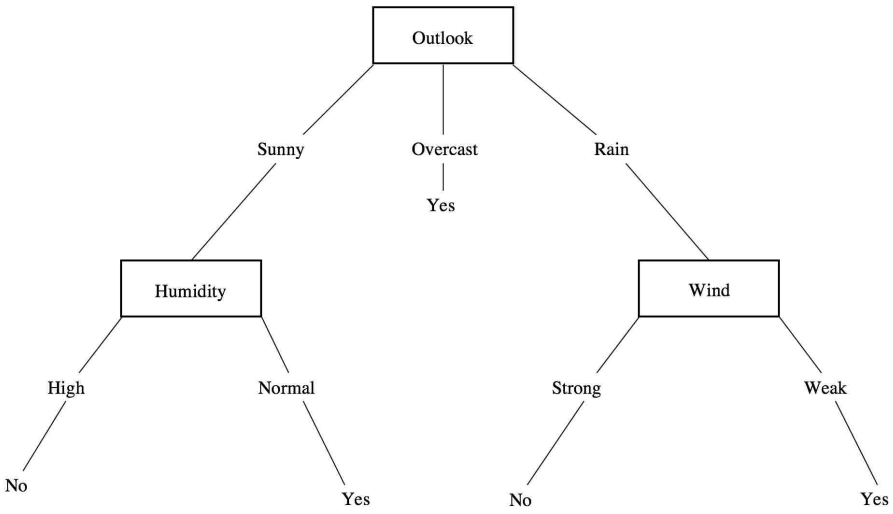


Figure 5.3: Decision Tree for Playing Tennis [106]

Feature selection is the first step to building the tree stump, and the easiest way to do that is to build a tree for each feature and consider the ones that achieve the highest number of occurrence in the training data. The decision stump is built by assigning a label to the leaves based on the most frequent label in the training data. For example, the word "hello" will be assigned a chatting label if it appears in the text. To build the decision tree for the training data requires consideration of the overall best decision stump, and attach the stumps to each other. For example, to grow the tree in Figure 5.3, we can replace the leftmost leaf with the decision stump "Temp" that has two outcomes, when "Hot", then the label will be "No" and when "Cool", the label will be "Yes" as shown in Figure 5.4.

Figure 5.5 shows a diagram that reflects how the decision tree applies its rules to the user input to report the label. From the diagram, one can infer that the decision tree works best when its decision rules apply to the user input, which means that the decision tree had encountered similar information in the training data that will facilitate the process of applying its rules.

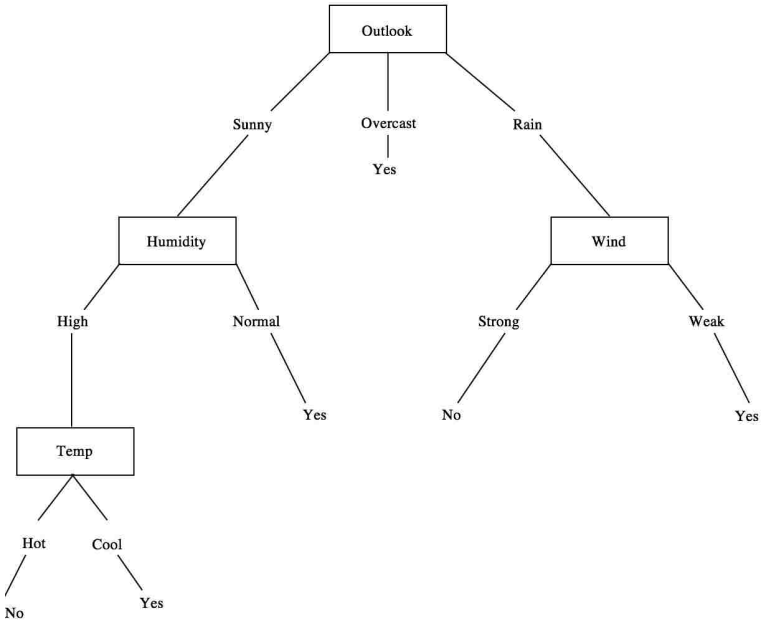


Figure 5.4: Decision Tree for Playing Tennis After Adding a Temp Stump

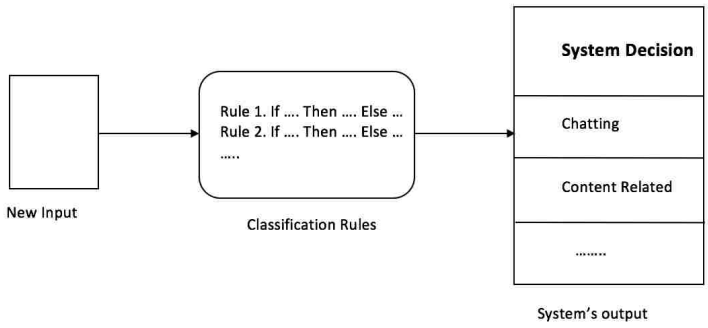


Figure 5.5: Decision Tree Example

5.4.5 Max Entropy Classifier

The Max Entropy classifier is a probabilistic estimator that is widely used in natural language tasks. Max Entropy is also a general technique to estimate the probability distribution for a given data. For text classification, Max Entropy is used to estimate the distribution of a class label of a given document. Nigam et al.[98] reported that Max Entropy is a promising technique for text classification, as using this classifier was able to reduce 40% of the classification errors that occurred by using Naïve Bayes classifier on one of their databases. However, Max Entropy performs slightly worse than the Naïve Bayes classifier on a different dataset. The authors reported that this performance was because of an overfitting problem to the training data.

Max Entropy classifier is closely related to the Naïve Bayes classifer. Max Entropy considers the dependency among the features by using search-based optimization to find weights for the features that maximize the likelihood of the training data [94]. Also, Max Entropy uses search mechanism to search for a set of parameters that maximize the total likelihood of the training dataset that will improve the classifier performance. On the other hand, Naïve Bayes classifier uses probability to represent the features of the training set.

Max Entropy classifier uses iterative optimization to choose the model features that are operated by first initializing all the model's parameters to the same values, and then applying a repeated refinement on those features as new information is revealed to bring them to the ideal solution. For example, to classify a text to any of four available classes: If we know nothing about the text, we will assume a uniform distribution and assume each class has a 25% chance to be the correct label. However, if we knew that the document contains the word "football" and the first class is about a sport, the chance for this class to be chosen will be higher (40%) and the other three classes lower (20%).

Additionally, unlike the Naïve Bayes classifier, Max Entropy classifier allows the user to decide the combination between the labels and the features in a way that one feature can have more than one label or multiple features assigned to one label. The combination of the labels and the features is called *joint-feature* [3].

LCC uses different classifiers to determine the label for a given data that operates using different approaches to assign the label. This can help improve the overall outcome of the classification process, as it chooses the label for a given input by consulting all the classifiers and applying a majority vote to decide on the label. It is also reported in the literature that combining classifiers that have acceptable individual performance and a good level of diversity improves the accuracy of the overall classification process [110].

5.5 Knowledge Representation

The knowledge base (KB) unit can only be accessed by the learning unit and the Q/A unit, as shown in Figure 5.1. Those units are discussed later in this Chapter. We proposed using two forms of representation for the LCC knowledge base: a flat text file that holds the original content of the knowledge base (facts in a plain text), and a semantic network that is constructed based on that text file. Those concurrent representations are kept continuously updated as a conversation progresses with the user. The text file used for creating the KB contains sentences extracted from a website [5]; these sentences include random facts related to various topics such as cars and food. Those sentences are used to generate a semantic network where each node contains one sentence. Nodes that contain similar information are connected with an edge as discussed later in this section. The sentences are encoded using Unicode (UTF-8, an 8-bit variable-width encoding, which maximizes compatibility with ASCII) to be compatible with the input of the sematic network.

Both representations are updated regularly during the learning process. We decided to use two forms of representations for two reasons: 1) the updated text file will be more easily used in other applications, as it can be fed directly to other systems, for example, using the learned knowledge for a system like Alexa, and 2) the semantic network is used to speed up the process of finding matches in the knowledge base with the user input. There are two ways to update the knowledge base: 1) by adding a new piece of information to the knowledge base and 2) by changing existing information. In the case of adding new information, the text file will be updated by appending the new information to the end of the file; the semantic network will be reconstructed from the updated file to allow the new node to build connections with the existing nodes. The new node is assigned a new numerical label: one plus the number of the last node label used. For example, if the last sentence/node in the previous document, before adding the new node, has an ID = 10, the new node will be assigned an ID = 11. To update the knowledge base is to change an existing information/sentence in the knowledge base with the user input. In this case, the new node is placed in the position of the previous information and receives its ID. Yet, the semantic network will still be rebuilt, as the new information can have different connections with the existing nodes than the previous node. A diagram that explains this process is shown in Figure 5.6.

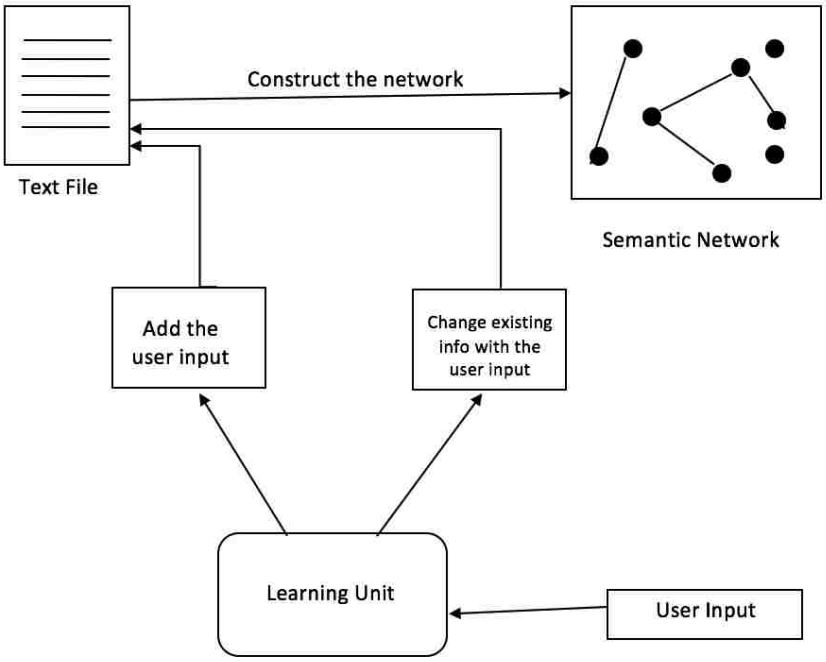


Figure 5.6: A Diagram to Show the Process of Updating the Knowledge Base of LCC

The semantic network used here is a modified version of a standard semantic network where the nodes represent words and edges reflect the relation between them. In our modified version of the network, the nodes represent the sentences rather than words, and the edges hold the similarity score between the sentences contained in the connected nodes. This approach is chosen because it is more appropriate to relate sentences that include similar information. The network is implemented using the Textacy[1] library [7]. The network assigns a number/ID to each sentence, starting with zero for the first sentence in the file, and incremented by one until the end of the file (the last sentence). The edges between the nodes are weighted using the Jaccard similarity measure,

[1]Textacy is a Python library for performing higher-level natural language processing (NLP) tasks.

typically used to represent the similarity between two sentences/documents. A value 0 means the documents are completely dissimilar and 1 means they are identical. Values between 0 and 1 represent the degree of similarity. The Jaccard index is defined as the size of the intersection of the two sentences (number of similar words) divided by the size of the union of them (number of unique words in both sentences).

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{5.3}$$

Where *A* represents the words in the first sentence and *B* holds the words in the second sentence.

In order to calculate the similarity between two sentences, first, we need to apply lemmatization to return the words to their roots, which will make it easier to compare the words. For example, if we want to measure the Jaccard similarity between "this girl has a pretty dress" and "those girls have pretty outfits". By applying lemmatization, the words "this" and "those" become "this", "girl" and "girls" become "girl", and "has" and "have" become "has". Figure 5.7 shows a Venn diagram of the Jaccard similarity intersection and union of the two sentences. Therefore, the Jaccard similarity of this example is 4/(2 + 4 + 1) = 0.57.

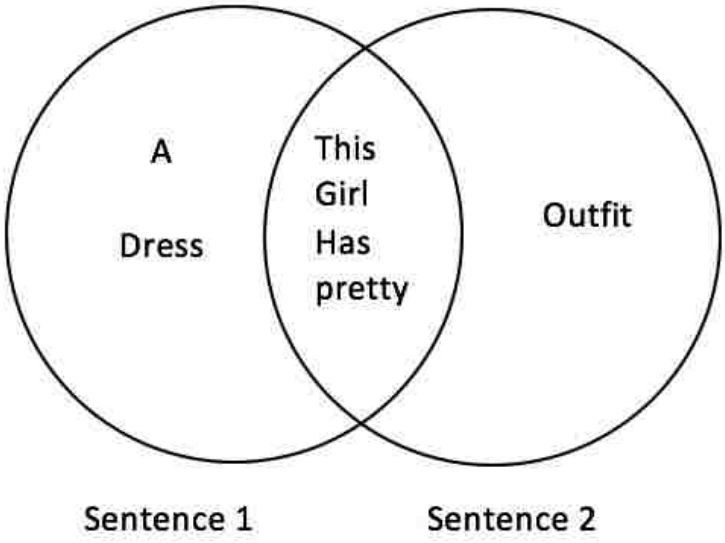


Figure 5.7: Venn Diagram to Show the Intersection and Union of Two Sentences.

The generated network connects nodes whose similarity measure is above 0.1. Therefore, the generated network is a sparse network, where nodes that have similarity between them were connected by undirected edges. The network can also include disjoint nodes when their similarity with other nodes in the network is below 0.1.

A nodes' information includes: the node ID, neighbors of that node and the edge weight between that node with its neighbor are stored using three dictionaries. For example, consider that u is a node and G is the graph, $G[u]$ returns an adjacency dictionary of u's neighbor say v, and $G[u][v]$ returns the edge weight between nodes u and v. This organization is proposed in the Textacy library, which allows faster lookup with reasonable storage for large networks.

The benefit of transforming the text file into a semantic network is to speed up the matching process between the user input and the existing information in the knowledge base, as it is faster to search

for similar knowledge and exclude other information without requiring an exhaustive search of the knowledge base. For example, assume the user input contains information about cars such as “my car has four wheels”. When LCC compares this input with the first sentence in the knowledge base, say “the sky is blue”, there is no similarity between the two sentences. Therefore, LCC will automatically exclude comparing the user input with all the neighbors’ nodes of “the sky is blue” sentence, which are connected with edges to this node, and move on to compare the input with the other nodes that have no connection with this node. An example of how the network represented is shown in Figure 5.8. Each of those nodes represents a sentence (shown in Figure 5.9). The weight on the edges represents the Jaccard similarity between every two nodes. In order to connect two nodes, we added a threshold which is the two sentences should share at least one noun and their Jaccard measure is above 0.1. Therefore, we can see that the network does not represent a fully connected graph, as the idea is to only connect nodes that have a strong relation between them to make the subsequent comparison with the user input faster. This is further discussed later in the learning algorithm.

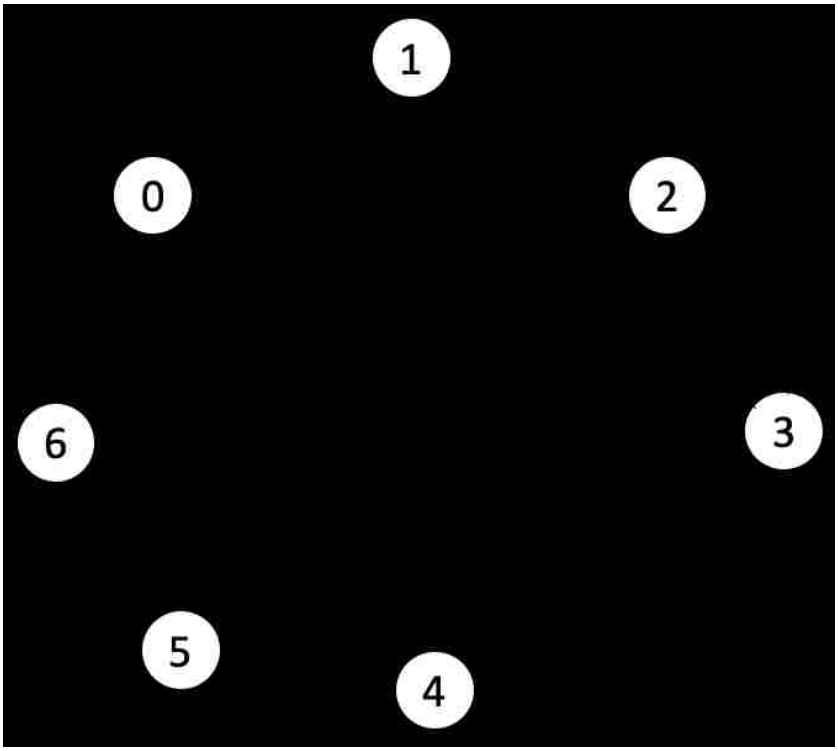


Figure 5.8: Example of the semantic network

0: cars have 4 wheels.
1: about 165,000 cars are produced every day.
2: it would take less than 6 months to get to the Moon by car at 60mph.
3: the first car accident occurred in 1891, in Ohio.
4: bananas are slightly radioactive.
5: bananas don't grow on trees.
6: bananas are the most popular fruits in the US.

Figure 5.9: The sentences for the nodes in Figure 5.8

The knowledge base (both the text file and the semantic network) is updated regularly to ensure that new changes in the knowledge base are reflected online. The idea behind rebuilding the network

every time a change occurrs is that a small change can result in different connections among the existing information. Therefore, it is advantageous to update the network regularly.

Because the network uses numbers to represent the sentences in the knowledge base, a dictionary that maps those ID's/numbers to their original sentences is created. This dictionary is also updated regularly during the learning process and is used to provide an easy and efficient lookup table to map the numbers back to their sentences.

5.6 Memory Function

The memory function/unit can be accessed only through the learning unit, as shown in Figure 5.1. In general, using a memory is closely related to the learning process because it is necessary to keep records of the input information and to remember the users who have previously interacted with the system. This allows new users to know previous modifications made to the knowledge base and to remind returning users what they had done earlier. In addition, this makes the conversation sound more human-like. Therefore, LCC saves information related to a user's trustworthy level, logs of previous interactions, and information related to previous modifications in the knowledge base. Those modifications are tracked during the learning process of LCC by assigning numerical values to information being confirmed, added, or modified. Yet, inputs that contain chatting statements and questions are not considered by the memory function. Therefore, LCC saves the following information:

- User's utterances during his/her interactions with the system.
- How many times a piece of existing information in the knowledge base has been changed and what were those changes.

- A numerical value associated with each of the sentences in the knowledge base. Those values are used to indicate whether a piece of information is being confirmed, added or modified. For now we are uaing those values only to display to the user what were the previous changes in the KB. We are planning to use those values in future research to improve the learning process. For example, LCC should investigate the source of information if a piece of information had changed multiple times.

The information is saved on text documents in form of dictionaries that consist of keys and values. For example, in the file that records the numerical values associated with the KB sentences, called memory-value dictionary, the key is the node/sentence ID that is used in the semantic network, and the value is the associated numerical value/number assigned by LCC. Figure 5.10 shows an example of how information will be saved in the memory-value dictionary. After mapping each sentence in the knowledge base to a unique ID, the memory-value dictionary will save the assigned value to the appropriate ID. For instance, the first sentence "cars have 4 wheels" is an existing sentence in the knowledge base that has not been changed or confirmed; therefore, the value associated to it that represents the initial assigned value is "0". The second sentence, "the sky is blue" is modified information, therefore, the associated value is "-1". The last sentence "humans can think", is an information that has been added, therefore, the assigned value is "0.5". Those values were only chosen to distinguish among the information that had been modified, added, or confirmed. If the associated value is negative, that means this information had been modified; positive integer values are assigned to information that had been confirmed and positive fraction integer to indicate information that had been added. More details about those values will be discussed in the learning section.

For the file that records the modification of previous information, the key is also the node ID, while the value is a list of previous versions of the original information. An example is shown in

Figure 5.11. It is important to note that this dictionary only holds the information of the modified sentences. Therefore, in our example, the dictionary holds the information of the second sentence only because it is the only one modified in this example (as it has a -1 value associated with it in Figure 5.10). We used this approach to make the size of the document more manageable.

Yet, for the user's previous interactions file, the key is the username, and the value is a set of his/her stated information. We only save information that is considered content-related to save space in the memory and make the file size manageable. An example of how the file will look is shown in Figure 5.12.

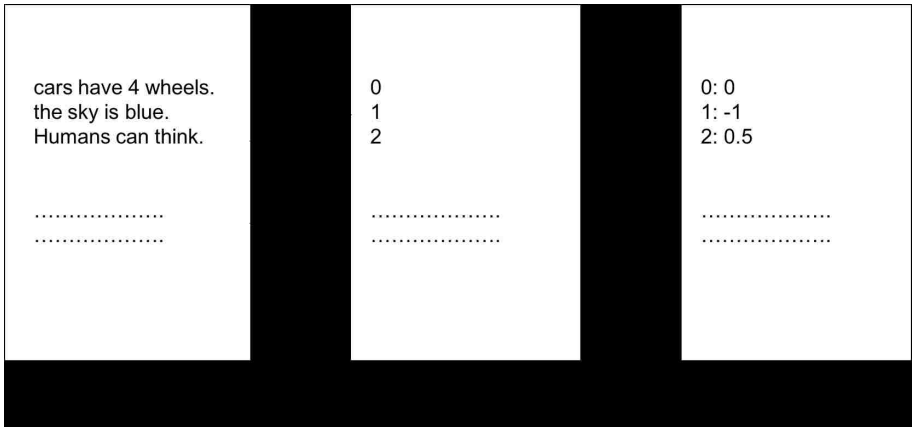


Figure 5.10: Memory Representation of the Numerical Associated Values

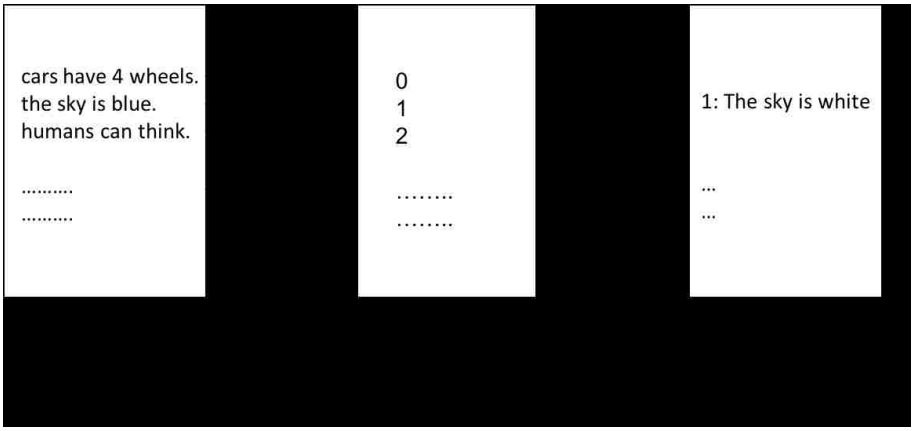


Figure 5.11: Memory Representation of Previous Versions of the Modified Information

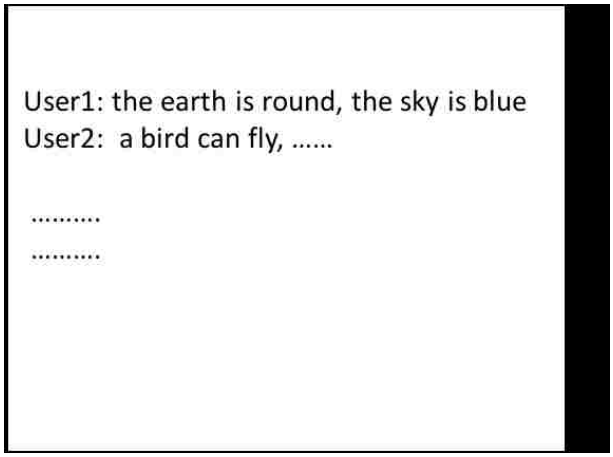


Figure 5.12: Memory Representation of Users Said Content-Related Information

The updated information is saved in the corresponding text file when the user ends his/her interactions with the system.

5.7 Chatbot

As shown in Figure 5.1, LCC uses a chatbot to produce responses for statements classified as chatting, or for when the user enters content-related information but he/she has been classified as an unauthorized user. The chatbot is built with the ChatterBot library [1] that uses searching and classification algorithms to produce different types of answers. ChatterBot uses an example-based model and the responses are selected by choosing the best matching answer to the user input. The training data are stored in the form of questions and answers. The program can also be trained from the user input, as it saves the user's input and the statements generated by the system as responses.

ChatterBot consists of four adapters, 1) input adapter that accepts the user input and passes it to the chatbot; 2) storage adapter that stores the information in a database; 3) logic adapters that are responsible for selecting the answers; 4) output adapter that returns the answer.

ChatterBot has multiple logic adapters that use Naïve Bayes classifiers to determine whether a given input by the user meets some criteria to guarantee a response by the system. It also uses search algorithms to help the system choose a response by measuring the similarity of an existing statement to the input statement, and the frequency of having similar responses that have occurred before.

The logic adapter plays the role of dialogue manager; therefore, it is responsible for finding an answer to the user input. The database in Chatterbot is divided into categories: For example, computers, food, greetings, health, etc. An example of how the knowledge base is organized for the computer topic is shown in Figure 5.13. The logic adapter first searches the database for a known statement that is similar or close to the user input. Later, it selects a known response to that statement, where there are usually several of them. For instance, if the user asks "what is a computer?", the logic adapter will first search the knowledge base to find the best match, which in

this case will be an exact match to the first question shown in Figure 5.13. Later, the logic adapter will respond to the user input with one of the responses on the left that will be chosen at random.

The chatbot also can learn from the user responses by storing the user's input, and uses them later in its future responses. More information can be obtained from the Chatterbot documentation [1].

5.8 Learning Algorithm

The core of our research is how to update a computer agent's knowledge based on a conversation with a human. This is done by the *learning unit* shown in Figure 5.1. Therefore, the main task of the learning process is to find a correlation between the user input and the best match in the knowledge base. Hence, to obtain the best results, LCC combines three different approaches of similarity measures to find the best match in the knowledge base for the user input. One of those measures is bounded on a character-based similarity measure, while the other two are knowledge-based similarity measures. Character-based similarity measures the distance between two strings of characters based on the contiguous chain of characters that are present in both strings. In this research, we apply Fuzzy String Matching that uses Levenshtein distance (discussed in Chapter 4) to compute the similarity between text strings [138].

Statement	Responses
What is a computer?	- A computer is an electronic device which takes information in digital form and performs a series of operations based on predetermined instructions to give some output. - The thing you're using to talk to me is a computer. - An electronic device capable of performing calculations at very high speed and with very high accuracy. - A device which maps one set of numbers onto another set of numbers.
What is a super computer?	- Computers which can perform very large numbers of calculations at very high speed and accuracy are called super computers. - A supercomputer is a computer which operates at several orders of magnitude greater speed and capacity than everyday general-purpose computers, like the one you are talking to me on. - You know, the big iron!
What is an operating system?	- Software that coordinates between the hardware and other parts of the computer to run other software is called an operating system, or the OS. - Windows, MacOS, Linux, UNIX... all of them are types of OSes. - Android and iOS are operating systems for mobile devices. - - Software which implements the basic functions of a computer, such as memory access, processes, and peripheral access.
.....	....

Figure 5.13: An Example of How the Database in Chatterbot is Organized.

Knowledge-based similarity is a semantic measure that calculates the degree of similarity between two words using various information derived from an external semantic network. Similarity measure based on WordNet and Word alignment matching can be considered examples of this measure. The idea behind using different types of similarity measures is to allow the system to overcome the limitations inherent in each measure, as well as create a collaborative environment that narrows down the search space of the system knowledge base to find the best match to the user input.

The similarity measures were used not only to find the best match but also to minimize the

search space. The matching algorithms were not used to compare the user input with every single node/sentence in the knowledge base; rather, the user input is compared to a subset of the knowledge base. This subset is chosen after using one of the similarity measures, the fuzzy string matching, and excludes the nodes that have edges with the current node whose similarity to the user input is below a benchmark threshold. These similarity measures are discussed next.

5.8.1 Similarity Measure Based on WordNet

We used a modified version of the work by Mihalcea et al. [89] that uses WordNet to measure the semantic score between two segments/sentences. WordNet is a large lexical English database that groups nouns, verbs, adjective and adverbs into sets of cognitive synonyms called *Synsets*. The work of Mihalcea et al. is one of the first approaches that goes beyond the simple matching method of word-to-word comparison for estimating the similarity using segments with more than three words. This approach uses a function of the semantic similarity at the word level to model the semantic similarity of the whole segment. The similarity measure combines the metrics of word-to-word similarity into a formula to measure the similarity between two sentences. There were multiple word-to-word semantic similarity measures suggested in [89]. In our work, we used *Pointwise Mutual Information* that makes use of data collected by information retrieval *(PMI-IR)* and measures the degree of statistical dependency between two words. PMI-IR is measured as follows:

$$PMI - IR(w_1, w_2) = log_2 \frac{p(w_1 \& w_2)}{p(w_1) * p(w_2)} \tag{5.4}$$

Where w_1 and w_2 are the two words to measure the PMI-IR between them.

The complete process to measure the semantic similarity between the two segments/sentences operates as follows: for each word w in the first segment S_1, the system tries to identify the word in the second segment S_2 that has the highest semantic similarity (maxSim(w, S_2)), according to

one of the word-to-word similarity measures. The same process is applied in the other direction, i.e., decide the most similar words in the first segment S_1 starting with the second segment S_2. The results are summed and normalized over the length of each text segment. The results of the measure are later combined using a simple average of the measures in both direction. The average is computed because the computed similarity measure is not the same in both directions. This is a drawback in their work, as it is more practical to have the same value in both directions; however, the difference between the two values is generally small, and averaging the values makes the result more acceptable as it considers both directions. The final formula to calculate the semantic similarity is as follows:

$$Sim(S_1, S_2) = \frac{1}{2}\frac{\sum_{w \in \{S_1\}} maxSim(w, S_2)}{S_1\ length} + \frac{\sum_{w \in \{S_2\}} maxSim(w, S_1)}{S_2\ length} \tag{5.5}$$

The similarity value can be any value between 0 and 1 inclusive where 1 means an exact match and 0 indicates no match between the two sentences.

In their work, Mihalcea et al. added a weight to the similarity measure using TF-IDF[2]. For our research, the extra computation of calculating TF-IDF is removed, as our desire is to have a uniform distribution by giving each word the same weight and importance. Another reason to not use TF-IDF is that it is most beneficial on documents that contain large paragraphs, as it emphasizes the importance of words that occur more often. Our work, on the other hand, involves relatively short and simple sentences, making TF-IDF an overkill.

WordNet can only compare words with the same part of speech tags i.e., it compares nouns with

[2]TF-IDF stands for Term Frequency-Inverse Document Frequency, a numerical measure that reflects the importance of a word in a document or corpus. The term frequency refers to the number of times a specific term appears in the document. For example, if we are looking for the most relevant document that has the term "the blue ocean", we can count the number of times each word appears in the document. This process is known as the term frequency. The inverse document frequency tends to give more weight to the more relevant words (less-common) in the search. In our example, because the word "the" is a very frequent word, it will be given less weight and the more relevant words "blue" and "ocean" will be given more weight [107].

nouns, verbs with verbs, etc. Therefore, to measure the degree of similarity between the user input and the knowledge base contents, part-of-speech tagging (POS) is used. Also, WordNet can only group nouns, verbs, adjectives, and adverbs; as a result, other parts-of-speech tags are ignored during the comparison (such as articles and pronouns).

To show an example of how the similarity measure is computed between two sentences, we show the results of calculating the similarity measure between: S1: "a cup of caffeinated coffee significantly improves blood flow", and S2: "a cup of coffee can enhance blood flow". The similarity measure starts with the words in the first sentence (S1) and compares it with the words in the second sentence (S2). These are reported in Table 5.1. While the results of the similarity measure starting with the words in S2 and comparing it with those in S1 are shown in Table 5.2.

As reported, the results were not identical in both directions, as we can see in comparing the words "enhance" and "better" in tables 5.1 and 5.2. This is a result of the limitation of this approach that we had discussed earlier; however, the simplest way to overcome this problem is by taking the average of the values in both tables. Therefore, the total similarity score (calculated using equation 5.5) of Table 5.1 is 0.825 and the score of Table 5.2 is 0.829. By adding the results of both directions and averaging them, we get 0.827 which is the semantic similarity measure between the two sentences.

Furthermore, we can see that the word "better" was not a part of the first sentence but it is the synset of the word "improve" because this measure uses the words' most common synset in calculating the similarity score. Therefore, this similarity measure was able to relate them to each other. Lastly, from both tables, we can see that this measure only considers the nouns, verbs, adverbs, and adjectives in the comparison.

Table 5.1: Word Similarity Scores Between Text1 and Text2 Starting with S1

S1	S2	maxSim($w \in S_1, S_2$)
cup	cup	1.0
coffee	coffee	1.0
significantly		None
better	enhance	0.125
blood	blood	1.0
flow	flow	1.0

Table 5.2: Word Similarity Scores Between S1 and S2 Starting with S2

S2	S1	maxSim($w \in S_2, S_1$)
cup	cup	1.0
coffee	coffee	1.0
enhance	better	0.143
blood	blood	1.0
flow	flow	1.0

5.8.2 Word Alignment Matching

The second similarity measure used here is adopted from the Semantic Evaluation (*SemEval 2015*) competition [2]. The basis of the SemEval competition is to measure the semantic similarity between two sentences using a scale from 0 - 5, where 5 indicates identical pairs and 0 means no similarity. The system used in our research, proposed by Sultan et al. [120], was ranked 1^{st} dur-

ing the SemEval competition[3]. It operates by aligning semantically-similar words across the two sentences and it depends on extracting word similarity and extracting contextual similarity. To identify word similarity, three levels are considered as follows:

- Exact word match, where the similarity score is 1.
- A degree of similarity that matches non-identical words. In order to do that, the authors used a database called Paraphrase Database (PPDB) [4] to identify such relations. This level assigns scores between 0 and 1 which represent a degree of similarity.
- No similarity exists, which receives a value of 0.

An example of the alignment process is shown in Figure 5.14 where five words are aligned between the two strings, including words that have a similar meaning, "awarded" and "received". Therefore, the number of alignments between the two strings is five.

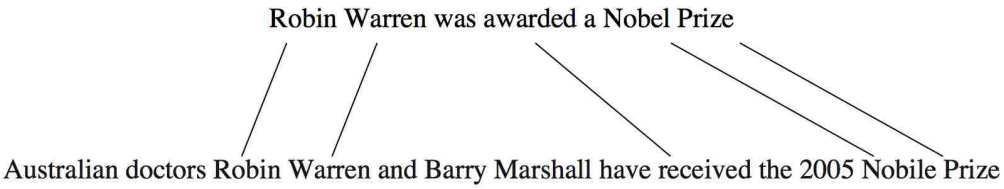


Figure 5.14: Example of the Alignment Process [2]

Extracting contextual similarity depends on two resources: syntactic dependencies and words occurring within a small window of the area of the two words to be aligned (three words from each

[3]SemEval is an ongoing series of evaluations of computational semantic analysis systems. The evaluation involves exploring the natural meaning of the language. This task is not intuitive to machines as it is for humans [2].

[4]http://paraphrase.org

side, left and right) [119]. Their approach also matches misspelled words when the difference is one letter using the Levenshtein distance.

Their work operates in a form of a pipeline that aligns words within the same category, as shown in their general architecture in Figure 5.15. Alignment of identical word sequences is the simplest form of alignment, as it aligns sentences that have identical words in the sequence and contextually similar words in both sentences. *Named Entity* is aligned separately to allow the alignment of full and partial names such as initials and abbreviations. The Stanford Named Entity Recognizer [42] is used to identify the names in both sentences that need to be aligned during named entity alignment. *Content Words* is used to match contextually similar words. *Stop words* used the same approach in matching contextually similar words, but because they are the last things that become aligned, it does not consider the neighboring words (on the left and right) in the alignment process, and it aligns word-to-word only.

Figure 5.15: Word Alignment Architecture [119]

The similarity measure is computed as follows:

$$Sim(S_1, S_2) = \frac{n_c^a(S_1) + n_c^a(S_2)}{n_c(S_1) + n_c(S_2)} \tag{5.6}$$

Where $n_c^a(S_i)$ and $n_c(S_i)$ are the total number of words and the number of aligned words in sentence S_i respectively. The words are aligned only when there is some semantic contextual similar-

ity between them.

5.8.3 Fuzzy String Matching

This similarity measure has been carried over from the baseline LCCv0 project discussed in Chapter 4. This measurement uses Levenshtein distance to calculate the difference between two sequences/sentences. The Levenshtein Distance [31] is the minimum number of edit operations (insertion, deletion or substation) of a single character that need to be applied to transfer one string to the other.

$$D_{s_1,s_2}(i,j) = min \begin{cases} D_{s_1,s_2}(i-1,j)+1 & \text{deletion} \\ D_{s_1,s_2}(i,j-1)+1 & \text{insertion} \\ D_{s_1,s_2}(i-1,j-1)+1_{s_{1i}\neq s_{2j}} & \text{substitution} \end{cases} \tag{5.7}$$

For example, to transfer the word *mitten* to *sitting*, we require three edits. This can be done by 1: mitten $\rightarrow$ sitten (substitution of "s" for "m"); 2: sitten $\rightarrow$ sittin (substitution of "i" for "e"); and 3: sittin $\rightarrow$ sitting (insertion of "g" at the end).

There are different variations of this measure: *token sort ratio*, *simple ratio*, and *partial ratio*. Token sort ratio sorts the strings that need to be compared in alphabetical order. It does not consider the position of the words in the comparison, as it resorts the strings and applies fuzzy string matching on them. The simple ratio compares the similarity between two strings having the same length. The partial ratio can compare two strings with different lengths by finding a substring of the shorter string length in the longest string. For example, "I have an appointment tomorrow" and "I have an appointment tomorrow with the doctor at 3:00 pm" have a partial ratio of 100% as the first sentence is a substring of the second.

In our approach, we used token sort ratio to measure the fuzzy matching similarity rather than simple ratio or partial ratio. This was done because simple ratio requires both strings to be of the same length, which is inpractical for the purpose of LCC; and the partial ratio looks to find sub-strings rather than matching the whole string. Token sort ratio has a drawback of not considering the position of the words in the comparison, but it takes into account all the words in both strings. For Instance, for the above example that reports a partial ratio of 100% between "I have an appointment tomorrow" and "I have an appointment tomorrow with the doctor at 3:00 pm", the token sort ratio between them is 69%. This ratio is more acceptable in our matching purposes because we are not looking to find a substring in both strings. Also, from the example above, the ratio 69% is more acceptable to represent the relationship between the two strings than 100% as we only use 100% to represent an exact match between two strings.

5.8.4 Combining Similarity Measures

The three similarity measures discussed above were combined to identify the most similar information in the knowledge base to the user input. A block diagram that illustrates this process is shown in Figure 5.16. LCC algorithm first minimizes the search space by ignoring any unnecessary comparison between the user input and the nodes/sentences in the knowledge base. This is done by ignoring the neighbors of the current node that have no similarity with the user input. This approach assumes that the neighbors will also have a low probability for a relationship with the user input, which can then be ignored.

Therefore, one way to visualize how the similarity measures are used in the LCC system is to use a hierarchy where the knowledge base is filtered and the search space is pruned until the algorithm finds the best match (the highest score) among the three similarity measures, or determines that no match exists, in which case it adds the information as a new knowledge.

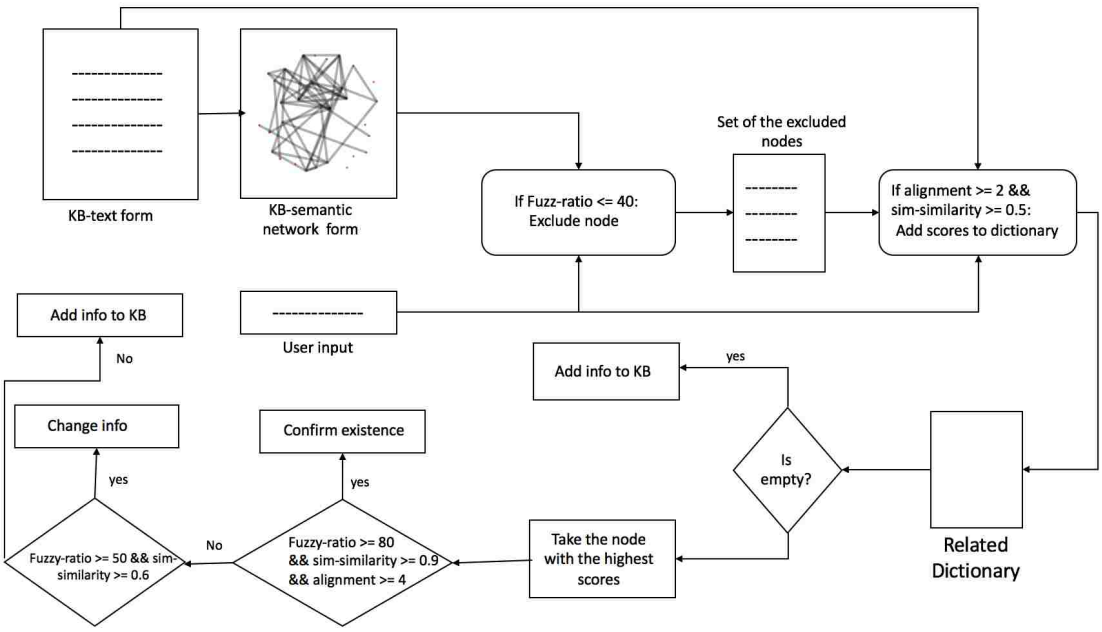


Figure 5.16: A block diagram that illustrates how the similarity measures work together

Fuzzy string matching is the simplest similarity measure and therefore the fastest to compute among the three similarity measures of interest here. Hence, this measure is used to prune the search space, as it does not require complicated computations to find the similarity score between the two compared sentences. Yet, selecting a threshold against which to compare the obtained similarity score was a critical task, as we did not want to have a very tight threshold that would significantly narrow down the search space. This will result in excluding nodes that could have a potential similarity to the user input. On the other hand, we did not want to choose a threshold that keeps the search space similar in size to its previous state (before pruning). In this case, the pruning would be unnecessary as the search space would only shrink by a small amount.

After trying multiple variations of what could be a good threshold to exclude or consider the nodes, we picked the value 40. We used this value as a threshold to exclude nodes after experimentally determining that 40 is a good threshold to exclude nodes. Because, during the experiments, using

a value less than 40 results in having many unnecessary nodes whose similarity with the user input is minimal. Choosing a higher threshold on the other hand, made the system exclude nodes that can be similar to the user input. For example, the fuzzy string ratio between “bananas don’t grow on trees” and “bananas are yellow” is 40. Using a threshold of more than 40 will assume those two sentences are not related; however, they are slightly related as they both deal with bananas. Another example that proves that a threshold of a less than 40 is not preferred is measuring the fuzzy matching score between “about 165,000 cars are produced every day” and “honey is the only food that will never rot” which is 36. It is clear that those two sentences relate to completely different topics; therefore, the value 40 is an acceptable threshold to use in this research. To visualize how the pruning process works, assume that the knowledge base contains information about foods, cars, and planets. When the user enters something related to cars, the learning process will start pruning the search space by comparing the fuzzy string matching between the user input and the current node in the knowledge base with the threshold 40. Nodes that have scores lower than 40 are added to the excluded list and all their neighbors that have edges with the current nodes are considered as potential unrelated information. Next, the learning process starts comparing the threshold (40) with the obtained scores between the user input and the neighbors of the excluded node. Computing the fuzzy ratio between the user input and the neighbors of the excluded node is used to eliminate any errors that can result from excluding neighbors that have a potential similarity with the user input. For example, a node that contains information about restaurants can have a connection with a node that contains information about delivery cars that is related to the user input assuming he/she enters information related to cars. Therefore, even if the user input does not have a relation with the statement about restaurant but it has a connection with its neighbor (the delivery cars). Hence, comparing the fuzzy similarity scores with the neighbors of the excluded node is considered necessary.

After the pruning process, the search for the best match continues by excluding the nodes that have

less than two aligned words, as it is not practical to consider nodes that have one alignment only. The nodes that have at least two aligned words and their semantic similarity measure above or equal to 0.5 are added to a dictionary called the *related dictionary*. The value of 0.5 is chosen to be the semantic similarity threshold for considering two sentences semantically related as suggested by Mihalcea et al.[89]. Those nodes/sentences are added with their scores to the related dictionary, where the key is the node number (ID) similar to the number used in the semantic network, and the values are the three similarity scores (fuzzy string matching, semantic similarity measure and the alignment score) between the node and the user input.

After collecting the nodes with the highest scores, LCC determines which nodes are related to the user input and ignores the others by choosing the node that has the highest similarity scores among the others. It does that by extracting the node with the highest score using the Max python function. Then, LCC determines whether the user's information (comparing to the dominant node) is new, a modification of existing information or similar to existing information in the knowledge base. An empty related dictionary indicates that there is no similar information in the knowledge base to the user input. If the dictionary is not empty, LCC takes the node with the highest score and starts checking whether the user input compared to the dominant node in the dictionary similarity scores are within the exact match thresholds, new information thresholds or modified information thresholds as described next.

In order for LCC to consider the user input as an existing information, the scores between the dominant node in the knowledge base and the user input should satisfy all the following thresholds:

- The fuzzy string matching score should be equal or above 80.
- The semantic similarity score is more than 0.9.
- There are at least four aligned words.

Those thresholds were chosen after performing a series of trial and error tests to determine the best scores that can yield acceptable results most of the time. The ranges of the matching scores were chosen to be tight, but we allowed a small gap to consider information to be identical when a slight difference between the user input and the most similar information in the knowledge base exists. For example, if the user enters "bananas are radioactive" and the knowledge base contains "bananas are slightly radioactive." should be considered the same as they both refer to the same thing, which is bananas are radioactive. During our experimental design to select the thresholds, we decided to choose those thresholds that match closely relevant information, as LCC has no way to understand the meaning of the sentences; rather, it depends on the matching scores to determine the closeness between two sentences.

The threshold of 80 means that the fuzzy string matching should match at least 80% of the information in both the user input and the most similar information in the knowledge base. The semantic similarity measure being 0.9 or above reflects a close semantic relation between the two compared sentences. For the fuzzy matching score, aligning four words is chosen to eliminate the effect of matching stop words, or words that do not hold important information such as: is, are, the, etc. However, the length of the sentences should include more than four words in order for this measure to be accurate.

Yet, when the fuzzy score > 50 and the semantic similarity score > 0.6, LCC considers that as a close relation to the user input with some modifications; therefore, LCC asks the user if he/she wants to exchange information. Those values are chosen not to be so high as the values for the exact information, and also not too low because having lower values of the fuzzy string matching and the semantic similarity measures can result in incorrectly assuming the information to be related. For example, "English is the language spoken in most US cities" and "US has many cities" are related as both refer to something in the US and its cities; however, they hold entirely different information. Therefore, they should not be considered as closely related, rather they should be

considered as two different pieces of information and add it to the knowledge base.

Lastly, if the dominant node scores in the related dictionary are not within the previous ranges, LCC considers the information as a new information.

During the learning process, the system assigns numerical values to the action taken (adding, exchanging and confirming information). Those values are assigned so the system can know what information has been changed, added or confirmed. A value of 0.5 is given when the piece of information is deemed to be new information and is added to the dictionary that keeps a record of the state of the information; a value of 1 is assigned each time information is confirmed to be an existing information. A value of -1 is given when the information is changed.

This learning algorithm is shown in Algorithm 3. This algorithm takes four input arguments: 1) The user text input. 2) The semantic network (graph) that has been created based on the knowledge base information. 3) The value dictionary that holds the numerical values associated with each node. 4) The archive dictionary that contains previous versions of the information associated to the modified nodes.

The learning process starts by creating the dictionary (dict) that maps the information in the knowledge base to their IDs. Lines 3 - 9 of the algorithm exclude the nodes and their neighbors that have fuzzy matching scores lower or equal to 40 compared to the user input. After shrinking the search space, LCC starts computing the similarity scores for the remaining nodes with the user input, one node at a time except for the excluded nodes (line 11). Line 12 calculates the alignments between the current node and the user input. Line 13 compares the number of alignments of the current node with the threshold (two). The node with two or more aligned words is passed to line 14, when the semantic similarity measure of the current node and the user input is computed and compared with 0.5 the score threshold. The nodes that pass these thresholds are saved in the related dictionary that contains the node ID and its associated similarity measures. Line 17 checks whether the related

```
Learning(input, graph, value, archive)
dict = dictionary(len(graph));
create mySet;
for g in range (len(graph)) do
    if g not in mySet then
        if fuzz.token-sort-ratio(question,dict[g]) ≤ 40 then
            for nb in graph[g] do
                if fuzz.token-sort-ratio(question,dict[nb]) ≤ 40 then
                    mySet.add(nb)
            end
end
for g in range (len(graph)) do
    if g not in mySet then
        alignments = align(dict[g], question);
        if (len(alignments[0]) >= 2) then
            if symmetric-sentence-similarity(question, dict[g]) >= 0.5 then
                related[g] = symmetric-sentence-similarity(question, dict[g]),
                fuzz.token-sort-ratio(question,dict[g]),len(alignments[0]) ;
end
if len(related.keys()) == 0 then
    add input to textFile;
    graph =create-net();
    update dict;
    value[g] = 0.5;
else
    maxValue = max(related, key = related.get);
    if related[maxValue][1] >= 80 and related[maxValue][0] >= 0.9) and
    (related[maxValue][2] >= 4 then
        value[maxValue] = value[maxValue] + 1.0;
        print (response of confirmation) ;
    else if related[maxValue][1] >= 50 and related[maxValue][0] >= 0.6 then
        Exchange information request ;
        if "yes" then
            change info;
            graph =create-net();
            value[maxValue] = value[maxValue] - 1.0;
            print (confirm the changing process)
        else
            print (information did not change)
        end
    end
    else
        add input to textFile;
        graph =create-net();
        update dict;
        value[g] = 0.5;
    end
end
```

Algorithm 3: LCC Learning Algorithm

dictionary is empty, which implies no similar information in the knowledge base exists. In this case, the information contained in the user input is added to the knowledge base and the graph is updated. The value and the dictionary (dict) in lines 17 - 21 are also updated. Line 23 extracts the dominant node with the highest scores and compares its scores with the thresholds of exact information matching. If the node passes the thresholds (lines 24 - 26), the rvalue dictionary is updated and a message to the user is sent to confirm the information.

Lines 27 - 37 deal with changing information when the node scores for the fuzzy string matching $>= 50$ and the semantic similarity is $>= 0.6$. Lines 39 - 42 add the information to the knowledge base as a new information because the scores were lower than the previous thresholds.

Figure 5.17 reports an example of how the LCC learning process finds a match to the user input. In this example, the user input "cars have 6 wheels" will be replacing existing information in the knowledge base. Therefore, the process starts by ignoring all the information that does not relate to the topic. This is done by comparing the fuzzy string matching between the user input and the current node/sentence above the threshold (40). Nodes/sentences that do not meet this threshold with their neighbors will be excluded from continuing in the comparison process. Later, the remaining nodes in the knowledge base are compared to the user input to determine how closely they are related to the knowledge base. The nodes that have similarity values ubove the thresholds (semantic similarity > 0.5 and they have more than two alignments with the user input) are added to the *related dictionary*. The dominant nodes in the related dictionary are compared later to evaluate whether they are information that needs to be added, modified or confirmed. In our example, the information is changed with the most similar information in the knowledge base, which is the first sentence.

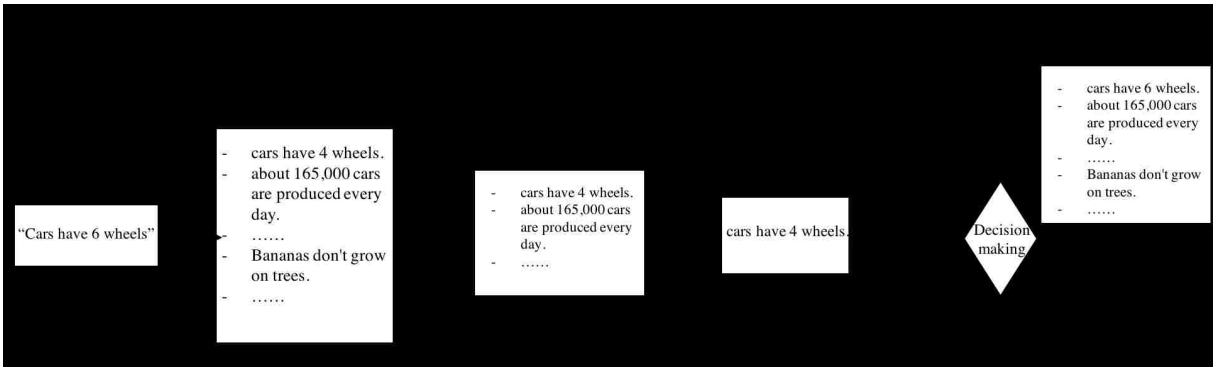


Figure 5.17: A Block Diagram of LCC Learning Component to Show an Example of How the Learning Process is Performed

5.9 Question/Answering System

The last unit from the LCC architecture that we want to discuss is the Q/A unit (shown in Figure 5.1). To answer users' questions related to the knowledge base, we used a similar approach to the learning algorithm to find the most suitable answer to the user question. Yet, rather than modifying the knowledge base when the user input contains different or new information, LCC outputs a message to the user indicating whether it has the same information when the similarity scores are within the range of confirming information (the fuzzy string matching score $>= 80$, the semantic similarity score > 0.9, and there exists at least four aligned words between the user input and the information in the knowledge base when the user input contains more than three words). When the user input contains slightly different information from what is in the knowledge base, LCC deals with it as partial matching when the thresholds are (fuzzy score in between 50 and 80 not inclusive, and the semantic similarity score in between 0.6 and 0.9 not inclusive). In this case, LCC will reflect that it has related information to the user input and will print out that information. When the matching scores are not within the previous thresholds range, LCC output a message indicating that it does not know the answer to the user question.

A block diagram that shows the process of how LCC handles user's questions input is shown in 5.18. As shown, LCC first needs to have the user input to be classified as related information and make sure it has a question mark. Later, if it determines that it is a question, it passes it to the question/answering unit where it compares the similarity scores with the scores obtained from comparing the user input with the most related information in the knowledge base. LCC also ignores the nodes and their neighbors when the fuzzy string matching is less than 40 (this is not shown in the diagram). After LCC determines whether the question exists in the knowledge base, related to information in the knowledge base, or no similar information is found, it responds to the user input with a message reflecting that finding.

For example, if the knowledge base contains information such as "cars have four wheels", when the user asks, "Do cars have four wheels?", the answer will be something like, "Yes, I have similar information". When the user asks, "Do cars have six wheels?", the system answers with, "I know that cars have four wheels". Yet, when the user asks about information that does not have any match in the knowledge base, the system will answer, "I do not know".

LCC can only answer a yes/no type of questions, so it cannot answer questions such as "how many parts does the car have?". We know that this a limitation of the question answering system. However, we should emphasize that LCC is focused mainly on the learning process. We leave this for future research.

Both trustworthy and untrustworthy users have access to the Q/A unit and the chatting unit. The knowledge base is not updated with any user-provided input in either unit.

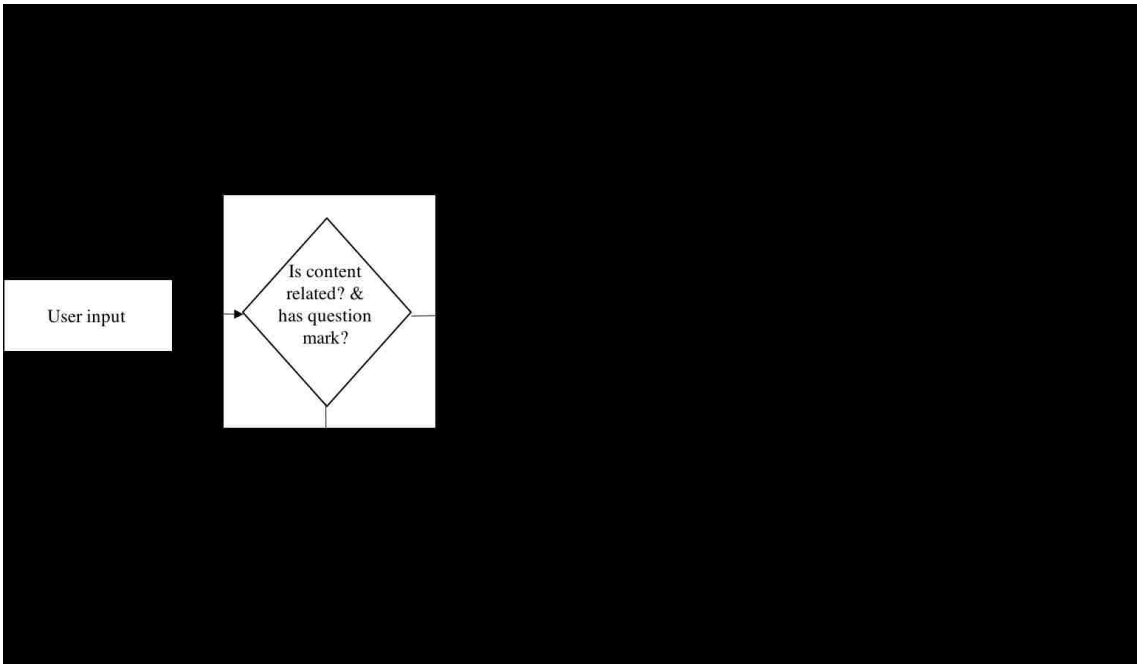


Figure 5.18: A Block Diagram to Show an Example of How LCC Answers the Users Questions

5.10 Summary

In this chapter, machine learning from a casual conversation is introduced and detailed descriptions of the components that LCC uses are presented. LCC is an interactive system that aims to allow authorized users to update the system's knowledge base through an easy, natural interactive conversation. Unauthorized users can only access the chatbot and the question answering components of LCC.

LCC accepts unrestricted natural language as the user input, analyzes the user input to pass it to the appropriate component to act accordingly on the user input. To optimize the process of searching for the most similar information in the knowledge base, LCC seeks to minimize the time required for the system to evaluate the user input by pruning the search space by excluding the nodes that have little or no similarity with the user input as indicated by different similarity

measures. The benefit of enabling the computer agent to recognize content-related information and reflect that in its knowledge base is to create a smart system that can learn from a conversation similar to how humans learn from each other. LCC also limited unauthorized access by prohibiting a untrustworthy user from adding or changing information in the knowledge base.

CHAPTER 6: LCC PROTOTYPE SYSTEM

The LCC system is designed to be an interactive conversational system that benefits from conversing with a human to update its knowledge base and beliefs regarding the knowledge that exists in its knowledge base. This chapter discusses the overall prototype of the LCC system, the modules that compose the prototype system, and how human interaction takes place with this prototype.

6.1 Prototype Overview

The prototype architecture consists of several modules that interact together to form the overall LCC system. All the modules were written in the Python language as it facilitates the process of implementing machine learning and natural language processing algorithms. The user interface was kept simple as, LCC uses the command line for an easy interaction with it. The command line was sufficient for the purpose of testing the prototype, as we do not need more than a window to allow the user to enter his/her text and the system to respond to it.

6.1.1 Launching LCC

In order to run the LCC system, first, we connect to the Stanford NLP server by using a Python interface as the Stanford Core NLP tools had been written in Java-8. The Python interface can be found in [6]. The connection to the Stanford server is necessary to operate the word alignment similarity measure as discussed in section 5.8.2. After connecting to the server as shown in 6.1, the LCC system runs in another window from the shell command line.

Figure 6.1: Establishing the Connection with Stanford Core NLP Server

After running the LCC system, the classifiers are trained and tested using the data that are provided in Comma-Separated Values (CSV) format, as shown in Figure 6.2. These data are gathered from an online article about random facts [5] and from data extracted from reviews on various topics as part of a project at the University of Illinois [49]. In this figure, LCC reports the accuracy of the testing data for the Naïve Bayes classifier, decision tree, and Max Entropy classifier. The way that Max Entropy classifier was set up is to report the log likelihood and the accuracy for 100 iterations. The training for Max Entropy was interrupted after the 30^{th} iteration as the log likelihood and the accuracy reached a steady state. Then, LCC launches the chatbot to be trained, as shown in Figure 6.3. During the launch, the semantic network of the knowledge base is constructed from the text file. The launching operation takes around 5-10 minutes on a MacBook Pro Core i5 processor. Upon completion of the launch, LCC starts by greeting the user and asking the user to enter "1" or "2" to determine his/her trustworthy label for testing purposes, as shown in Figure 6.3. In this figure, the user labeled herself as an untrustworthy person; therefore, LCC displays a message indicating that she has limited access to the LCC system with no access to the learning component.

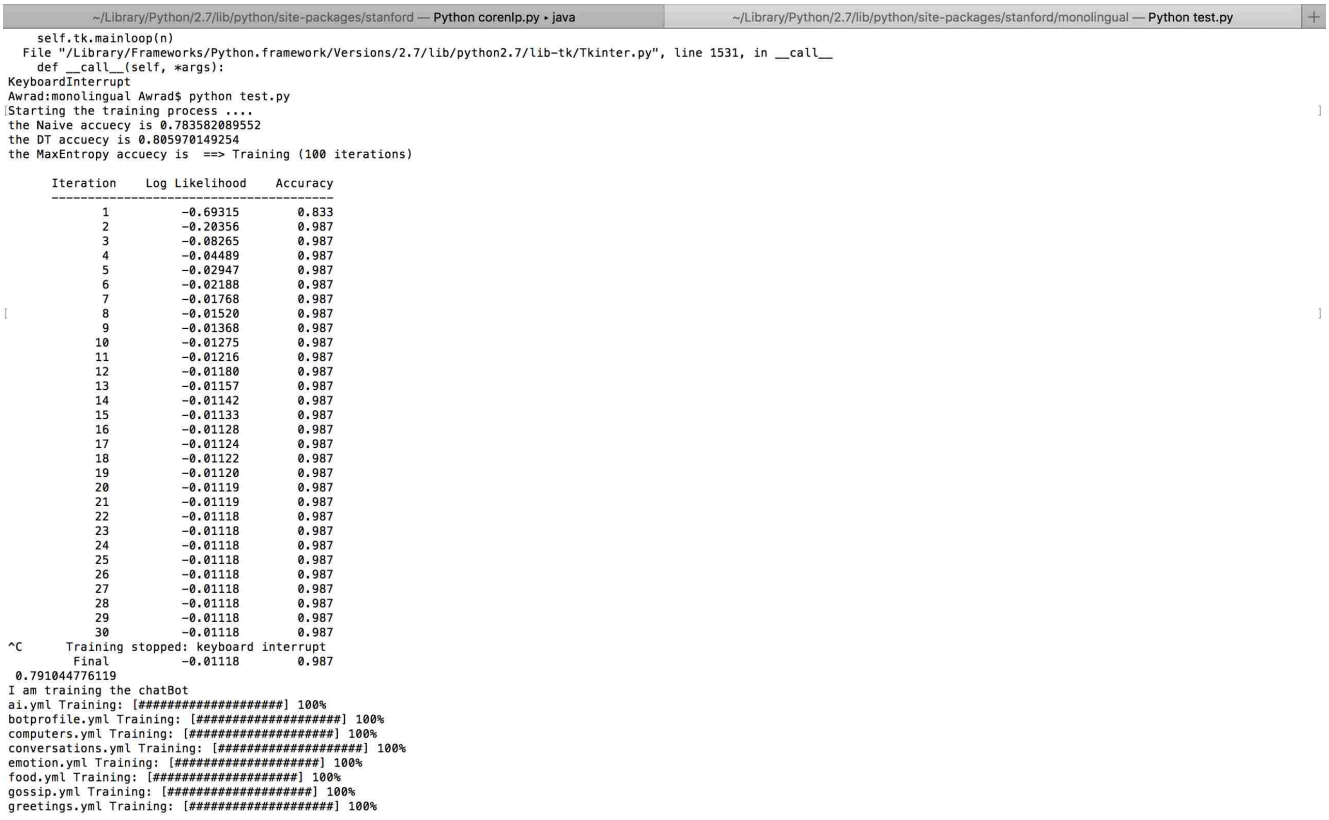


Figure 6.2: Training and Testing Naïve Bayes Classifier, Decision Tree Classifier (DT) and Max Entropy Classifier Respectivly.

```
I am training the chatBot
ai.yml Training: [####################] 100%
botprofile.yml Training: [####################] 100%
computers.yml Training: [####################] 100%
conversations.yml Training: [####################] 100%
emotion.yml Training: [####################] 100%
food.yml Training: [####################] 100%
gossip.yml Training: [####################] 100%
greetings.yml Training: [####################] 100%
history.yml Training: [####################] 100%
humor.yml Training: [####################] 100%
literature.yml Training: [####################] 100%
money.yml Training: [####################] 100%
movies.yml Training: [####################] 100%
politics.yml Training: [####################] 100%
psychology.yml Training: [####################] 100%
science.yml Training: [####################] 100%
sports.yml Training: [####################] 100%
trivia.yml Training: [####################] 100%
Hi there, Welcome to LCC system. How are you today?
I am good
wonderful
Can you please provide me with your trustworthy level? 1: for trustworthy users, 2 otherwise:
2
Sorry you are unqualified to enter the learning mode of my system, you only can access the chatbot or the QA part of the system
```

Then, LCC establishes a conversation with the user by either asking a question or providing a chatting statement chosen at random from a list of possible conversation-initiating outputs. A flow diagram that illustrates this process is shown in Figure 6.4.

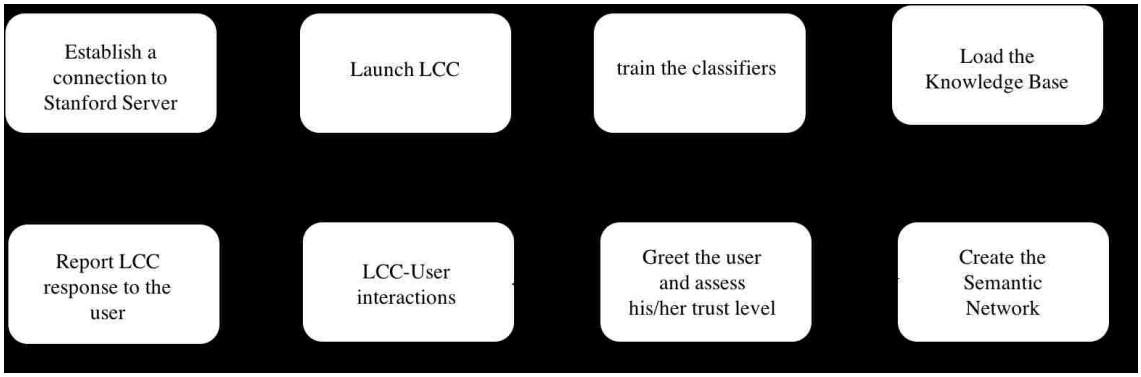


Figure 6.4: Flow Diagram of LCC Operating Process

6.1.2 Conversation Flow

The conversation with LCC is an interactive process, where the human user and LCC take turns in the dialogue. A block diagram of the LCC architecture is shown in Figure 5.1. There are two possibilities for the user input to be interpreted by LCC: 1) a chatting statement and 2) a content related statement that can be either a question related to LCC's knowledge base or information that needs to be considered in the learning process. The process of classifying the user input is performed in the classification unit. After deciding the proper unit that can correctly handle the user input, LCC activates the selected unit and transfers the user input to it. During the processing of the user input, the other units stay in idle states. For example, if the user input had been directed to the learning unit, the chatbot and the question-answering unit are in the idle state. This is important to allow the system to naturally transit between units depending on the user input without requiring the user to exit the current unit or to feel that he/she must stay within the same type of dialogue.

LCC uses word variation to report responses that involve greeting and reporting actions that involve the learning information, i.e., confirming existing information, exchanging request and adding request. LCC chooses from random lists that are associated to each of those responses to give a human-like variety of responses.

6.2 Interacting with LCC

After the system has been launched as described earlier, the user uses an interactive screen that accepts the user input and reports the system responses by displaying it to the same screen. The user can exit the conversation by typing the word "exit" during his/her turn. When the user enters an input, the system prints a message indicating that it is searching for the answer if it is related to its knowledge base or to the question-answering system before reporting the answer. The reason for printing this message is to let the user know that the system is processing the input, as the system may take a few seconds before reporting the answer to the user depending on the size of the knowledge base (disscuseed later in section 7.3). Interacting with LCC is simple because there are no restrictions on the structure of the text input that the user can use as long as the user writes in English and spells mostly correctly. An example of how the conversation flow takes place with LCC is shown in Figure 6.5.

6.3 Summary

In this chapter, we cover the prototype description of LCC that includes how the user can launch the system and the components that need to be established before running LCC. We also described the conversational flow of LCC, and the interface used to communicate with the user. We provide an example that shows how LCC responds to the user input.

```
Hi there, Welcome to LCC system. How are you today?
Is apple a fruit?
amazing
Can you please provie me with your trustworthy level? 1: for trustworthy users, 2 otherwise:
1
Please enter something related to my knowledge
 Is apple a fruit?
 I am checking for the answer
Yes, I have the same info
cars have 6 wheels sometimes
 I am checking my KB....
I am about to exchange info, do you still want to take this action? and change the information of  cars have 4 wheels.
with cars have 6 wheels sometimes  ?
Please enter "YES" or "NO"
NO
ok information is stayed as it was
how are you today?
glad to know
know what?
yes
humans can think
 I am checking my KB

I added this info as no similar information exists
exit
bye for today
Awrad:monolingual Awrad$
```

Figure 6.5: A Short Example that Shows How LCC interacts with a User.

CHAPTER 7: TESTING AND RESULTS

This chapter covers the evaluation of the LCC prototype system. A series of tests were performed to assess the performance of the system as well as to show the validity of the system for use by test subjects. This chapter breaks down the evaluation process into three different assessments: The testing procedure involves 1) Test 1: a series of functional tests performed by the developer of the LCC to test various aspects of the system; 2) Test 2: a two-step user testing process that involves asking humans test subjects to use and/or evaluate the LCC performance; and 3) An experimental and theoretical assessment the time complexity of LCC to determine how scalable the system is when used in real-world applications, and the impact of increasing the size of the knowledge base on its runtime performance.

7.1 Test 1: Functional Testing

Functional testing involves evaluating the influence of using input patterns of varying levels of difficulty on the system's accuracy. The functional tests also assess the impact of using the two levels of decison-making in the LCC classification process.

7.1.1 Test A: Difficulty Level Testing

7.1.1.1 Description and Procedure

To evaluate the performance of the LCC system, we conducted a series of tests with 90 statements/sentences that played the role of user inputs. Those sentences were placed into three groups based on their level of difficulty. Each group contained 30 sentences. Within each of those groups,

each sentences was labeled by the developer as to the expected correct action that needed to be performed by LCC prototype. The actions here represent the available modes of the LCC system (add information, confirm information, exchange information, answer a question and respond to a chatting statement). These developer-assigned labels were considered the ground truth (expected) labels that were later used to determine the correctness of the action taken by LCC in response to each corresponding sentence. The accuracy measure was calculated by comparing the label with the obtained (actual) action by the LCC prototype.

The LCC prototype was given an initial knowledge base consisting of thirty sentences. This initial knowledge base is shown in Appendix A. These sentences were obtained from a website [5] that include random facts related to various topics; we used cars and foods. The classifiers were trained and tested as described in section 5.4.1. The test sentences were created by the developer with full knowledge of the contents of the KB. The test sentences were composed with a difficulty level consistent with the difficulty level specified for each group.

The way the test sentences are conducted does not require a reset of the knowledge base every time a statement or question from the available testing set is entered. This is because we designed the testing sentences with respect to entering the statements sequentially. For example if we modify a statement such as “cars have 4 wheels” to “cars have 6 wheels”. The following question can be “Do cars have 6 wheels?”. This creates a more challenging test, as the coming testing sentence/question can depend on our expectation of the label assigned to the previous testing sentence. However, this is not always the case , as we also used sentences that are not affected by the previous modifications that took place in the KB as a result of the prior statements presented. We thought that this would be a more robust procedure to test as it emulates how normal humans can use the system.

The difficulty levels were assigned as follows: Group 1 included sentences that were considered easy (i.e., very similar to sentences in the knowledge base) and which the LCC prototype should

easily handle correctly. For example, “The first car accident occurred in Ohio”. The expected label considers the information as similar to existing information contained in the knowledge base “The first car accident occurred in 1891, in Ohio”. An example of information that is expected to be added is “Parking is a problem in the US”. Because no similar information exists in the knowledge base, the LCC system should add this info to the knowledge base. An example of information that LCC should direct to the chatbot is “Do you like coffee?” as the classifiers had trained to consider similar information as chatting statements. An example of information that is expected to be an exchange request is “Bread is the most stolen food in the world” as the knowledge base contains “Cheese is the most stolen food in the world”.

Group 2 includes sentences that pose a challenge to LCC. These sentences do not use exact words to what LCC has in its KB; therefore, there is less certainty as to whether LCC is able to direct the input to the correct component. For example, the test case “Decaffeinated coffee enhances the flow of blood” should be related to “Drinking a cup of caffeinated coffee significantly improves blood flow” and LCC should issue an exchange request. An example of a chatting input that makes it uncertain whether LCC will be able to classify it as chatting statement is “I like to travel to Europe”. LCC can misclassify this statement and instead of considering it as chatting statement, it can be added to its knowledge base because the classification algorithms have not seen a similar statement before.

Group 3 includes information that is so challenging that it will likely “break” the LCC prototype. This is done to explore the limits of our LCC prototype. In engineering, this would be considered “destructive testing”. Therefore, we are uncertain as to how the system will deal with it. For example, “who is Henry Ford?”. We know that the knowledge base contains information about “Henry Ford made a car out of soybeans”, however; the only similar thing between the two statements in the name Henry Ford and it is uncertain whether the system can respond to this input with “I don’t know” or consider it as chatting statement. Another example is “In 1978, the chickens were more

flabby". We knew that the knowledge base contains "Chicken contains 266% more fat than it did 40 years ago" but it is uncertain whether the LCC system can relate the information to its existing knowledge, as both statements use completely different words and structure.

During the test design, the number of test sentences in each type of label was not the focus; instead, the main focus was on the level of difficulty of each input. However, based on our previous observations of the system performance, we intended to create more sentences that we labeled as exchanging request and chatting statements because we knew that LCC has a higher probability to mishandle those inputs.

7.1.1.2 Results and Discussion

The resulting accuracy for each of the tests in the three groups are reported in Table 7.1.

Table 7.1: Accuracy Measurement of the Three Groups

Group	No. of test sentences	Accuracy
Group 1	30	96.7%
Group 2	30	70.0%
Group 3	30	43.3%

As expected, the accuracy for Group 1 was the highest among the other two groups, nearing 100%. It only mishandled one input that was expected to be a question for the chatting statement: "can we chat?" that the classifiers labeled it as related information. Therefore, the LCC prototype directed the input statement to the Q/A system (because it also saw the question mark).

As expected the accuracy decreased for Group 2, as the LCC system was able to handle correctly

70% of the 30 test sentences. Based on our expectation, this is considered an acceptable result considering the increasing level of difficulty of the second group. A good point of comparison would be to compare the results of LCC with those of others found in the literature. However, we could not find any comparable tests in the literature. Therefore, using the standard academic threshold of 70% as minimal passing, we used 70% as the minimum for acceptable results in our work.

The accuracy was further reduced for the third group, as it was only able to handle less than half of the provided inputs. This dramatic drop was expected, as the inputs were purposely misleading and were likely to be mishandled by LCC. Yet, most of the errors happened when the classifiers mistakenly classified the input to be chatting when it was not. For example, the sentence "I am confused and worried" that was labeled as chatting statements, was classified by LCC as content-related. Other test statements that were considered content-related were classified by LCC as chatting for example, "do you think a spoon is an utensil?" had been classified as chatting question, while the label was a content-related question.

An alternative way to assess the results gathered from this test was to combine the three test sentences in the three groups and report the accuracy over all the test sentences according to the type of action expected.

Table 7.2: Accuracy Measurement Among the Five Categories

	Exchange	Same	Add	Chat	QA	Total
Count	24	15	18	21	12	90
Accuracy	82.6%	73.3%	72.2%	75.0%	83.3%	73.3%

Table 7.2 reports the accuracy and the number of test sentences under each label. LCC was able

to handle correctly 73% of the simulated inputs, which can be considered an acceptable average considering the level of difficulties of the third group.

7.1.1.3 Analysis of Failed Tests

To better understand where LCC failed to handle the input data, we examined the inputs that LCC failed to handle correctly. For the information to be exchanged, the inputs that were mishandled were the test sentences considered as existing information in the knowledge base because the similarity scores between both statements were within the range of the exact match. For example, "There are two and a half billion cars currently in use on earth" was considered by LCC as similar infromation to "There are 1 billion cars currently in use on earth". This can be fixed by tightening the required match scores on inputs that need to be considered as exact matches; however, doing this affects the matching of exact information in a way that LCC only accepts exact information that uses the same words as the information in the knowledge base as similar matches (confirm information). For example, "automobiles have 4 wheels" and "cars have 4 wheels" would be considered as different information and an exchange request will be executed by LCC.

LCC sometimes mishandled exact information by either considering these inputs as chatting statements because of mislabeling the test sentence by the classifiers, as in "trees do not grow bananas", or it tried to incorrectly exchange the test sentence with similar information, such as "almost half of the Americans eat sandwich every day" should be similar to an existing information in the knowledge base "49% of U.S. Adults eat one sandwich a day". However, LCC cannot understand that "49%" and "almost half" are the same thing, which made it issue an exchange request. Four cases were reported to be mishandled fell within this category.

For information that needs to be added, the problem was always misclassifying the input as chatting information by the three classifiers that contribute in the majority vote; however, all the misclas-

sified test sentences were from Group 3. There were six such cases from the total of 18. For example, “I don’t know if cats can think”: this test sentence has the part “cats can think” that should be considered as content-related; however, it had been classified as a chatting statement by all three classifiers, which resulted in not activating the second level of the classification process (see section 5.4 for details). So, it misclassified the test sentences to be chatting information and LCC was not able to remove the first part from the input “I don’t know if”. A similar problem arose in five chatting statements that were misclassified and directed to the Q/A engine. For example,“are you interested in eating a sandwich?” Also, there were four input sentences that were directed to the chatbot instead of the Q/A system. One example is “Do you know that Henry Ford made a car from soybeans?”. This happened because LCC classifiers have not been trained on similar data during the training process. The idea of using sentences/questions that we knew LCC had not trained on them before was to examine how LCC would act when the user enters questions/statements that the used classifiers were not aware of them.

7.1.1.4 Summary of Test A

From the observed results, we can say that the accuracy over all the labels was found to be acceptable as the lowest obtained accuracy was above 72%, which reflects the number of test inputs that LCC was able to handle correctly, considering that is the 70% acceptable passing rate. Also, we can infer that the main problem that faces LCC is the limitation of the classification process. Although we sought to improve the classification algorithm by deploying the second level, it was not always a solution because it was not always activated; it was only activated when there is no unanimity in the output label of the contributed classifiers in the first stage of majority vote.

7.1.2 Test B: Classifiers Testing

7.1.2.1 Description and Procedure

To evaluate the performance of the classification approach used in LCC, and to measure the impact of using Ensemble learning and the two-level classification process, we compared the classification results of the Ensemble learning with the individual performance of each of the classifiers that contribute in the majority vote in the Ensemble learning. The two levels of the Ensemble learning are discussed in section 5.4. LCC combines the results of both of these levels. This means that when there is unanimous agreement on the label of the user input, LCC considers the majority vote among the three classifiers in the sentence level. However, when there is no unanimity in the first stage, thereby triggering the second stage, LCC uses only the results form the second level and neglects the results of the first level.

To report the results, we used several metrics that are commonly used to evaluate the classifiers as suggested by [105]. During our discussion of those metrics, TP refers to True Positive statements classified as (content-related), TN refers to True Negative statements (chatting), FP refers to False Positive (chatting statements classified as content-related) and FN refers to False Negative (content-related statements classified as chatting). Those metrics are as follows:

- True positive rate (TPR): measures the proportion of the actual positives (related information to the KB) that were correctly identified; Therefore, the higher the value, the better the classifier. This is also known as sensitivity or recall.

$$TPR = \frac{TP}{TP + FN} \tag{7.1}$$

- False positive rate (FPR): represents the number of negative (chatting) statements that are

wrongly categorized as positive statements (content-related). This implies that the desired value needs to be small, as that will minimize the number of examples that had been classified by mistake to be content-related while they were originally labeled as chatting statements.

$$FPR = \frac{FP}{TN + FP} \tag{7.2}$$

- Accuracy is the measure of the closeness of a measured label by the classifier to the actual label.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.3}$$

- Error-Rate, which is the basically the complement of the accuracy measure.

$$ErrorRate = 1 - Accuracy \tag{7.4}$$

- Precision, also known as *positive predictive value*, it intends to answer what is the correct ratio of the actual positive values/label over all the positive identification. The higher the value, the better the classifier, as it measures the number of sentences that had been classified as related information over all the sentences that had been classified as related information. It is calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{7.5}$$

The test involves comparing the performance of each of the three classifiers (Naïve Bayes classifier, Decision Tree classifier and Max Entropy classifier) that contribute to the majority vote for both levels.

The performance was examined using the discussed metrics on two datasets: the first dataset (Dataset-1) contains 90 sentences that were manually labeled by the developer as either content

related statements or chatting statements. Dataset-1 is shown in Appendix E, was labeled by the developer, and is broken down as follows:

- 30 are chatting statements.
- 30 are pure content-related statements.
- 30 are content-related statements that are preceded by chatting expressions to measure the ability of the LCC prototype to separate the chatting expression from the content-related portion of the dialog.

The results are reported in Table 7.3. The second dataset (Dataset-2) was the same one used in section 7.1.1 (shown in Appendix B). The sentences in the knowledge base were also labeled as either content-related or chatting statements manually by the developer. The classifiers testing do not involve using the data original grouping described in Section 7.1. This is because in the classification testing, we only consider binary labeling (content-related or chatting). The results of this data are reported in Table 7.4.

Table 7.3: Comparison of Classifiers Performance (Scaled to a Range Between 0 and 1) on Dataset-1

Classifier	TPR	FPR	Accuracy	Error Rate	Precision
LCC	0.53	0.03	0.68	0.32	0.97
Naïve Bayes	0.28	0	0.52	0.48	1
Decision Tree	0.47	0.57	0.46	0.54	0.62
Max Entropy	0.42	0.4	0.48	0.52	0.68

7.1.2.2 Results and Discussion

The results reported in Table 7.3 for Dataset-1 show that LCC has better performance when compared to Naïve Bayes, Decision Tree, and Max Entropy classifiers individually. LCC's accuracy and TPR (recall) are higher than the values of the other classifiers. The error rate is lower than the values calculated for the other classifiers. However, the FPR of the Naïve Bayes classifier is slightly lower than the one reported by LCC. Yet, this slight difference is not so important, as the gap between the precision and the recall values reported for the LCC system is smaller than the gap in the Naïve Bayes classifier. As reported, the overall performance of LCC on this dataset was better than for the individual classifiers. This good performance is a result of activating the second level of the classification process multiple times when there was not unanimous agreement on the label assignment.

Table 7.4: Comparison of Classifiers Performance (Scaled to a Range Between 0 and 1) on Dataset-2

Classifier	TPR	FPR	Accuracy	Error Rate	Precision
LCC	0.85	0.26	0.83	0.17	0.93
Naïve Bayes	0.8	0.16	0.81	0.19	0.95
Decision Tree	0.76	0.26	0.76	0.24	0.92
Max Entropy	0.92	0.32	0.87	0.13	0.92

In Table 7.4, Max Entropy classifier has a better overall performance compared to the other approaches. LCC was the second best performance. Also, the FPR for LCC was lower than the one reported for Max Entropy. LCC performance depended mostly on the majority vote on the first level because the second level of the classification process was not activated as many times as in

Dataset-1. This was a result of having the classifiers either being in total agreement for the label of the given input (even when it was not the correct label), or because LCC could not extract the nouns in the input as a result of the errors in speech tagging when it does not recognize the noun in the sentence and gives it a different label.

The results of both tables reflect that LCC performance is bounded by the classifiers that contribute in the majority vote. It only outperforms the individual classifiers when it activates the second level at the proper time, which was the case in Dataset-1. The benefit of the second level in LCC can be seen mostly when the text input contains complex sentences. Also, when the part of speech tagging does not detect the nouns in the input, the LCC second level is not activated, as it depends on extracting the nouns in the sentence and feed them to the ensemble learning classifiers to determine their label. In general, the classifiers are affected by the quality and quantity of the training data. The training data that had been used for training the classifiers contained 522 sentences. We picked this size to make the training time for the classifiers acceptable (all the classifiers are trained within 15 minutes). The training process can be improved by including more training data; however, this would affect the training time by making it longer. Additionally, to have better training data, we need to use non-noisy data. Hence, the data used here are annotated manually to avoid having noisy data. LCC cannot guarantee a correct label for the given input when the user's input contains new information that the classifiers had not trained on.

7.2 Test 2: User Testing

The objective of this test was to determine what humans think about the LCC system and whether or not it was able to deal with the user input correctly. This assessments was done by creating two groups of test subjects. The first group, called Group A, allows subjects to interact directly with the LCC prototype using typed text. Later, they were asked to evaluate LCC performance by

completing a survey. The second test (Group B) was designed to ask anonymous testers to evaluate the performance of the LCC system by presenting them with logs of the interactions with subjects from Group A and asking them to examine the interactions. Group B subjects were subsequently asked to complete a survey to evaluate the system performance.

The expected outcome of this test should reveal whether test subjects from Group A were able to change information in the knowledge base when they entered the LCC prototype as trustworthy users and should not be able to modify the knowledge base when they entered in as untrustworthy users. The testers from Group A should be able to initiate a conversation with LCC and inquire for knowledge that already exists in the knowledge base. We believe that Group B subjects should be able to sense the benefit of having such a system and be able to give their opinion regarding its performance. The benefit of testing with the second group is to expose LCC to a larger test population and obtain their feedbacks and opinions.

7.2.1 Group A testing procedure

The first group of test subjects (Group A) consisted of 30 persons aged 18 and above who had never interacted with LCC before. The procedure used was as follows: a test subject was first informed about the system. Then, a demo of how someone can interact with the system was shown to the subjects by the developer, such as how a user can add information, ask a question about something, chat, confirm and exchange information. All subjects in this group were given access to the knowledge base to give them an idea about what information LCC contains. The knowledge base used in the testing included 30 sentences and is shown in Figure 7.1. This number was selected to avoid overwhelming the user by having to review a large document, as well as to achieve fast response to the user input. The testers were asked whether they had any questions before they started using the system, and were also told to let the developer know whether they had questions

during the test. The developer, after giving all the needed instructions, stepped back to allow the subject to freely interact with the LCC prototype system.

The test subjects were asked to not provide any information about their identities. We labeled the test subjects surveys and logs by numbers from 1-30. Therefore, every test subject had a number and this is how we connected the surveys with the user logs.

Each subject was asked to log in twice, once as a trustworthy user and then later as an untrustworthy user. Both terms were explained to the subjects before they started, and they were told what they should expect in each mode. In the untrustworthy version, the subjects were asked to test whether they were still able to modify the knowledge base or not. The subjects were also informed that their logs would be saved to be used as part of the testing assessment with the second group of testers. The only requirement to perform the test is being over 18 and able to use a computer keyboard. The subjects were given a survey and asked to complete it after finishing the test. The survey includes 13 statements shown in Table 7.5 that can be divided into three groups:

- General statements to test what the subject thinks about conversing with LCC.
- Statements to assess LCC performance when he/she logged in as a trustworthy user.
- Statements to assess LCC performance when he/she logged in as an untrustworthy user.
- One open-ended question to allow the subjects to provide feedback and any thoughts they might have regarding LCC.

The subjects were asked to use a 5-point Likert scale to evaluate the statements in the survey, where 1 represents completely disagree/totally unsatisfied and 5 means strongly agree/totally satisfied with the statement. The subjects can use any integer value within this range that represents a degree of satisfaction or agreement. The subjects were also informed that the test can take between

20-30 minutes. Finally, the subjects were told to perform at least 10 dialogue turns for each of the trustworthy and the untrustworthy tests before exiting the conversation. This is important to ensure having valid conversations for the second group of subjects.

7.2.2 Group A Testing Results

To report the results collected from Group A users, we calculated the mean and the standard deviation for each of the questions shown in Table 7.5 among the 30 test subjects.

7.2.2.1 Analysis of Survey Results for Group A Tests

The below statements are meant to collect general information to know what the subjects thought about using LCC. Those statements do not test the system's functionality, but rather they were intended to gauge the subjects' opinion in using LCC. We now discuss each of the question/statement in the survey for Group A.

1. "I found the conversation with the system to be natural, as if conversing with a human."

   This statement is used to measure the naturalness of the conversation and how well it is compared to talking with a real human. The obtained average value (3.00) is neutral. This was expected, as talking with a computer agent is generally not nearly as natural as conversing with humans. Besides, the focus of LCC was primarily on how to learn from a human and only secondarily about creating a natural conversation, which is left for our future research. The standard deviation (1.15) was high, reflecting significant variance in the answers.

2. "I found the system to be intuitive and easy to use."

   This statement in the survey was designed to determine the user opinion about how easy

Table 7.5: Group A Questionnaire Survey

	Strongly disagree				**Strongly agree**
1. I found the conversation with the system to be natural, as if conversing a human.	1	2	3	4	5
2. I found the system to be intuitive and easy to use	1	2	3	4	5
3. I found the user interface to be acceptable for the purpose of this program.	1	2	3	4	5
4. As a trustworthy user, I found that the system correctly added new information to its knowledge base when appropriate.	1	2	3	4	5
5. As a trustworthy user, I found that the system correctly neglected to add information that is already in the knowledge base when appropriate.	1	2	3	4	5
6. As a trustworthy user, the system correctly identified the most similar information in the knowledge base and requested my approval to replace it before doing so.	1	2	3	4	5
7. The system was able to correctly distinguish between chatting statements and content-related statements.	1	2	3	4	5
8. As a trustworthy user, I found the system was able to answer my questions when it ends with question mark.	1	2	3	4	5
9. As a trustworthy user, I found that the system correctly transitions back and forth between chatting, question-answering system and processing content related information.	1	2	3	4	5
10. As a trustworthy user, I found that the system was able to correctly separate the chatting information from the related information.	1	2	3	4	5
11. As an untrustworthy user, I found the system correctly prohibited me from adding and updating information in its knowledge base.	1	2	3	4	5
12. As an untrustworthy user, I found that the system answered my questions correctly according to its knowledge base.	1	2	3	4	5
13. Please indicate any final thoughts and comments about the system.					

Table 7.6: Means and Standard Deviations for Group A Survey

No.	Statement	Mean	STD
1	I found the conversation with the system to be natural, as if conversing with a human	3.00	1.15
2	I found the system to be intuitive and easy to use	4.40	0.81
3	I found the user interface to be acceptable for the purpose of this program	4.37	0.8
4	I found that the system correctly added new information to its knowledge base	4.20	0.93
5	LCC correctly neglected to add information that is already in the knowledge base	4.47	0.97
6	The system correctly identified the most similar information in the knowledge base	4.30	1.02
7	The system correctly distinguish between chatting statements and content-related statements	3.33	1.24
8	I found the system was able to answer my questions	4.33	0.84
9	transitions back and forth among the users' different inputs	3.83	1.12
10	LCC was able to correctly separate the chatting information from the related information	3.59	0.95
11	LCC prohibited me from adding and updating information in its knowledge base	4.90	0.40
12	LCC answers my questions	4.24	0.99

it was to use the system by regular users who do not have information about the internal architecture of the LCC system. The average among the collected responses was above 4 points on the Likert scale (4.40). This can be considered as good agreement with the statement. The standard deviation (0.81) reflects that the testers' answerers were centered around the mean as most of them either agreed or totally agreed with this statement.

3. "I found the user interface to be acceptable for the purpose of this program."

   This statement is used to determine what people think about the LCC interface. The mean of 4.37 reflects that the subjects generally think that the interface is acceptable for the purposes of this project. The standard deviation value (0.8) reflects that the collected responses were closely grouped around the mean value.

7.2.2.2 Trustworthy User Testing

To test whether the trustworthy subject can have the full functionality while using the LCC system, the survey includes seven statements that evaluate whether LCC was able to make the correct

decisions about how to handle the subjects' inputs.

4. "As a trustworthy user, I found that the system correctly added new information to its knowledge base when appropriate."

   The test subjects were encouraged to confirm that the information has been added to the knowledge base by either asking a question related to the added information or by trying to add the information again, in this case, the system should indicate that similar information already existed. The average of the subjects' answers of 4.20 reflects the subjects' general agreement with the statement. The standard deviation of 0.93 suggests that the data were not widely scattered around the mean.

5. "As a trustworthy user, I found that the system correctly neglected to add information that is already in the knowledge base."

   To evaluate whether LCC was able to confirm information already existing in the knowledge base, the subjects were asked to add information that already existed in the knowledge base. The mean of 4.47 indicates that the subjects were predominantly satisfied (25 of 30 voted 4 or 5 in the Likert scale). The standard deviation of 0.97 meaning that most answers were reasonably close to the mean value.

6. "As a trustworthy user, the system correctly identified the most similar information in the knowledge base and requested my approval to replace it before doing so."

   This statement is used to test what the subjects thought of LCC's abitity to change existing information in the knowledge base. Among the 30 subjects, 26 gave this statement 4+[1] points on the Likert scale. The mean value of their responses was 4.30 (see Table 7.6). This reflects that the subjects were satisfied with LCC's performance when modifying existing information.

[1] 4+ refers to assigning values of 4 and 5 on the 5-points Likert scale

7. "As a trustworthy user, the system was able to correctly distinguish between chatting statements and content-related statements."

   This statement is meant to test how well LCC was able to correctly classify the user input. The mean of the collected answers was above 3 (3.33) (see Table 7.6) and the standard deviation was high, (1.24) meaning that the answers were spread out on the Likert scale. LCC did not perform as well as we hoped because the classifiers had many incorrect decisions, mostly when the subjects use statements that LCC classifiers never encountered before. That produced a random label for the given input. We are planning to address this issue in future research by allowing LCC to continuously update its training data using the user input.

8. "As a trustworthy user, I found the system was able to answer my questions when it ends with a question mark."

   This question is designed to learn whether or not LCC was able to answer content-related questions. The question mark is important to be included in the question, as LCC does not have any other way to distinguish between statements and questions. Other forms of distinguishing mechanisms will be left for future work. However, when the statement contains chatting information, it will be directed to the chatbot. The mean (4.33) (see Table 7.6) was above 4 and the standard deviation (0.84) indicated that the subjects' answers were grouped around the mean value.

9. "As a trustworthy user, I found that the system correctly transitions back and forth among chatting, question-answering and processing content related information."

   This statement is used to measure the flexibility of LCC to respond to different types of statements, and its ability to transition between the statements naturally. Twenty-six test subjects had reported 3+ points on the Likert scale, and 20 of them gave 4+ points. The reported mean was 3.83 (see Table 7.6). Yet, the standard deviation is high (1.12) reflecting

widespread responses on the Likert scale.

10. "As a trustworthy user, I found that the system was able to correctly separate the chatting information from the related information."

    This question is meant to test whether LCC is able to separate chatting information from content-related information in complex sentences that contain both. After checking the subjects' logs, we saw that most of the subjects did not use such statements in the testing process. We think that this happened because the subjects interpreted this statement as similar to statement 7 in the survey. Therefore, the mean and the standard deviation were close to the values for statement 7 (3.59 mean and 0.95 STD as per Table 7.6.

The overall results of Group A indicate user satisfaction in the performance of the LCC system, where the means for all the statements in the survey were 3 and above.

7.2.2.3 Untrustworthy User Testing

The human test subjects were then asked to register as untrustworthy users to test two concepts: was the system successful in prohibiting them from changing information? and was the system able to answer their questions related to the information in the knowledge base?

11. "As an untrustworthy user, I found the system correctly prohibited me from adding and updating information in its knowledge base."

    LCC should protect the information in the knowledge base when it detects that the user is untrustworthy; therefore, this statement is intended to evaluate that. The mean value (4.9) (as in Table 7.6) was close to 5 with a very low standard deviation (0.4). This strongly suggests that the system was able to protect the information in the knowledge base successufly.

12. “As an untrustworthy user, I found that the system answered my questions correctly according to its knowledge base.”

    This statement was designed to measure how successful LCC was in answering content-related information when the user is logged as untrustworthy. The mean value was 4.24 and STD was 0.99 (see Table 7.6). Twenty-six (26) of the test subjects voted with 4+ for this statement. This reflects the satisfaction of the majority of the test subjects.

7.2.2.4 Summary

In general, the responses by Group A test subjects were positive. Furthermore, among the twelve survey’s statements, eight of them reported means of 4, and above.

7.2.3 Group B Testing Procedure

We designed group B testing to expose LCC to more humans and to report their opinions about the system and its performance. The test subjects in this group were asked to evaluate the system performance by reading the dialogue logs of two conversations from Group A test subjects that were chosen randomly from the 30 collected conversation logs. The Group B testers were also given the content of the knowledge base before and after the conversation so they could compare between the two. The Group B survey contains nine statements only. We removed the statements related to the system interface evaluation and statements mentioning untrustworthy users to make it easier for the participant as the survey was completed online and the Group B subjects did not interact with the developer. The survey is shown in Table 7.7. However, the wording for those statements in common in the surveys for Group A and Group B subjects were identical. Two randomly-chosen dialogues are shown in Figures 7.2 and 7.4. The KB before the interactions is

shown in Figure 7.1. The knowledge base after the first dialogue is in Figure 7.3; and the KB after the second dialogue is in Figure 7.5. One hundred (100) human test subjects participated in group B testing.

Table 7.7: Group B Questionnaire Survey

	Strongly disagree				**Strongly agree**
1. I found the conversation with the system to be natural as if conversing with a human.	1	2	3	4	5
2. I found that the system correctly added new information to its knowledge base when appropriate.	1	2	3	4	5
3. I found that the system correctly neglected to add information that is already in the knowledge base when appropriate.	1	2	3	4	5
4. The system correctly identified the most similar information in the knowledge base and requested my approval to replace it before doing so.	1	2	3	4	5
5. The system was able to correctly distinguish between chatting statements and content-related statements	1	2	3	4	5
6. I found the system was able to answer questions.	1	2	3	4	5
7. I found that the system correctly transitions back and forth between chatting, question-answering system and processing content-related information.	1	2	3	4	5
8. I found that the system was able to correctly separate the chatting information from the content-related information.	1	2	3	4	5
9. Please indicate any final thoughts and comments about the system.					

cars have 4 wheels.
about 165,000 cars are produced every day.
it would take less than 6 months to get to the Moon by car at 60mph.
The first car accident occurred in 1891, in Ohio.
The average car has 30,000 parts.
There are 1 billion cars currently in use on earth.
92% of all new sold cars in Brazil.
Volkswagen owns Bentley, Bugatti, Lamborghini, Audi, Ducati and Porsche.
The average American spends about 38 hours a year stuck in traffic.
The first car accident occurred in 1891, in Ohio.
There are more cars than people in Los Angeles.
The inventor of the cruise control was blind.
Ferrari only builds 14 cars each day.
Henry Ford made a car out of soybeans.
Most new cars fake engine noise through speakers.
95% of a car's lifetime is spent parked.
Drinking a cup of caffeinated coffee significantly improves blood flow.
New Yorkers drink almost 7 times more coffee than other cities in the US.
54% of the Americans drink coffee every day.
Brazil has been the largest producer of coffee for the last 150 years.
Chicken contains 266% more fat than it did 40 years ago.
Honey is the only food that will never rot.
Cheese is the most stolen food in the world.
Scientists can turn peanut butter into diamonds.
49% of US Adults eat one sandwich a day.
India has the world's lowest meat consumption per person.
Bananas are slightly radioactive.
Bananas don't grow on trees.
Bananas are produced by herbaceous plants.
Bananas are the most popular fruits in the US.
Humans share 50% of their DNA with bananas.

Figure 7.1: The Initial Knowledge Base for Both Conversations/Dialogues

System: How can I help you today?
User: How many people live in New York?
System: I am checking for the answer
hmmm, I cannot guess
User: New Yorkers drink less coffee than other cities
System: I am checking my KB....
I am about to exchange info; do you still want to take this action? and change the information of New Yorkers drink almost 7 times more coffee than other cities in the US.
with New Yorkers drink less coffee than other cities?
Please enter "YES" or "NO"
User: no
System: ok information is stayed as it was
User: Pokemon go encourages exercise
System: I am checking my KB....
I added this info
User: Hi
System: what is wrong
User: Nothing. Just wanted to talk to you
System: what do you work on
User: Well I have a calculus 3 quiz later today
System: I am checking my KB
OK, info is now added
User: Do you know how to do calculus?
System: hal is the famous artificial intelligence from "2001".
User: Did you enjoy 2001?
System: I am checking for the answer
I don't have similar information
User: I didn't think 2001 was that good
System: Is today a good day?
User: Yes it seems like a good day
Might be a busy day too
System: what do you mean?
User: There is a lot to do today
System: I am checking my KB
OK, info is now added
User: Bananas sometimes grow on trees
System: I am checking my KB
I am about to exchange info; do you still want to take this action? and change the information of Bananas don't grow on trees.
with Bananas sometimes grow on trees?
Please enter "YES" or "NO"
User: yes
System: I am exchanging information here
User: are bananas slightly radioactive?
System: I am checking for the answer
hmm, I know that Bananas are slightly radioactive.
User: exit
System: Ok, bye

Figure 7.2: First Sample Conversation log (randomly-chosen from the 30 logs generated by Group A subjects)

cars have 4 wheels.
about 165,000 cars are produced every day.
it would take less than 6 months to get to the Moon by car at 60mph.
The first car accident occurred in 1891, in Ohio.
The average car has 30,000 parts.
There are 1 billion cars currently in use on earth.
92% of all new sold cars in Brazil.
Volkswagen owns Bentley, Bugatti, Lamborghini, Audi, Ducati and Porsche.
The average American spends about 38 hours a year stuck in traffic.
The first car accident occurred in 1891, in Ohio.
There are more cars than people in Los Angeles.
The inventor of the cruise control was blind.
Ferrari only builds 14 cars each day.
Henry Ford made a car out of soybeans.
Most new cars fake engine noise through speakers.
95% of a car's lifetime is spent parked.
Drinking a cup of caffeinated coffee significantly improves blood flow.
New Yorkers drink almost 7 times more coffee than other cities in the US.
54% of the Americans drink coffee every day.
Brazil has been the largest producer of coffee for the last 150 years.
Chicken contains 266% more fat than it did 40 years ago.
Honey is the only food that will never rot.
Cheese is the most stolen food in the world.
Scientists can turn peanut butter into diamonds.
49% of US Adults eat one sandwich a day.
India has the world's lowest meat consumption per person.
Bananas are slightly radioactive.
Bananas sometimes grow on trees.
Bananas are produced by herbaceous plants.
Bananas are the most popular fruits in the US.
Humans share 50% of their DNA with bananas.
Pokemon go encourages exercise.
quiz later today.
a lot to do today.

Figure 7.3: The Knowledge Base After the First Conversation

System: Please enter something related to my knowledge
User: how many wheels does a car have?
System: I am checking for the answer
I have related information to what you are mentioning cars have 4 wheels.
User: how are you?
System: I am doing well.
User: what did you do today?
System: I do not have any emotions.
User: are you a boy or girl?
System: I am checking for the answer
I don't know
User: an average car has how many parts?
System: I am checking for the answer ....
I don't have similar information
User: queen is an old rock band
System: I am checking my KB....
I added this info
User: bananas grow on trees
System: I am checking my KB....
I have the same info
User: bananas dont grow on trees
System: I am checking my KB....
I have the same info
User: bananas grow on acorns
System: I am checking my KB....
I am about to exchange info; do you still want to take this action? and change the information of Bananas don't grow on trees.
with bananas grow on acorns?
Please enter "YES" or "NO"
User: NO
System: ok information is stayed as it was
User: exit
System: ok, bye for today

Figure 7.4: Second Sample Conversation log (randomly-chosen from the 30 logs generated by Group A subjects)

cars have 4 wheels.
about 165,000 cars are produced every day.
it would take less than 6 months to get to the Moon by car at 60mph.
The first car accident occurred in 1891, in Ohio.
The average car has 30,000 parts.
There are 1 billion cars currently in use on earth.
92% of all new sold cars in Brazil.
Volkswagen owns Bentley, Bugatti, Lamborghini, Audi, Ducati and Porsche.
The average American spends about 38 hours a year stuck in traffic.
The first car accident occurred in 1891, in Ohio.
There are more cars than people in Los Angeles.
The inventor of the cruise control was blind.
Ferrari only builds 14 cars each day.
Henry Ford made a car out of soybeans.
Most new cars fake engine noise through speakers.
95% of a car's lifetime is spent parked.
Drinking a cup of caffeinated coffee significantly improves blood flow.
New Yorkers drink almost 7 times more coffee than other cities in the US.
54% of the Americans drink coffee every day.
Brazil has been the largest producer of coffee for the last 150 years.
Chicken contains 266% more fat than it did 40 years ago.
Honey is the only food that will never rot.
Cheese is the most stolen food in the world.
Scientists can turn peanut butter into diamonds.
49% of US Adults eat one sandwich a day.
India has the world's lowest meat consumption per person.
Bananas are slightly radioactive.
Bananas don't grow on trees.
Bananas are produced by herbaceous plants.
Bananas are the most popular fruits in the US.
Humans share 50% of their DNA with bananas.
queen is an old rock band.

Figure 7.5: The Knowledge Base After the Second Conversation

7.2.4 Results of Group B

To report the results, we compared the test subjects' responses collected for this group to the results obtained from the corresponding statement from Group A. Similar to Group A, the results were measured by calculating the mean and the standard deviation among the 100 participants. These results are shown in Table 7.8.

Table 7.8: Means and Standard Deviations for Group B (100 participant) Survey

No.	Statement	Mean	STD
1	The conversation is natural	2.72	1.44
2	LCC correctly added new information	3.78	1.08
3	LCC correctly neglected to add information that is already in the knowledge base	3.93	1.18
4	LCC correctly identified the most similar information in the knowledge base	4.16	1.00
5	LCC correctly distinguish between chatting statements and content-related statements	3.55	1.19
6	LCC was able to answer questions	3.69	1.14
7	LCC correctly transitions back and forth between the different engines	3.61	1.17
8	LCC correctly separate the chatting information from the content-related information	3.56	1.19

1. "I found the conversation with the system to be natural as if conversing with a human."

   The mean (2.72) was slightly below 3.0 (see Table 7.8) because 53 of the test subjects voted for 1 or 2 points on the Likert scale. The standard deviation of 1.44 was the highest among the questions/statements in the Group B survey. This means that the collected answers were spread out among the range. By comparing the results obtained here and the results for Group A in Table 7.6, we can see that subjects in Group A who interacted with the system reported higher values than those who did not. Additionally, the standard deviation was lower than those of Group B. We believe that this is because the fact that the second group did not have a chance to get a full sense of the system performance and capability, only getting access to two conversations. Yet, It was inpractical to include additional logs in the second testing as this would have required the subjects to take an unreasonable amount of time to complete the test.

2. “I found that the system correctly added new information to its knowledge base when appropriate.”

   By comparing the results of Table 7.8 with the results for Statement 4 from Table 7.6, we can see that the mean for Group A (4.2) was higher than that for Group B (3.78) and with a lower standard deviation of (0.93) for Group A comparing to (1.08) for Group B.

3. “I found that the system correctly neglected to add information that is already in the knowledge base when appropriate.”

   The test subjects were asked to evaluate whether LCC was successful in confirming existing information. Sixty-eight participants reported 4+ Likert points. Comparing the results with the same statement for Group A in table 7.6, we can notice that Group A mean (4.47) was higher, and that the standard deviation (0.97) was lower than that reported for Group B (3.93 and 1.18 respectively). We also believe that this is a result of the inability of Group B subjects to actually interact with the system, which would have given them a better sense for its performance. Yet, the results here are still acceptable, as the mean is close to 4.0 on the Likert scale.

4. “The system correctly identified the most similar information in the knowledge base and requested my approval to replace it before doing so.”

   Seventy-five (75) of the participants reported a 4+ on the Likert scale, which reflects general positive agreement with this statement. Comparing the results of Group B (mean of 4.16 and STD of 1.00) with those of Group A (mean of 4.3 and STD of 1.02) in Table 7.6, we can see that the results were close, indicating that most users approved of the system performance in replacing information.

5. “The system was able to correctly distinguish between chatting statements and content-related statements.”

Fifty-two (52) of the test subjects gave this statement 4+ on the Likert scale; therefore, the mean of this statement in Group B (3.55) (see Table 7.8) is slightly higher than the mean obtained from Group A (3.33) in Table 7.6. The standard deviation is lower in Group B (1.19) comparing to the standard deviation of Group A (1.24) which reflects that the answers were closer to the mean value. This may be a result of having limited access to Group A logs that results in not having a very sensitive decision about the system performance.

6. "I found the system was able to answer questions."

   This statement was used to measure the accuracy and ability of the system to answer questions in general. The mean was above three (3.69) (see Table 7.8), where 53 of the users voted 4+ points on the Likert scale and 29 users voted for 3 points. In both conversations logs, the LCC system had some issues handling the subjects' inputs as their inputs/questions went beyond a yes/no type of answer. This limits LCC's ability to match information between the provided input and what is in the knowledge base. For example, "an average car has how many parts?". Another example of where LCC failed to respond correctly is when it produced random answers when the test subject asked a chatting question such as in "Do you know how to do calculus?". LCC did not have anything similar to this question; therefore, it responded with a random answer through its chatbot, which was "hal is the famous artificial intelligence from 2001". We believe that such responses affected the suhjects in Group B opinions about the system's performance in general.

7. "I found that the system correctly transitions back and forth between chatting, question-answering system and processing content-related information."

   Fifty-seven (57) of the test subjects voted 4+ points on the Likert scale. The mean value for Group B (3.61) (see Table 7.8) was slightly lower than the one for Group A (3.83) (shown in Table 7.6), with a slightly higher standard deviation (1.17) comparing to the one for Group A (1.12).

8. “I found that the system was able to correctly separate the chatting information from the content-related information.”

   We believe that this statement was the hardest for the Group B test subjects to understand, as they probably interpreted the statement as Statement 5 (see Table 7.8). This is similar to what happened with Group A (statement 10 in Table 7.6). The mean and the standard deviation for Group B are (3.56 and 1.19) (see Table 7.8). Comparing the results with the results in Table 7.6 (3.59 and 0.95), we can see that the mean and the standard deviation were very close to the values reported for Group A.

7.2.4.1 Observation

There is one observation worth mentioning for Group B testing. In this group, the surveys for the first 51 participants reported higher accuracy for all the questions compared to the accuracy reported for the entire Group B. One possible explanation for this difference is that the first 51 human subjects were directly requested to participate by the Computer Science Department at the University of Central Florida via an email that asked for volunteer test subjects. The rest of the participants (49) were students who were asked by their professor in a particular course to whom we had reached out to allow her students to participate in the survey as part of their course requirements. The results dropped noticeably after those students joined the survey. We think that this was because the students may have felt obligated to do the survey and therefore did not pay close attention to reading the logs or answering the survey questions. The mean and standard deviation values for the first 51 participants are reported in Table 7.9.

Table 7.10 reports the p-values of the first part of Group B (the 51 participants) and the whole Group B (the 100 participants). Although the means reported in Table 7.9 show slightly better values for the first part of Group B, The p-values reported in Table 7.10 do not show a significant

difference between the means for the majority of the questions (except for the first and last one), as all the p-values are above the significant threshold of 0.05.

A more informative computation would be to compare the first part of Group B (51 participants) with the second part of Group B (49 participants); however, this insight was noticed after all the 100 entries had been received and intermixed with the first 51. Given that the survey is anonymous and were not timed tagged, a separation from the second 49 participants of the 100 became impossible.

Table 7.9: Means and Standard Deviations for the first 51 Human Subjects from Group B

No.	Statement	Mean	STD
1	The conversation is natural	3.32	1.51
2	LCC correctly added new information	4.06	1.16
3	LCC correctly neglected to add information that is already in the knowledge base	4.26	1.07
4	LCC correctly identified the most similar information in the knowledge base	4.28	0.90
5	LCC correctly distinguish between chatting statements and content-related statements	3.90	1.22
6	LCC was able to answer questions	3.86	1.12
7	LCC correctly transitions back and forth between the different engines	3.96	1.09
8	LCC correctly separate the chatting information from the content-related information	3.96	1.10

Table 7.10: P-value Between the first 51 participants of Group B and Group B total participants (100)

No.	Statement	Mean (51)	Mean (100)	P-value
1	The conversation is natural	3.32	2.72	0.02
2	LCC correctly added new information	4.06	3.78	0.14
3	LCC correctly neglected to add information that is already in the knowledge base	4.26	3.93	0.09
4	LCC correctly identified the most similar information in the knowledge base	4.28	4.16	0.47
5	LCC correctly distinguish between chatting statements and content-related statements	3.90	3.55	0.09
6	LCC was able to answer questions	3.86	3.69	0.38
7	LCC correctly transitions back and forth between the different engines	3.96	3.61	0.07
8	LCC correctly separate the chatting information from the content-related information	3.96	3.56	0.04

7.2.5 Statistical Significance Correlation

To determine the statistically significance of the correlation between the data collected for each question in common to Group A and Group B, we calculated the p-values as shown in Figure

7.11. The significance level used here to determine whether the comparisons of the means of each common question Group A and Group B statistically significant was 0.05 with a corresponding confidence level is 95%.

The p-values for the means of all the questions in common (except for questions 3 and 6) are higher than the 0.05 threshold of significance. This indicates that we cannot conclude that a significant difference between the two groups exists for those questions. However, there exists a significant difference between the two groups for questions 3 and 6.

Table 7.11: P-value Between Group A (30 Participants) and Group B (100 Participant)

No.	Statement	Mean (Group A)	Mean (Group B)	P-value
1	The conversation is natural	3.00	2.72	0.33
2	LCC correctly added new information	4.20	3.78	0.06
3	LCC correctly neglected to add information that is already in the knowledge base	4.47	3.93	0.02
4	LCC correctly identified the most similar information in the knowledge base	4.30	4.16	0.50
5	LCC correctly distinguish between chatting statements and content-related statements	3.33	3.55	0.38
6	LCC was able to answer questions	4.33	3.69	0.01
7	LCC correctly transitions back and forth between the different engines	3.83	3.61	0.36
8	LCC correctly separate the chatting information from the content-related information	3.59	3.56	0.89

7.2.6 Qualitative Results- Thoughts and Feedback of the Human Test Subjects

We collected the test subjects' feedback regarding their interaction with the system for Group A and their opinions after reading the logs for Group B. For Group A, only 16 participants wrote feedback. The feedbacks can be found in Appendix F. Many of the test subjects of both groups reported that they thought the system was useful and they were interested in using it, such as in "Great program! Super convenient" and "I thought the system was pretty intelligent. It didn't feel very human-like to me, but I could tell that efforts were there to make it more human". Some suggested improving aspects of the system, such as the chatbot responses need to be somehow related to the knowledge base, and improve the system's ability to answer questions. An example

is “There is room for improvement, but it functions well”. The feedback was generally positive regarding the system performance in the learning process. Test subjects thought that this was the most robust part of LCC “the system was organized and robust! it did not permit me to change information as untrustworthy”. We agree with the testers, as the focus of this dissertation research was on the learning process, while other parts were only auxiliary, such as the question answering system and the chatbot.

For Group B, among the 100 test subjects, 60 of them wrote feedback. Some feedback from the second group contained questions such as, “Is this something like Alexia, Siri or google home?” and our answer is clearly no. The idea behind the LCC system is to enable the human to teach the computer agent new information or updating old information. LCC does not work as a personal assistant as Alexa and Siri but rather, it can be used by those systems to allow the users to teach the agent and improve its existing knowledge.

7.2.7 User Testing Summary

This set of tests was designed to allow other humans besides the developer to interact with LCC and report their opinions. The testing was performed using two groups, the first group (Group A) interacted directly with the LCC system, while the second group (Group B) reviewed logs of two dialogues from Group A selected randomly. The overall mean score obtained from Group A were higher than those of the second group, with significantly lower standard deviation. This may have been a result of the limited access to the data provided to Group B participants. One way to improve the user understanding is to provide more examples and dialogue logs; however, this can be time-consuming for the test subjects who have to read all the logs.

The obtained results for both groups reflect that the subjects were not strongly satisfied with the naturalness of the dialogue. We note that the focus of this research is learning content related

information and not creating a natural dialogue with the human. However, improving the LCC's ability to produce more natural responses is a consideration for future research. On the other hand, results related to the learning process reflect a strong agreement by the test subjects that the system performed well. Yet, the system needs significant improvement to the chatbot and the question answering system to have more robust responses.

7.3 Complexity and Scalability

To measure the runtime complexity of the LCC algorithm, we first provide a theoretical analysis of the runtime over the learning algorithm and then provide an empirical analysis by measuring the runtime over various knowledge base sizes, as this is the main factor in increasing LCC's runtime.

7.3.1 Theoretical Runtime Analysis

The runtime of the LCC algorithm depends on the learning process, as it is the longest process in the system that involves many computations. Therefore, to analyze the runtime of LCC, we need to analyze the runtime of each of the steps of the algorithm shown in Algorithm 4 (required here for the convenience of the reader).

- Part 1: Create mySet (ignore set): Lines 2 – 12: deals with creating mySet that contains the nodes that LCC excludes from the search for the most related node to the user input. Those nodes are excluded when their similarity using the fuzzy string matching is below 40. At first glance on this part of the code, this operation takes $O(n^2)$, where n is the number of nodes in the knowledge base because this part has two nested for loops. However, upon studying this operation further by considering the different types of networks that the LCC

```
Learning(input, graph, value, archive)
dict = dictionary(len(graph));
create mySet();
for g in range (len(graph)) do
    if g not in mySet then
        if fuzz.token-sort-ratio(question,dict[g]) ≤ 40 then
            for nb in graph[g] do
                if fuzz.token-sort-ratio(question,dict[nb]) ≤ 40 then
                    mySet.add(nb)
            end
end
for g in range (len(graph)) do
    if g not in mySet then
        alignments = align(dict[g], question);
        if (len(alignments[0]) >= 2) then
            if symmetric-sentence-similarity(question, dict[g]) >= 0.5 then
                related[g] = symmetric-sentence-similarity(question, dict[g]),
                fuzz.token-sort-ratio(question,dict[g]),len(alignments[0]) ;
end
if len(related.keys()) == 0 then
    add input to textFile;
    graph =create-net();
    update dict;
    value[g] = 0.5;
else
    maxValue = max(related, key = related.get);
    if related[maxValue][1] >= 80 and related[maxValue][0] >= 0.9) and
    (related[maxValue][2] >= 4 then
        value[maxValue] = value[maxValue] + 1.0;
        print (response of confirmation) ;
    else if related[maxValue][1] >= 50 and related[maxValue][0] >= 0.6 then
        Exchange information request ;
        if "yes" then
            change info;
            graph =create-net();
            value[maxValue] = value[maxValue] - 1.0;
            print (confirm the changing process)
        else
            print (information did not change)
        end
    end
    else
        add input to textFile;
        graph =create-net();
        update dict;
        value[g] = 0.5;
    end
end
```

Algorithm 4: LCC Learning Algorithm

knowledge base can contain with respect to their relation to the user input, we have the following additional insights:

1) A dense network is when each node is connected with edges to every other node in the network. In this case, if the user input is not related to the current node (the calculated fuzzy ration is < 40), LCC will start comparing the calculated fuzzy ratio between the current node's neighbors and the user input again with the value 40 using the inner for loop.

If all the neighbors' fuzzy ratio with the user input is below 40, this will exclude all the nodes from continuing, therefore, it will speed up the process and consider the user input as new information with no further calculation. Therefore, the worst-case runtime will be $O(n^2)$ but the outer loop will not do more than a quick loop through the nodes that are already excluded. Yet, in the case of the user input has a fuzzy ratio of > 40 that will make the inner loop not triggered at all and therefore the runtime will be *O(n)*.

2) A sparse network is when each node is isolated. This case considers the current node has no neighbors during evaluating (line 4), therefore, the inner loop will not be executed. However, when the nodes are clustered into groups that will make the run time increased but no more than a fraction of *n*. Therefore, in practice, the runtime of this part of code never reaches $O(n^2)$.

- Part 2: Calculate the related scores: Calculating the related score of the remaining nodes that have a relation with the input takes *O(n)*. This part of the code shown in lines 13 – 19. Calculating the similarity scores will not take more than $O(w^2)$, where *w* is the number of words in the user input in the current node in the KB. In practice, since the length of the input cannot exceed 50 words at most, this number can be considered a constant compared to the knowledge base size. Therefore, we can say that the worst runtime would take *O(n)*.

- Part 3: Determine the correct action after extracting the dominant node: Line 26 finds the

node that has the highest similarity scores. Lines 27 – 46 evaluates the scores and sees if the information in the KB needs to be confirmed, modified, or the user information is new information that needs to be added. The longest operation of this part of the code is constructing the graph which takes *O(n)*.

Therefore, the overall runtime for this algorithm is linear on average. This is because the first part of the code is used to trim the network and make the subsequent calculations faster.

7.3.2 Empirical Runtime Analysis

Scalability is an important factor that reflects how well a system can perform when the size of the data increases. Therefore, to measure the scalability, we measure how long it takes LCC to respond to the user input. Hence, producing a response by the system is directly proportional to the size of the knowledge base because the larger the knowledge base is, the more search that is needed to relate the user input to the most relevant information in the knowledge base.

Therefore, the scalability is measured using different sizes of the knowledge base, ranging from 30 sentences to 960 sentences. Those sentences are closely related and have many connections in the network. Appendix G include the knowledge base of size 960. The other knowledge bases were subsets of this knowledge base.

We measured the time it took for each of the operations to produce a response: adding, exchanging, confirming, question-answering and chatting. The results are averaged over five runs. For exchanging, confirming and Q/A operations, we chose the input to be related to information that was located in different parts of the knowledge base. Figure 7.6 shows the means of the available operations (adding, exchanging, confirming, question-answering and chatting) averaged over 5 runs. The x-axis represents the number of sentences in the knowledge base, while the y-axis

represents the time it took the LCC system to respond to the user input in seconds.

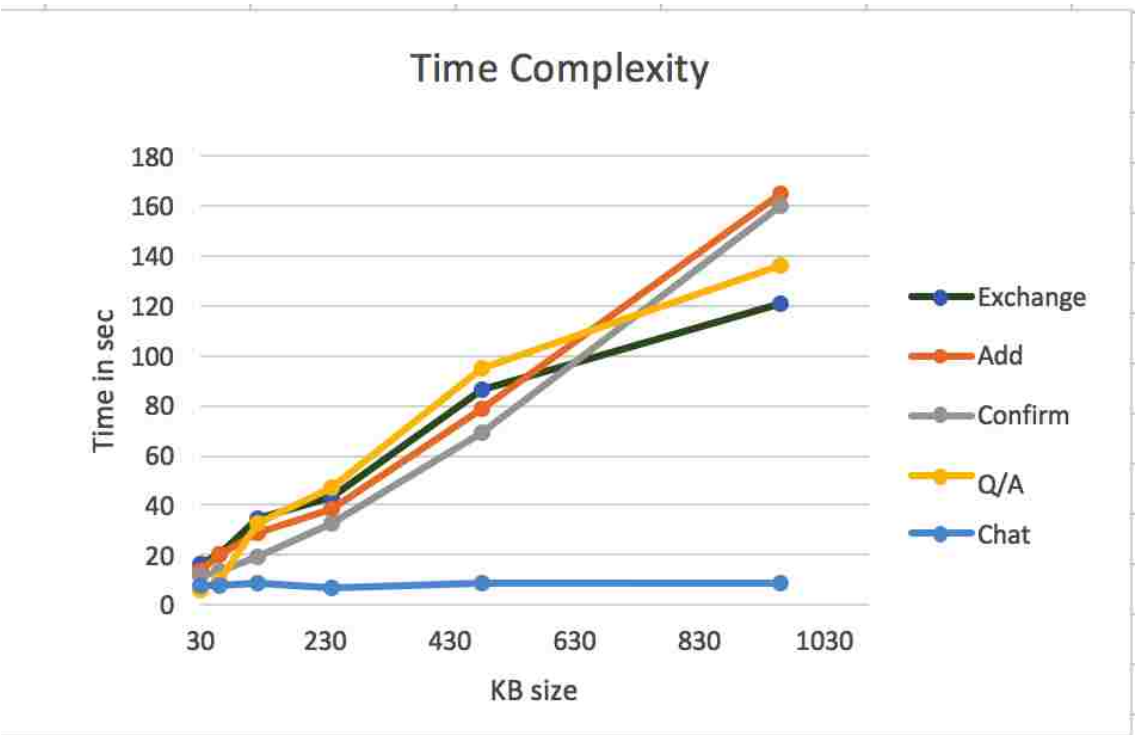


Figure 7.6: Scalablity of the LCC system

The system spent almost the same amount of time responding to the chatting statements across all the sizes of the knowledge base. This is expected, as the chatting engine has no relation to the size of the knowledge base. Therefore, the time complexity to respond to a chatting statement is a constant O(1) with respect to KB size.

LCC takes longer to respond to Q/A request, confirm the request, exchange request and add request, as the knowledge base increases in size as shown in Figure 7.6. However, this increase is acceptable considering the increasing number of sentences in the knowledge base that LCC uses to make a comparison with the user input. We considered using different positions in the knowledge base (beginning, middle, and end) to perform those operations on them. The time complexity of responding to the user input in any of those labels was found to be linear.

We reported the standard deviation for each of the operations to analyze the runtime further. For the chatting responses, the standard deviation and the mean across the different sizes of the knowledge base are reported in Table 7.12. The obtained results show almost a constant standard deviation for all the chatting responses. The runtime for the chatting responses was not affected by the knowledge base size, as it uses a different knowledge base to produce the answers (discussed in Section 6.3).

However, this is not the case with the other operations as shown in Tables 7.13, 7.14, 7.15, and 7.16. In all these tables, the standard deviation increases as the knowledge base size increases. This reflects a scattered runtime around the mean values. Yet, this is a result of the dependency on the number of nodes in the knowledge base to which the user input is related and the density of the network.

One uncommon behavior can be seen in Table 7.14. The standard deviation of the last value when the knowledge base size is 960 is smaller than the standard deviation values reported for smaller sizes of the knowledge base, as the time that took LCC to add new knowledge was within the same range (160 - 167) across the five runs, which was not the case with the previous adding operations for smaller sizes. We think that part of this issue came from the type of inputs that needed to be added, which required an approximately similar number of comparison with information in the knowledge base across the five runs.

7.4 Summary

In summary, to test and evaluate the performance of LCC, a series of testing procedures were used. We first performed functionality testing where the developer tested the accuracy of LCC using the different levels of difficulties in the user input. The results were expected, as we knew that there

would be a decrease in the accuracy as the difficulty level increased.

Another experiment involved testing the classification process used in LCC and compared the results to the results obtained from the individual performance of the classifiers used in the majority vote (Naïve Bayes classifier, Decision tree, and Max Entropy). The results show that LCC classification process is not always the best as it depends on different factors including, the user input difficulty level, the number of classifiers in the majority vote agreed on the label and finally, on the errors associated to the part of speech tagging process.

For the humans testing, we asked two groups of participants to evaluate the performance of LCC. Both groups indicated general satisfaction will LCC. However, the first group results reflect that Group A were more satisfied with LCC performance than were Group B. We expected that evaluating LCC with limited access to information would not be easy on the second group, as they did not have the opportunity to directly interact with the system.

The third type of testing is used to measure the complexity and scalability of the LCC system. The practical analysis of the runtime shows the runtime is linear on average.

The overall obtained results were acceptable. However, LCC needs several improvements that will be addressed in our future research to make it function better.

Table 7.12: The Mean and the Standard Deviation of the Chatting Operation

KB size (Sentences)	Mean (Sec.)	STD
30	7.84	1.26
60	8.02	3.56
120	8.60	1.96
240	6.77	1.03
480	8.95	4.66
960	9.09	1.70

Table 7.13: The Mean and the Standard Deviation of the Confirming Operation

KB size (Sentences)	Mean (Sec.)	STD
30	11.44	12.31
60	13.49	8.64
120	18.92	9.87
240	32.57	20.40
480	79.02	55.78
960	164.75	45.67

Table 7.14: Standard deviation of the Adding operation

KB size (Sentences)	Mean (Sec.)	STD
30	13.72	10.03
60	20.56	5.07
120	29.05	10.13
240	38.86	15.43
480	79.02	44.39
960	164.75	2.26

Table 7.15: Standard deviation of the Exchanging operation

KB size (Sentences)	Mean (Sec.)	STD
30	16.61	10.02
60	20.33	6.98
120	34.16	7.90
240	42.83	15.47
480	85.95	22.35
960	120.50	43.61

Table 7.16: Standard deviation of the Q/A operation

KB size (Sentences)	Mean (Sec.)	STD
30	5.59	1.02
60	9.90	2.63
120	33.10	10.28
240	46.62	22.23
480	94.64	43.35
960	136.23	31.56

CHAPTER 8: SUMMARY, CONCLUSIONS AND FUTURE RESEARCH

In this chapter, we present an overall summary of the completed research of learning from a casual conversation. Later, we discuss the overall conclusions obtained from the research and the test results. Lastly, we discuss our future work to improve this research.

8.1 Overall Summary

In this dissertation, we presented Learning from a Casual Conversation (LCC) as a new type of behavioral learning that involves direct human interaction with a learning agent to allow it to learn new knowledge. This research explored the idea of how a computer agent learns new knowledge from natural conversation, similar to how humans learn from each other through extended dialogues. This research can have an impact on improving computer agent knowledge as it allows the computer agent to learn new information using natural language without involving a programmer to update the computer agent's knowledge base.

Existing behavioral learning mechanisms focus more on learning to accomplish specific tasks that require exact commands from the user who usually needs to be an expert in the area. Our research focuses on teaching the agent how to learn declarative knowledge/information using simple natural language to communicate the new information as part of a natural conversation.

LCC operates by combining multiple components to generate the overall architecture of the system. The most important and fundamental components are classification, learning, and knowledge base units. The classification unit is responsible for labeling the user input as either a chatting statement or content-related information. The learning unit determines what is to be learned from the user input. The knowledge base unit holds the knowledge of the agent.

Other components were added to make the conversation sound natural and to improve the process of learning, such as creating a chatbot to respond to a user's chatting statements and a question-answering unit to answer a user's content-related questions.

The core of this research is the learning unit, where we employed three different similarity measures to help LCC decide what information in the knowledge base is the closest to the user's input. The learning process has the ability to confirm existing information, revise information to match the user input, and add information when no similar information is found in the knowledge base.

The LCC system was tested using various methods that included testing the functionality of the system and how it performs while increasing the difficulty level of the simulated user input. LCC was also tested by human test subjects who were divided into two groups: Group A subjects were asked to physically interact with the system and then fill out a survey regarding their experience; Group B subjects were asked to evaluate the system by reviewing dialogue logs gathered from Group A subjects, and then complete a survey similar to the survey for Group A (except for few modifications). The system generally performed successfully. It worked as expected; however, it sometimes failed to classify the user input correctly, which resulted in directing it to the wrong components. These failures happened mostly in the classification process, when the user input was misclassified as chatting statements when they were not and vice versa. Improving this process is left for future work. The overall results were acceptable, which indicates that this approach is a significant step forward toward extending a computer agent's capability to learn through a novel means.

There are many possible applications for LCC, as it can be used for computer agent to expand its learning process. For example, it can be used in social companions where the human user can share new information with the robot using simple natural language. Additionally, we can see it being applied to autonomous agents, such as a robot driving a car. LCC can also be applied to help

teach human-robot assistants such as Alexa and Siri how to update their knowledge base without requiring an update from their designers.

8.2 Conclusions

This dissertation brings insight to a new area of research in behavioral learning. The hypothesis of this dissertation was that a learning mechanism could be built that allows a computer agent to learn through casual conversation with humans. This hypothesis was validated by creating the LCC system and testing it.

During the process of developing the LCC system, one interesting problem that appeared was how to handle the user's input when he/she uses unrestricted grammar. We managed to resolve this problem by deploying different components used to classify the user input and direct it to the correct unit.

Many interesting and novel approaches were used in the design of LCC, such as using two levels in the classification process to improve the overall decision of choosing a correct label for the user input (chatting or content-related statement). However, this approach did not always lead to optimal results, as it mostly depended on the user input and the training data for the classifiers.

Another interesting approach was using two ways to represent the knowledge base: one was to create an easy way to extract the learned information using only text; the other way was to use a semantic network representation on the sentence level of the text file used to improve the process of finding the most related information to the user input. We think that this approach was very successful as it improved the runtime of the algorithm when finding the best match to the user input. The computational cost could be much higher if no such approach had been used, as without the semantic network and ability to exclude the nodes that have low similarity with the user input,

LCC would have to compare the user input with every single bit of information in the knowledge base. The running time will be $O(n^2)$ in worst case, and $O(n)$ in average case, where *n* is the number of nodes. Therefore, we can consider this approach to be successful.

Another interesting aspect of this research was setting up the thresholds in the learning process to compare the obtained similarity measure between the user input and the information in the knowledge base. We performed a series of trial and error experiments to find the best thresholds that could yield an acceptable performance in most of the cases of user input.

In general, we think the overall architecture of the LCC system served well the objective of learning from casual conversation.

8.3 Future Work

Learning form casual conversation has several extension paths for the future. One possible improvement is to use voice instead of text to communicate with the system. This will allow for an easier and a more efficient communication with the system; Moreover, it can help users with disabilities who can not use a keyboard to communicate with LCC.

Trust evaluation of the users and of the source of information would be an important enhancement to LCC, mainly for when the system could potentially be used by untrustworthy users. One possible way to do it is to evaluate the user trust level based on her previous interactions with the system and by searching for the information he/she is providing online. However, this will be applicable to information that is only considered as known facts.

Feedback to improve the learning process would significantly enhance the learning algorithm of LCC. We established this approach by assigning values that indicate whether the information is

new, exchanged or confirmed. Our plan is to extend this approach to allow the user to assign feedback in form of rewards and penalties that reflect his/her satisfaction with LCC performance when taking the correct decision regarding the current provided user input. Such a process will make the system continuously improve its performance, which would improve its future decisions.

We are considering improving the classification process as the current approach makes a noticeable number of misclassifications. We suggest using online training for the classifiers because the current training was performed offline. Regular online training would allow the classifiers to learn the new patterns that the users use, which should improve the classification process in general.

In order to improve the overall flow of the dialogues and to make the conversation sound more natural, the chatbot and the question answering component need to be improved. One way to enhance the question-answering system is by improving the system's ability to answer questions that are not of a yes/no form. For example, if the user input is "how many wheels does a car have?", LCC should be able to answer "4". Furthermore, we plan to upgrade the chatbot to respond with more meaningful responses when it does not have similar information in its knowledge base. This will be done by upgrading the database of the chatbot to include more topics and create general answers that can fit with any statements when the chatbot does not recognize the user input. For example, answer with "I see" or "hmmm" when it does not know what the user is talking about.

We are also interested in applying LCC to different domains to determine whether LCC is easily combined with other systems, such as Alexa and social companions.

Improving the LCC runtime is also something we are considering in our future research. The current average time it takes LCC to respond to the user is linear with a high slope, which is still too long when using a large knowledge base. Our future work involves applying more optimizing algorithms to improve the slope, to make it smoother, or even to make the runtime logarithmic in the average case and linear for the worst case.

Other minor improvements include designing a better user interface that can be customized based on the application, such as suggesting the information that needs to be updated if it is out of date. For example, the number of times a medicine needs to be taken by the user.

APPENDIX A: LCC INITIAL KNOWLEDGE BASE

cars have 4 wheels.
about 165,000 cars are produced every day.
it would take less than 6 months to get to the Moon by car at 60mph.
The first car accident occurred in 1891, in Ohio.
The average car has 30,000 parts.
There are 1 billion cars currently in use on earth.
92% of all new sold cars in Brazil.
Volkswagen owns Bentley, Bugatti, Lamborghini, Audi, Ducati and Porsche.
The average American spends about 38 hours a year stuck in traffic.
The first car accident occurred in 1891, in Ohio.
There are more cars than people in Los Angeles.
The inventor of the cruise control was blind.
Ferrari only builds 14 cars each day.
Henry Ford made a car out of soybeans.
Most new cars fake engine noise through speakers.
95% of a car's lifetime is spent parked.
Drinking a cup of caffeinated coffee significantly improves blood flow.
New Yorkers drink almost 7 times more coffee than other cities in the US.
54% of the Americans drink coffee every day.
Brazil has been the largest producer of coffee for the last 150 years.
Chicken contains 266% more fat than it did 40 years ago.
Honey is the only food that will never rot.
Cheese is the most stolen food in the world.
Scientists can turn peanut butter into diamonds.
49% of US Adults eat one sandwich a day.
India has the world's lowest meat consumption per person.
Bananas are slightly radioactive.
Bananas don't grow on trees.
Bananas are produced by herbaceous plants.
Bananas are the most popular fruits in the US.
Humans share 50% of their DNA with bananas.

APPENDIX B: DIFFICULTY LEVEL TESTING DATA

Difficulty level	Sentences	Expected label
3	apple is a computer company	add
2	The average Americans have at least two cars per household	add
2	sandwich is a daily food for adults	add
1	Car parking is a problem in the US	add
1	life is good	add
1	humans can think	add
1	apple is a fruit	add
1	the moon is the nearest plant to the earth	add
3	There are more people than cars in Los Angeles	add
3	laughing is good for the heart	add
1	Cheese is made out of milk	add
1	humans are similar to cats by being sensitive	add
3	I am not sure if this thing will be finished	chatting
3	Can you describe what you have in your kb?	chatting
3	I am confused and worried	chatting
2	Do you like banana?	chatting
2	what do you think about life?	chatting
2	Do you believe in yourself?	chatting
2	let’s talk about you	chatting
2	Are you a robot?	chatting
2	decide on a name?	chatting
2	are you interested in eating a sandwich?	chatting
1	Can we talk about something else?	chatting
1	How is life?	chatting
1	Do you like coffee?	chatting
1	what is your name?	chatting

Difficulty level	Sentences	Expected label
1	could you tell me a story?	chatting
1	can you tell me a joke?	chatting
1	can we chat?	chatting
3	I ate a sandwich in my car	chatting
2	Do you eat apple?	chatting
3	I don't know if cats can think	cut chat, add info
3	I have seen a bird that can fly	cut chat, add info
3	Do you like artichoke? It is a vegetable	cut chat, add info
3	I had always heard that Toyota had the best quality cars .	cut chat, add info
3	Like everyone else, I have the problem with the transmission .	cut chat, add info
3	I have heard that warsaw is a town in poland.	cut chat, add info
3	I am not sure but do cars have 4 wheels	cut chat, exchange
3	I have heard that the earth has only one plant	cut chat, exchange
3	It is hard to say if a car can have more than 4 wheels	cut chat, exchange
3	The first car accident occurred in Ohio and Los Angeles have more cars than humans	Exchange request
3	herbaceous planets produces banana and herbs	Exchange request
3	Since 1868, Brazel was the largest producer of coffee	Exchange request
2	Volkswagen owns Bentley, Bugatti and Lamborghini	Exchange request
2	a car can have up to 4 wheels	Exchange request
2	chicken contains much more fat than in previous 40 years	Exchange request
2	Apples are the most popular fruit in the world	Exchange request
2	old cars used to have 6 wheels	Exchange request
2	92% of all new sold cars in Brazil and Maxico	Exchange request
1	a car can have multiple wheels	Exchange request
1	cars have 30000 parts	Exchange request
1	new cars fake engine noise	Exchange request

Difficulty level	Sentences	Expected label
1	a cup of caffeinated coffee can improve blood flow	Exchange request
1	Bread is the most stolen food in the world	Exchange request
3	Los Angeles is crowded with cars and peoples	Exchange request
3	Ferrari is a car company that builds fourteen car daily	Exchange request
2	There are two and a half billion cars currently in use on earth.	Exchange request
2	Soybeans is used to make a car by Ford	Exchange request
2	peanut butter can be turned into diamonds	Exchange request
1	Bananas grow on trees	Exchange request
3	Los Angeles reported having many cars comparing to people	same
3	40 years ago, the chickens were having less fat	same
2	herbaceous plants produced bananas	same
2	Bananas and humans share 50% of their DNA together	same
1	Drinking a cup of caffeinated coffee significantly improves blood flow.	same
1	Bananas are radioactive	same
1	Bananas share 50% of their DNA with humans	same
1	Brazil is the largest coffee producer for the last 150 years	same
2	I know that apple is a fruit	same
3	Do you think banana can be grow on trees?	QA
3	do you think a spoon is an utensil?	QA
2	Do you believe that Henry Ford made a car out of soybeans?	QA
1	can you find banana on trees?	QA
2	how long it takes to get to the moon by car at 60mph?	QA
2	Do human drive cars?	QA
1	Is Honey the only food that never rot?	QA

Difficulty level	Sentences	Expected label
1	Are banana the most popular fruits in US?	QA
2	Are Apple the most popular fruits in US?	QA
1	can humans think?	QA
2	Do New Yorkers drink 7 times more coffee than other cities in the US?	QA
3	Do you know that the first car accident occurred in 1891?	QA
2	There are one billion cars currently in use on earth	same
2	many Americans drink coffee every day.	same
1	cars have four wheels	same
1	The first car accident occurred in Ohio	same
3	trees do not grow bananas	same
3	speakers can be used to fake engine noice	same
3	Tress cannot have bananas	same
3	almost half of the Americans eat sandwich every day	same
2	I like to go to the moon by car	chatting

APPENDIX C: CLASSIFIERS TRAINING DATA

```
train = [
('hi how are you', 1),
('hello', 1),
('hey, whats up',1),
('I have to be home in ten minutes',1),
('how was your day?', 1),
('you doing ok?',1),
('hi have you been keeping busy?',1),
('hi',1),
('I am bored',1),
('nothing important', 1),
('tell me about life', 1),
('are you happy in your life?', 1),
('how it is going', 1),
('you are silly', 1),
('I did not see you for long time', 1),
('you are stupid', 1),
('nice one', 1),
('are you okay',1),
('she is sick', 1),
('hi there', 1),
('she is blue', 1),
('come here', 1),
('what is your name',1),
('what is your faviorate color?', 1)      ,
('my name is Mike', 1),
('An average ear of corn has an even number of rows, usually 16.',2),
('Chocolate was once used as currency.',2),
('Ketchup was used as a medicine',2),
('Apples float in water',2),
('Dynamite is made with peanuts',2),
('a salad is a food',2),
('canola oil is type of rapeseed oil', 2),
('one can of soda contains 10 teaspoons of sugar', 2),
('coffee beans are fruit pits', 2),
('ancient carrots were purple.', 2),
('one can of soda contains 10 teaspoons of sugar.', 2),
('almonds are member of peach family', 2),
('apples are fruits', 2),
('banana is a hurb',2),
('How dare you?', 1),
('banana has 105 calories', 2),
('alyssa is a girls name', 2),
('pluto is a planet', 2),
('progress is good', 2)  ,
('earth is round', 2),
('buddhists value compassion', 2 ),
('toronto is in canada', 2),
('warsaw is a town in poland', 2),
('I am coming with you', 1),
```

```
('life ends with the death', 2),
('life is hard', 2),
('omnipotent means all powerful', 2),
('lightning can kill you',2),
('you sit on a chair', 2),
('you need security in life', 2),
('you stop at red and go at green', 2),
('you can find germany in europe', 2),
('you can drink tea hot or cold', 2),
('you can cook food in a microwave',  2),
('you big',1),
('rain will cause people to become wet', 2),
('rain is wet', 2),
('bread sometimes means money', 2),
('bread is a food', 2),
('sheep have been cloned', 2),

('men have ever landed on the moon', 2)   ,
('you have been angry', 1),
('you have human ever cried', 1),
('you have seen yourself in a mirror', 1),

('people have been known to beat rugs', 2),

('all have humans got a brain', 2),
('computers have changed our culture', 2),
('computers have helped word processing', 2),

('aeroplanes are used for transportation', 2),

('5x5 equal 25', 2),
('york lies in usa',2),
('can we go to the movie?', 1),
('airplanes without motors are gliders', 2),

('human have two ears', 2),
('freindship is a two way street', 2),
('einstein was a genius', 2),
('argentina is a country', 2),
('a room has four walls', 2),
('I cannot understand', 1),
('a radio receives waves', 2),
('a mirror reflects light', 2),

('a turtle has a shell', 2),
('a flame generates heat', 2),
('a triangle has 3 sides', 2),
('a triangle has three sides', 2),
('a triangle has three corners', 2),
('a motorbike has two wheels', 2),
```

```
('a year has 52 weeks',2),
('a bike has two wheels', 2),
('a car produces pollution', 2)   ,
('a car helps you transport faster', 2),
('a car has four wheels', 2),

('a car has an engine', 2),

('a car moves faster than a human', 2),
('a day has 24 hours', 2),
('a day consists of 24 hours', 2),

('a boat floats in water', 2),

('a butterfly has wings', 2),
('a genius has a high iq', 2),
('a lamp generates light', 2),
('water boils at 100 degree celsius', 2),
('water covers most of the earth', 2),
('water froms the ocean contain salt', 2),
('water makes things wet', 2),
('milk haves calcium', 2),
('hitler was a dictator',2),
('I wish I can do it',1),
('life is hard',2),
('omnipotent means all powerful',2),
('I did not see him for long time',1),
('you sit on a chair',2),
('you need security in life',2),
('you stop at red and go at green',2),
('you can find germany in europe',2),
('you can drink tea hot or cold',2),
('you can cook food in a microwave',2),
('you big',2),
('rain will cause people to become wet',2),
('rain is wet',2),
('bread sometimes means money',2),
('bread is a food',2),
('sheep have been cloned',2),
('men have got penis',2),
('men have ever landed on the moon',2),
('you have been angry',2),
('you have human ever cried',2),
('you have seen yourself in a mirror',2),
('you have seen rain in a sunny day',2),
('we have visited the moon',2),
('people have been known to beat rugs',2),
('people have ever walked on the moon',2),
('humans have visited the moon',2),
('humans have been on the moon',2),
```

```
('humans have ever been on the moon',2),
('humans have ever exploded nuclear bombs',2),
('all have humans got a brain',2),
('computers have changed our culture',2),
('computers have helped word processing',2),
('aeroplanes are used for transportation',2),
('5x5 equal 25',2),
('york lies in usa',2),
('rose is red',2),
('airplanes without motors are gliders',2),
('human have two ears',2),
('freindship is a two way street',2),
('einstein was a genius',2),
('argentina is a country',2),
('a room has four walls',2),
('a chair has 4 legs',2),
('a radio receives waves',2),
('a mirror reflects light',2),
('are you kidding?',1),
('a turtle has a shell',2),
('a flame generates heat',2),
('a triangle has 3 sides',2),
('a triangle has three sides',2),
('a triangle has three corners',2),
('a motorbike has two wheels',2),
('a year has 52 weeks',2),
('it is fun',1),
('a car produces pollution',2),
('a car helps you transport faster',2),
('a car has four wheels',2),
('a car has wheels',2),
('a car has a engine',2),
('a car has 4 wheels',2),
('a car moves faster than a human',2),
('a day has 24 hours',2),
('a day consists of 24 hours',2),
('a baby bears have a mother',2),
('a boat floats in water',2),
('a boat floats on the water',2),
('a boat floats',2),
('a butterfly has wings',2),
('a genius has a high iq',2),
('a lamp generates light',2),
('a milkshake contains milk',2),
('a rainbow contains many colors',2),
('a rainbow contains many colours',2),
('a rainbow has many colours',2),
('a rainbow has colours',2),
('a rainbow consists of many colours',2),
('a square has four sides',2),
```

```
('a square has four corners',2),
('a square has 4 corners',2),
('a zebra has stripes',2),
('a airplane flys',2),
('a zoo cages animals',2),
('a zoo has animals in it',2),
('a hexagon has 6 sides',2),
('a teacher teachs people',2),
('a musician plays a instrument',2),
('a coin has two sides',2),
('a speaker produces sound',2),
('a paste has a thick consistency',2),
('a lantern provides light',2),
('a fan moves air and have blades',2),
('a woman loves roses as a gift',2),
('a house contains rooms',2),
('a house boats float',2),
('a house has a roof',2),
('a rancher raises cattle',2),
('a week consists of seven days',2),
('a mouse has babies',2),
('a fish needs oxygen',2),
('a man runs with his legs',2),
('a man runs with his legs',2),
('a man runs using his legs',2),
('a man stands on two feet',2),
('a man has a penis',2),
('a man sleeps when he is tired',2),
('a phone rings',2),
('a bicycle has wheels',2),
('a wolf has four legs',2),
('a rabbit needs to eat',2),
('a rabbit eats carrots',2),
('a ruler has followers',2),
('a tree has leaves',2),
('a tree has roots',2),
('a piano has white and black keys',2),
('a piano has strings',2),
('a strawberry tastes good',2),
('a snail moves slowly',2),
('a thing likes soul exist',2),
('a basketball bounces when dribbled',2),
('a basketball bounces',2),
('a horse has four legs',2),
('a horse runs faster than a chicken',2),
('a horse breathes air',2),
('a horse has four legs',2),
('a horse has a tail',2),
('a bear craps in the woods',2),
('a person who needs sleep get tired',2),
```

```
('a person has feelings',2),
('a person has hair',2),
('a person has a brain',2),
('a person has 2 legs',2),
('a bird flies with its wings',2),
('a bird has feathers',2),
('a bird has a head',2),
('a bird has two wings',2),
('a redlight means stop',2),
('a wasp stings cause pain',2),
('a living hearts pump blood',2),
('a watched pots ever boil',2),
('a human needs water to live',2),
('a human feels',2),
('a human drinks when s he is thirsty',2),
('a human has a nose',2),
('a human hands have bones in it',2),
('a human requires water',2),
('a human has a liver',2),
('a human has legs',2),
('a human has two nostrils',2),
('a human has two hands',2),
('a trustworthy mans keep secrets',2),
('a pentagon has five sides',2),
('a quadriped has four legs',2),
('a kangaroo has a pouch',2),
('a baromter measures air pressure',2),
('a fart smells bad',2),
('a whale weighs more than a ant',2),
('a whale lives in the ocean',2),
('a wristwatch has a band',2),
('a humming birds fly',2),
('a hat keeps your head warm',2),
('a hat gos on your head',2),
('a unicorn has four legs',2),
('a television displays video images',2),
('a lake contains water',2),
('a dog barks',2),
('a dog has four legs',2),
('a dog has four feet',2),
('a dog has more legs than a human',2),
('a bookstore sells books',2),
('a toothache hurts',2),
('a breast has nipples',2),
('a light bulbs produce heat',2),
('a boy has a penis',2),
('a train runs on railroad tracks',2),
('a happy cats purr',2),
('a chair has four legs',2),
('a minute has 60 seconds',2),
```

```
('a rectangle has four sides',2),
('a broken legs hurts',2),
('a duck quacks',2),
('a giraffe has four legs',2),
('a giraffe has a long neck',2),
('a boxer uses his fists',2),
('a headache hurts',2),
('a cow has four legs',2),
('a cow eats grass',2),
('a camera shops sell cameras',2),
('a camera has a shutter',2),
('a normal humans being have two eyes',2),
('a normal foots have five toes',2),
('a normal persons have two arms',2),
('a equal as',2),
('a plant needs water to grow',2),
('a proton has a positive charge',2),
('a thermos keeps your coffee hot',2),
('a room has four sides',2),
('a room has a ceiling',2),
('a room has a door',2),
('a hurricane has heavy winds',2),
('tomato juices stain white cloth',2),
('medicine heals the sick',2),
('lion eats meat',2),
('eleven minus two equal nine',2),
('fashion changes constantly',2),
('ten times five equal fifty',2),
('book contains knowledges',2),
('man haves a penis',2),
('man haves two legs',2),
('man tastes food with his tongue',2),
('spam exists',2),
('internet exists',2),
('iomega makes zip dirves',2),
('four pluss four equal eight',2),
('cheese melts when it is heated',2),
('clinton likes lewinski sucking him',2),
('microsoft contains the letter i',2),
('microsoft manufactures software',2),
('microsoft sells windows',2),
('microsoft makes software',2),
('microsoft makes windows nt',2),
('microsoft windowss crash',2),
('red means stop',2),
('red highlights danger',2),
('red ons traffic lights mean stop',2),
('pie rhymes with sky',2),
('hoover makes vacuum cleaners',2),
('ram stands for random access memory',2),
```

```
('vomit comes from the stomach',2),
('yanni plays the piano',2),
('iron rusts',2),
('fauna differs from flora',2),
('monday comes after sunday',2),
('monday follows sunday',2),
('china represss human rights',2),
('china haves a large population',2),
('gravity pulls us down',2),
('gravity affects everything on earth',2),
('gravity varys from planet to planet',2),
('gravity exists on earth',2),
('gravity exists',2),
('ice melts when it becomes warmer',2),
('ice melts and become water',2),
('ice melts',2),
('ice creams contain water',2),
('ice creams melt in the sun',2),
('ice creams melt in the heat',2),
('ice creams melt in heat',2),
('ice creams taste good',2),
('ice turns into water when it melts',2),
('ice floats in water',2),
('rotten meats smell bad',2),
('honda makes the accord',2),
('honda makes automobiles',2),
('alcohol makes people act silly',2),
('1 days consist of 24 hours',2),
('1 pluss one equal 2',2),
('1 timess 9 equal 9',2),
('1 timess 8 equal 8',2),
('1 timess 5 equal 5',2),
('1 timess 3 equal 3',2),
('1 timess 2 equal 2',2),
('electricity flows through wires',2),
('peircing hurts',2),
('federal expresss deliver packages',2),
('eating bleachs make you sick',2),
('eating beanss induce farting',2),
('whisky contains alcohol',2),
('beer contains alcohol',2),
('beer comes in bottles',2),
('beer haves water in it',2),
('beer makes you drunk',2),
('poetry exists',2),
('beauty varys based on the observer',2),
('summer haves more daylight',2),
('everybody needs water to live',2),
('everybody dies',2),
('everybody makes mistakes',2),
```

```
('shampoo cleans hair',2),
('rabbits eats carrots',2),
('mayonnaise contains eggs',2),
('11 pluss 11 equal 22',2),
('wood contains fibrous material',2),
('wood comes from a tree',2),
('wood comes from trees',2),
('wood burns in a fire',2),
('wood comes from trees',2),
('sushi contains raw fish',2),
('fried chickens taste good',2),
('africa exists',2),
('sea waters have salt in it',2),
('lava comes from volcanoes',2),
('effect follows cause',2),
('sunny weathers make you happier',2),
('fish lives in the water',2),
('stress weakens the immune system',2),
('rum contains alcohol',2),
('anybody lives in new jersey',2),
('pizza huts make pizza',2),
('pizza contains cheese',2),
('pizza tastes good',2),
('bmw makes cars',2),
('sunlight feels warm',2),
('sunlight helps plants grow',2),
('sunlight makes plants grow',2),
('sunlight makes you feel warm',2),
('sunlight causes warmth',2),
('about 165,000 cars are produced every day.',2),
('it would take less than 6 months to get to the Moon by car at
60mph.',2),

('The average car has 30,000 parts.',2),

('There are 1 billion cars currently in use on earth.',2),

('92% of all new sold cars in Brazil',2),

('Volkswagen owns Bentley, Bugatti, Lamborghini, Audi, Ducati and
Porsche.',2),

('The average American spends about 38 hours a year stuck in
traffic.',2),

('The first car accident occurred in 1891, in Ohio.',2),

('There are more cars than people in Los Angeles.',2),

('The inventor of the cruise control was blind.',2),
```

```
('Ferrari only builds 14 cars each day.',2),

('Henry Ford made a car out of soybeans.',2),

('Most new cars fake engine noise through speakers.',2),

('Each car has an engin',2),

('95% of a cars lifetime is spent parked.',2),

('Drinking a cup of caffeinated coffee significantly improves blood
flow.',2),

('New Yorkers drink almost 7 times more coffee than other cities in
the US.',2),

('54% of the Americans drink coffee every day.',2),

('Brazil has been the largest producer of coffee for the last 150
years.',2),

('Chicken contains 266% more fat than it did 40 years ago.',2),

('Honey is the only food that will never rot.',2),

('Cheese is the most stolen food in the world.',2),

('Scientists can turn peanut butter into diamonds.',2),

('49% of U.S. Adults eat one sandwich a day.',2),
('India has the worlds lowest meat consumption per person.',2),

('Bananas are slightly radioactive.',2),

('Bananas don not grow on trees.',2),

('Bananas are produced by herbaceous plants.',2),

('Bananas are the most popular fruits in the U.S.',2),

('Humans share 50% of their DNA with bananas.',2),
('how are you?',1),
('describe yourself?', 1),
('how is life?', 1),
('Would you mind getting me a drink?', 1),
('do you like tea?', 1),
('do you like coffee?', 1),
('do you like to go to school?', 1),
('do you believe in me?', 1),
```

```
('how do you know I am a person?', 1),
('I ate a sandwich in my house', 1),
('thank you', 1),
('how do you know that?', 1),
('Where is he going?', 1),
('see you soon', 1),
('are you interested in watching a movie?', 1),
(' I am worried', 1),
('I think I am alright', 1),
('trees do not grow bananas', 1),
('Do you have information about chicken?', 1),
('do you know me?', 1),
('no', 1),
('yes', 1),
('not sure', 1),
('end the chat please' , 1),
('good night', 1),
('good evening', 1),
('What are your interests', 1),
('Why cant you eat food?', 1),
('Where are you from', 1),
('Who is your mother', 1),

('a boat floats in water',2),
('a butterfly has wings', 2),
('a genius has a high iq', 2),
('advertising sells more products', 2),
('fire needs oxygen to burn', 2),
('irc stands for internet relay chat', 2),
('light travels quickly', 2),
('ocean contains salt water', 2),
('school helps children learn things', 2),
('smoking tobaccos cause cancer', 2),
('iq stands for intelligence quotient', 2),
('yellow is a color', 2),
('there was a revolution in france', 2),
('hitler ruled germany', 2),
('mexico is a country', 2),
('glue is sticky', 2),
('we put cigarette butts in a ashtray',2),
('playing is in traffic dangerous', 2),
('anchorage is in alaska', 2),
('school is a good place to learn', 2),
('lava is molten rock', 2),
('windows 2000 is an operating system', 2),
('can I sleep?', 1),
('can you go shopping', 1),
('can we play', 1),
('I am sleepy', 1),
('banana is my favoriate', 1),
```

```
('I will go to do shopping', 1),
('how is life?', 1),
('I like to watch movies', 1),
('I do not like to play video games', 1),
('Do you like to see a movie?', 1),
('I will see you tomorrow', 1),
('why you are here?', 1),
('family is important', 2),
('I like to drink water', 1),
('Lamo', 1),
('can we chat?', 1),
('I like you', 1),
('Is it too early?', 1),
('I am leaving', 1),
('lets talk about it', 1),
('I am done', 1),
]
```

APPENDIX D: CLASSIFIERS TESTING DATA

```
test =[
('You are sweet', 1),
('I feel amazing!', 1),
("I can't believe I'm doing this.", 1),
('Just for a minute', 1),
('Wear the belly before you go.', 1),
('It is just a party. Daddy.', 1),
('Oh, God.  It is starting.', 1),
(' If Kat is not going, you are not going.', 1),
('Daddy, people expect me to be there!', 1),
(' It is just a party. Daddy, but I knew you had forbid me to go', 1),
('What are you talking about?', 1),
('Cameron, I am a little busy', 1),
('Good ones, yes, Mr ', 1),
('hello, are you here', 1),
('How are you', 1),
('hi my name is Jamie',1),
('I have not seen you for long time', 1),
('lets go shopping',1),
('you are welcome',1),
('banana is a fruit', 2),
('cars are fast', 2),
('The "new car smell" is composed of over 50 volatile organic
compounds.', 2),
('Up to 19 girls can be crammed into a smart car.', 2),
('The average car has 30,000 parts.', 2),
('The average AmericanHonking your car horn, except in an emergency,
is illegal in NYC. spends about 38 hours a year stuck in traffic.',
2),
('The odds of dying in a car accident are around 1 in 5,000.', 2),
('It is a criminal offense to drive around in a dirty car in Russia.',
2),
('Car wrecks are the number one cause of death for Americans under
35.', 2),
('Up to 80% of an average car is recyclable.', 2),
('The average Bugatti customer has about 84 cars, 3 jets and one
yacht.', 2),
('Traffic accidents kill 1.25 million people per year.', 2),
('The top speed at the world first real automobile race in 1895 was
just 15 mph.', 2),
('a pot has a lid',2),('a men has a head',2),
('a green traffics light mean go',2),
('a green lights mean go',2),
('a sperm whales have a big head',2),
('a billionaire has a lot of money',2),
('a skateboard has wheels',2),
('a hungry child cries',2),
('a computer computes',2),
('a computer consumes energy',2),
('a computer uses power to run',2),
```

```
('a computer uses electricity',2),
('a computer has a memory',2),
('a computer has a cpu',2),
('a computer monitors use electricity',2),
('a computer needs a memory',2),
('a sunburn causes pain',2),
('a cat has four legs',2),
('a cat has fur',2),
('a cat has a tail',2),
('a cat has a brain',2),
('a cat has whiskers',2),
('a sentence contains words',2),
('a sentence consists of words',2),
('a banana grows in a tree',2),
('a box sometimes have a lid',2),
('a box has corners',2),
('a joke makes you laugh',2),
('a monitor screens emit radiation',2),
('a monitor displays a image',2),
('a red traffics signal mean stop',2),
('a clock keeps track of time',2),
('a clock shows the time',2),
('a clock runs clockwise',2),
('a clock tells time',2),
('a alligator swims',2),
('a stapler staples',2),
('a human needs oxygen',2),
('hi world', 1),
('what', 1),
('hmmm', 1),
('not really', 1),
('They do not!',1),
('They do to!',1),
('I hope so.',1),
('She okay?',1),
('Lets go.',1),
('Wow',1),
('Okay you are gonna need to learn how to lie.',1),
('No',1),
('no', 1),
(' that is not funny', 1),
('how do you know that?', 1),
('cannot you guess?', 1),
('I am not sure',1),
('are you depressed?', 1),
('a rice is a white food', 2),
('a beans is a food',2),
('apple taste good',1),
('wine has a color',2),
('hi there',1),
```

```
('tell me a story', 1),
('tell me about yourself', 1),
('am i nice', 1),
('Can we talk', 1),
('it is funny', 1),
('tell me a story', 1),
      ('tell me about yourself', 1),
('am i pretty', 1),
('Can we talk', 1),
('it is funny', 1),
('tell me a story', 1),
('Can we go to see a movie', 1),
('6 is more than 2', 2),
('an orange is a fruit', 2),
('cars can be fast', 2),
('I am going to bed', 1),
('lets talk', 1),
(' I am sad', 1),
('I do not have friendss', 1),
('I agree', 1),
('there are many cars in the us', 2),
('look there', 1),
('life is good', 2),
('a desert is a course', 2),
('a fish is a food', 2) ,
('a cherimoya is a fruit', 2),
('a nectarine is a frui', 2),
('an artichoke is a vegetable', 2),
('an eggplant is a vegetable', 2),
('a red pepper is a vegetable', 2),
('a soda is a cold drink', 2),
('a toothpick is an object', 2),
('a coffee is a drink', 2),
('I am back', 1),
('I am leaving',1),
('not yet',1),
('he is my son', 1),
('who are you?', 1),
('I like swimming', 1),
('I miss you', 1)
]
```

APPENDIX E: LCC CLASSIFIERS TESTING PROCEDURE DATA

The keyboard is extremely comfortable to use .
there are too many problems with the transmission
Toyota's quality is slipping .
Horrible quality of interior .
the sound quality is very good .
Gas mileage is only about 22 around town .
The hotel is in a great location
Best location in Fisherman's Wharf .
This hotel is in a great location in terms of a walking distance to key Fisherman's Wharf attractions .
a spoon is a utensil
a food can be stoved
a rice is a white food
sugar is a sweet food
a drink is a food
avocado is a fruit
a coconut is a fruit
a pineapple is a yellow fruit
a beet is a vegetable
drinks are liquids
property color has possible values red and white
hi, do you know that apple is a fruit
Like everyone else, I have the problem with the transmission .
I had always heard that Toyota had the best quality cars .
I've been really impressed with the sound quality .
do you think a spoon is a utensil?
I am not sure but do cars have 4 wheels
this is right, rice can be brown or white
Hi there, how are you? Orange is a type of fruits
Do you know that Toronto is in Canada
I have heard that Warsaw is a town in Poland.
The sky is blue
the earth is round
apples have multiple colors
a lantern provides light
a fish needs oxygen
a bicycle has wheels
a wolf has four legs
a rabbit eats carrots
a green traffics light mean go
a computer consumes energy
I don't know if cats can think
I have seen a bird that can fly
hi, a whale weighs more than a ant
I heard that some birds cannot fly
Do you know that some fish eat other fish
I have heard that math is fun
my teacher said the earth has a atmosphere
can you believe that a snail moves slowly
Yes this is true, a bird flies with its wings

My mother said a wasp stings cause pain
I know that a watermelon is a green fruit
Do you like artichoke? It is a vegetable
hmmm, a coffee is a hot drink
that is awesome, a property taste has possible values sweet sour bitter salty
yes it is, a fig is a fruit
Yes I know that a mango is a fruit
I don't care, I know the sky is blue
Oh how do you know that some people can die from chocking
I don't know if lentil has many colors
No that is not true, the earth is round

APPENDIX F: PARTICIPANTS COMMENTS

I found the system pretty good in checking its database. I found the system would have strange replies at times when chatting.

The system is sometimes confused with differently phrased questions that are not facts

very quick response time. it was able to identify chatting from information related questions.

The KB was easily updated when logged in as a trusted user

Chatting with the AI was humor to me especially when it asked me if I was a robot

I think this system is interesting

the system was organized and robust! it did not permit me to change information as untrustworthy

I really liked it. I thought some of the funny responses were more human-like

Understand my statement after have misspelled a word, was human to me especially when it asked me if I was a robot

The system was easy to chat with, but it is still not completely natural if it is being compared to conversing with a human. It is too descriptive or uses more words than a regular conversation

I thought it was a great idea, I think this program could be a valuable asset in the industry if the KB can be more extended

to make the system friendlier, you could add fun facts. I like the system, this is the stuff that makes AI fun!

A promising piece of work

the system needs a larger KB and faster responses

It is a interesting idea and it could be very useful if the computer could learn the new information by having a dialogue. However, from the dialogues, I could see that the computer is not a human,

it could not understand what information was needed to be memorized and wha information should have been replaced. All of the new information that

helpful to scientists, it will memorize not only the important information, but also everything is else that is said in a dialogue.
Also, because of the unuseful information, the memory of computer

I really thought this was informational, first it seemed a bit silly but I read it and overall thought it was interesting and if it wasn't for me reading this then I would not have known any of this.

I think the system is an intriguing way of handling and communication information

the system should work to become more human like in its preset responses, as well as less redundant with the recall of information.

The system was able to accurately add data during the conversations especially when it's related to the agenda of the user but as for the
conversing part, it still needs work as well as some parts which it could not pull up accurate answers to the questions, and was only able to pull related information instead

Some of the conversations did not feel very natural.

While it worked correctly at times, the system was often inconsistent

when attempting to distinguish chatting and content-related info.

the system somewhat kept conversation but would answer with responses that is not about that topic and maybe change topics

I felt that the system represent something very similar to an Alexa or
a Siri type. When asked question it didn't always have the answers,
but it tried. And I'm trying it almost seemed to have human like
conversations.

Overall I would compare the system to a siri or alexa type. They don't
have all the answers, but they try. In trying they occasionally carry

on in what seems like a human conversation.

In the second conversation the system said it had the "same info" when the user inputs "bananas grow on trees" and "bananas don’t grow on trees" . The system should have asked about an information exchange after the first comment instead of agreeing.

The dialogue was maybe a little bit too restrictive at times but overall good development

I believe the system did have good adaptive processing in adding information once corrected but it did not flow as a normal human to human conversation

It did not feel like I was having a natural conversation with a human. The program did well answering some questions and separating the chatting information from the content-related information a few times.

I thought it was a very simple machine that didn't understand the chatting conversation but did understand the content-related information

I thought the system was adding random information that was not important like the fact they had a lot to do that day. Thats not a fact to fit in with the rest of the list, however it did ask to change information at good times when it conflicted to what was already on the list.

Very innovative, I encourage more "conversation" with the system
The flow of the systems conversation with the user seemed a little too artificial and calculated. For example, the system would announce its actions regarding new information it acquires, rather than continuing the flow of natural conversation.

I thought the system was pretty intelligent. It didn't feel very human-like to me, but I could tell that efforts were there to make it

more human.

I did not think that the conversation resembled anything near what I normal conversation would be like and that the system failed to answer questions that were related to the data that was already programmed into it.

The system has potential to eventually mimic another human but at this moment, it doesn't have enough information saved to do so and is far from being able to mimic human characteristics like emotion.

I think that the system was more focused on answering questions that were not opinion based but rather facts that can be looked up. The system was also great at carefully adding information when commanded to do so.

After reading the dialogue, between the system and user. I think the system has some flaws like any other technology does but any bugs found can sometimes be fixed with the help of feedback like this
The dialogue logs would be easier to read if the system replied with proper grammar. When the user asked about calculus the system responded with a random fact about "2001" which seems like an error.

Great program! Super convenient.

its a great idea that should be implemented

I think the system is far from mimicking the vernacular of two humans, but the system is promising. I think it would be quite cool and unique to have a system that you can have regular conversation with.
The system works well. There are a few things that could be fixed but outside of that it works.

Although the system didn't seem very smooth in conversation, it was able to understand the difference between basic greetings and content related information for knowledge storage

I think that the system needs improvement on the overall feel conversing with a human as well as to be able to distinguish between conversation and facts.

The system is able to correctly add new information to its database. However, it sometimes adds irrelevant information that is just conversational.

A bit scattered but pretty useful overall!

I think that besides updating the data available and including more information for the system to pull from, the system that was used in the trials works very well. With some minor improvements I can definitely see this system making the automated calls I have to sit through now more efficient.

The system seemed helpful for some questions, but for others it only added to the confusion.

There is room for improvement, but it functions well.

While the system is able to properly add and request confirmation to add info to it's KB, I noticed it had difficulty differentiating between conversation and Q+A, such as when it added "a lot to do today" to it's KB and incorrectly answering the user's question about calculus.

Perhaps to enhance to natural experience of conversing to an A.I, the developer may consider changing phrases such as "I am checking for the answer" or "I am checking my KB" for content information already in the system. If the systems decides to add new information, it can prompt the user to do so; to display these phrases during every interaction will not likely reach the experience to that of a human.
I understand the importance and use of Artificial Intelligence however, I just don't believe the tone of the system will ever measure up in being equal of giving that feeling of a human-human

conversation.

Siri is way smarter than this “system"

Though the system was able to identify and relate some of the chatting information to the content in its data base, it was not able to identify some of the language used by the user such as when the user said "hi," the system did the register the thought as to how an actual human would.

The system added additional information told by the user after each conversation but also deleted some information to after each conversation.

In order for this system to be successful it may possibly need a wider vocabulary and know the usage of different types of languages such as modern-day slang and jargon.

The system is definitely something to be proud it. It can be more human like by further developing how the AI recognizes context and facts and also how AI searches data.

It is impressive but, needs more work transitioning smoothly and separating the person feelings from general facts.

The system reminds me of the early stages of Siri and so the system can only get smarter with the more information that is added.

It was disappointing that the system couldn't answer question about Hal from 2001, especially since the system brought it up!

If the user mislead the system, then the system will saved the wrong info! In this case, the system may depend on trusted online information or other information already saved due to previous conversation for other user so the system can compare what is the right info to be learned depending on the majority of users' common info

Fun and interesting.

The system did not have any protections to identify incorrect content exchanges and allowed the human to add incorrect information

Data sets might have to be fed properly into the system
is it something like Alexa or google home?

Great system, but needs a few clarifications to help conversation flow better

To make the system appear more human, I would include more basic conversational answers (i.e. "I am a robot, not a boy or a girl." and some kind of greeting in response to "Hi.") and make the system's responses shorter.

It missed the question about car parts, replying that it had no similar information, when it really had exactly the same answer. During chatting, it occasionally made totally unrelated comments in a non-human way. Aside from that, it’s really good! Good luck with your research!

First conversation in Table 2 was very stilted, the Calculus -> 2001 conversation was the worst example of it.

It disregard that it has the information about the first car crash twice and neglected to answer the car parts' question even though it had that information.

I thought that system was very interesting.

This is a fascinating project. Its english could use work... would you be able to feed it information outside of dialogue? In the form of long writings perhaps? Keep up the great work!

The system seems to be very good at identifying factual statements, but not at holding natural conversations. Especially in the first conversation, the system stored info that there was a lot to do today with no context and no indication that the system needed to call upon that later.

APPENDIX G: LCC KNOWLEDGE BASE THAT CONTAINS 960 SENTENCES AND ITS ASSOCIATED SEMANTIC NETWORK

cars have 4 wheels.
about 165,000 cars are produced every day.
it would take less than 6 months to get to the Moon by car at 60mph.
The first car accident occurred in 1891, in Ohio.
The average car has 30,000 parts.
There are 1 billion cars currently in use on earth.
92% of all new sold cars in Brazil.
Volkswagen owns Bentley, Bugatti, Lamborghini, Audi, Ducati and Porsche.
The average American spends about 38 hours a year stuck in traffic.
The first car accident occurred in 1891, in Ohio.
There are more cars than people in Los Angeles.
The inventor of the cruise control was blind.
Ferrari only builds 14 cars each day.
Henry Ford made a car out of soybeans.
Most new cars fake engine noise through speakers.
95% of a car's lifetime is spent parked.
Drinking a cup of caffeinated coffee significantly improves blood flow.
New Yorkers drink almost 7 times more coffee than other cities in the US.
54% of the Americans drink coffee every day.
Brazil has been the largest producer of coffee for the last 150 years.
Chicken contains 266% more fat than it did 40 years ago.
Honey is the only food that will never rot.
Cheese is the most stolen food in the world.
Scientists can turn peanut butter into diamonds.
49% of US Adults eat one sandwich a day.
India consumes the lowest amount of meat per person.
Bananas are slightly radioactive.
Bananas don't grow on trees.
Bananas are produced by herbaceous plants.
Bananas are the most popular fruits in the US.
Humans share 50% of their DNA with apples.
property taste has possible values sweet sour bitter salty.
edibles are objects.
foods are edibles.
drinks are edibles.
dishes are edibles.
people can eat foods.
foods have a taste.
drinks have a taste.
dishes have ingredients.
water is a drink.
meals are activities.
a course is a concept.
meals have courses.
an appetizer is a course.
a toothpick is an object.
a menu is an object.

grilled is a method.
fried is a method.
baked is a method.
a desert is a course.
a lunch is a meal.
a breakfast is a meal.
a dinner is a meal.
a spoon is a utensil.
a knife is a utensil.
a fork is a utensil.
a food can be grilled.
a food can be cooked.
a food can be stoved.
a meal has courses.
a monitor displays a image.
a red traffics signal mean stop.
a clock keeps track of time.
a clock shows the time.
a clock runs clockwise.
a clock tells time.
a alligator swims.
a stapler staples.
a human needs oxygen.
a human needs water to live.
a human feels.
a human drinks when s he is thirsty.
a human has a nose.
a human hands have bones in it.
a human requires water.
a human has a liver.
a human has legs.
a human has two nostrils.
a human has two hands.
a trustworthy mans keep secrets.
a pentagon has five sides.
a quadruped has four legs.
a kangaroo has a pouch.
a barometer measures air pressure.
an onion smells bad.
a whale weighs more than a ant.
a whale lives in the ocean.
a bridge cross a body of water.
a moon revolves around a planet.
a male has a hand.
a refrigerator cools objects.
a calculator has buttons.
a cookbook contains recipes.
a bookshelf holds books.
a jellyfish lives in the sea.
a massage feels good.

a circle has a radius.
a circle consists of 360 degrees.
a trucker drives a 18 wheeler.
a flag waves in the breeze.
a skunk smells bad.
a good jokes make people laugh.
a crying babies attract attention.
a parent has children.
a battery runs down.
a roof gets wet when it rains.
a wise mans listen to advice.
a wise sons bring joy to his father.
a book has pages.
a book has mass.
a book consists of chapters.
a nail has a head.
a diamond cuts glass.
a tricycle has three wheels.
a table supports other objects.
a thermometer measures temperature.
a forest consists of multiple trees.
the human hearts have 4 chambers.
the planet earths orbit the sun.
the planet Jupiter exist.
the wind sometimes blow outside.
the wind blows.
the sun provides light and heat.
the sun generates heat.
the sun produces light.
the sun shines.
the sun rises in the east.
the sun rises in the morning.
the sun rises every morning.
the sun rises daily.
the sun emits light.
the sun bigger then earth.
the sun sustains life on earth.
the sun comes up in the east.
the sun comes up every morning.
the sun comes up every day.
the sun comes out in the daytime.
the sun shines today in Hungary.
the sun shines on the moon.
the sun shines.
the sun burns hot.
the sun rises in the east.
the sun emits uv rays.
the sun warms the earth.
the sun burnings.
the sun rises in the east.

the sun rises from the east.
the sun radiates heat.
the sun creates heat.
the ocean contains fish.
the ocean contains salt.
the ocean contains salt water.
the ocean appears to be blue.
the rabbit jumps.
the internet has a future.
the great walls stand in china.
the postman delivers letters.
the normal humans have two thumbs.
the sky looks blue on a clear day.
the brain contains neurons.
the rush tos work causes stress.
the desert has sand.
the number 100s contain two zeros.
the world needs love.
the world rotates.
the world spins.
the world exists.
the army has generals.
the galaxy has end.
vanilla ices cream taste good.
England haves a monarchy.
life needs energy.
life requires water on earth.
life haves a price.
life exists beyond earth.
life exists in universe.
life exists.
fleece keeps you warm.
trust builds stronger relationships.
matter consists of tiny particles.
matter exists.
frog jumps and eat fly.
aspirin helps a headache.
airplanes flys.
this questions have six words.
this questions exist. do you exist.
sunburned skins get red.
Chris Duarte play blues.
travel broadens the mind.
new York lie in america.
new yorks city have a subway system.
100 minuss 1 equal 99.
glue sticks things.
bad breaths make kissing unpleasant.
lightning causes forest fires.
tomorrow comes after.

tomorrow follows.
nokia makes telephones.
sony produces electronic products.
sony producess hardware.
sony makes good products.
sony makes electronic devices.
3 timess 6 equal 18.
winnie the pooh like honey.
gym sounds like jim.
50 pluss 1 equal 51.
spam exists.
internet exists.
iomega makes zip dirves.
four pluss four equal eight.
cheese melts when it is heated.
clinton likes lewinski sucking him.
microsoft contains the letter i.
microsoft manufactures software.
microsoft sells windows.
microsoft makes software.
microsoft makes windows nt.
microsoft windowss crash.
red means stop.
red highlights danger.
red ons traffic lights mean stop.
pie rhymes with sky.
hoover makes vacuum cleaners.
ram stands for random access memory.
vomit comes from the stomach.
yanni plays the piano.
iron rusts.
fauna differs from flora.
monday comes after sunday.
monday follows sunday.
china represss human rights.
china haves a large population.
gravity pulls us down.
gravity affects everything on earth.
gravity varys from planet to planet.
lava comes from volcanoes.
effect follows cause.
iron rusts.
fauna differs from flora.
monday comes after sunday.
monday follows sunday.
china represss human rights.
china haves a large population.
gravity pulls us down.
gravity affects everything on earth.
gravity varys from planet to planet.

gravity exists on earth.
gravity exists.
ice melts when it becomes warmer.
ice melts and become water.
ice melts.
ice creams contain water.
ice creams melt in the sun.
ice creams melt in the heat.
ice creams melt in heat.
ice creams taste good.
ice turns into water when it melts.
ice floats in water.
rotten meats smell bad.
honda makes the accord.
honda makes automobiles.
alcohol makes people act silly.
1 days consist of 24 hours.
1 pluss one equal 2.
1 timess 9 equal 9.
1 timess 8 equal 8.
1 timess 5 equal 5.
1 timess 3 equal 3.
1 timess 2 equal 2.
electricity flows through wires.
peircing hurts.
federal expresss deliver packages.
eating bleachs make you sick.
whisky contains alcohol.
beer contains alcohol.
beer comes in bottles.
beer haves water in it.
beer makes you drunk.
poetry exists.
beauty varys based on the observer.
summer haves more daylight.
everybody needs water to live.
everybody dies.
everybody makes mistakes.
shampoo cleans hair.
rabbits eats carrots.
mayonnaise contains eggs.
11 pluss 11 equal 22.
wood contains fibrous material.
wood comes from a tree.
wood comes from trees.
wood burns in a fire.
wood comes from trees.
sushi contains raw fish.
fried chickens taste good.
africa exists.

sea waters have salt in it.
lava comes from volcanoes.
effect follows cause.
sunny weathers make you happier.
ancient means old.
fish lives in the water.
stress weakens the immune system.
rum contains alcohol.
anybody lives in new jersey.
pizza huts make pizza.
pizza contains cheese.
pizza tastes good.
bmw makes cars.
sunlight feels warm.
sunlight helps plants grow.
sunlight makes plants grow.
sunlight makes you feel warm.
sunlight causes warmth.
bacon comes from pork bellies.
bacon comes from a pig.
bacon tastes good.
two plus two equal four.
10 plus 10 equal 20.
10 plus 9 equals 19.
10 lies between 1 and 100.
10 divided by 2 equal 5.
love helps you be happy.
love exists.
love hurts sometimes.
bill Clintons lie.
bill gates have a lot of money.
bill gates have lots of money.
bill gates make a lot of money.
time flows in one direction only.
time flows.
time moves in one direction only.
my husbands love me.
juice comes from fruit.
regular coffees contain caffeine.
plants haves life.
amazon dots com sell books.
amazon sells books.
saturday comes after friday.
x equals x.
.
advertising sells more products.
java makes use of a virtual machine.
oil comes out of the ground.
swimming helps keep people fit.
turtles lives longer than spiders.

petrol burns.
saltwater contains salt and water.
getting runs over by a car hurt.
getting yours arms chopped off hurt.
getting as knife in the head hurt.
harley davidsons make motorcycles.
square haves 4 sides.
lennon begins with the letter l.
santa clauss wear red.
santa wears red.
friday follows thursday.
fire needs oxygen to burn.
fire gives off light.
fire gives off heat.
fire produces smoke.
fire produces heat.
fire harms humans.
fire burns paper.
fire burns human flesh.
fire burns things.
irc stands for internet relay chat.
eight rhymes with hate.
dell makes computers.
every livings human have a head.
every livings thing die sometime.
every livings thing die.
every countrys have a flag.
every years consist of 12 months.
every animals have only one anus.
every triangles have 3 sides.
every triangles have 3 corners.
snow melts into water.
snow falls most often in winter.
marijuana eases pain.
light travels quickly.
light travels faster then sound.
light travels faster than sound.
light travels as a particle.
light travels as a wave.
light helps humans see.
light sometimes behave like a wave.
light carries energy.
night alternates with day.
night follows day.
www stands for world wide web.
hamburger comes from cows.
heartbreak hurts.
February sometimes have 28 days.
February comes after January.
paint comes in colors.

brewing coffees smell good.
cutting yourself hurt.
glass breaks easier than steel.
glass breaks.
rain feels wet.
rain helps the grass grow.
rain helps the people.
rain falls down.
rain falls from the sky.
rain falls from clouds.
rain falls.
rain comes from the sky.
rain comes from the clouds.
rain comes out of clouds.
rain makes grass grow.
rain makes he ground wet.
blood looks red.
blood flows through human veins.
blood carrys oxygen in the body.
pink floyds make music.
nike sells shoes.
one pluss one equal two.
one gets wet in rain.
one buys food at a supermarket.
one centurys consist of 100 years.
one likes to be happy.
sugar tastes sweet to you.
sugar tastes sweet to humans.
sugar tastes sweet.
a hour has 60 minutes.
a umbrella keeps you dry.
a octopus has eight arms.
a effect needs a cause.
a elephant exists.
a ocean contains salt water.
a ocean has salt water in it.
a ocean consists of water.
a object hiddens by a wall exists.
a architect designs buildings.
a orgasm feels nice.
a ant has 6 legs.
jerusalem locatads in israel.
feces smells.
humour relieves stress.
alice needs a server backup system.
alice consumes bandwith.
alice makes mistakes.
alice becomes boring after awhile.
consuming alchols makes you drunk.
being puncheds hurt.

being strucks by lightning hurt.
6 timess 7 equal 42.
6 timess 6 equal 36.
porsche makes sports cars.
nintendo makes video games.
metal conducts electricity.
human feelingss changes with time.
human sings.
human haves two eyes.
human haves two hands.
human wastes smell bad.
norway haves its own flag.
word propers mean correct.
exercise improves health.
chevrolet manufactures automobiles.
firestone makes tires.
lightening contains electricity.
un stands for united nations.
salt corrodes metal.
mankind loves itself.
paper comes from a tree.
paper comes from wood.
paper burns.
segregation exists.
hair loss is in my genes.
everyone dies.
equator passes through Africa.
air contains oxygen.
Wednesday comes after Monday.
people haves feelings.
people likes eating.
transsylavania exists.
education begins at home.
uk stands for united kingdom.
Pamela Anderson's have large breasts.
ham comes from pigs.
Harvard provides a good education.
hot solders on your skin hurt.
sound travels in the form of waves.
silk feels soft.
cats eats mouses.
exercise improves human health.
school helps children learn things.
23 22s equal 1.
2 means the same as two.
2 plus 2 equal four.
2 plus 2 equal 4.
2 plus 2 equal 2 times 2.
2 xs 2 equal 4.
2 timess 4 equal 8.

2 timess 3 equal 6.
lying ins the sun make you warmer.
coffee contains caffeine.
coffee contains caffeine.
coffee contains caffeine.
coffee wakes a person up.
coffee haves caffeine in it.
each humans have a heart.
whiskey contains alcohol.
laughing feels good.
laughing makes you feel good.
applause follows a good performance.
fm means frequency modulation.
fm stands for frequency modulation.
September haves thirty days.
superman flys.
twenty plus twenty equal forty.
pc stands for personal computer.
tv means television.
tv exists.
cow has four legs.
vodka contains alcohol.
aol stands for America online.
Serena William play tennis.
jerry springers host a talk show.
arithmetic deals with numbers.
clay hardens when heated.
cyanide kills humans.
gravity effects all objects on earth.
spring comes before summer.
yesterday occurs in the past.
real loves exist.
e comes after d in the alphabet.
e equals mc 2.
cd stands for compact disc.
cd stands for compact disc.
cd stands for compact disc.
arithmetic involves numbers.
0 pluss 0 equal 0.
0 0s equal 0.
wine contains alcohol.
wine comes in bottles.
wine comes from grapes.
wine tastes different from beer.
watch shows time.
it feels good to masturbate.
it feels good to laugh.
it feels good to stretch.
it feels good to scratch a itch.
it feels good to master a new skill.

it feels good to win.
it snows in the winter.
it gets dark at night.
it gets wet when it rains.
it takes two to tango.
it takes two to make a couple.
it rains during the monsoon.
it rains in England.
it rains in America.
it rains in Seattle.
it rains over the ocean.
it gets dark at night.
it hurts to cut yourself.
it hurts to cut into your body.
it hurts to fall down.
it hurts to fall on your head.
it hurts to fall.
it hurts to be cut with a knife.
it hurts to lose a friend.
it hurts to stick your hand in fire.
it hurts to touch fire.
it hurts when you break your arm.
58 pluss 2 equals to 60.
looking at the sun hurt your eyes.
earth revolves around the sun.
earth has only one moon.
earth has a natural satellite.
earth has life.
earth rotates around the sun.
earth haves a moon.
earth haves 1 moon.
earth haves one moon.
earth haves life on it.
earth spins on its axis.
earth moves.
humanity haves a future.
Cisco systems make ip routers.
men enjoys looking at naked ladies.
lunch comes before dinner.
ford makes cars.
rubber stretches.
rubber bounces.
I love fish.
sun feels good on your face.
sun rises in east.
sun rises in the east.
sun shines onto earth.
sun makes software.
sun gives light.
chlorophyl makes plants green.

five plus five equal ten.
free softwares exist.
war lys harm people.
orange juices come from oranges.
9 pluss 9 equal 18.
9 equals nine.
letters forms words.
rainbow has 7 colors.
windows crashs.
windows over crash.
history describes past.
smoking gives you cancer.
smoking tobaccos cause cancer.
smoking harms your health.
diamond cuts glass.
popcorn pops when heated.
arizona iceds tea contain caffeine.
wool comes from sheep.
europe consists of many countries.
swiss cheess have holes.
tuesday immediatelys follow monday.
Tuesday follows Monday.
racism exists on earth.
racism exists.
racism hurts the world.
Boeing builds airplanes.
Boeing makes aircraft.
iq stands for intelligence quotient.
frustration causes stress.
soccer involves much running.
biology involves chemistry.
coffee contains caffeine.
apple computers make the macintosh.
most steaks come from cattle.
most beers contain alcohol.
most of fruits grow on trees.
most dogs have a tail.
biting ons your tongue hurt.
water freezes at zero celsius.
water freezes at 0 degrees celsius.
water helps plants grow.
water contains oxygen.
water extinguish fire.
water flows downhill.
water flows.
water falls from the sky.
water boils at 100 degree celsius.
water covers most of the earth.
water quench thirst.
water refracts light.

water from the ocean contain salt.
water puts out fires.
water makes things wet.
water turns solid at 32 degrees f.
all dies on Russian submarine Kursk.
all lives end eventually.
milk needs to be refrigerated.
milk contains calcium.
milk comes from cows.
milk haves calcium.
Stephen kings write horror novels.
sand contains quartz.
nasa launch spaceships.
having as shower make you clean.
having as orgasm feel good.
having problems irritate one.
tennis requires a ball.
yellow is a color.
violence is bad.
authors write books.
fear is a emotion.
staying up all night makes you tired.
needle is used to sew clothes.
fire is hot.
drinking water is healthy.
pumpkins grow on ground.
hockey is a sport.
rio de janeiro is in brazil.
music is made of sounds.
music is a art.
snail run slow.
bananas are fruit.
madonna is female singer.
child marriage is bad.
did nazis kill jews during the war.
did nazis kill jews.
did bob marley like reggae.
did shakespear write romeo and juliet.
did bill clinton cheat on his wife.
did a iceberg sink the titanic.
did reggae music originate in jamaica.
did people invent the computers.
did walt disney create mickey mouse.
did homer write the odyssey.
did ernest hemingway write novels.
did thomas edison invent the light bulb.
did iraq invade kuwait in august 1990.
did dinosaurs exist.
did the printing press change the world.
did the holocaust happen.

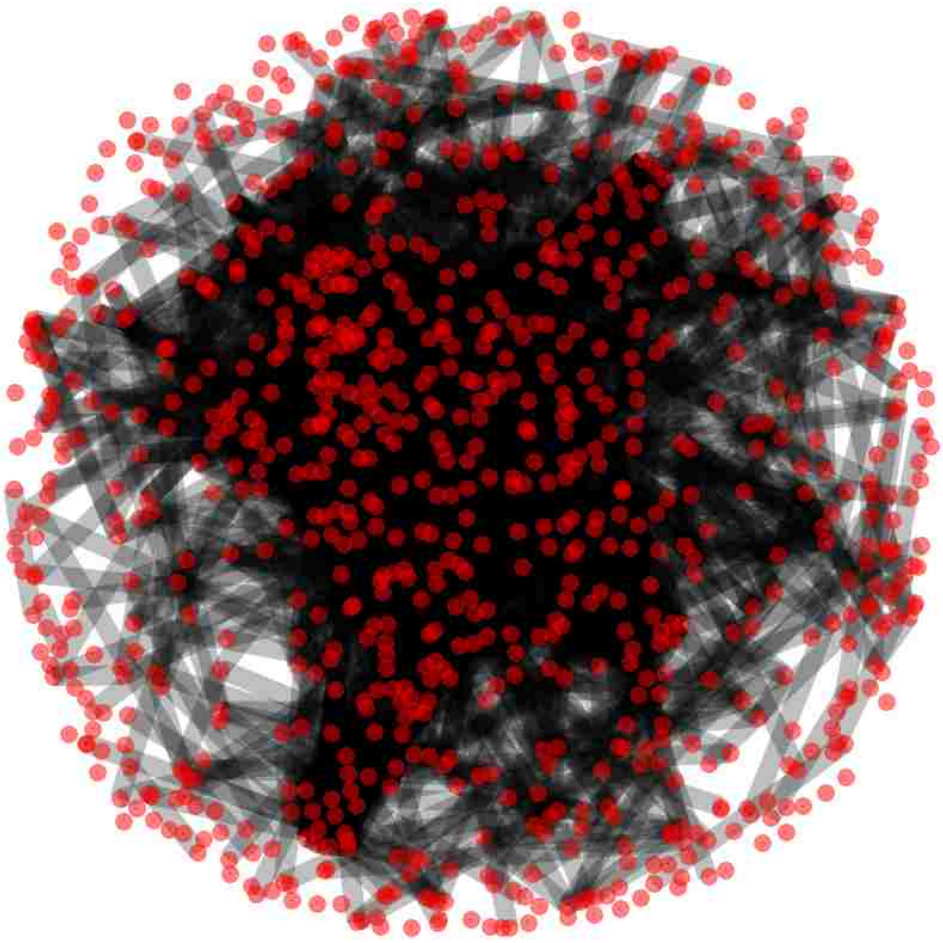
did the japanese lose world war ii.
did the allies win world war 1.
did the ancient greeks drink water.
did the ancient egyptians worship cats.
did the north win the civil war.
did the beatles sing yellow submarine.
did the beatles break up.
did the village people sing ymca.
did the roman empire fall.
did the dinosaurs exist.
did the titanic sink.
did the titantic sink.
did the man reach the moon.
elvis was a human.
was elvis known as the king.
mark twain was a author.
john lennon was a famous singer.
john lennon was a member of the beatles.
was john f kennedy assasinated.
john wayne was a american actor.
john wayne was a actor.
rimbaud was was a french poet.
t shirts can be red.
t shirts come in many sizes.
color of sky is blue.
grapes are fruit.
gandhi was a indian leader.
could a chicken cross a road.
could elivs presley sing.
could websites be easier to use.
soccer is a ball game.
sould you find a car in a garage.
animals need water to live.
animals breathe oxygen from the air.
would i throw a ball with my hand.
would a person weigh less on the moon.
you would find pants in a closet.
you would find a fireplace in a house.
you would find a car in a garage.
you would find a closet in a house.
you would find a microwave in a kitchen.
you would find a toilet in a bathroom.
you would find a dining room in a house.
you would find a living room in a house.
you would find a bedroom in a house.
you would find a bathtub in a bathroom.
you would find a sink in a bathroom.
you would find a bed in a bedroom.
you would find a lamp in a bedroom.
you would find a blender in a kitchen.

you would find a lawn mower in a garage.
you would find a oven in a kitchen.
you would find sheets on a bed.
you would want to win a million dollars.
you would enjoy a cold drink.
you would be my friend.
you would like to be loved.
you would like to have lots of money.
would one plus one equal 2.
would world peace be a good thing.
would reading increase your vocabulary.
mountains are taller than hills.
uk stands for united kingdom.
uk stand for united kingdom.
saki is made from rice.
whales are mammals.
doesfire burn dry wood.
creditcards are made of plastic.
oil is slippery.
isa e mail cheaper than telephone.
jupiter is the largest planet.
jupiter is a planet.
colombia is in south america.
photographic negatives contain silver.
pink is a color.
congo is in africa.
fish swim under water.
my uncles son is my cousin.
my fathers sister is my aunt.
my fathers brother is my uncle.
my mothers brother is my uncle.
it is dark at night.
it is good to survive.
it is good to take care of humans.
it gets cold in alaska.
albert einstien was a scientist.
uranus is a planet.
cars have four wheels.
cars have wheels.
cars have 4 wheels.
cars are used for transportation.
monopoly is a boardgame.
ibm makes computers.
cat is larger than mouse.
electricity can kill.
rome is the capital of italy.
day by day you are getting older.
women have two breasts.
women are pretty.
6 is greater than 5.

britain is a country.
make a mistake.
gnu is not unix.
when the sun is up is it daytime.
when it rains do things get wet.
when a human bleeds is the blood red.
when you are cut do you bleed.
when swimming should sharks be avoided.
when water freezes does it become ice.
four plus four equals eight.
must you eat to live.
must people have eyes to see.
must all animals eat to survive.
beer is a alcoholic beverage.
in a dark room is white still white.
in general do we need light to see.
in general is urine yellow.
dollars are a kind of currency.
hitler ruled germany.
if i break my arm will it hurt.
if i drop a apple will it fall.
if something is large is it big.
if there is no light is it dark.
if a vaccume cleaner sucks is that good.
if a comb is in water is that comb wet.
if a person is asleep do they breathe.
if you get hit by a car will it hurt.
if you cut me do i bleed.
if you water plants will they live.
if facing east is behind you is west.
telephones are used for communication.
delhi is capital of india.
mars is a planet.
mars is called the red planet.
mars is known as the red planet.
14 is twice as many as 7.
rabbit run fast.
what names is your friends.
wired is a magazine.
one plus one will be two.
one and one will be two.
mexico is a country.
mexico city has a lot of people.
nokia makes cellular phones.
sea water is salty.
sea water is salted.
wheels are round.
i scream in pain when i burn my face do.
i think. i exist.
i enjoy music.

gossiping is addictive.
six is bigger than five.
clothing can be made out of cotton.
whiskey is a alcoholic drink.
stone thrown upwards falls down.
honey bee live in a group.
winter is a season.
lava is molten rock.
vascodagama was a voyager.
black is dark.
purple is a color.
gold is heavier than tin.
diesel fumes are carconogenic.
glue is sticky.
jazz is a type of music.
jazz is a kind of music.
brains are found inside heads.
trout live in water.
nights are colder than days.
hair help to protect our head.
2 plus 4 minus 3 plus 2 equals 5.
2 plus 2 is four.
2 plus 2 equals 4.
2 3 5 7 are the prime numbers.
2 times 2 equals 4.
2 times 100 equals 200.
mother gives birth to children.
we put cigarette butts in a ashtray.
we live only.
we should give peace a chance.
river amazon is the largest river.
friendship is important.
newyork is in us.
by asking questions do we learn answers.
water is heavie than air.
water is neccesary for survival.
water is wet at room temperature.
water is wet.
water is colourless.
down is the opposite of up.
arkansas is in the united states.
sushi is a japanese food.
maintenance of is a machine good.
biology is a science.
biology is a scientific field of study.
biology is a branch of science.
gasoline is toxic.
gasoline is highly flameable.
gasoline is a power source.
gasoline is a fuel.

gasoline is a petroleum product.
gasoline is flammable.
cook is a profession.
compassion is worthwhile.
playing is in traffic dangerous.
playing is a piano difficult.
playing is good for the mind.
playing is games fun.
playing is fun.
britney is spears good looking.
switzerland is in europe.
switzerland is is a european country.
lucky strike is a brand of cigarettes.
gigahertz is larger than megahertz.
princess is di still dead.
nbc is a tv network.
italian is spoken in italy.
italian is the language spoken in italy.
b is a letter of the english alphabet.
b is a letter.
anchorage is in alaska.
grilling is one method of cooking.
yen is is the currency of japan.
cyanide is poisonous to humans.
cyanide is poisonous.
humankind is mortal.
school is a good place to learn.
school is good to life.
school is good.
liquid is nitrogen cold.
melbourne is in australia.
marijuana is a drug.
marijuana is sometimes called pot.
racism is a global problem.
big ben is a clock.
big is the opposite of small.
christmas is in december.
christmas is a holiday.
is is true that fish live in the water.
is is true that lemons taste sour.
is is dark at night.
is is ok to eat meat.
pepsi is a cola.
pepsi is a type of soft drink.
pepsi is a type of drink.
chess is a board game.
chess is a boardgame.
chess is a mind game.
chess is a two player game.



APPENDIX H: UCF IRB LETTER

University of Central Florida Institutional Review Board
Office of Research & Commercialization
12201 Research Parkway, Suite 501
Orlando, Florida 32826-3246
Telephone: 407-823-2901 or 407-882-2276
www.research.ucf.edu/compliance/irb.html

Approval of Human Research

From: **UCF Institutional Review Board #1**
FWA00000351, IRB00001138

To: **Awrad Mohammed Ali**

Date: **August 14, 2018**

Dear Researcher:

On 08/14/2018 the IRB approved the following human participant research until 08/13/2019 inclusive:

Type of Review:	UCF Initial Review Submission Form Expedited Review
Project Title:	Performance assessment of a machine learning system that learns from casual conversation with humans (LCC)
Investigator:	Awrad Mohammed Ali
IRB Number:	SBE-18-14188
Funding Agency:	
Grant Title:	
Research ID:	N/A

The scientific merit of the research was considered during the IRB review. The Continuing Review Application must be submitted 30days prior to the expiration date for studies that were previously expedited, and 60 days prior to the expiration date for research that was previously reviewed at a convened meeting. Do not make changes to the study (i.e., protocol, methodology, consent form, personnel, site, etc.) before obtaining IRB approval. A Modification Form **cannot** be used to extend the approval period of a study. All forms may be completed and submitted online at https://iris.research.ucf.edu .

If continuing review approval is not granted before the expiration date of 08/13/2019, approval of this research expires on that date. When you have completed your research, please submit a Study Closure request in iRIS so that IRB records will be accurate.

Use of the approved, stamped consent document(s) is required. The new form supersedes all previous versions, which are now invalid for further use. Only approved investigators (or other approved key study personnel) may solicit consent for research participation. Participants or their representatives must receive a copy of the consent form(s).

All data, including signed consent forms if applicable, must be retained and secured per protocol for a minimum of five years (six if HIPAA applies) past the completion of this research. Any links to the identification of participants should be maintained and secured per protocol. Additional requirements may be imposed by your funding agency, your department, or other entities. Access to data is limited to authorized individuals listed as key study personnel.

In the conduct of this research, you are responsible to follow the requirements of the Investigator Manual.

This letter is signed by:

Signature applied by Gillian Morien on 08/14/2018 10:07:27 AM EDT

Designated Reviewer

LIST OF REFERENCES

[1] Chatterbot-machine learning, conversational dialog engine. https://chatterbot.readthedocs.io/en/stable.

[2] International workshop on semantic evaluation (semeval-2015). http://alt.qcri.org/semeval2015.

[3] Learning to classify text. https://www.nltk.org/book/ch06.html.

[4] Naïve bayes text classification. https://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html.

[5] Random facts. https://www.factslides.com.

[6] stanford-corenlp-python. https://github.com/dasmith/stanford-corenlp-python/.

[7] Textacy online documentation. https://media.readthedocs.org/pdf/textacy/latest/textacy.pdf.

[8] E. Abravanel and S. A. Ferguson. Observational learning and the use of retrieval information during the second and third years. *The Journal of genetic psychology*, 159(4):455–476, 1998.

[9] S. Alexandrova, M. Cakmak, K. Hsiao, and L. Takayama. Robot programming by demonstration with interactive action visualizations. *Robotics: science and systems*, 2014.

[10] A. M. Ali and A. J. Gonzalez. Machine learning from conversation. *Proceedings of the 29th Florida Artificial Intelligence Research Society (FLAIRS) Conference*, pages 2–7, 2017.

[11] J. R. Anderson. Act: A simple theory of complex cognition. *American Psychologist*, 51(4):355, 1996.

[12] J. R. Anderson, D. Bothell, J. M. Fincham, and J. Moon. The sequential structure of brain activation predicts skill. *Neuropsychologia*, 81:94–106, 2016.

[13] J. R. Anderson, M. Matessa, and C. Lebiere. Act-r: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, 12(4):439–462, 1997.

[14] C. Apté, F. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *Association for Computing Machinery (ACM) Transactions on Information Systems (TOIS)*, 12(3):233–251, 1994.

[15] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[16] R. Artstein, S. Gandhe, A. Leuski, and D. Traum. Field testing of an interactive question-answering character. *European Language Resources Association (ELRA) Workshop on Evaluation*, pages 36–40, 2008.

[17] C. G. Atkeson and S. Schaal. Robot learning from demonstration. *Proceedings of the International Conference on Machine Learning (ICML)*, volume 97, pages 12–20, 1997.

[18] R. E. Banchs, R. Jiang, S. Kim, A. Niswar, and K. H. Yeo. Aida: artificial intelligent dialogue agent. *Proceedings of the Special Interest Group in discourse and dialogue (SIGDIAL) Conference*, pages 145–147, 2013.

[19] R. E. Banchs and H. Li. Iris: A chat-oriented dialogue system based on the vector space model. *Proceedings of the Association for Computational Linguistics (ACL)*, pages 37–42, 2012.

[20] J. Bang, H. Noh, Y. Kim, and G. G. Lee. Example-based chat-oriented dialogue system with personalized long-term memory. *International Conference on Big Data and Smart Computing (BigComp)*, pages 238–243. IEEE, 2015.

[21] M. A. Bauer. Programming by examples. *Artificial Intelligence*, 12(1):1–21, 1979.

[22] D. C. Bentivegna and C. G. Atkeson. Learning from observation using primitives. *Robotics and Automation*, volume 2, pages 1988–1993. IEEE, 2001.

[23] D. C. Bentivegna, C. G. Atkeson, and G. Cheng. Learning tasks from observation and practice. *Robotics and Autonomous Systems*, 47(2):163–169, 2004.

[24] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus. Interpreting and executing recipes with a cooking robot. *Experimental Robotics*, pages 481–495, 2013.

[25] M. Brady. *Robot motion: Planning and control*. MIT press, 1982.

[26] L. Breslow, A. Harrison, and J. Trafton. Linguistic spatial gestures. *Proceedings of the 10th international conference on cognitive modeling*, pages 13–18. Citeseer, 2010.

[27] J. Brownlee. Classification and regression trees for machine learning. `https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/`.

[28] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé. Reinforcement learning from demonstration through shaping. *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3352–3358, 2015.

[29] R. Cantrell, J. Benton, K. Talamadupula, S. Kambhampati, P. Schermerhorn, and M. Scheutz. Tell me when and why to do it! run-time planner model updates via natural language instruction. *Human-Robot Interaction (HRI)*, pages 471–478. IEEE, 2012.

[30] D. Carruth, B. Robbins, M. Thomas, A. Morais, M. Letherwood, and K. Nebel. Symbolic model of perception in dynamic 3d environments. Technical report, MISSISSIPPI STATE UNIV MISSISSIPPI STATE CENTER FOR ADVANCED VEHICULAR SYSTEMS, 2006.

[31] A. Chacón, S. Marco-Sola, A. Espinosa, P. Ribeca, and J. C. Moure. Thread-cooperative, bit-parallel computation of levenshtein distance on gpu. *Proceedings of the 28th of Inter-*

national Conference on Supercomputing, pages 103–112. Association for Computing Machinery (ACM), 2014.

[32] H. L. Chieu and H. T. Ng. A maximum entropy approach to information extraction from semi-structured and free text. *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2002:786–791, 2002.

[33] K. J. Cios, R. W. Swiniarski, W. Pedrycz, and L. A. Kurgan. Unsupervised learning: clustering. *Data Mining*, pages 257–288, 2007.

[34] A. Coates, P. Abbeel, and A. Y. Ng. Learning for control from multiple demonstrations. *Proceedings of the 25th international conference on Machine learning*, pages 144–151. Association for Computing Machinery (ACM), 2008.

[35] A. Cohen. Fuzzywuzzy: Fuzzy string matching in python. `https://www.geeksforgeeks.org/fuzzywuzzy-python-library/`, 2015.

[36] W. Dai, G.-R. Xue, Q. Yang, and Y. Yu. Transferring naïve bayes classifiers for text classification. *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, volume 7, pages 540–545, 2007.

[37] J. A. Danowski. Wordi version 3.0: Semantic network analysis software. University of Illinois at Chicago, 2013.

[38] T. G. Dietterich. Ensemble learning. *The handbook of brain theory and neural networks*, 2:110–125, 2002.

[39] L. F. D'Haro, S. Kim, K. H. Yeo, R. Jiang, A. I. Niculescu, R. E. Banchs, and H. Li. Clara: a multifunctional virtual agent for conference support and touristic information. *Natural Language Dialog Systems and Intelligent Assistants*, pages 233–239, 2015.

[40] M. Elvir, A. J. Gonzalez, C. Walls, and B. Wilder. Remembering a conversation–a conversational memory architecture for embodied conversational agents. *Journal of Intelligent Systems*, 25, 2016.

[41] H. K. Fernlund, A. J. Gonzalez, M. Georgiopoulos, and R. F. DeMara. Learning tactical human behavior through observation of human performance. *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(1):128–140, 2006.

[42] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[43] D. H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2(2):139–172, 1987.

[44] T. Fitzgerald and A. Goel. A case-based approach to imitation learning in robotic agents. *Case-Based Reasoning Workshop on Case-Based Agents*, 2014.

[45] M. W. Floyd, M. V. Bicakci, and B. Esfandiari. Case-based learning by observation in robotics using a dynamic case representation. *Proceedings of the Florida Artificial Intelligence Research Society (FLAIRS) Conference*, 2012.

[46] M. W. Floyd, B. Esfandiari, and K. Lam. A case-based reasoning approach to imitating robocup players. *Proceedings of the Florida Artificial Intelligence Research Society (FLAIRS) Conference*, pages 251–256, 2008.

[47] M. W. Floyd, J. Turner, and D. W. Aha. Using deep learning to automate feature modeling in learning by observation: A preliminary study. *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2017.

[48] M. Forbes, M. J.-Y. Chung, M. Cakmak, and R. P. Rao. Robot programming by demonstration with crowdsourced action fixes. *Proceedings of the Second of the Association for*

the Advancement of Artificial Intelligence (AAAI) Conference on Human Computation and Crowdsourcing, pages 67–76, 2014.

[49] K. Ganesan, C. Zhai, and J. Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. *Proceedings of the 23rd international conference on computational linguistics*, pages 340–348. Association for Computational Linguistics, 2010.

[50] D. Goldwasser and D. Roth. Learning from natural instructions. *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[51] J. Grizou, I. Iturrate, L. Montesano, P.-Y. Oudeyer, and M. Lopes. Interactive learning from unlabeled instructions. *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.

[52] J. Grizou, M. Lopes, and P.-Y. Oudeyer. Robot learning simultaneously a task and how to interpret human instructions. *IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pages 1–8. IEEE, 2013.

[53] D. H. Grollman and O. C. Jenkins. Learning robot soccer skills from demonstration. *Development and Learning, 2007. ICDL 2007. IEEE 6th International Conference on*, pages 276–281. IEEE, 2007.

[54] S. Guadarrama, L. Riano, D. Golland, D. Go, Y. Jia, D. Klein, P. Abbeel, and T. Darrell. Grounding spatial relations for human-robot interaction. *International Conference on Intelligent Robots and Systems (IROS)*, pages 1640–1647. IEEE, 2013.

[55] M. Halbrügge and N. Russwinkel. The sum of two models: how a composite model explains unexpected user behavior in a dual-task scenario. *Proceedings of the 14th international conference on cognitive modeling*, pages 137–143, 2016.

[56] T. Hastie, R. Tibshirani, and J. Friedman. Overview of supervised learning. *The Elements of Statistical Learning*, pages 9–41. 2009.

[57] R. Higashinaka, K. Imamura, T. Meguro, C. Miyazaki, N. Kobayashi, H. Sugiyama, T. Hirano, T. Makino, and Y. Matsuo. Towards an open domain conversational system fully based on natural language processing. *Proceedings of the 25th International Conference on Computational Linguistics*, pages 928–939, 2014.

[58] R. Hill. Modeling perceptual attention in virtual humans. *Proceedings of the 8th Conference on Computer Generated Forces and Behavioral Representation*, pages 563–573. Citeseer, 1999.

[59] V. C. Hung and A. J. Gonzalez. Context-centric speech-based human–computer interaction. *International Journal of Intelligent Systems*, 28(10):1010–1037, 2013.

[60] K. Ikeuchi, Z. Yan, Z. Ma, Y. Sato, M. Nakamura, and S. Kudoh. Describing upper body motions based on the labanotation for learning-from-observation robots. *International Journal of Computer Vision*, 126(12):1415–1429, 2016.

[61] T. Joachims. *Learning to classify text using support vector machines: Methods, theory and algorithms*. Kluwer Academic Publishers, 2002.

[62] B. E. John and Y. Lallement. Strategy use while learning to perform the kanfer-ackerman air traffic controller task. *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*, pages 337–342, 1997.

[63] P. Khandelwal, S. Zhang, J. Sinapov, M. Leonetti, J. Thomason, F. Yang, I. Gori, M. Svetlik, P. Khante, V. Lifschitz, J. K. Aggarwal, R. Mooney, and P. Stone. Bwibots: A platform for bridging the gap between ai and human-robot interaction research. *The International Journal of Robotics Research*, 36(5-7):635–659, 2017.

[64] D. E. Kieras and D. E. Meyer. An overview of the epic architecture for cognition and performance with application to human-computer interaction. *Human–Computer Interaction*, 12(4):391–438, 1997.

[65] S. Kim, C. Lee, S. Jung, and G. G. Lee. A spoken dialogue system for electronic program guide information access. *Proceedings of Robot and Human interactive Communication*, pages 178–181. IEEE, 2007.

[66] Y. Kim, J. Bang, J. Choi, S. Ryu, S. Koo, and G. G. Lee. Acquisition and use of long-term memory for personalized dialog systems. *Workshop on Multimodal Analyses enabling Artificial Agents in Human-Machine Interaction*, pages 78–87. 2014.

[67] K. R. Koedinger and V. Aleven. Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, 19(3):239–264, 2007.

[68] I. Kruijff-Korbayová, G. Athanasopoulos, A. Beck, P. Cosi, H. Cuayáhuitl, T. Dekens, V. Enescu, A. Hiolle, B. Kiefer, and H. Sahli. An event-based conversational system for the nao robot. *Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems*, pages 125–132, 2011.

[69] G. Kuhlmann, P. Stone, R. Mooney, and J. Shavlik. Guiding a reinforcement learner with natural language advice: Initial results in robocup soccer. *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) workshop on supervisory control of learning and adaptive systems*, 2004.

[70] J. E. Laird. Extending the soar cognitive architecture. *Frontiers in Artificial Intelligence and Applications*, 171:224, 2008.

[71] J. E. Laird. *The SOAR cognitive architecture*. MIT press, 2012.

[72] J. E. Laird, A. Newell, and P. S. Rosenbloom. Soar: An architecture for general intelligence. *Artificial intelligence*, 33(1):1–64, 1987.

[73] J. E. Laird and E. Nielsen. Coordinated behavior of computer generated forces in tacair-soar. *AD-A280 063*, 1001:57, 1994.

[74] J. E. Laird and P. S. Rosenbloom. The anatomy of a general learning mechanism. 1986.

[75] S. Lee. Toward continual learning for conversational agents. *CoRR 1712.09943*, 2017.

[76] S. Lee. Accumulating conversational skills using continual learning. 2018.

[77] J. F. Lehman, R. L. Lewis, and A. Newell. Integrating knowledge sources in language comprehension. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, pages 461–466, 1991.

[78] A. Leuski, R. Patel, D. Traum, and B. Kennedy. Building effective question answering characters. *Proceedings of the 7th Special Interest Group in discourse and dialogue (SIGDIAL) Workshop on Discourse and Dialogue*, pages 18–27. Association for Computational Linguistics, 2009.

[79] J. Li, A. H. Miller, S. Chopra, M. Ranzato, and J. Weston. Learning through dialogue interactions. 2016.

[80] T. Li, Y. Li, F. Chen, and B. A. Myers. Programming iot devices by demonstration using mobile apps. *Proceedings of the International Symposium on End User Development*, pages 3–17, 2017.

[81] P. Lison. Model-based bayesian reinforcement learning for dialogue management. *Association for Computing Machinery (ACM)*, 2013.

[82] D. Lonsdale, M. Hutchison, T. Richards, and W. Taysom. Lg-soar: Parsing for information. *Talk given at the 21st North American SOAR workshop, Ann Arbor/MI*, 2001.

[83] D. Lonsdale, C. Tustison, C. Parker, and D. W. Embley. Formulating queries for assessing clinical trial eligibility. *International Conference on Application of Natural Language to Information Systems*, pages 82–93. Springer, 2006.

[84] S. Marsella, M. Tambe, J. Adibi, Y. Al-Onaizan, G. A. Kaminka, and I. Muslea. Experiences acquired in the design of robocup teams: A comparison of two fielded teams. *Autonomous Agents and Multi-Agent Systems*, 4(1-2):115–129, 2001.

[85] M. L. Mauldin. Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition. *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*, volume 94, pages 16–21, 1994.

[86] A. McGregor, M. Hall, P. Lorier, and J. Brunskill. Flow clustering using machine learning techniques. *Passive and Active Network Measurement*, pages 205–214, 2004.

[87] M. L. McNeal and D. Newyear. Introducing chatbots in libraries. *Library technology reports*, 49(8):5–10, 2013.

[88] C. Meriçli, S. D. Klee, J. Paparian, and M. Veloso. An interactive approach for situated task specification through verbal instructions. *Proceedings of the international conference on Autonomous agents and multi-agent systems*, pages 1069–1076. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[89] R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, volume 6, pages 775–780, 2006.

[90] D. K. Misra, J. Sung, K. Lee, and A. Saxena. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, 35(1-3):281–300, 2016.

[91] S. Mohan and J. E. Laird. Learning goal-oriented hierarchical tasks from situated interactive instruction. *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*, pages 387–394, 2014.

[92] J. Montana and A. Gonzalez. Towards a unified framework for learning from observation. *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI) on Agents Learning Interactively from Human Teachers*, 2011.

[93] F. Morbini, D. DeVault, K. Sagae, J. Gerten, A. Nazarian, and D. Traum. Flores: a forward looking, reward seeking, dialogue manager. *4th International Workshop on Natural Interaction with Robots, Knowbots and Smartphones*, pages 313–325. 2014.

[94] K. P. Murphy. Naïve bayes classifiers. *University of British Columbia*, 18, 2006.

[95] A. Nagpal and S. P. Panda. Career path suggestion using string matching and decision trees. *International Journal of Computer Applications*, 117, 2015.

[96] S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, H. Hirukawa, and K. Ikeuchi. Learning from observation paradigm: Leg task models for enabling a biped humanoid robot to imitate human dances. *The International Journal of Robotics Research*, 26(8):829–844, 2007.

[97] A. I. Niculescu, R. Jiang, S. Kim, K. H. Yeo, L. F. D'Haro, A. Niswar, and R. E. Banchs. Sara: singapore"s automated responsive assistant, a multimodal dialogue system for touristic information. *Proceedings of the 11th International Mobile Web Information Systems Conference*, pages 153–164. 2014.

[98] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. *Proceedings of International Joint Conference on Artificial Intelligence IJCAI-99 workshop on machine learning for information filtering*, volume 1, pages 61–67, 1999.

[99] L. Nio, S. Sakti, G. Neubig, T. Toda, and S. Nakamura. Combination of example-based and smt-based approaches in a chat-oriented dialog system. *Proceedings of International Conference on Emerging Infectious Diseases (ICE-ID)*, 2013.

[100] S. Ontanón, K. Mishra, N. Sugandh, and A. Ram. Online case-based planning. *Computational Intelligence*, 26(1):84–119, 2010.

[101] S. Ontañón, J. L. Montaña, and A. J. Gonzalez. A dynamic-bayesian network framework for modeling and evaluating learning from observation. *Expert Systems with Applications*, 41(11):5212–5226, 2014.

[102] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up: sentiment classification using machine learning techniques. *Proceedings of the Association for Computational Linguistics (ACL) Conference on Empirical methods in natural language processing*, volume 10, pages 79–86, 2002.

[103] J. Planells, L. F. Hurtado, E. Segarra, and E. Sanchis. A multi-domain dialog system to integrate heterogeneous spoken dialog systems. *Proceedings of International Speech Communication Association (INTERSPEECH)*, pages 1891–1895, 2013.

[104] D. Pomerleau and T. Jochem. Rapidly adapting machine vision for automated vehicle steering. *IEEE expert*, 11(2):19–27, 1996.

[105] F. J. Provost and T. Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. *Knowledge Discovery and Data Mining (KDD)*, volume 97, pages 43–48, 1997.

[106] S. Ramaswamy. Multiclass text classification a decision tree based svm approach. *CS294 Practical Machine Learning Project*, 2006.

[107] J. Ramos. Using tf-idf to determine word relevance in document queries. *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142, 2003.

[108] D. Reitter and C. Lebiere. How groups develop a specialized domain vocabulary: A cognitive multi-agent model. *Cognitive Systems Research*, 12(2):175–185, 2011.

[109] A. Ritter, C. Cherry, and W. B. Dolan. Data-driven response generation in social media. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 583–593. Association for Computational Linguistics, 2011.

[110] D. Ruta and B. Gabrys. Classifier selection for majority voting. *Information fusion*, 6(1):63–81, 2005.

[111] P. E. Rybski, K. Yoon, J. Stolarz, and M. M. Veloso. Interactive robot task training through dialog and demonstration. *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 49–56. Association for Computing Machinery (ACM), 2007.

[112] C. Sammut, S. Hurst, D. Kedzier, and D. Michie. Learning to fly. *Proceedings of the 9th International Workshop on Machine Learning*, pages 385–393, 2014.

[113] B. Schloss, M. D. Chang, A. Vempaty, A. Acharya, R. Kokku, L. Wilde, and N. Mukhi. Research questions to support conversational learning in the era of ubiquitous, mobile agents. International Society of the Learning Sciences, Inc.[ISLS]., 2018.

[114] B. A. Shawar and E. S. Atwell. Using corpora in machine-learning chatbot systems. *International Iournal of Corpus Linguistics*, 10(4):489–516, 2005.

[115] T. Shibata, Y. Egashira, and S. Kurohashi. Chat-like conversational system based on selection of reply generating module with reinforcement learning. *Proceedings of the 5th International Workshop Series on Spoken Dialog Systems*, pages 124–129, 2014.

[116] T. A. Sidani and A. J. Gonzalez. A framework for learning implicit expert knowledge through observation. *Transactions of the Society for Computer Simulation*, 17(2):54–72, 2000.

[117] G. Stein and A. J. Gonzalez. Building high-performing human-like tactical agents through observation and experience. *Proceedings of the Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(3):792–804, 2011.

[118] H. Sugiyama, T. Meguro, R. Higashinaka, and Y. Minami. Open-domain utterance generation for conversational dialogue systems using web-scale dependency structures. *Proceedings of the Special Interest Group in discourse and dialogue (SIGDIAL) Conference*, pages 334–338, 2013.

[119] M. A. Sultan, S. Bethard, and T. Sumner. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association for Computational Linguistics*, 2:219–230, 2014.

[120] M. A. Sultan, S. Bethard, and T. Sumner. Dls @ cu: Sentence similarity from word alignment and semantic vector composition. *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 148–153, 2015.

[121] R. Sun. The clarion cognitive architecture: Extending cognitive modeling to social simulation. *Cognition and multi-agent interaction*, pages 79–99, 2006.

[122] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[123] L. Torrey, T. Walker, J. Shavlik, and R. Maclin. Using advice to transfer knowledge acquired in one reinforcement learning task to another. *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pages 412–424, 2005.

[124] D. Traum, A. Roque, A. Leuski, P. Georgiou, J. Gerten, B. Martinovski, S. Narayanan, S. Robinson, and A. Vaswani. Hassan: A virtual human for tactical questioning. *Proceedings of the Special Interest Group in discourse and dialogue (SIGDIAL)*, volume 2008, 2007.

[125] A. M. Turing. Computing machinery and intelligence. *Mind*, pages 433–460, 1950.

[126] M. van Lent and J. Laird. Learning by observation in complex domains. *Workshop on Knowledge Acquisition, Modeling, and Management*, 1001:18–23, 1998.

[127] M. van Lent and J. E. Laird. Learning procedural knowledge through observation. *Proceedings of the First International Conference on Knowledge Capture*, pages 179–186. Association for Computing Machinery (ACM), 2001.

[128] S. Volkova, P. Choudhury, C. Quirk, B. Dolan, and L. S. Zettlemoyer. Lightly supervised learning of procedural dialog systems. *Proceedings of the Association for Computational Linguistics (ACL)*, volume 1, pages 1669–1679, 2013.

[129] R. S. Wallace. *The anatomy of ALICE*. Artificial Intelligence Foundation Inc., 2004.

[130] X. Wang. Learning by observation and practice: An incremental approach for planning operator acquisition. *Proceedings of the International Conference on Machine Learning (ICML)*, pages 549–557, 1995.

[131] B. Watson. An overview of the deep-qa project. *IBM Corporation*. AI Magazine, 2010.

[132] J. Weizenbaum. Eliza a computer program for the study of natural language communication between man and machine. *Communications of the Association for Computing Machinery (ACM)*, 9(1):36–45, 1966.

[133] J. E. Weston. Dialog-based language learning. *Advances in Neural Information Processing Systems*, pages 829–837, 2016.

[134] J. Woo and N. Kubota. Conversation system based on computational intelligence for robot partner using smart phone. *Systems, Man, and Cybernetics (SMC)*, pages 2927–2932. IEEE, 2013.

[135] T. Yoshimura. Casual conversation technology achieving natural dialog with computers. volume 15. Nippon Telegraph and Telephone DO COmmunications over the MObile network (NTT DOCOMO) Technical Journal, 2014.

[136] K. Yoshino, S. Mori, and T. Kawahara. Spoken dialogue system based on information extraction using similarity of predicate argument structures. *Proceedings of the Special Interest Group in discourse and dialogue (SIGDIAL) Conference*, pages 59–66. Association for Computational Linguistics, 2011.

[137] J. Young and N. Hawes. Learning by observation using qualitative spatial relations. *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, pages 745–751, 2015.

[138] L. Yujian and L. Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007.

[139] H. Zhang, H. Yu, and W. Xu. Listen, interact and talk: Learning to speak via interaction. *In NIPS Workshop on Visually-Grounded Interaction and Language*. Cornell University, 2017.