2020-07-31

ClarQue: Chatbot Recognizing Ambiguity in the Conversation and Asking Clarifying Questions

Shreeya Himanshu Mody
Brigham Young University

ClarQue: Chatbot Recognizing Ambiguity in the Conversation and Asking Clarifying Questions

Shreeya Himanshu Mody

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

David Wingate, Chair
Kevin Seppi
Jacob Crandall

Department of Computer Science

Brigham Young University

ABSTRACT

ClarQue: Chatbot Recognizing Ambiguity in the Conversation and Asking Clarifying Questions

Shreeya Himanshu Mody
Department of Computer Science, BYU
Master of Science

Recognizing when we need more information and asking clarifying questions are integral to communication in our day to day life. It helps us complete our mental model of the world and eliminate confusion. Chatbots need this technique to meaningfully collaborate with humans. We have investigated a process to generate an automated system that mimics human communication behavior using knowledge graphs, weights, an ambiguity test, and a response generator. It can take input dialog text and based on the chatbot's knowledge about the world and the user it can decide if it has enough information or if it requires more. Based on that decision, the chatbot generates a dialog output text which can be an answer if a question is asked, a statement if there are no doubts or if there is any ambiguity, it generates a clarifying question. The effectiveness of these features has been backed up by an empirical study which suggests that they are very useful in a chatbot not only for crucial information retrial but also for keeping the flow and context of the conversation intact.

Keywords: knowledge graph, Stanford NLP annotators, chatbot, clarifying questions, ambiguity

ACKNOWLEDGMENTS

I would like to start by sincerely thanking my advisor, Dr. David Wingate, whose insights, guidance, and faith have been crucial to me. His consistent support and encouragement have been extremely important in finishing this thesis. Additionally, I would also like to thank Dr. Kevin Seppi whose inputs and attention to detail have been important for the progress of this thesis. Thanks to Dr. Jacob Crandall for his insightful critiques and ever-cheerful smile. My sincere thanks to the CS department staff for always being there to help.

I cannot thank my family enough for their constant moral support and sacrifices. This thesis would not have been possible without their persistent faith in me. Thank you, everyone!

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction and Motivation

Consider this snippet of dialogue between two people:

P1: Hello! How was your day?

P2: It was good! I came home early from work.

P1: Oh that is wonderful!

P2: I am going out to celebrate with my friends.

P1: Is it your birthday?

P2: I got a promotion at work.

P1: Congratulations! Where are you celebrating?

P2: At Olive Garden.

P1: Ah! I love that place. Congratulations and enjoy!

When two people are having a conversation, there is an assumption that common knowledge exists and both the participants are aware of it. We have multiple ways of transferring our knowledge or mental picture of the world to the other person, one of which is natural language speech so that they have the context of what we are conveying. In the above conversation between Person 1 and Person 2, the shared knowledge is that it is usually good to come home early, therefore, P1 responds that it is wonderful. However, sometimes we forget to mention important details like P2 mentions that he/she is celebrating with friends but misses to inform the cause for the celebration. P1 knows from common knowledge that we celebrate when something remarkable happens or when a milestone is achieved and starts thinking of such events that can be celebrated with friends. There are a few occasions that

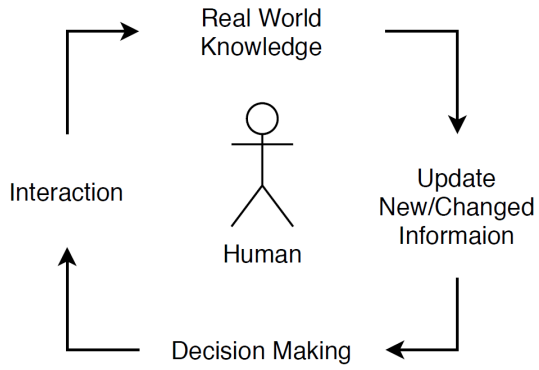


Figure 1.1: We start building mental model of the real world by interacting with others and our surroundings. We keep the model updated with newer or changed information as we receive it and based on this model we make decisions about how to interact with others.

can be celebrated with friends, so P1 uses his/her mental probabilistic model to figure out which event P2 could be celebrating with friends and asks if it is his/her birthday. On the other hand, it might not always be possible to find one answer as there might be too many correct answers and in such scenarios asking a clarifying question directly seems more fruitful. In the conversation above, since there are many places where P2 could be celebrating his/her promotion, instead of guessing, P1 chooses to directly ask where the celebration is.

From the conversation above and Figure 1.1, we see that humans build a mental model of the real world while interacting with others as well as their surrounding environment. Craik [5] and Norman [18] suggest that this model is an explanation of our thought process of how things work in the real world. In this mental model, we have various connections between different words, for example, the word accident is related to bad-painful-hospital, couple is related to two-pair-relationship, ice is related to cold-water-snow-refrigerator, and so on. We use this knowledge from our mental model that has been acquired over the years to interact with others and constantly keep updating it as we find new or changed or missing pieces of information. We carefully analyze the words and the context that have been communicated to us and try to fit them into our existing model. We tend to use Bayesian inference [9] to "invert" the model of the speaker, which means that we assume that the speaker chose his/her words optimally to mean something specific. Based on the words used, we can

logically infer what the speaker exactly means. On the other hand, sometimes we might not clearly understand the meaning of what has been told to us, or we might come across something that we do not have enough information about to communicate further. In such scenarios, we usually ask each other questions about the missing pieces of information to eliminate ambiguity and update the information in our mental model. Once we obtain this new information, we keep the flow of the conversation going.

A chatbot with such reasoning and deduction capabilities would be a great asset in every field. There are many chatbots that are very good at mimicking the human behavior of communication used for various singular tasks like booking tickets, making reservations at restaurants, hotels, theaters, ordering food, checking out books, getting health tips, getting tutored for classes, and many more [25]. Their presence is spread over almost all the major aspects of our lives like medical, educational, recreational, financial, social, and spiritual. Their need will only increase with time to improve customer service and they will be required to respond to natural language excellently by multi-tasking (i.e., have relevant and appropriate responses, have a good flow of conversation, and switch to a different topic as required).

A lot of these chatbots care about singular tasks and only need information that is relevant to it. They find these pieces of information by asking questions to the users. In most cases, these responses (including questions) are hardcoded [6] into the system which means that the chatbot is not learning to deduce reasoning from what the user has already mentioned previously nor is it learning to ask questions. They look for the keywords in the user input to check which questions from the list of questions have not yet been answered. There are workflows where the questions like "did you mean *xyz*?" (Google Search) are asked when the input of the user does not closely match with anything in the knowledge base. Even though this question helps the chatbot in finding better results for the query, the drawback of this method is that *xyz* might be completely irrelevant to the topic that the user is talking about. There might hardly be any intersection between what the generated question is and what the user wants to communicate. There are also comprehension questions generated from

a given input paragraph or line [11]. The problem with this method of question generation is that their answers are already present in the given input. It does not generate questions whose answers can add newer information to the knowledge base.

In the era where artificial personal assistants are constantly used by so many people to complete numerous day to day and other tasks, there are many who care and depend on the competency of chatbots. The above-mentioned skill will be useful for companies that use chatbots for customer service. Understanding the natural language and asking good questions can make them more effective in interpreting the specific needs of the customer. It will be useful for product suggestion businesses, where customers do not know what they want to buy specifically, so the chatbot keeps asking various questions to make a model based on the user's interest. Conversational Artificial Intelligence chatbots like Alexa and Siri might be able to enhance their assistance for us if they could ask us a good clarifying question that would remove uncertainty. It will be immensely useful for social media platforms where people, for example, prefer to tweet and other platforms where people prefer to chat rather than call a number to request service or give feedback. Chatbots are also used for sales pitch i.e., users find out about interesting sales, ideas, and merchandise, and only the interested users advance to a human assistant. This application demonstrates how apart from helping the customers, they are also able to help businesses by filtering potential users that will buy their services. These are the people who are concerned about the competency and effectiveness of the chatbots so that they can earnestly help the users and themselves without needing a lot of employee involvement. A lot of research and effort has been invested in this area of developing chatbots because it not only saves time and money but also helps in building a bond of trust with the customers. They can deploy multiple instances of these chatbots to help a large magnitude of customers at the same time. Good service is always attractive to all consumers and this technique can be used to market their products and help them grow their business.

The difficulty we are trying to resolve is that the chatbots must be able to ask questions when they have doubts and difficulty in inferring what the user means or when they have incomplete knowledge about the new input. We need this skill in the chatbots so that they can facilitate the users more efficiently. In case an ambiguity or confusion arises while chatting, we desire that the chatbot can ask intelligent questions by using the logical reasoning skills that will help them in receiving sufficient knowledge to eradicate uncertainty.

In this work, we have investigated a process that enables the chatbot to decide on whether it has the required information or whether it needs more, and in the latter case, ask a good clarification question to the user. Analysis of the results received from the empirical study, suggests that these features are a very important part of a conversation and are required for information retrieval and to keep the conversation flowing.

Chapter 2

Related Work

One way of solving a natural language processing problem is to use a neural network to read and store the data in a latent space, manipulate it, and then retrieve newly generated data back in a human-readable format by using the autoencoders [24]. There are a couple of ways to store knowledge or data: First way is turning the data into vector representation like one-hot encoding or WordToVec encoding [17] or Glove encoding [20] in a latent space. Socher et al. [26] created a system that transfers a large amount of data to an embedded space using vector representations. This allows the natural arrangement of the data in a latent space and it is easier to manipulate the data in this space than it is in its original/natural form.

Another way is to store the data in a knowledge graph that contains nodes and edges. The nodes are various words which can be people, places, things, actions, abstractions, describing words, etc., and they are connected to other nodes by edges that represent a relationship between them. Wikidata [30], Conceptnet [27], and DBpedia [13] are some of examples of open source pre-built knowledge graphs. In a knowledge graph, some nodes are closer to each other compared to others which indicates that those nodes are more closely tied or connected in natural language than the others. The edges of collates (co-occurring words), synonyms, antonyms, or words that are a part of a larger group are shorter than the edges between words that have fewer shared properties. Figure 2.1 shows that the nodes "planet" and "earth" are closely connected than the nodes "space" and "insects" which is a reflection of how related these words are in natural language. Given the natural language

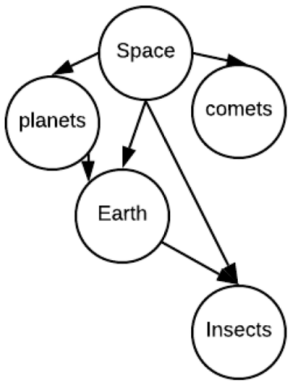


Figure 2.1: The edges between objects like planets-earth-comets are shorter because they are a part of a larger group "space", while the edge between the objects insects-space is larger as there are fewer common properties between them.

processing tools, it is convenient to work with words in natural form, hence, we will use this method to store the data for our chatbot.

Abundant prior work has been done in the field of question generation. Heilman et al. [10], Rus et al. [23], Olney et al. [19] focused on generating comprehension questions that tend to generate questions based on the input which can be a sentence or a paragraph. However, the questions generated with these methods are not clarifying questions, rather they are made so that their answers lie within the paragraph. These questions are useful for educational purposes to generate comprehension test questions. Rao et al. [21] used the data from Stack-Exchange and ten other similar posts that have candidate questions and answers in them and all the questions and answers are ranked by users based on their importance to that post. They implemented a model that generates candidate questions from the database and each question has a utility connected to it which is a measure of how complete the post would be if they managed to acquire a particular answer to the question. Their neural network is inspired by the expected value of perfect information (EVPI) [3] that is a measurement of how important would acquiring some information be. In the end, the question with the highest utility is asked to the user. We will adapt the idea of getting scores for most important questions from this method that would help our chatbot in finding answers from the users.

Given a set of input keywords, Zheng et al. [32] implemented "keywords to question model" (K2Q) that generates a list of candidate questions and refinement words that help the user to ask a better question. Ding et al. [7] trained the neural network to generate a keyword query from the user's natural language question. Then they use a neural network to generate keyword-question pairs when a huge dataset of questions is fed as input. Lastly, they train the neural machine translation model with keywords to generate candidate questions that the user can choose from. Our chatbot will generate the keywords that need to be present in the response and use a natural language generation template to produce a natural language sentence/answer/question.

Chapter 3

ClarQue: Chatbot That Deduces When and What Clarifying Questions to Ask

Chatbots can reply in a variety of ways such as with a sentence, with an answer to a question, with a question, etc. Apart from some of these basic features, the main aim of ClarQue is to determine if there is uncertainty or doubts in the given input and if there is uncertainty then ask clarifying questions regarding it to the user. This chapter talks about each segment of ClarQue in detail that starts from taking input and ends at generating a response.

3.1 Architecture

ClarQue is an integration of multiple separately functioning components. When these components all work together, the chatbot can generate a relevant response to the input. The architecture that demonstrates an entire cycle of ClarQue from the input to the response generation is shown in Figure 3.1. Here are all the components of ClarQue:

- The cycle starts when ClarQue gets prior knowledge (only the very first time before interacting with a user) and user input in the text format. It uses natural language processing tools to annotate the words with useful tags like parts of speech and named entity. Other tools also help in building a dependency tree that represents the dependency of a word on the rest of the words.
- There are two custom knowledge graphs, first, the prior knowledge graph that contains prior facts provided right before the chatbot is first deployed, and the second is the user knowledge graph which contains the knowledge that is only relevant to the user. Weights

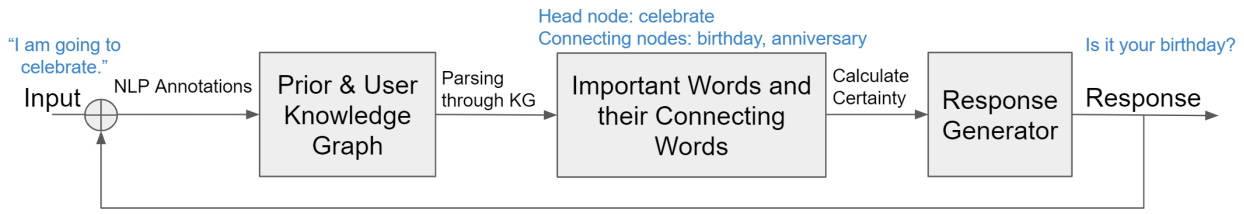


Figure 3.1: ClarQue's workflow that includes all the major components required to generate a response: Prior Knowledge and user input, NLP annotation tools, generating and updating knowledge graphs, finding important nodes and questions, ambiguity test, and response generator.

of the nodes are a representation of how frequently they appear in the knowledge graph and the user inputs. The tagged words are added to the custom knowledge graph as nodes along with custom edges.

- Prior knowledge is constant and does not get updated as the communication progresses. On the other hand, the user knowledge is updated with every new input if needed along with the weights.
- ClarQue searches for important words in the input sentence and finds other words that are connected to it through the edges in the knowledge graphs.
- It uses the weights of these connected words and the edges found in the knowledge graph for the ambiguity test to decide if it is facing uncertainty or not. In this way, it can decide how it needs to respond to the user. If the chatbot decides that it has enough information based on these tests, it will reply with an answer or a statement. If ClarQue faces uncertainty, it will respond with a question.
- Once the decision is made on what kind of response it wants to produce, ClarQue uses a Response Generator to compose responses that are close to natural language.
- Some times it is necessary to use the chatbot's previous response and the new user input together to update the knowledge, e.g., when the chatbot's previous response was a question and the user's current input is an answer, it will use both the responses to save the new information.

3.2 Methodology

Based on the architecture, the workflow of ClarQue is divided into five steps that are (i) annotating words of the input sentence, (ii) generating and updating knowledge and weights, (iii) parsing the knowledge to find important and relevant nodes and edges, (iv) checking for ambiguity, and (v) generating a response. A bit of data preprocessing is needed before the chatbot starts its workflow. The input is checked for any spelling mistakes and autocorrect is used to rectify them. Punctuation marks are crucial in forming dependencies between words in a sentence so they are kept in place for the annotators so they can tag the words correctly. This section explains all the components of the workflow as well as how they work in synchronization together in detail.

3.2.1 Annotation

One of the ways for the users to interact with a chatbot is through natural language text, therefore, the natural language must be recognized and understood by the machines so that they can interact with us efficiently. To understand the context and the meaning of the natural language, humans first recognize the parts of speech, the dependency structure, etc. between the words. Similarly, in natural language processing (NLP), the text annotation tools help in tagging the words meaningfully in the sentence. Moreover, this step helps in making the words usable for machines so that they can understand the structure and logic of the string of words like we do.

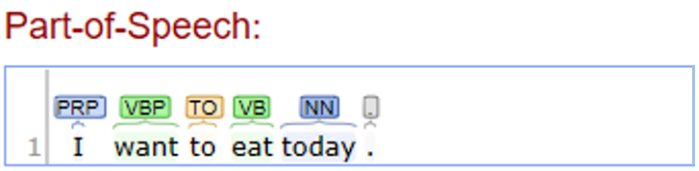
To annotate the input, we used the Stanford annotators [14] like the Part-Of-Speech tagger [29], [28], the Named Entity Recognizer [8], the Basic and Advanced Dependency parser [4], the Constituency parser, and the lemmatizer. As soon as the chatbot receives an input, it is passed to these annotators so that they can build an informative structure that helps ClarQue in understanding the meaning and the context of the sentence.

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	TO	"to"	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential 'there'	*there*	VB	verb base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb gerund	*eating*
JJ	adjective	*yellow*	VBN	verb past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, sing.	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	*#*
PDT	predeterminer	*all, both*	"	left quote	' or "
POS	possessive ending	*'s*	"	right quote	' or "
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one's*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

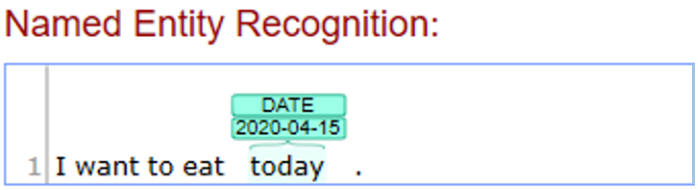
Figure 3.2: The Penn Treebank Project tags for parts of speech used by the Stanford POS tagger.

POS Tagger

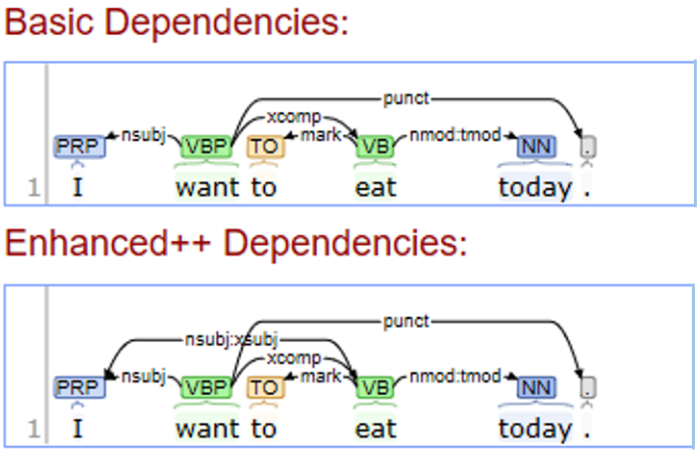
The Part-of-Speech (POS) annotator labels each word in a sentence according to their parts of speech i.e. adjective, determiner, noun, verb, etc. It is useful for building parse trees and understanding the role of a word in a sentence. The Stanford POS tagger follows the Penn Treebank Project [15] tags for annotation as shown in Figure 3.2. When ClarQue receives an input sentence from the user such as **User:** *I want to eat today*, the POS tagger will annotate the words as shown in Figure 3.3a. These tags signify which parts of speech each word belongs to that helps the chatbot in understanding the relationship between the words.



(a) Stanford POS tagger: annotating the input sentence with parts of speech.



(b) Stanford NER tagger: annotating the input sentence with named entities.

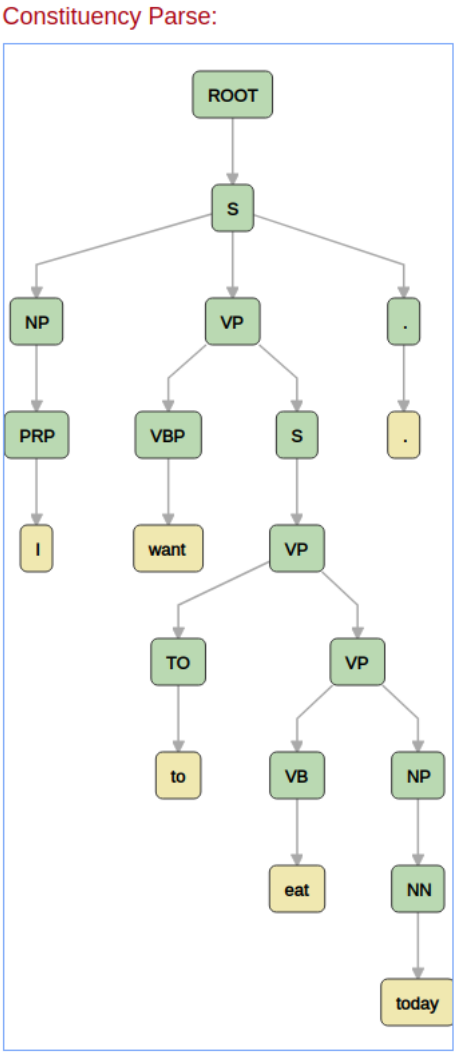


(c) Stanford Basic and Advanced Dependency parser: annotating the dependencies between words of the input sentence.

Figure 3.3: Various Stanford taggers annotating the input sentence

NER Tagger

The Named Entity Recognition (NER) annotator labels the words in a sentence that might be named entities. We used the Stanford NER tagger that can identify these seven classes in the input: locations, persons, organizations, times, money, percent, dates. This annotator is immensely helpful in information retrieval tasks when the chatbot is supposed to find an answer to a question. Using the same example input sentence, the NER tagger will annotate



(d) Stanford Constituency parser: annotating the input sentence with a syntactic structure.

Figure 3.3: Various Stanford annotators tagging the input sentence (Conti.)

the named entities as shown in Figure 3.3b. It recognizes the word "today" and classifies it to the dates class. These annotations are useful as it puts stress on informative words.

Dependency Parser

A dependency parser analyzes the grammatical structure of a sentence and converts it into a dependency tree by extracting relationships between two dependent words (a word that can modify the root word) in a sentence. Figure 3.4 shows that the parser identifies the word "moving" as the root word. The **root** is the most important word in the whole sentence:

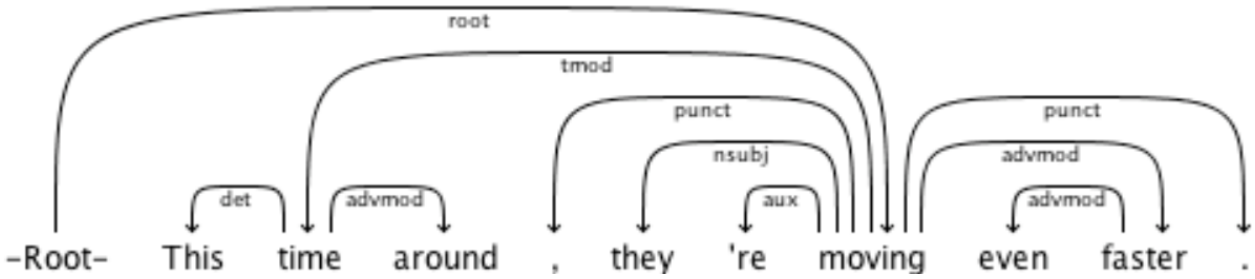


Figure 3.4: Stanford Dependency parser identifying the root word and the relationships between the dependent and the governing words.

it is directly or indirectly the head of every other word in the tree. The arrow from the word "moving" to the word "faster" illustrates that "faster" modifies "moving" and the tag "advmod" (adverb modifier) assigned to the arrow describes the exact nature of dependency between them.

We used the Basic and the Advanced Stanford dependency parsers for ClarQue. [16] lists all the dependencies that the Stanford dependency parser recognizes between words. Figure 3.3c demonstrates the results of the dependency parser on our example input sentence. The arrow from the word "eat" towards "today" indicates that "today" modifies "eat" and the relation "nmod:tmod" (noun modifier: time modifier) is assigned to the arrow representing the dependency of the verb on noun modifier. The Enhanced Dependencies usually give more detailed relationships between words than the Basic Dependencies.

Constituency Parser

Constituency parser [12] recognizes sub-phrases in a sentence and assigns a syntactic structure to it. We used the Stanford constituency parser mainly to retrieve the tense and the aspect of the input so that later ClarQue can communicate with the same. Figure 3.3d shows a tree for the ongoing example input sentence. The first root verb tag (VP) is assigned to "want" which provides information about the tense (i.e. past, present, future) of the sentence. ClarQue will keep traversing down further to the VPB tag to extract the aspect of the sentence (i.e.

simple, perfect, progressive, perfect progressive). The chatbot saves both the values until it generates a response and applies the same to the output.

3.2.2 Data Processing

The primary reason behind ClarQue performing data processing is to avoid duplication of words with inflectional endings in the knowledge graph. It helps in making a huge dataset manageable and also makes parsing and updating the knowledge base easier.

Lower Case

All the data received by ClarQue after the annotation is converted to lowercase. It is important to perform this step after annotations because the parsers work better with the natural language and proper grammar. If this step is performed before annotations, there might be loss of information or misinformation with lowercase words as it might fail to recognize proper nouns and especially the pronoun 'I', which will eventually lead to misleading tags.

Lemmatization

All the words are sent to a Stanford lemmatizer which reduces the word to its root form to remove inflectional endings and to return the base form of a word which is known as the lemma. For instance, for the words "went", "going", "gone" the lemma is "go" and for words "apples" and "apple" the lemma is "apple". These lemmas are saved in the knowledge base instead of the actual words from the sentence. There is a trade-off for using this tool where we lose information like tense, singular/plural forms, etc., of the input but it makes it easier to add to and update the knowledge graph (We have taken care of the loss of information due to tense by using the constituency parser.)

3.2.3 Knowledge Graph

Knowledge graph (KG) acquires and integrates information in space through data-interlinking in such a manner that parsing through combinations of information will lead to new information. It is a systematic way of putting facts and relevant information together in the form of nodes that are connected to other nodes by defined relations.

We prepared our custom knowledge graph as opposed to using a pre-existing KG so that it gives us control over the size of the prior knowledge and also over the domain of conversation. It gives us the flexibility of defining the relationships/edges between the nodes based on our needs. Moreover, we can select what metrics (weights) to use to define which nodes are more relevant in the graph compared to others.

Custom Knowledge Graph Format

The knowledge graph has head nodes that are connected to other nodes through pre-defined edges. A head node is usually the main subject of the input sentence (i.e. the **root** word, which is determined by using the NLP dependency parsers). The predefined relationships that we used for the head nodes are the most common question words like *who, what, why, how, whose, when, where, whom, which*. Every node that needs to be connected to a head node will be placed under one of these edges. These edges were selected on the basis that a sentence is usually a combination of words that when broken down, can easily be placed under one of these question words. Every head node that will be added to the knowledge graph will have the format as shown in Figure 3.5.

Building a Knowledge Graph

Once the chatbot has all the annotations, it will investigate the dependency tree that is received from the dependency parser. ClarQue will look for the **root** word or the main driving word of the sentence to start building the graph. For some sentences, there may be more than one subject and they all should be their separate head nodes since they are driving

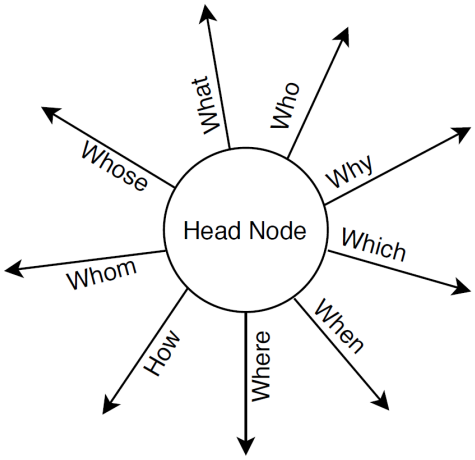


Figure 3.5: Custom knowledge graph format for a head node and its custom edges.

the sentence. To find more subjects, ClarQue will look for a dependency called "nsubj" (a nominal subject which is the syntactic subject of a clause) and the word at the tail of the arrow is the driving word which will become a separate head node. In Figure 3.3c, Enhanced Dependencies show that there are two words in the sentence with this dependency: "want" and "eat" and hence they will form separate head nodes. These head nodes will be connected to other nodes from the sentence with the relations/edges we defined. To fill the other nodes, the chatbot will go through the rest of the dependencies that these driving words have. With the dependency tree and the POS and NER taggers, the chatbot will determine which words can be placed under the edges of the head nodes in the knowledge graph. To make sure that the words are allocated under the correct edge, we set some rules such as if the dependency between the two words is "nsubj" and the POS tag of the word at the arrowhead is a pronoun (PRP) or the NER tag belongs to person or organization then it belongs under the "who" edge of the head node. More rules are mentioned in Table 3.1 for a head node when it is a verb. The rules from the table can be applied in this manner: If (Dependency and (POS or NER)), then return edge. Rules slightly differ when the head node is a noun as shown in Table A.1 in Appendix A.

Edge	Dependency	POS	NER
who	nsubj	PRP	PERSONS or ORGANIZATIONS
what	amod nmod or nsubj or dobj xcomp	NN* NN* VB	MONEY
why	advcl	VB	
how	amod	JJ* or RB*	
whose		PRP$	
when	nmod nsubj or advcl	NN*	TIMES or DATES TIMES or DATES
where	advcl case	NN* IN	LOCATIONS
whom	dobj or advcl or nmod dobj	PRP PRP	PERSONS
which		JJ*	

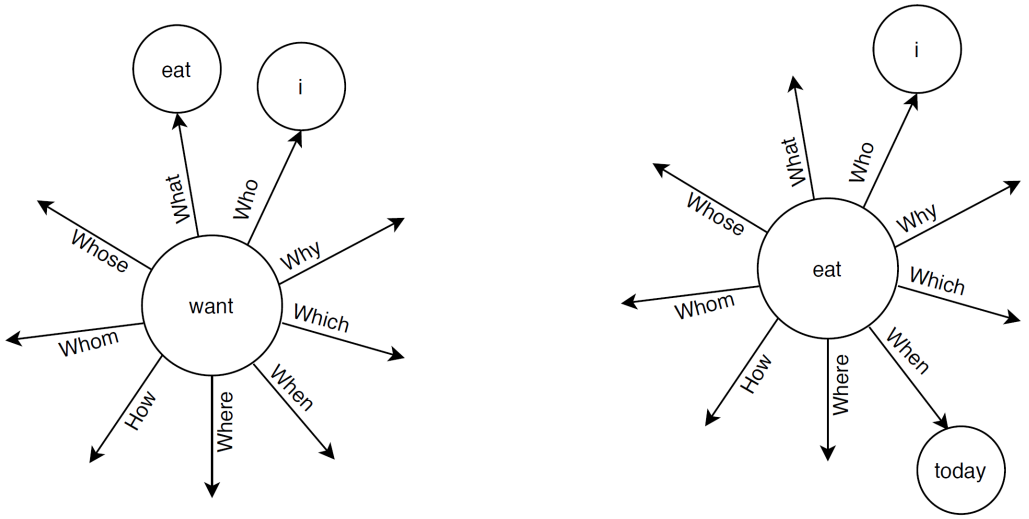
Table 3.1: Rules to assign nodes to edges when the head node is a verb: If (Dependency and (POS or NER)), then edge.
* implies any version of the tag like plural, singular, etc.

For the example sentence: *I want to eat today*, the chatbot recognizes "want" and "eat" as the words at the tail of "nsubj" dependency by parsing the tree, therefore, these words will be considered as the head nodes. While parsing through the rest of the words in the tree, ClarQue will keep applying the rules from Table 3.1 and identify the edges for them as shown in Table 3.2. Once the tree is completely parsed through, ClarQue will build the knowledge graph with the nodes and the edges. Figure 3.6 represents what the KG looks like after the assignment of edges.

The relations that remain empty after the formation of the graph are filled with a '_'. Since a particular head node is only added once to the knowledge graph and the other connecting nodes keep updating, this is a necessary step to preserve the context of the KG as well as the history of inputs. If in future the head node is "want" again, then ClarQue can keep all the connected nodes in order by using '_'. For example, if the input sentence is

Headnode	Node	Dependency	POS	NER	Edge
want	I	nsubj	PRP		who
	eat	xcomp	VB		what
eat	I	nsubj	PRP		who
	today	nmod:tmod	NN	DATES	when

Table 3.2: Rules that apply to *I want to eat today* to add to KG



(a) Head node "want" connected with other nodes through the edges

(b) Head node "eat" connected with other nodes through the edges

Figure 3.6: Converting the words from the input into head nodes and connecting nodes through edges in the knowledge graphs.

I urgently want a book, the KG will be updated as shown in Figure 3.7. With this format the chatbot will not mix up the nodes with previous inputs or extract mixed information like *I urgently want to eat.* This way when more input sentences are received, ClarQue can preserve their context and history. Even though there is a loss of information after adding the nodes to the KG (since not every single word from the input is added to KG and from the annotations), this format helps the chatbot in maintaining and extracting the correct context of the input when needed.

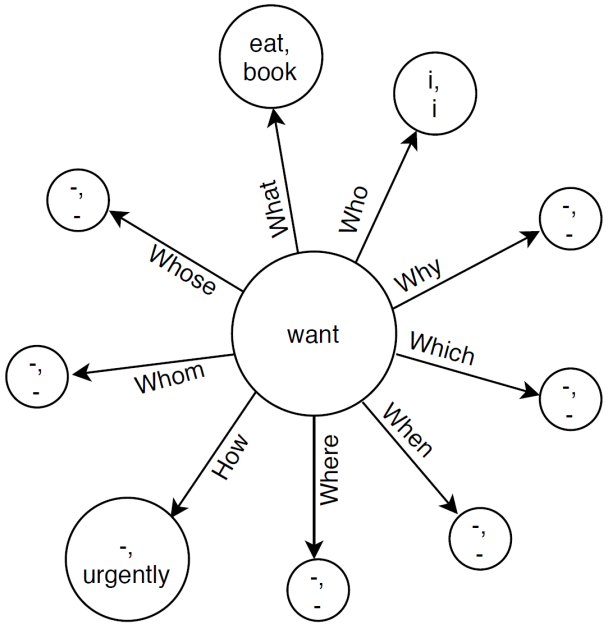


Figure 3.7: Updating KG while preserving the context and the history

Knowledge Graphs for ClarQue

ClarQue tries to mimic human behavior by having similar elements/processes as humans to communicate with users. Just as we have a notion of a mental model of the world or prior knowledge that is universal and shared by many people, ClarQue is given prior knowledge which is fed to it before it is ready to communicate with a human. It will construct the prior knowledge graph first from a text file we provided which consists of facts. These facts can be any information that we want our chatbot to know about before it starts chatting with a human.

When we start a conversation with person Z, we make a separate smaller mental model that is just for Z as shown in Figure 3.8. We build a different model for every person we talk to which consists of information that is relevant to them and the world. For instance, a model of a person informs us that they are interested in sports and so we discuss topics related to sports with them. As we talk, we gather newer information and adjust the mental model for that person accordingly. This knowledge cannot be expected to be universal knowledge, i.e., not everyone is expected to know these facts, hence it is kept separate from prior knowledge.

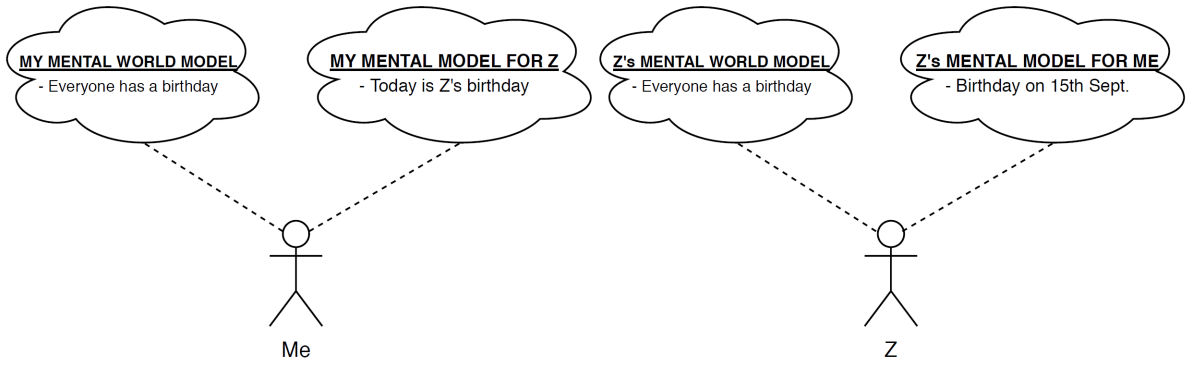


Figure 3.8: A mental model of the world and a mental model of a person.

Imitating human behavior, ClarQue keeps a separate user knowledge base for each user that is relevant only to them. It will use both the prior and the user knowledge combined to communicate with us and keep updating the user knowledge after every input as newer information is fed.

Weights

It is not sufficient to only have the interconnected knowledge stored in some space. It needs to be accompanied by a numeric value that represents what part of knowledge is preferred over the other part so that the chatbot can decide where to look at in the KG. We chose the frequency of words occurring in the input to be the weights as it conveys how relevant the words are together and what is more important in the pool of connected nodes. These weights will be used by ClarQue as mentioned in Section 3.2.4 Parsing the Knowledge Graph to check if there exists any uncertainty or not.

Identical to the number of knowledge graphs, there are two weights that the chatbot will keep track of. The first one is the prior knowledge weights that will keep track of the frequency of words in prior knowledge (PK) and the second is the current user knowledge weights that will keep track of the frequency of words in the user knowledge (UK). The UK weights are updated after every input. Assuming at this point that we already have both these knowledge graphs, the nodes will have weights as shown in Table 3.3. By using these

Head nodes	PK Weights	UK Weights
want	8	3
book	0	3
water	2	0
dance	4	4
her	1	7
seat	2	1

Table 3.3: Weights of head nodes in prior knowledge graph and user knowledge graph.

weights ClarQue can determine which nodes are comparatively more important than the other. If the node is present in the user knowledge the chatbot will use those weights first as they will be more relevant to the current user. If it is not present in the UK, then the chatbot will look for it in the PK and use those weights. If the node is not present in either of the knowledge graphs, the weight defaults to 0.

3.2.4 Parsing the Knowledge Graph

Once the knowledge graphs are constructed and the weights are updated, ClarQue will focus on the user input and its context. It will parse the input sentence and look for the important words and phrases and will check if they are present in the knowledge graphs. If they are present, the chatbot will inquire if that head node has nodes connected to it through the meaningful edges. Parsing the KG is split into two parts: searching for important words in the input and checking which nodes are connected to them through meaningful edges.

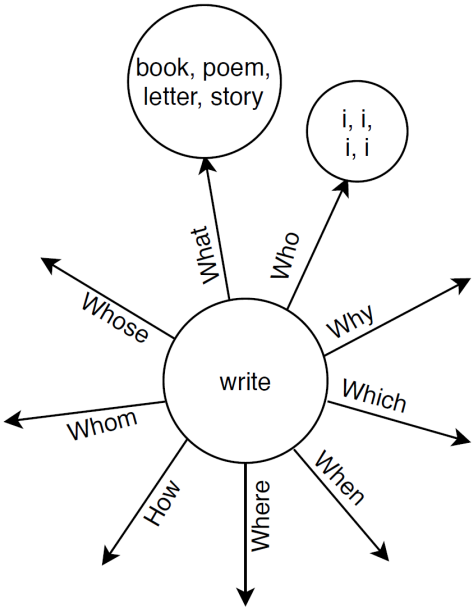
Finding Important Words

When an input is given by a user, it is crucial to understand which words are valuable in the sentence to check if enough information is present or not. The chatbot needs to recognize and extract these important words from the sentence. To accomplish this task, ClarQue uses Rapid Automatic Keyword Extraction (RAKE) library [22] which determines key phrases in a body of text by analyzing the frequency of word appearance and its co-occurrence with

other words in the text and presents them with weights showing which keywords have higher importance. ClarQue only needs to use RAKE if the input is a statement since responding to it requires the chatbot to have enough knowledge about it. When the input is a question, the chatbot can traverse the knowledge graphs for an answer and does not need to find valuable words. For the example input: *I want to eat today*, the RAKE library returns ["eat today": 4.0, "want": 1.0], which demonstrates that the phrase "eat today" has higher importance than the other phrases/words in the sentence. Since the nodes saved in the KGs are single words, ClarQue divides the phrases into single words by splitting them and returning: ["eat": 4.0, "today": 4.0, "want": 1.0] so that comparing these words with the KG nodes becomes straightforward.

Finding Meaningful Edges

Since our edges are question words, the chatbot can check to see which question words are most related and relevant to the important words found through RAKE. ClarQue uses Datamuse API [1], which is a word-finding query engine for a wide range of features, including auto complete on text input fields, search relevancy ranking, assistive writing apps, word games, and more. The chatbot uses this API to find question words that lead an follow these important keywords/head node found in the previous section. For instance, one of the keywords RAKE recognized from our example input was "eat". Using the relevancy ranking (finding words that often go with the word of interest) and a filter that only returns words that start with {wh*, how} (as our edges start with "wh" with an exception of "how"), the Datamuse API comes up with words: ["what": 6217, "when": 5456]. This truncates other edges so that the chatbot can focus on the most important edges of the head node. ClarQue makes sure that the words returned by the API are filtered so that only the question words that represent one of the edges are present. The chatbot also looks for the answers to these question words in the input sentence first and does not add them to the questions list if they have already been answered in the input. In other words, for the keyword "want",



(a) KG for head node: write

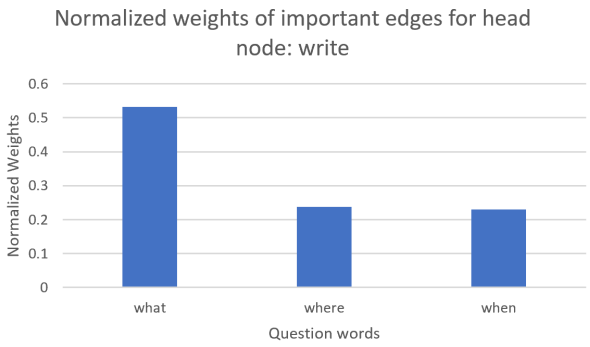
Figure 3.9: KG and normalized weight distribution for the head node: write

RAKE returns ["what": 2134]. The input already answers "what" the user wants i.e. to eat, therefore, this edge will not be added to the potential questions list. If there were other question words such as "when" or "where", then those would have been added to the potential questions list. ClarQue has enough information about "want" head node and hence it will move on to the next important node and its edges.

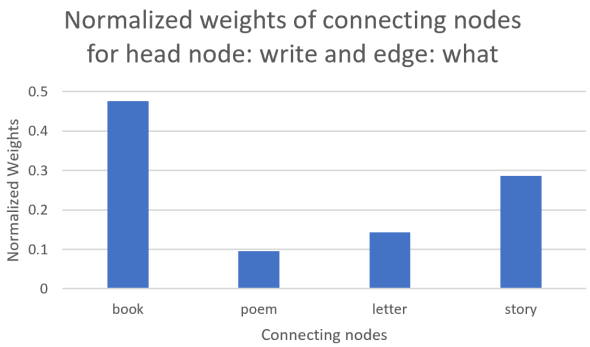
Nodes Connected through Meaningful Edges

After ClarQue has collected the list of all meaningful edges of the important words from the input, it will proceed towards checking which connecting nodes are present under those edges. The chatbot recognizes two scenarios that might happen during this process:

*S*cenario 1: The list of important edges received is probably long and/or there are outliers (words with comparatively lower weights) present. In this scenario, ClarQue needs to focus on edges that have even higher importance amongst themselves by sending them to the Ambiguity test. For instance, given the knowledge in Figure 3.9a, the list of edges for the head node "write" returned by the Datamuse API is [what, when, where]. The edges "where"



(b) Important edges



(c) Connecting nodes

Figure 3.9: KG and normalized weight distribution for the head node: write (Conti.)

and "when" belong to the lower weighing outliers when compared to the weight of "what" as shown in Figure 3.9b, and hence according to the Ambiguity test, ClarQue should focus on that edge.

*S*cenario 2: The list of edges is received and ClarQue starts looking for connecting nodes under each edge. It is again probable that the list of nodes received is very long and/or there are outliers present. In this scenario also, the chatbot should focus only on the nodes that have even higher importance amongst themselves by sending them to the Ambiguity test. Since knowledge can be missing, there are two ways that this can unfold:

*S*cenario 2a: The list of nodes under the important edge might be empty. This means that there is information missing regarding the head node and the edge and the chatbot should ask a question regarding them to eliminate doubts. If "where" was deemed as an important edge by the Ambiguity test, then the chatbot would not receive an empty list of nodes since it has no connected nodes.

*S*cenario 2b: If the nodes under the edges are present, it means that the information is also present but the chatbot needs to determine if that information is enough to move on to the next edge or does it need to ask a question about this edge. In our example, using the connecting nodes for "write" through "what" edge, the chatbot gets the nodes and their weight distributions as shown in Figure 3.9c. The Ambiguity test will decide which nodes the chatbot should focus on.

3.2.5 Certainty

When we are presented with some statements, we refer to our knowledge and make sure we have enough information before deciding how to respond. Similarly, once ClarQue has the list of keywords provided to it, it needs to make sure that the information it has is enough or not. It needs to decide whether there are any doubts and ambiguity or does it have everything it needs to move on. To make this decision, ClarQue passes the keywords and their respective scores to the Ambiguity test.

Ambiguity Test

The Ambiguity test will help ClarQue in deciding whether it has enough information or if it needs more information regarding the nodes and the edges. This test only requires a list of keywords (nodes or edges) and the weights associated with them, which is why ClarQue can use this test for both the list of important edges and the list of nodes under an edge. ClarQue will start by normalizing the weights of the keywords and then passing them through multiple smaller tests.

*T*est 1: Finding Highend Outliers: ClarQue uses the InterQuartile Range (IQR) to find the outliers on the higher end as it is mostly interested in any obvious higher weighted keywords. Given a list of sorted normalised weights $W = [w_1, w_2, w_3, w_4, w_5, w_6, w_7]$, the chatbot splits it into three parts: $[(w_1, Q1, w_3), (Q2), (w_5, Q3, w_7)]$ where Q2 is the median of all the normalised weights and is considered the second quartile; Q1 is the median of the first quartile; and Q3 is the median of the third quartile. The outliers on the higher side are found by setting a threshold value T_{high} and any value above it is considered a high-end outlier. The threshold value is obtained by finding the IQR value (Equation 3.1) and adding it to the Q3 after multiplying it with 1.5 (a constant used to discern outliers) as shown in Equation 3.2,

$$IQR = Q3 - Q1 \tag{3.1}$$

$$T_{high} = Q3 + 1.5(IQR) \tag{3.2}$$

$$f(w_i) = \begin{cases} k_i, & \text{if } w_i \geq T_{high} \\ Test2, & \text{otherwise} \end{cases} \tag{3.3}$$

where k_i is the keyword whose weight the chatbot is checking and w_i is the normalized weight of the keyword. If none of the keywords are selected in Test 1, then ClarQue passes the keywords and their weights to Test2.

Test 2: Finding keywords that dominate other keywords: If none of the keywords have weights greater than or equal to 40% (as it shows clear dominance when compared to other keywords' weights), then the keywords and their weights are sent to Test 3 (Equation 3.4); otherwise only the keywords with weights greater than or equal to 40% are returned.

$$f(w_i) = \begin{cases} k_i, & \text{if } w_i \geq 0.4 \\ Test3, & \text{otherwise} \end{cases} \tag{3.4}$$

where k_i is the keyword whose weight the chatbot is checking and w_i is the normalized weight of the keyword.

Test3: Finding Entropy (S): If there are no dominant keywords, then ClarQue will look at the entropy of the weights (W). Entropy gives the measure of randomness in the list of weights. The entropy can be found using

$$S = -\sum_{i=1}^{N} w_i log w_i \tag{3.5}$$

where w_i is the normalized weight of the ith keyword. The entropy is highest when all the weights are different $S_{max} = log_2(L)$, where L is the number of all unique weights. The entropy is lowest when all the weights are same $S_{min} = log_2(1)$. When there is no randomness and all keywords have almost equal weights (uniform distribution), then ClarQue has a

difficult time picking valuable keywords. Therefore it decides to ask a question rather than guessing which keywords to use in the response. On the other hand, when the entropy is high, the chatbot picks the keyword with the highest weight. When it belongs to neither of these categories, ClarQue will return the top three keywords.

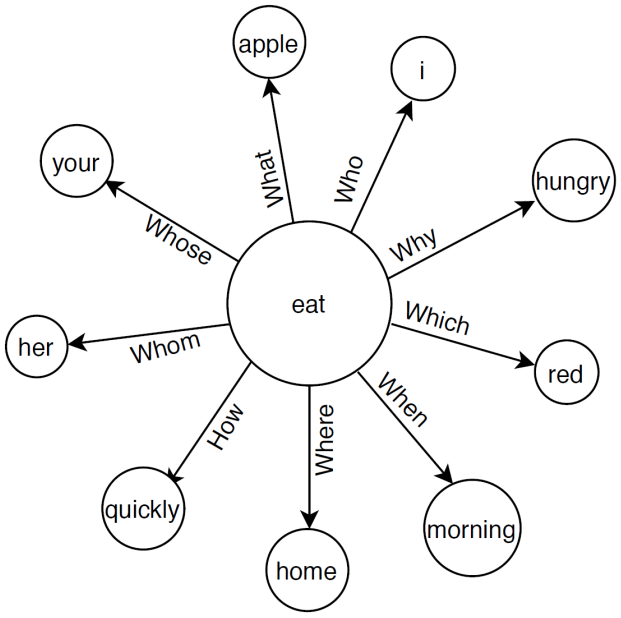
$$f(S) = \begin{cases} k_{max}, & \text{if S} \geq 0.9(S_{max}) \\ k_{max\_1}, k_{max\_2}, k_{max\_3}, & \text{if } 0.3(S_{max}) \geq S \leq 0.9(S_{max}) \\ q, & \text{if S} \leq 0.3(S_{max}) \end{cases} \tag{3.6}$$

where k_{max} is the keyword with the maximum score, $k_{max\_1}, k_{max\_2}, k_{max\_3}$ are the keywords with maximum weights, q is the edge (question) ClarQue is looking into.

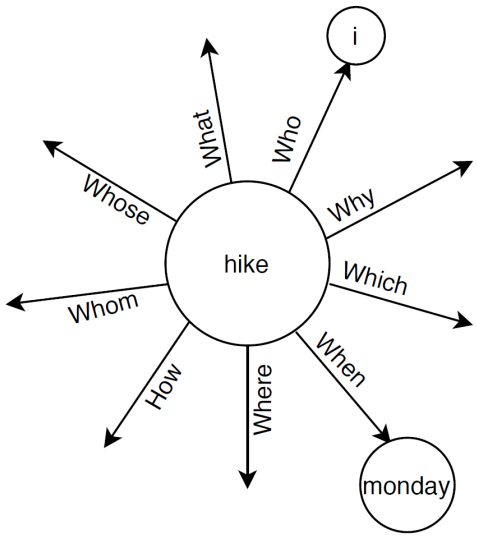
For the list of edges passed to this test, it returns a shortened list of more important edges so that ClarQue can parse through the nodes that are connected to the head node under the given edge. For the list of connecting nodes passed to this test, it returns a shortened list which will be used for generating a response.

3.2.6 Response Generator

This section talks about how ClarQue generates an output based on all the previous steps and decisions. ClarQue can respond with a statement, an answer to a question, or ask a question. At this point, the chatbot has all the keywords that will be used in composing a response. To make it more understandable, we generate templates that use these keywords to form a response. For these templates we used Natural Language Generation library (nlglib) [2] that transforms data into natural language. For ClarQue, the data comes in the form of keywords (head node, connected nodes, and edges) that were returned from the Ambiguity test. Each way that a chatbot can reply has a slightly different template that will be mentioned in the subsections below. A general template is made by placing the placeholders for the keywords at fixed positions in a clause. Here is an example of the general template which has a placeholder for every keyword under every edge connected to the head node:



(a) Generating a head node for word "eat"



(b) Generating a head node for word "hike"

Figure 3.10: Adding input to the KG to generate head nodes

Pronoun*(who)* **Adverb***(how)* **Verb***(head node)* **Possessive Pronoun***(whose)* **Adjective***(which)* **Noun***(what)* **Preposition Determinant Noun***(where)* **Preposition Determinant Noun***(when)* **Preposition Pronoun***(whom)* **Conjunction Verb***(why).*

The Nlglib will apply the tense, aspect, person, negation, question type, etc. to this template if they are provided. Assuming there is a piece of knowledge that uses this template to form a normal sentence using first person and past simple tense, as shown in Figure 3.10a, it would look like this:

ClarQue: I quickly ate your red apple at home in the morning with her because hungry.

It is not mandatory that all the nodes be present under the edges to generate a response. ClarQue will simply skip the part of the template for which the nodes are not present. Using the same general template and fewer nodes as shown in Figure 3.10b, shorter sentences like *I will hike on monday*, can be generated.

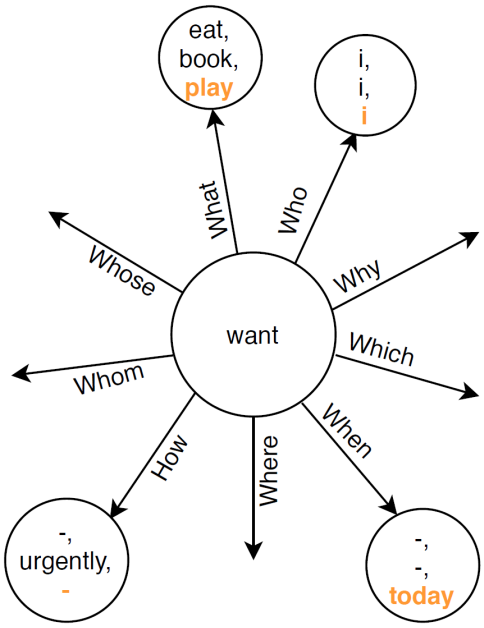


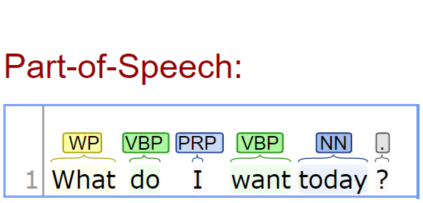
Figure 3.11: Adding input to the KG to generate head node for "want"

Replying with a statement

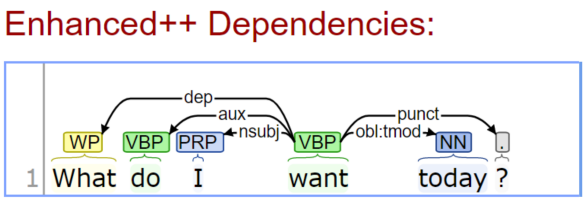
When the chatbot determines that there is no uncertainty regarding the important words from the input and there are no questions that need to be asked, it will respond with a statement such that it acknowledges the user's input. For instance, the input is *I want to play today,* that gets saved in the KG as shown in Figure 3.11. Based on the Ambiguity test if there is no uncertainty then ClarQue will use the general response template to generate a response. ClarQue adds a word of affirmation before the generated response: *Ok! You want to play during today.* Here *Ok* is added as the first word in the sentence to show acknowledgment and then the general template is used to finish the statement. There are other affirmations used as well like *Alright* and *Ok* without the entire sentence so the chatbot doesn't sound repetitive.

Replying with an answer

Users can even ask questions to ClarQue. It is only capable of answering the queries whose answers are present in the prior or user KG. It annotates the query like a normal input and



(a) POS annotator tagging a user query



(b) Enhanced Dependency annotator tagging a user query

Figure 3.12: Stanford annotators tagging a user query.

uses the labels of the words to filter only the relevant head nodes and its connected nodes to find an answer to the question (ClarQue works best when the user questions are specific). Assuming the input *I want to draw today* is saved in KG and the user asks a question to the chatbot: *What do I want today?* For this query, ClarQue will look for the **root** word from the dependency parser which is: "want", and the question word tag (WP) from the POS tagger: "what" as shown in Figure 3.12. For the next step, ClarQue will try to match the root word of the query with the head nodes in the KG. Once that is found, it will match the question word to the edges under which the answer might be present. Figure 3.13 shows how the input was added to the KG and the highlighted nodes under the highlighted edge: "what" shows the possible answers for this query. Since the question is more specific regarding "who" wants and "when" they want instead of in general question "What is wanted?", ClarQue can further narrow these answers by using filters. ClarQue will only consider the nodes under the edge of interest "what" that correspond to 'I' (who) and "today" (when), as they are also a part of the query. Therefore only "play" and "draw" nodes will be sent to the Ambiguity test along with their weights from the KG. Depending on the weights and the result of the Ambiguity test, the response will contain at least one of these words: play and draw. When ClarQue has the answer to the question it will use the general response template with slight changes to respond with *You want to draw, play during today.*

If the user question is *What does she want today?*, ClarQue will go through the same process but during filtering, it would determine that under the "who" edge in the KG there is no node called "she". Hence, in this case, the chatbot does not know the answer to the

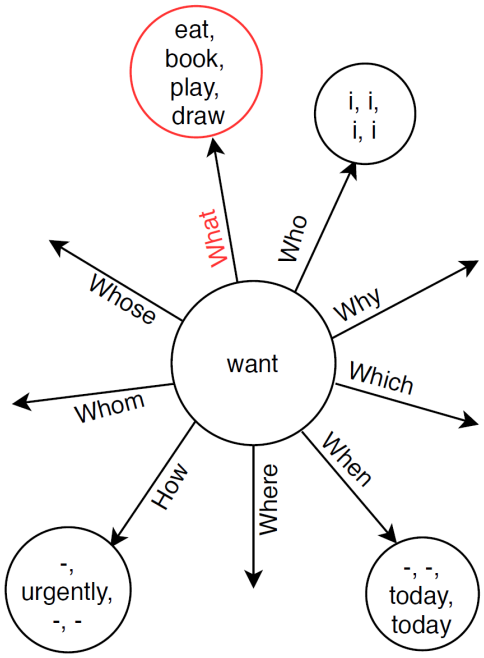


Figure 3.13: Filtering: the nodes under the highlighted edge "what" are the potential answers for the query *What do I want today?*

question because it does not have the knowledge about it. Therefore, it will not use any template and respond with a version of *I don't know!* The other response to this scenario is *I am not sure.*

ClarQue is also capable of replying to a yes or no question. For example, if the user question is *Do I want to play?* or more specifically *Do I want to play today?*, the chatbot will filter the nodes and if they match in the KG and the query, it will respond with an affirmation added to the general response template, *Yes. You want to play during today.* The next question may be *Do I want to draw tomorrow?* and the chatbot detects that "tomorrow" does not match the time to draw in the KG, so it will respond with *No. You do not want to draw during tomorrow.*

Asking question

Additionally, ClarQue can respond by asking a clarifying question to the user. It will ask questions in three scenarios (i) When an important edge does not have any nodes connected.

For example, the input is *I sang a song*, ClarQue determines "which" to be an important edge but there are no nodes present under that edge. Hence, ClarQue will ask the user *Which song did you sing?* This response is generated using a question template where the interrogation parameter is set to true (ii) When the user has already mentioned a certain piece of information previously while chatting and it is brought up again. For example, the user had previously mentioned that they like to eat food. If the user brings it up again *I like to eat*, then the chatbot will respond with *food?* (iii) When the users ask a very broad question and the chatbot needs further information to narrow down the answer like in our example in above section *What do I want?* When ClarQue looks at the relation "what" it has four answers "eat", "book", "play", and "draw". It also has other edges that have connected nodes in the KG which are relevant to these four nodes. To narrow down the answer, ClarQue will find those filled edges and find the most meaningful edge using the Ambiguity test and ask a question regarding that edge which in this case is *When?*

Switching First and Second Person

The final step before responding to the user is changing the person of the response. All the knowledge is from the world's and user's point of reference so the words in the KG are from their perspective. If while responding the chatbot uses the same voice, it would sound like an echo of the user talking. Hence it is important to switch the voice before responding. ClarQue will recognize the voice of the user input and change it appropriately before the response is sent back by passing the output person as a parameter to the templates.

3.3 Validation

There are many different ways in which a chatbot can reply to given user input. It could be a simple yes or no, a complete answer, a statement, or a question, which makes measuring any response difficult as there is no standard to how one must reply.

	ClarQue	**PBot**	**Qbot**
Finds important words from the input	x	x	x
Deduces what question to ask	x	x	
Deduces when to ask a question	x		
Chooses random question when no question needs to be asked		x	x
Questions asked	based on ambiguity	randomly 43% of the time	every time

Table 3.4: Features and abilities of ClarQue and the two baseline chatbots (PBot and QBot).

ClarQue can recognize important words from the input and based on whether or not there is any uncertainty regarding them, it will derive what clarifying questions to ask instead of randomly picking a question that might have nothing to do with the conversation. To compare how our chatbot performs we made two other baseline chatbots. The first one, QBot, always asks a question regarding the input irrespective of whether there is any uncertainty or not. Just like ClarQue, this bot finds important words from the input and finds corresponding questions. It removes the questions that are already answered in the input from the list and instead asks a question that has not been answered yet. If there are no more questions to be asked, the chatbot just randomly selects a question from the edges and asks about it. The purpose of this chatbot is to show how inconvenient it would be for the users if the chatbot is unable to use the knowledge it posses and just keeps asking clarifying questions all the time. The questions might result in responses that are repetitive or unnecessary or even annoying at times.

The second chatbot, PBot, is very similar to ClarQue, which can reply with a sentence, answer questions, and can ask questions but only at random times. We deprived it of its reasoning ability to deduce if there is any uncertainty or not. We pass a probability (p is 43%) of generating a question to this chatbot which means that it will ask questions at random times in the conversation for 43% of the times. This number was obtained by parsing through human-to-human chats corpus: Chitchat dataset [31] and figuring out how much percentage of the entire conversation were questions. This means that the PBot cannot figure

out when to ask a question on its own and relies on a random generator but it can still identify important questions to be asked. If the chatbot has all the information and there is no ambiguity, then it will randomly select a question from the edges and ask about it. The purpose of this chatbot is to show that without the reasoning ability of when to ask a question and when to reply in other ways, the conversation might become confusing and annoying.

We will also compare ClarQue's performance with a human conversation to observe how close it gets to it in certain areas (appropriateness, relevance, conversational flow, and annoyance) of communication. The goal is to see where our chatbot stands when compared to these two baselines chatbots and how close it gets to the human to human conversation. Table 3.4 encapsulates all the abilities and features of the three chatbots for comparison.

Chapter 4

Results

In this chapter, we present the performance evaluation of our chatbot compared to the performances of the other two baseline chatbots: PBot and QBot, and to human performance. We first discuss the questions that were asked by us in the empirical study that was conducted to evaluate the performance of all the chatbots followed by the reporting of all the results achieved from the study and then analyzing them.

4.1 Empirical Study

To evaluate and validate the performance of ClarQue, we conducted an empirical study to determine how useful the features (deducing when to ask a question and which question to ask) of our chatbot are compared to the other three baselines. We started by generating Dialogue Sets that contain conversation between P1 (a human) and P2 (the four contenders: human, ClarQue, PBot, Qbot). We added a human-to-human conversation to determine how far or close ClarQue's performance would be when compared to human performance. We want to verify how well ClarQue performs in field of relevance (how related the responses are), appropriateness (is this response acceptable), conversational flow (does it allow the conversation to flow smoothly), and if it gets annoying (because of its repetitiveness, unrelated, or inappropriate nature) while chatting. Therefore, to compare the performance of ClarQue and other baselines, questions related to these fields were added in the survey. One out of three Dialogue Sets that were generated and used for the study is shown in Figure 4.1 (the

She eats eggs in the morning.
She eats bread in the evening.

Table 4.1: Prior knowledge provided to the chatbots for first Dialogue Set.

	Human	**ClarQue**	**PBot**	**QBot**
Relevance	3.96 ± 1.13	3.61 ± 1.18	2.94 ± 1.45	2.92 ± 1.55
Appropriate	3.91 ± 1.26	3.58 ± 1.21	2.82 ± 1.52	2.80 ± 1.65
Flow of Conversation	3.85 ± 1.28	3.57 ± 1.29	2.80 ± 1.53	2.81 ± 1.63
Annoyance	0.77 ± 1.09	1.24 ± 1.28	2.28 ± 1.68	2.41 ± 1.82

Table 4.2: Mean values and standard deviations of the performance by the contenders in the four fields for all the three Dialogue Sets.

other two Dialogue Sets can be found in Appendix B). We provided prior knowledge to the chatbots for this Dialogue Set before the conversation started which is given in Table 4.1.

We asked 50 surveyors to rate every single response by P2 on a 6-point Likert Scale (0 - not at all, 1 - barely, 2 - somewhat, 3 - fair enough, 4 - good, 5 - extremely) for these four questions (i) How related is P2's response to P1's input and the rest of the conversation? (ii) How appropriate is P2's response? (Would you respond like this?) (iii) Is this response helpful in keeping the flow of conversation going? (iv) How much did this response annoy you? For each of the four fields, we have 50 ratings from the survey x 3 Dialogue Sets x 5 responses by P2 for each Dialogue Set which gives us (N = 750) scores per field. Overall performance by P2 in all four fields is shown in Figure 4.2. The mean value and the standard deviation of ratings received are shown in Table 4.2.

Since our data is ordinal and non-parametric, we ran a Mann-Whitney U statistical test on ClarQue vs. other baselines one at a time. Our null hypothesis H_0 is that there is no statistical difference between the performance of ClarQue and the other baselines (human, PBot, QBot) in the four fields. Our alternate hypothesis H_a is that there is a statistical difference between the performance of ClarQue and other baselines in the four fields. The p values of these pairs after conducting the Mann-Whitney U test are shown in Table 4.3.

P1: She loves chocolate.

P2: Oh! Which one?

P1: Snickers.

P2: Ya, that is a very good chocolate.

P1: Do you like chocolate?

P2: I love chocolate very much.

P1: What did she eat?

P2: In the evening?

P1: Yes.

P2: She ate bread.

(a) Dialogues between P1 (a human) and P2 (a human)

P1: She loves chocolate.

P2: Which chocolate does she love?

P1: Snickers.

P2: Ok! She loves the chocolate.

P1: Do you like chocolate?

P2: I don't know.

P1: What did she eat?

P2: When?

P1: Evening.

P2: She ate the bread at evening.

(b) Dialogues between P1 (a human) and P2 (ClarQue)

P1: She loves chocolate.

P2: Ok! She loves the chocolate.

P1: Do you like chocolate?

P2: I don't know!

P1: What did she eat?

P2: When?

P1: Evening.

P2: She ate the bread at evening.

P1: I went to a beautiful park today.

P2: Where did you go in the park today?

(c) Dialogues between P1 (a human) and P2 (PBot)

P1: She loves chocolate.

P2: Which chocolate does she love?

P1: She loves Snickers.

P2: Which snicker does she love?

P1: Do you like chocolate?

P2: What chocolate?

P1: What did she eat?

P2: Where did she eat?

P1: She ate in the park.

P2: What did she eat in the park?

(d) Dialogues between P1 (a human) and P2 (QBot)

Figure 4.1: First Dialogue Set used for the empirical study

4.2 Analysis

In this section, we will analyze the scores gathered from the survey and the results achieved by running analytical tests on these scores in the empirical study.

4.2.1 ClarQue vs. Human

The Mann-Whitney U test indicates that the relevance of the conversation was significantly greater for human ($Mdn = 4$, $SD = 1.13$) than for ClarQue ($Mdn = 4$, $SD = 1.18$), $p < .001$, the appropriateness of the conversation was significantly greater for human ($Mdn = 4$,

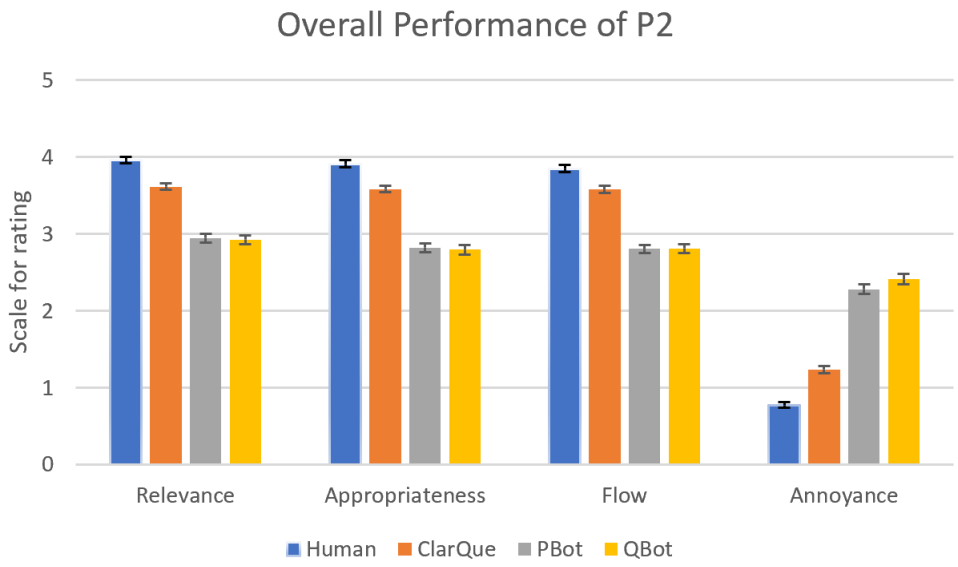


Figure 4.2: Overall performance of all the contenders and their standard error in the four fields for the three Dialogue Sets

ClarQue vs.	**Human**	**PBot**	**QBot**
Relevance	$< .001$	$< .001$	.059
Appropriate	$< .001$	$< .001$	.023
Flow of Conversation	$< .001$	$< .001$	.001
Annoyance	$< .001$	$< .001$	$< .001$

Table 4.3: p values of ClarQue vs. other baselines (Humans, PBot, and QBot) achieved by performing Mann Whitney U Test.

$SD = 1.26$) than for ClarQue ($Mdn = 4$, $SD = 1.21$), $p < .001$, the flow of the conversation was significantly greater for human ($Mdn = 4$, $SD = 1.28$) than for ClarQue ($Mdn = 4$, $SD = 1.29$), $p < .001$, and the annoyance in the conversation was significantly greater for ClarQue ($Mdn = 1$, $SD = 1.28$) than for human ($Mdn = 0$, $SD = 1.09$), $p < .001$. Therefore we can reject the null hypothesis that the difference in the all the performances of Human and ClarQue is due to random sampling, and conclude that the populations are distinct by accepting the alternate hypothesis.

4.2.2 ClarQue vs. PBot

The Mann-Whitney U test indicates that the relevance of the conversation was significantly greater for ClarQue *(Mdn* = 4, *SD* = 1.18) than for PBot *(Mdn* = 3, *SD* = 1.45), $p < .001$, the appropriateness of the conversation was significantly greater for ClarQue *(Mdn* = 4, *SD* = 1.21) than for PBot *(Mdn* = 3, *SD* = 1.52), $p < .001$, the flow of the conversation was significantly greater for ClarQue *(Mdn* = 4, *SD* = 1.29) than for PBot *(Mdn* = 3, *SD* = 1.53), $p < .001$, and the annoyance in the conversation was significantly greater for PBot *(Mdn* = 2, *SD* = 1.68) than for ClarQue *(Mdn* = 1, *SD* = 1.28), $p < .001$. Therefore we can reject the null hypothesis that the difference in the all the performances of ClarQue and PBot is due to random sampling, and conclude that the populations are distinct by accepting the alternate hypothesis.

4.2.3 ClarQue vs. QBot

The Mann-Whitney U test indicates that the relevance of the conversation was not significantly different for ClarQue *(Mdn* = 4, *SD* = 1.18) and QBot *(Mdn* = 3, *SD* = 1.55), $p = .059$. On the other hand, the appropriateness of the conversation was significantly greater for ClarQue *(Mdn* = 4, *SD* = 1.21) than for QBot *(Mdn* = 3, *SD* = 1.65), $p = .023$, the conversational flow in the conversation was significantly greater for ClarQue *(Mdn* = 4, *SD* = 1.29) than for QBot *(Mdn* = 3, *SD* = 1.63), $p = .001$, and the annoyance in the conversation was significantly greater for QBot *(Mdn* = 3, *SD* = 1.82) than for ClarQue *(Mdn* = 1, *SD* = 1.28), $p < .001$. Therefore we can reject the null hypothesis that the difference in appropriateness, flow and annoyance of ClarQue and QBot is due to random sampling, and conclude that the populations are distinct by accepting the alternate hypothesis. In the field of relevance we accept the null hypothesis and reject the alternate hypothesis.

Chapter 5

Conclusions and Future Work

Deducing when we have enough information and when to ask a question is extremely important for a meaningful conversation. Additionally, asking the right kinds of questions when needed helps us in completing our mental model of the world. Asking clarifying questions is beneficial for understanding the needs of both the speaker and the listener. It can help collect incomplete information and in finishing the tasks. We developed three chatbots with various levels of deduction powers: ClarQue can decide about when it has enough information and when it must ask a question along with deducing which question to ask, PBot can deduce which question to ask whenever prompted, and QBot will always ask a question. The effectiveness of the deduction has been verified through an empirical study which shows that as the use of chatbots has increased in today's world, it is imperative for a chatbot to have this compelling feature so that it can have a purposeful conversation with humans. Based on the results of the overall performances of all contenders in all the Dialogue Sets in Table 4.2 and the p values in Table 4.3, it is evident that the human performance trumps all the chatbots in every field when having a conversation. From among the chatbots, ClarQue performs the best in all the fields. PBot, who asks questions at random times, performs close to the QBot, who always asks questions whether there is any uncertainty or not (implying: Human $>$ ClarQue $>$ PBot $\sim$ QBot).

Our research work contributes to the area of conversational AI. The proposed method is an initial attempt towards making a chatbot capable of deciding how to respond by using natural language processing tools, knowledge graphs and weights, and the Ambiguity test.

ClarQue can accept prior knowledge of any size that belongs to one or multiple domains. It can store all the knowledge in the form of nodes and edges and can answer queries, ask questions, and reply with affirmations and negations.

The proposed workflow of ClarQue attempts to close the gap between the chatbots having a difficult time comprehending natural language and wanting to help humans with their tasks by mimicking human behavior while communicating. With the increasing features in chatbots these days, our chatbot adds value by determining what information it already has, what information is needed, and what questions to ask the users to collect that information.

5.1 Future Work

In the future, we would like to work on constructing a better structure for our custom knowledge graph that can manage multiple information related to a node under an edge while preserving the history and context of the input. Currently, when a lot of information is added under one edge of a node for a single input, it messes up the history which in turn will mess up the response of the chatbot.

Next, we also want to improve the quality of questions by expanding the types of questions to compound words that can be asked base on the existing edges like *with what, by whom, about whom, how many, how good, how bad, etc.* This will add more variety to the questions and help the chatbot in collecting more knowledge.

Finally, something that would add a lot of value to this workflow is adjusting the weights of nodes depending on the sentiment of the input. The chatbot can temporarily increase the weights of nodes that match the sentiment on the sentence, for example, if the input is *I am celebrating*, which has a happy sentiment, then the chatbot can temporarily increase the weights of nodes that have happy sentiment like *birthday, anniversary, promotion, etc.* By using this feature during the decision making step, ClarQue can precisely construct its response which is more relevant to the input. Moreover, negative scoring for negative input can also be added. Both of these techniques will improve the flow of conversation.

References

[1] Datamuse api. URL https://www.datamuse.com/api/.

[2] Natural language generation python library. URL https://pypi.org/project/nlglib/.

[3] Mordecai Avriel and A C. Williams. The value of information and stochastic programming. *Operations Research*, 18:947–954, 10 1970. doi: 10.1287/opre.18.5.947.

[4] Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014. doi: 10.3115/v1/d14-1082.

[5] Kenneth J. W. Craik. *Nature of Explanation*. Cambridge University Press, 1967.

[6] Alexandre Sawczuk da Silva, Xiaoying Gao, and Peter Andreae. Wallace: Incorporating search into chatting. In Duc-Nghia Pham and Seong-Bae Park, editors, *PRICAI 2014: Trends in Artificial Intelligence*, pages 842–848. Springer International Publishing, 2014. ISBN 978-3-319-13560-1.

[7] Heng Ding and Krisztian Balog. Generating synthetic data for neural keyword-to-question models. *CoRR*, abs/1807.05324, 2018. URL https://arxiv.org/abs/1807.05324.

[8] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL 05*, pages 363–370, 2005. doi: 10.3115/1219840.1219885.

[9] Noah D. Goodman and Andreas Stuhlmüller. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in cognitive science*, 5 1:173–84, 2012.

[10] Michael Heilman. *Automatic Factual Question Generation from Text.* PhD thesis, Carnegie Mellon University, 2011.

[11] Michael Heilman and Noah A. Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North*

American Chapter of the Association for Computational Linguistics, pages 609–617, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N10-1086.

[12] Daniel Jurafsky and James Martin. Statistical constituency parsing, Oct 2019. URL https://web.stanford.edu/~jurafsky/slp3/14.pdf.

[13] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sren Auer, and et al. Dbpedia a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167195, 2015. doi: 10.3233/sw-140134.

[14] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014. URL http://www.aclweb.org/anthology/P/P14/P14-5010.

[15] Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert Macintyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The penn treebank. *Proceedings of the workshop on Human Language Technology - HLT 94*, 1994. doi: 10.3115/1075812.1075835.

[16] Marie-Catherine Marneffe and Christopher Manning. Stanford typed dependencies manual - stanford nlp group, Sep 2008. URL https://nlp.stanford.edu/software/dependencies_manual.pdf.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013. URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[18] D. A. Norman. *Some Observations on Mental Models*, page 241244. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987. ISBN 0934613249.

[19] Andrew Olney, Arthur C. Graesser, and Natalie K. Person. Question generation from concept maps. *D&D*, 3:75–99, 2012.

[20] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

[21] Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. *CoRR*, abs/1805.04655, 2018. URL http://arxiv.org/abs/1805.04655.

[22] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. *Text Mining*, page 120, Apr 2010. doi: 10.1002/9780470689646.ch1.

[23] Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. Question generation shared task and evaluation challenge – status report. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 318–320. Association for Computational Linguistics, 2011. URL http://aclweb.org/anthology/W11-2853.

[24] Jrgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85117, Jan 2015. ISSN 0893-6080. doi: 10.1016/j.neunet.2014.09.003. URL http://dx.doi.org/10.1016/j.neunet.2014.09.003.

[25] Bayan Shawar and Eric Atwell. Chatbots: Are they really useful? *LDV Forum*, 22: 29–49, 01 2007.

[26] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 926–934. Curran Associates, Inc., 2013. URL http://papers.nips.cc/paper/5028-reasoning-with-neural-tensor-networks-for-knowledge-base-completion.pdf.

[27] Robert Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. *CoRR*, abs/1612.03975, 2016. URL http://arxiv.org/abs/1612.03975.

[28] Kristina Toutanova and Christopher D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*

held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics -, pages 63–70, 2000. doi: 10.3115/1117794.1117802.

[29] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL 03*, pages 252–259, 2003. doi: 10.3115/1073445.1073478.

[30] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, September 2014. ISSN 0001-0782. doi: 10.1145/2629489. URL `http://doi.acm.org/10.1145/2629489`.

[31] Myers Will, Etchart Tyler, and Fulda Nancy. Conversational scaffolding: An analogy-based approach to response prioritization in open-domain dialogs. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence (ICAART)*, 2020.

[32] Zhicheng Zheng, Xiance Si, Edward Y. Chang, and Xiaoyan Zhu. K2q: Generating natural language questions from keywords with user refinements. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, page 947955, 2011. URL `http://aclweb.org/anthology-new/I/I11/I11-1106.pdf`.

Appendix A

Noun Head Nodes

Edge	Dependency	POS	NER
who	nsubj	PRP	PERSONS or ORGANIZATIONS
	nmod	PRP	PERSONS or ORGANIZATIONS
what	amod	NN*	MONEY
	nmod or nsubj or dobj	NN*	
	xcomp	VB	
	iobj		
why	advcl	VB	
how	amod	JJ* or RB*	
whose		PRP$	
	nmod		
when	nmod	NN*	TIMES or DATES
	nsubj or advcl		TIMES or DATES
where	advcl	NN*	LOCATIONS
	case	IN	
whom	dobj or advcl	PRP	PERSONS
	iobj	PRP	
which		JJ*	

Table A.1: Rules to assign nodes to edges when the head node is a noun: If (Dependency and (POS or NER)), then edge.
* implies any version of the tag like plural, singular, etc.

Appendix B

Dialogue Sets Generated for the Survey

I have a novel.
I read a comic novel in the morning.

Table B.1: Prior knowledge provided to the chatbots for second Dialogue Set

I walked in a park in the morning.
I walk in the gym in the evening.

Table B.2: Prior knowledge provided to the chatbots for third Dialogue Set

P1: Do you read novels?

P2: Yes, I read science fiction novels.

P1: I love science fiction.

P2: Who is your favorite author?

P1: HG Wells. Have you read his novels?

P2: I read The Time Machine.

P1: That is a great one.

P2: Which novels do you read?

P1: I read comic novels.

P2: They are pretty fun!

(a) Dialogues between person (P1) and another person (P2)

P1: Do you read novels?

P2: Yes. I read the novel.

P1: I like fiction novels.

P2: When do you like the novel?

P1: I like to read the novels in the evening.

P2: Ok! You read the novel at evening.

P1: Which novel did you read?

P2: Comic.

P1: Comic novels are pretty fun!

P2: Ok!

(b) Dialogues between person (P1) and ClarQue (P2)

P1: Do you read novels?

P2: Yes. I read the novel.

P1: I like fiction novels.

P2: Ok!

P1: Which novels did you read?

P2: Comic.

P1: Comic novels are pretty fun!

P2: Where is comic novel fun?

P1: Everywhere.

P2: Ok! Comic novel fun at everywhere.

(c) Dialogues between person (P1) and PBot (P2)

P1: Do you read novels?

P2: Which novel do I read?

P1: I do not know, you tell me.

P2: What do you know?

P1: I like fiction novels.

P2: When do you like the novel?

P1: I read novels in the evening.

P2: Where do you read the novel at evening?

P1: I read novels in my room.

P2: When do you read the novel?

(d) Dialogues between person (P1) and Qbot (P2)

Figure B.1: Second Dialogue Set used for the empirical study

P1: I went for a walk in the morning.

P2: Where did you go?

P1: I went to a trail.

P2: I love morning walks.

P1: Ya it refreshes me.

P2: True, I like watching the sunrise.

P1: Where do you go for a walk?

P2: I go to a park.

P1: Do a lot of people go there?

P2: Not many people come there early morning.

(a) Dialogues between person (P1) and another person (P2)

P1: I went for a walk in the morning.

P2: Where did you walk?

P1: Trail.

P2: Ok! You walked to a trail.

P1: Do you walk?

P2: Where do I walk?

P1: Beach.

P2: No. I do not walk at beach.

P1: Where do you walk in the morning?

P2: I walk at the park during morning.

(b) Dialogues between person (P1) and ClarQue (P2)

P1: I went for a walk in the morning.

P2: Alright!

P1: Do you walk?

P2: Where do I walk?

P1: Beach.

P2: No. I do not walk at beach.

P1: Do you like to walk in the morning?

P2: I am not sure.

P1: Where do you walk in the morning?

P2: I walk at the park during morning.

(c) Dialogues between person (P1) and PBot (P2)

P1: I went for a walk in the morning.

P2: Where did you go at walk during morning?

P1: I went to a trail.

P2: Which trail did you go?

P1: It is called YTrail

P2: When is it called?

P1: Do you go for hikes?

P2: Where do I go?

P1: You can go to this one.

P2: When can I go?

(d) Dialogues between person (P1) and Qbot (P2)

Figure B.2: Third Dialogue Set used for the empirical study

Appendix C

Conversations with ClarQue

These are the conversational examples with ClarQue given the prior knowledge. ClarQue considers the tense, aspect, person, spelling mistakes, prior knowledge, and user knowledge before responding to the user.

Prior Knowledge:

She wants fruit in the evening. She wants a cake in the evening. She wants a paper in the evening. She wanted a book in the morning. Jessica went to the park at 3pm. It is a beautiful day. The sun is shining bright. The green grass is refreshing. She saw ducks in the pond. She wants a pen. She ate a salad. She eats fish. She eats chocolates. This morning is beautiful. Jeff will go to the mall today. He took the black book with him. He will want water in the afternoon. I am dancing with him. Today is my birthday. Today is a Wednesday. I will celebrate my birthday today. I will sleep because I am tired. The small house has four windows. The windows are clean. I cleaned the windows.

C.1 Question-Answering with ClarQue

User: What does she eat?
ClarQue: She eats the chocolate, salad, and fish.
U: She loves chocolate.
C: Which chocolate does she love?
U: Dark.
C: Ok! She loves the dark chocolate.
U: What will she eat?
C: She will eat the chocolate.
U: What does she want?
C: When?
U: Evening.
C: She wants the fruit, cake, and paper at evening.

U: She drinks the juice.
C: When does she drink the juice?
U: Morning.
C: Ok! She drinks the juice during morning.
U: What did she drink in the morning?
C: She drank the juice during morning.
U: What will he want?
C: He will want the water.
U: Did she want a flower?
C: No. She did not want the flower.
U: Does she want a book?
C: Yes. She wants the book.
U: When does she want a pen?
C: I am not sure.
U: Where did Jeff go?
C: Jeff went at the mall.
U: When did Jeff go to the mall?
C: Jeff went at the mall during today.
U: What did he take?
C: He took the book.
U: What did he take with him?
C: He took the book with him.
U: Why did I sleep?
C: You slept because tired.
U: I like to eat.
C: What do you like to eat?
U: food
C: Ok! You eat the food.
U: I like to eat.
C: Food?
U: Yes!

C.2 Normal Conversation with ClarQue

U: She wants a laptop.
C: Ok.

U: What did she want?
C: She wanted a laptop.
U: I danced at home.
C: Did you dance at the home with him?
U: Yes!
C: Alright.
U: I will go.
C: Where will you go?
U: mall
C: Ok! You will go at the mall.
U: I will buy a dress.
C: Ok.
U: I will go to a park.
C: Ok! You will go in the park.
U: I will eat in the park.
C: Ok! You will eat food in the park.
U: I will go.
C: mall or park?
U: Park.
C: Ok.
U: It is sunny today?
C: I don't know!
U: I like walking.
C: Alright.
U: Do you like to walk?
C: No. I do not like to walk.
U: Why do you not like to walk?
C: I am not sure.
U: Jessica likes to walk on Monday.
C: Ok! Jessica likes a walk on monday.