An Investigation on Question Answering for an Online Feedable Chatbot

Sameera A'amer Abdul-Kader
Department of Computer Science and Electronic Engineering
University of Essex

A thesis submitted for the degree of
Doctor of Philosophy in Computing and Electronic Systems
November, 2017

I would like to dedicate this thesis to

my family

Acknowledgements

First, I would like to address my sincerest thanks to my supervisor Dr. John Woods, who supported and guided me through my PhD journey, his helpful guidance, advice, humanity, and care.

I also would like to thank the Ministry of Higher Education and Scientific Research (MOHESR) in Iraq for offering me a scholarship, which helped me to undertake my PhD degree.

I'd also like to thank my colleagues and friends at the School of Computer Science and Electronic Engineering, University of Essex for their help and support. Special thanks to my colleagues and friends Mrs. Thabat Thabet, Mrs. Maysa Almulla Khalaf, Miss. Nora Alkhamees, Mrs. Rabab Al-Zaidi, Mr. Mohammed Al-Khalidi and Dr. Abdullah Almuhaimeed for their endless support and advice.

Last but not least, I would like to express my gratitude to my family, my parents, and my husband Ali for their endless trust, support, encouragement and love, as well as critique and advice whenever I was lost or helpless in this journey.

Abstract

This thesis presents the design and implementation of a Chatbot that is able to answer questions about an entity it is learning about. This Chatbot is capable of automatically generating multiple genres using a unique technique to populate its SQL database from the Web. Our Online Feedable Chatbot can hold a conversation with the user regarding the information it has extracted from the Web. Our Online Feedable Chatbot attempts to create Question Answer pairs (QAPs) and acquire imperative sentences specially targeted at the entity it gives information about. A method to select the best response for a Chatbot query among a set of sentences using hybrid terms, syntactic, and semantic extracted features is developed as a response search system of our Online Feedable Chatbot. This tutor Chatbot can expand its training knowledge base by automatically extracting more QAPs and imperative sentences from the Web whenever the user needs to learn about a new entity and without any instructor's supervision, amendments, or control.

Contents

List of Figures

List of Tables

Chapter One

Introduction

1.1 Introduction

Commercial conversational agents have given a boost to the research area of Conversational Agents. Computer based chat systems have become one of the most common communication models used in the modern world [1]. Therefore, there are numerous chat systems available worldwide [1, 2]. Chatbots have been used in various scenarios for getting people interested in different subject areas for decades [2]. However, their ability to teach basic concepts and their engaging effect have not been measured [3, 4].

The World Wide Web (WWW) has developed into a rich information repository in a distributed manner. It is a good trade-off for the information revolution that eventual users are finding it challenging to find relevant information and services easily and quickly [5, 6].

A better compromise is needed among the conversational agents, tutor Chatbots, exploiting the huge information source on the internet and one of these main sources is the WWW. The artificial intelligence Chatbot, which is a category of spoken dialogue systems, is a technology that makes interaction between humans and machines using natural language processing [7] [8]. A general block diagram of a spoken dialogue system is shown in Fig.1.1. A Chatbot is basically structured in a database or knowledge base, a dialogue strategy, Automatic Speech Recognition (ASR) for input speech, and Text To Speech (TTS) conversion for output [9]. Researchers in the area of conversational systems mainly concentrate on improvement of the database and the dialogue strategy [10] [11] [12].

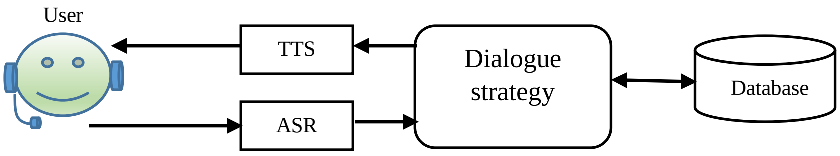


Fig. 1. 1: A general block diagram of a spoken dialogue system.

The main idea presented in this thesis is basically to create a Chatbot that can answer questions about the personality of a figure or the characteristics of an object. This Chatbot has the ability to answer questions in order to present the information it contains about the figure or object i.e. it acts as a tutor to provide the user with information about the figure or object without any instructor control or access to its information. This Chatbot has the capability to populate itself any time and its knowledge base is fillable from the web. The database of the proposed Chatbot can be empty when the user chooses the figure or the object they need to learn about. When the user chooses the figure (any figure randomly) or the object they desire, the Chatbot starts to populate its (SQL) database from the web using, mainly, the Wikipedia page related to that figure or object. After a few hours, the Chatbot Question Answer Pairs (QAPs) database is full of information about the desired person or object. Notice that the Chatbot's database was empty before choosing the property the user needs to learn, then the person or the object is initialised as a Chatbot after a while. Therefore, the Chatbot can be initialised any time the user switches to another object. The database of our Chatbot is accumulative and extendable as the user keeps choosing more entities to learn.

This thesis mainly focuses on how to find out new automatic techniques or methods to populate the knowledge base of the Online Feedable Chatbot from the web. Question Answer Pairs are one of the popular data types to fill in Chatbot's database. The first novel idea is generating factual question answer pairs from factual sentences gathered by a web spider; the raw text sentences are extracted from the HTML and pre-processed. Named Entity (Proper Name) Recognition (NER) is used in addition to verb tense recognition in order to identify the factual sentence category. Specific rules are built to categorize the sentences and then to generate questions based upon them.

Subjective assessment is used to evaluate the Question Answer Generation (QAG) system, and a comparison with other approaches is made. We did not extract questions; we generated questions. There is a measure of error in the generated questions and they are evaluated and given a score. We do not have 100% correct questions and this is explained in Source of Error in section 3.5.

The second idea is to use hybrid (Term, Syntactic, and Semantic) features to design a Question Answer (QA) system that uses multiple feature extraction to filter and quantify the response sentences to a Chatbot query before selecting the best answer for that query. The response sentences are extracted from the same source as in the first idea, which is Wikipedia. The system is evaluated and compared to other comparative systems. The third novel idea is to automatically extract imperative sentences using POS tags and verb tense type from the same source of text as in the first and the second ideas in order for them to be used in future work in actionable activities and added to the designed Chatbot. The footballer David Beckham is used as an example and the data used is acquired from a page about him on Wikipedia. The Chatbot is implemented and comparative systems are chosen and adapted to the data sets of Online Feedable Chatbot (OFC) and implemented for the purpose of comparison to our systems. Subjective assessments are used to evaluate our systems' outputs as well as the comparative systems'. The results of the subjective assessments report that our system performance was better than the comparative systems.

1.2 Motivation

The following questions motivated us to address the problem:

1. Can a Chatbot be a tutor to answer questions and give information about the personality of a figure or characteristics of an object?
2. How can the information about that figure or that object be collected?
3. Can Wikipedia be an information repository to extract adequate and reliable information?
4. What kind of information can the database be populated with?

5. How can we model appropriate methods to populate the database of our Chatbot?
6. What are the best methods to validate our outputs and evaluate our system?

The first step was to research previous studies about Chatbots in order to investigate the recent developments in the area of conversational system research. We also investigated information retrieval studies to track the recent improvements. Furthermore, we looked into the recent methods of Question Generation (QG) and QA systems. After these investigations, we created the idea of designing a tutor Chatbot that can answer questions and give information about the entity that the user selects randomly. Then we found the idea of populating the knowledge base of this Chatbot from the World Wide Web and specifically Wikipedia. Then we started to think about the type of information that can be inserted into our Chatbot database and we found the idea of automatically generating question answer pairs to be considered as part of the knowledge base of our Chatbot. In the meantime, we found a way to extract another type of information to populate the database and this was imperative sentences. Moreover, we thought of a method to find the best match for a question among a group of sentences as a response to a Chatbot query. Furthermore, our invented QA data set and Stanford data set (Stanford Question Answer Dataset) SQuAD was used to test our QA system in chapter 4.

In order to evaluate the outcomes of the system, we have chosen the idea of using subjective assessment, since the main interaction of the conversational agents is with people and their task is to talk to humans [13]. Therefore, we needed to evaluate our system by examining human satisfaction as humans are the eventual users for any Chatbot. We used human assessment for three of our methods: the QG system (Chapter 3), the Imperative sentence extraction system (Chapter 5), and the implementation of our Chatbot (Chapter 6). Other evaluation metrics were chosen for the QA system (Chapter 4) such as Precision, Recall, Mean Average Precision (MAP), and Mean Reciprocal Rank (MRR) [14]. Precision is also used to evaluate the results of the subjective assessments in the QG system, and Imperative sentence extraction system.

In order to implement the Chatbot, a plan for an algorithm has been derived and an analysis of the conversation form has also been written. The plan is divided into stages and the first stage of implementing our Online Feedable Chatbot (OFC) is presented in Chapter 6 and the other implementation stages are explained in section 7.3 in Chapter 7.

1.3 Research Objectives

1.3.1 Online Feedable Chatbot

The first main objective of this thesis is to design and implement a Chatbot that has information about the personality of a figure or the characteristics of an object. This Chatbot should:

1. Populate its database automatically from the internet. We found an idea to populate our Chatbot's database from the Wikipedia page of the target figure or the object [15].
2. Hold a simple conversation with the user like greetings and asking about name.
3. Answer questions on the entity it has information about.
4. Use Wikipedia to extract information about the figure or the object it answers questions about.
5. Update its database when a user needs information about a new entity. This means adding to the already available information (accumulative database).

1.3.2 Automatic Question Generation

The second main objective is to find the best methods to design and populate the database of the proposed Chatbot. The idea is to develop a QG system to produce QAPs for the proposed Chatbot database. This system should:

1. Extract plain text from the Wikipedia page of the figure or the object that a user desire using a web crawler.
2. Filter the extracted text from HTML and undesired codes using natural language processing (NLP) like POS tagging.
3. Acquire factual sentences from the text that was extracted from the Wikipedia page associated to the user selected entity.

4. Detect the proper names at the beginning of each factual sentence and the verb tense in order to use them for question generation.
5. Select the factual sentences with proper name subject and simple present and simple past verb tenses.
6. Generate *Wh* factual questions from the selected sentences.
7. Pair the generated questions with the selected factual sentences to make QAPs then put the QAPs into an SQL database.

1.3.3 Question Answer System

The third main objective of this thesis is to develop a model of response answer selection for a query from a database. This model investigates the best ways to extract features from the query and the response answers. We needed to find a method to obtain the best match between a query and a response answer and we designed a system for this purpose. This answer selection system should:

1. Extract raw text from the Wikipedia page of the desired entity using a web crawler.
2. Filter the extracted text using NLP.
3. Split the text into sentences using NLP word tokenizing and then POS tag each word.
4. After posing a question, Split the query sentence using NLP word tokenizing then POS tag the tokenized words.
5. Use POS tags to find syntactic match, Jaccard's coefficient or cosine similarity to find term match, and named entity or semantic cosine similarity to find semantic similarity between the query and the response sentences then aggregate the matching scores for each response sentence.

The highest scored sentence should be the best to match the query sentence.

1.3.4 Imperative Sentence Extraction

The fourth main objective of this thesis is to find a method to extract imperative sentences. A system is designed to support this method and this system should:

1. Do the same steps of text extraction, filtering, sentence then word tokenization, and POS tagging done in the previous two sections which are all named pre-processing.
2. Detect the verb tense in the beginning of each sentence.
3. Select the sentences that begin with simple present verbs and put them into the SQL database.

1.4 Contributions

The work presented in this thesis contributes to designing and building a Chatbot that depends mainly on its knowledge from data acquired from the Web. It also contributes to proposing a new method of automatic question generation, a new method to extract multiple features for a QA system, and a new method to automatically extract imperative sentences from the Web.

The main contributions are listed as follows:

- **Online Feedable Chatbot (OFC).** Proposing and building a tutor Chatbot that can feed its knowledge base from the Web. This Chatbot has the ability to:
    1. Make conversational chat with the user using the chatting database.
    2. Answer definition, information, and new queries about the figure or object.
    3. Populate its knowledge base with information from the web about a figure or an object whenever a user chooses. After a few hours of running, its SQL database is populated by QAPs about the user selected figure or object.
    4. Extend its database and categorise it into genres depending on the subjects added according to user request.

    The database of this Chatbot is built using SQL. New approaches are used to extract text from the web and generate QAPs as chat information for the OFC in addition to imperative sentences. The OFC is implemented and tested using human assessment.

- **Automatic Question Generation System for OFC.** We built a new automatic system to generate questions from factual sentences, which are extracted from the web using a web crawler and filtered using part of speech information. Specific hypotheses and rules are built for selecting the factual sentences and to produce questions from these sentences. We used named entity and verb tense features to carefully select the corresponding sentences and the generated questions have been categorised according to verb tense type. This part was published in [16]. We have run an experiment to implement the system and evaluated the resultant QAPs. We adapted a comparative system to our QG system and data set in order to compare it with our system. A subjective assessment has been used to validate and evaluate our system's output as well as the comparative system's. The evaluation results show that our QG system outperforms the comparative system by 5 percentage points. Moreover, our questions are more answerable than the comparative systems' according to the QA match scores. The resultant QAPs are stored in an SQL database as part of the knowledge base of OFC.

- **Question Answer System for the OFC.** We built a QA system for the OFC that uses hybrid features to select the best response for a query in a Chatbot. Multiple terms, syntactic, and semantic features are used to find the similarity between a query and a set of sentences extracted from the web, and filtered using part of speech POS tags. We used cosine similarity and Jaccard's Coefficient for the term similarity, POS tag for syntactic features, and named entity and semantic cosine similarity for semantic features. Four hybrid formulas are produced from these similarity features and compared to each other. The resultant selected sentences are re-ranked according to the obtained similarity scores from the highest to the lowest. We assume that the highest scored sentences are the best responses to the query. A comparative system has been adapted to our system and our data set so as to compare it with our four produced equations. Our produced QA data set and Stanford data set SQuAD were used to test our system. Precision, recall, mean average precision, and mean reciprocal rank have been calculated and graphs were produced so as to compare the five comparative systems (our four combinations and the comparative system). We found that the best combination to find the best response for a Chatbot query is; cosine similarity for term, POS tag for syntactic, and semantic cosine similarity for semantic and this part was published in [17]. It is planned to use this QA system as the response selection system for our Chatbot OFC from the SQL database or from the Web.

- **Automatic Imperative sentence extraction System for the OFC**. We built an automatic system to extract imperative sentences from the web. Verb tense type and a POS tag are used to extract direct imperative sentences from a group of sentences extracted from the Wikipedia page of the English footballer David Beckham using a web crawler and filtered using a part of speech tag. A comparative system was chosen and adapted to the data set of the OFC and implemented for the purpose of comparison to our system. A subjective assessment was implemented using human assessors to evaluate our system's output as well as the comparative system's. The subjective test result reports that our system outperforms the comparative system. The resultant imperative sentences are saved in an SQL database as part of the OFC database.

1.5 Thesis Outlines

The remainder of this thesis is structured as follows:

- Chapter 2 gives a background for some important terminologies, introduces a literature review of the state of the art regarding the problem domain, concentrating on the Online Feedable Chatbot, Tutor Chatbots, Automatic Question Generation, Question Answer Systems, and Extracting Imperative sentences. This provides a motivation for the subsequent chapters where we present the detailed theories and hypotheses, justifications, experimental work and evaluation results and analysis.

- Chapter 3 presents the QAG system's rules and hypothesis. It explains a generalised abstract design of the proposed system. It gives details about the experiment conducted to implement and evaluate the proposed system. It also demonstrates the evaluation results in comparison to the comparative system.

- Chapter 4 describes feature extraction methods used in term, syntactic, and semantic similarities. It proposes a method for a QA system to be used in the OFC. Moreover, it presents the proposed system and its implementation. It also provides the experimental results and how they are evaluated and compared to other systems.

- Chapter 5 explains the rules and the hypothesis to extract imperative sentences from the web. It also describes the proposed system to apply the corresponding hypothesis. Furthermore, it presents the experimental results and the evaluation way to assess the proposed system and to compare it with other studies.

- Chapter 6 gathers all the methods presented in the former chapters to design the OFC by implementing this Chatbot. The chapter provides diagrams and an explanation for the structure of the Chatbot and the discourse analysis. In addition, it reports the evaluation results using human assessors.

- Chapter 7 summarises the additive conclusions of the earlier chapters and identifies the challenges which can be faced through further development of our Chatbot in the future.

The next section is included to clarify which sections of this thesis have been published.

1.6 Publications

1. Abdul-Kader, Sameera A., and John Woods, "Survey on Chatbot design techniques in speech conversation systems." *Int. J. Adv. Comput. Sci. Appl.(IJACSA)* 6.7, 2015.
2. Abdul-Kader, Sameera A., and John Woods, "Question Answer System for an Online Feedable Chatbot " *Intelligent Systems Conference*, London, UK, 2017.
3. Abdul-Kader, Sameera A., and John Woods, "Automatic Web-based Question Answer Generation System for Online Feedable New-Born Chatbot". *IEEE Computing Conference*, London, 2018.

Chapter Two

Background

(Part of this chapter is published [18])

2.1 Introduction

Speech is one of the most powerful forms of communication between humans; hence, it is the ambition of the researchers in the field of human computer interaction to improve speech interaction between humans and computers. Speech interaction with modern networked computing devices has received increasing interest in the past few years with contributions from Google, Android and IOS. Spoken dialogue systems are becoming the primary interaction method with a machine because they are more natural than graphic-based interfaces [19]. Therefore, speech interaction will play a significant role in humanising machines in the near future [20].

Much research work has focused on improving recognition rates of the human voice and the technology is now approaching viability for speech based human computer interaction. Speech interaction splits into more than one area [21], including speech recognition, speech parsing, NLP (Natural Language Processing), keyword identification, Chatbot design/personality, artificial intelligence and so on.

Chatbots have been the new trend of the information technology world in the past few years because of the convenience and ease of their usage. A Chatbot is a computer program that has the ability to hold a conversation with humans using Natural Language Speech [18]. It can interact with humans and perform different tasks such as booking trains or flights, ordering food, dialling a person's number, telling the weather forecast, and teaching foreign languages. Commercial examples of such types of Chatbots are MS Cortana, Siri, and Google Assistant.

This chapter presents definitions and identification for the important terminologies used in this thesis. Then, a literature review categorised into detailed topics is presented.

2.2 Important Concepts

In order to know the fundamental Chatbot characteristics and strategies, we need to figure out some preliminary terminologies related to the area of Chatbots or conversational systems. Some of these terms are explained in the following subsections. Each section is explained separately and independently from the other as a title related to the work in this thesis.

2.2.1 Human Computer Speech Interaction

Speech recognition is one of the most natural and sought-after techniques in computers, and networked device interaction have only recently become possible (in the last two decades) with the advent of fast computing.

Speech is a sophisticated signal and happens at different levels: "semantic, linguistic, articulatory, and acoustic" [22]. Speech is considered to be the most natural among the aspects of human communication, owing to copious information implicitly existing beyond the meaning of the spoken words. One of the speech information extraction stages is converting speech to text via Automatic Speech Recognition (ASR) and mining speech information [23, 24]; then, the resulting text can be treated to extract the meaning of the words.

Speech recognition is widely accepted as the future of interaction with computers and mobile applications; there is no need to use traditional input devices such as the mouse, keyboard or touch sensitive screen and it is especially useful for users who do not have the ability to use these traditional devices [25, 26]. It can help disabled people with paralysis, for example, to interact with modern devices easily by voice only without moving their hands. This part is not going to be adopted in this thesis, but we found it important to explain it as a technique used to input user queries into a Chatbot.

Speech analysis can be divided into three stages: (i) voice recognition and conversion to text, (ii) text processing, and (iii) response and action taking. These stages are explained as follows:

Firstly, speaker independent speech passes through a microphone to a digital signal processing package built in the computer to convert it into a stream of pulses that contain speech information. Specific instructions are used to read input speech then to convert it into text [27]. This stage provides speech text for processing in the next stage. The diagram, which illustrates this stage, is shown in fig.2.3.

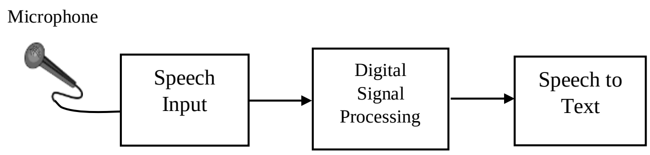


Fig. 2. 1: The stage of speech recognition and converting to text.

Secondly, the resulting text is split into separate words for tagging with part of speech labels according to their positions and neighbours in the sentence. Different types of grammar can be used in this stage to chunk the individual tagged words in order to form phrases [18]. Keywords can be extracted from these phrases by eliminating unwanted words in chinking operations. These keywords can be checked and corrected if they are not right. The phases of the text processing stage are shown in fig.2.4.

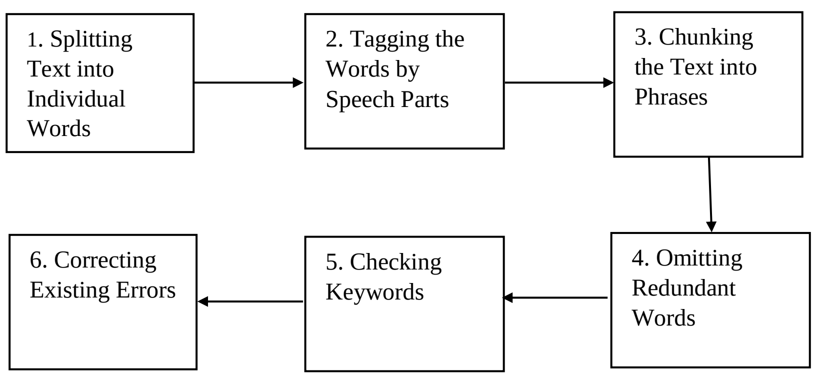


Fig. 2. 2: The stage of text processing.

Finally, a Chatbot can be built to give the desired intelligent response to a natural language speech conversation. The input to this Chatbot is keywords released from the speech text processing; the output is the programmed response, which will be, for example, an application running or any other text or speech response. Fig.2.5 shows a brief diagram of the third stage [28].

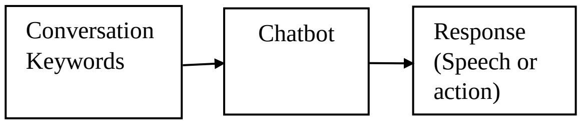


Fig. 2. 3: The stage of response and action taking

Conversation techniques between a human and a computer can be either chatting by typing text or speech dialogue using the voice. The processing of the information in both techniques is the same after converting speech to text in the case of speech dialogue. A diagram showing the main steps of analysis and processing required to perform human computer conversation is shown in fig.2.6 [18].

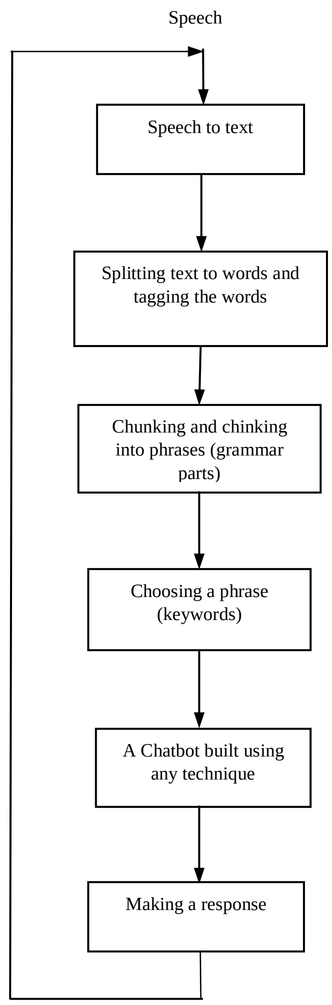


Fig. 2. 4: The main steps of analysis and processing to perform human computer conversation.

2.2.2 Chatbot Strategies

To give suitable answers to keywords or phrases extracted from speech and to keep conversation continuous, there is a need to build a dialogue system (programme) called a Chatbot (Chatter-Bot). Chatbots can assist in human computer interaction and they have the ability to examine and influence the behaviour of the user [29] by asking questions and responding to the user's questions. The Chatbot is a computer programme that mimics intelligent conversation. The input to this programme is natural language text, and the application should give an answer that is the best intelligent response to the input sentence. This process is repeated as the conversation continues [30] and the response is either text or speech.

Writing a perfect Chatbot is very difficult because it needs a very large database and must give reasonable answers to all interactions. There are a number of approaches to create a knowledge base for a Chatbot and these include writing by hand and learning from a language corpus. Learning here means saving new phrases and then using them later to give appropriate answers for similar phrases [31].

Designing a Chatbot software package requires the identification of the constituent parts. A Chatbot can be divided into three parts: Responder, Classifier and Graphmaster (as shown in fig. 2.1) [32], which are described as follows:

1. Responder: the part that plays the interfacing role between the bot's main routines and the user. The tasks of the responder are: transferring the data from the user to the Classifier and controlling the input and output.
2. Classifier: this is the part between the Responder and the Graphmaster. This layer's functions are filtering and normalising the input, segmenting the input entered by the user into logical components, transferring the normalised sentence into the Graphmaster, processing the output from the Graphmaster, and handling the instructions of the database syntax (e.g. AIML).
3. Graphmaster: this is the part for pattern matching that does the following tasks: organising the Chatbot's storage and holding the pattern matching algorithms.

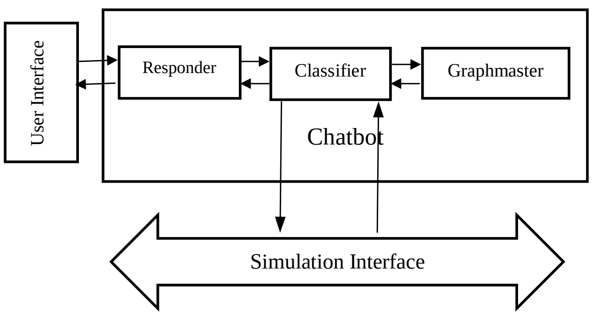


Fig. 2. 5: Components of Chatbot.

2.2.3 Chatbot as Part of Dialogue Systems

In the field of Artificial Intelligence, Turing was the first to pose the question, "Can a machine think?" [33], where thinking is defined as the ability held by humans. According to this question and this definition, Turing suggests the "imitation game" as a method to directly avoid the question and to specify a measurement of achievement for researchers in Artificial Intelligence [34] if the machine appears to be human. The imitation game is played between three people: (A) which is a man, (B) which is a woman, and (C) which is the interrogator and can be either a man or a woman. The aim of the interrogator here is to determine who the woman is and who the man is (A and B). The interrogator knows the two as labels X and Y and has to decide at the end of the game either "X is B and Y is A" or "X is A and Y is B". The interrogator also has the right to direct questions to A and B. Turing then questions what will happen if A is replaced with a machine; can the interrogator differentiate between the two? The original question "Can machines think?" can make the new question explicit [33, 35]. In this imitation game, the Chatbot represents the machine and it tries to mislead the interrogator into thinking that it is the human, or the designers who try to programme it to do so [36].

In 1990 an agreement was made between Hugh Loebner and The Cambridge Centre for Behavioural Studies to establish a competition based on implementing the Turing Test. A Gold Medal and $100,000 were offered by Hugh Loebner as a Grand Prize for the first computer that makes responses which cannot be distinguished from humans [18]. The important thing in this competition is to design a Chatbot that has the ability to drive a conversation. During the chat session, the interrogator tries to guess whether they are talking to a programme or a human. After a ten-minute conversation between the judge and a Chatbot on one side and the judge and a confederate independently on the other side, the judge has to nominate which one was the human. The scale of non-human to human is from 1 to 4 and the judge must evaluate the Chatbot in this range [36]. According to this judgement, the more human Chatbot is the winner. No Chatbot has ever achieved the gold medal and passed the test to win the Loebner Prize. However, some Chatbots have scored as highly as 3 out of the 12 judges believing they were human [18].

A Chatbot is a software system that is capable of interacting (chatting) with real people in natural language [37]. Chatbots can receive natural language input that is text or speech interpreted using speech recognition software, and can execute appropriate commands to engage in a desired behaviour. As intelligent agents, they are normally reactive, autonomous, social, and proactive. Chatbot is a category of conversational agents, which are software programs that mimic conversations with humans. They are typically not embodied in the forms of animals, avatars, humans, or humanoid robots (those programs are considered as "embodied conversational agents"). Conversational agents are a class of dialogue system and have been an interest of research in communications for decades. Interactive Voice Response (IVR) systems are dialogue systems as well. However, they are not considered as conversational agents because they implement decision trees [38, 39]. Types of dialogue system and their relations to each other are shown in fig.2.2 [40].

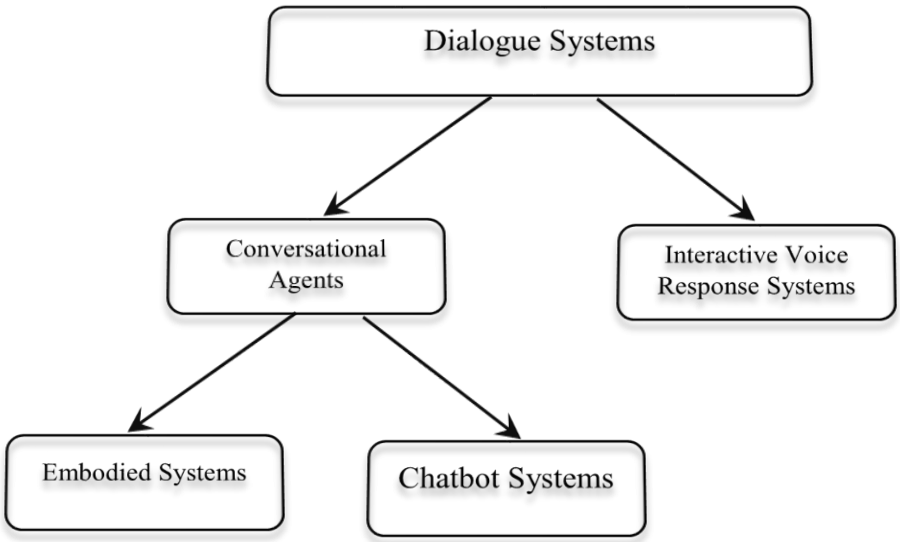


Fig. 2. 6: Dialogue systems types and their relations to each other.

Our implemented Chatbot in this research is a Chatbot system that is part of conversational agents from the category of dialogue systems.

To build a dialogue system (Chatbot) program, one of the most essential requirements is to design a sufficiently detailed database for that system. As the Chatbot bases its knowledge on statements or sentences and uses them to hold a conversation, it needs a large but not overlapping knowledge base. Chatbots can assist in human computer interaction and they have the ability to examine and influence the behaviour of the user by asking questions and responding to the user's questions. The Chatbot is a computer programme that mimics an intelligent conversation. The input to this program is natural language text, and the application should give an answer that is the most relevant intelligent response to the input sentence. This process is repeated as the conversation continues and the response is either text or speech [18]

Trying to build a Chatbot is a big challenge. The challenge is collecting and processing the data that is used to populate the Chatbot database because the only knowledge the Chatbot has access to is the information it has learnt itself. Therefore, the data fed into the Chatbot should be selected and filtered carefully using statistical and numerical means.

A Chatbot can learn general facts but is often focused towards a specific figure or object and its database can be updated from the web according to a user request (i.e. the user can choose the figure or the object they need).

2.2.4 Chatbot Fundamental Design Techniques and Approaches

To design any Chatbot, the designer must be familiar with a number of techniques [41] these techniques are listed below:

1. **Parsing**: this technique includes analysing the input text and manipulating it by using a number of NLP functions, for example, trees in Python NLTK. This technique is not used in this research.
2. **Pattern matching**: this is the technique that is used in most Chatbots and it is quite common in QA systems, depending on matching types, such as natural language enquiries, simple statements, or the semantic meaning of enquiries [42]. This technique is used in this work to select a suitable answer to a Chatbot query.
3. **AIML**: this is one of the core techniques that is commonly used in Chatbot design. More details about this technique and the language used are explained in Section 2.2.6 below. This technique is not employed in our work.

4. **Chat Script**: this is the technique that helps when no matches occur in AIML. It concentrates on the best syntax to build a sensible default answer. It gives a set of functionalities such as variable concepts, facts, and logical AND/OR [43]. We do not use Chat Script in this work.
5. **SQL and relational database**: this is a technique used recently in Chatbot design in order to make the Chatbot remember previous conversations. More details and explanation are provided in Section 2.2.7 below. This database type is used to build our Chatbot's database.
6. **Markov Chain**: this is used in Chatbots to build responses that are more applicable probabilistically and, consequently, are more correct. The idea of Markov Chains is that there is a fixed probability of occurrences for each letter or word in the same textual data set [44]. We do not use Markov Chain in our work.
7. **Language tricks**: these are sentences, phrases, or even paragraphs available in Chatbots in order to add variety to the knowledge base and make it more convincing. The types of language tricks are:
   - Canned responses;
   - Typing errors and simulating key strokes;
   - Model of personal history;
   - Non-Sequitur (not a logical conclusion).

   Each of these language tricks is used to satisfy a specific purpose and to provide alternative answers to questions [44]. We use the first two techniques (Canned responses and typing errors and simulating key strokes) in our Chatbot.

8. **Ontologies**: these are also named semantic networks and are a set of concepts that are interconnected relationally and hierarchically. The aim of using ontologies in a Chatbot is to compute the relation between these concepts, such as synonyms, hyponyms and other relations which are natural language concept names. The interconnection between these concepts can be represented in a graph enabling the computer to search by using particular rules for reasoning [44]. Ontologies are not used in this work.

2.2.5 SQL

A relational data base (RDB) is one of the database types used to build Chatbot knowledge bases. The technique has been used to build a database for a Chatbot, i.e. to enable the Chatbot to remember previous conversations and to make the conversation more continuous and meaningful. The most familiar RDB language is SQL (Structured Query Language), which can be used for this purpose.

SQL, or MYSQL has gained a high recognition in RDB because it is the most popular open source database system for nonprocedural data. Query blocks nesting to arbitrary depths is one of the most interesting features, and the SQL query is divided into five basic kinds of nesting. Algorithms are developed to change queries that include these basic nesting types into "semantically equivalent queries". Semantically equivalent queries are adjustable to achieve effective processing via existing query processing subsystems. SQL as a data language is implemented in ZETA; also as a calculus-based and block-structured language, it is implemented in System R, ORACLE, as well as SEQUEL [45, 46]. Some researchers, as can be seen in the next sections, have recently used SQL to generate a database that saves the conversation history in order to make a search for any word or phrase match easier. This technique gives continuity and accuracy to the dialogue because it enables the dialogue system to retrieve some previous information history.

2.2.6 Named Entity

One of the main elements in the QA is Named Entity Recognition to extract information from text. The NER consists of three groups:

1. Entity names.
2. Number expressions.
3. Temporal expressions.

Temporal expressions identify time entities, such as date, and time, and number expressions identify number entities like monetary values [47]. Numbered and temporal expressions are not the interest of this work. Entity names can annotate unique identifiers for the proper nouns that they represent: PERSON, ORGANIZATION, LOCATION (GPE), and FACILITY names in the text (see fig. 2.7 below) as in the NER.

The NER system normally recognize the string that represents the entity name then identifies it as the named entity specifying the type as in the example below:

The sentence:

"Rami Eid is studying at Stony Brook University in NY"

contains entities of proper names. Applying NER operation gives the following:

[[('Rami', 'PERSON'), ('Eid', 'PERSON')], [('Stony', 'ORGANIZATION'), ('Brook', 'ORGANIZATION'), ('University', 'ORGANIZATION')], [('NY', 'LOCATION')]].

The noticeable issue here is that we have two main entity names: 'Rami Eid' and 'Stony Brook University in NY'. This chained type of entity name is called a *Cascaded Entity Name* and the existing (Python) kinds of NER recognise one word entities [48]. Therefore, the cascaded names are separated during the NER application because of the limitations in the Stanford and NLTK NER modules. Hard code is needed in the implementation part to re-join the same name's separated words and obtain the following form of output:

[('Rami Eid', ' PERSON '), ('Stony Brook University', ' ORGANIZATION ')]

In this work, NER is used to detect the proper name subjects at the beginning of the sentences because we need to ask questions about the subject of the sentence; and identify their classes in order to determine question-words during the question generation process. NER is also used in this work as a feature extracted to examine the semantic similarity between a query and a group of response answers.

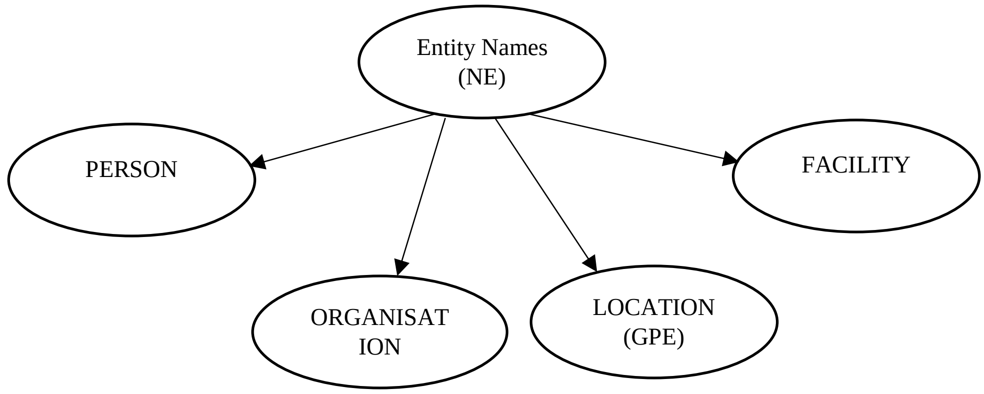


Fig. 2. 7: Entity Name classes as in the NLTK_NE library.

2.2.7 Question Answering

Question answering is a topic of Information Retrieval and an NLP domain interest. The authors in [49] think that it needs more cooperation between the communities of Knowledge Representation and NLP [49]. The Question Answering system is normally a mechanism embedded within sophisticated search engines that has featured in TREC 8 (Text Retrieval Conference 8) [50]. A QA system normally retrieves a particular piece of information from the web to select the optimum answer to a user query. The concepts and rules of TREC developed over a number of years to expand the range of question sets and to choose answers that are more accurate. The rules proposed in this thesis reverse the concept of putting or setting the question, analysing it, and then finding the best answer for it. It is proposed to retrieve the piece of information, classify the sentences after extracting, analyse the sentences, and then generate questions from the existing statement to give an answer. The analysis techniques used in both cases (Question lead to Answer or Answer lead to Question) are the same: part of speech tag, ngrams, and NER.

There are three levels of questions we will use when creating or answering questions from a piece of text [51] [52] .

1. Factual: In this level, the questions are answered explicitly by facts contained in the text. The answers to these questions are clear in the text. These questions

are really basic and literal. For example if we have the following text from the Cinderella story [51] [52]:

“Cinderella’s mother died while she was a very little child, leaving her to the care of her father and her two step-sisters, who were very much older than herself, for Cinderella’s father had been twice married, and her mother was his second wife. It happened, when Cinderella was about seventeen years old, that the King of that country gave a ball, to which all the ladies of the land, and among the rest, the young girl’s sisters, were invited. And they made her dress them for the ball, but never thought of allowing her to go there. When they were gone, Cinderella, whose heart was very sad, sat down and cried bitterly; but as she sat sorrowful, thinking of the unkindness of her sisters, a voice called to her from the garden, and she went out to see who was there. It was her godmother, a good old Fairy [53]”

The factual questions are:

- Who died when Cinderella was little?
- How many step-sisters does Cinderella have?
- Who did the King invite to the ball?

2. Interpretive: this type of question is textually implicit, needing analysis and interpretation of specific parts of the text. The reader needs to apply their knowledge to the text. The reader has to read between the lines (infer) the answer to the questions on this level. For example, for the same piece of text above, the questions would be [51] [52]:
    - Why doesn’t Cinderella’s step-family want her to attend the ball?
    - Do you think the prince would have fallen in love with Cinderella even if she hadn’t received a makeover?
    - In what ways does the Fairy Godmother assist Cinderella?
    - How do you think Cinderella feels about her step-family?

3. Evaluative: are more open-ended and go beyond the text. They are intended to stimulate a discussion of an abstract idea or issue. These questions ask So what? Why does it matter? For example from the text above the evaluation questions would be [51] [52]:

- Why do step-parents often have difficulty getting along with their step-children?
- Do you believe in love-at-first-sight?
- Has anyone ever given you help when you least expected it?

In this work, straightforward factual type questions are generated from the text extracted from the web and pre-processed using part-of-speech tag. Moreover, 'Wh' question words are specifically used to generate questions in this work.

Wh questions are the type of questions that usually start with a word beginning with 'wh' such as what, who, where, etc. but 'how' is also included. This type of question is used to ask for information [54] [55]. For example:

- Who are you talking to?
- What is it?
- Where do you live?

2.2.8 Part of Speech Tags

A large number of language processing systems use a part of speech tagger for pre-processing [56]. In corpus linguistics, part of speech tagging (POS tagging or PoS tagging or POST) is the process of labelling a word in a text according to a particular part of speech [57] depending on both its definition and its context (meaning the relationship of such a word with adjacent and related words in a phrase, sentence, or paragraph). The simplest form of this is commonly taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs, and so on [58]. It was done by hand in the past, whereas POS tagging is now performed in the context of computational linguistics, using algorithms which relate to discrete terms, and hidden parts of speech, in accordance with a collection of descriptive tags [58].

The tagger works by automatically recognising and treating its own weaknesses, by incrementally improving its performance. The tagger initially tags by allocating each word its most likely tag, estimated by inspecting a large tagged corpus, without reference to context [57]. POS tagging is used in this work for question answer comparison to find the similarity between them (Chapter 3 and Chapter 4) and for imperative sentence extraction (Chapter 5).

2.2.9 Semantic Role Labels

In the field of artificial intelligence, semantic role labelling, sometimes also called shallow semantic parsing, is a process in natural language processing that allocates labels to words or phrases in a sentence that specifies their semantic role in the sentence, like that of an agent, goal, or result [59]. It includes the detection of the semantic arguments related to the predicate or the verb of a sentence and their classification according to their specific roles. For example, given a sentence like "Layan sold the book to Mustafa", the task is to recognize the verb "to sell" as representing the predicate, "Layan" as representing the seller (agent), "the book" as representing the goods (theme), and "Mustafa" as representing the recipient [60]. A semantic analysis of this kind is at a lower level of abstraction than a syntactic analysis because it has more categories and groups with fewer clauses in each category. For instance, "the book belongs to Layan" would need two labels such as "possessed" and "possessor" whereas "the book was sold to Mustafa" need two other labels such as "goal" (or "theme") and "receiver" (or "recipient") in spite of that, these two clauses would be very similar as long as "subject" and "object" functions are examined [59]. A form of semantic role labels (WordNet) is used in this work to detect the similarity in meaning between a query and an answer (Chapter 4).

2.2.10 Word Embedding

This is the common name for a collection of feature learning and language modelling techniques in NLP where words or phrases from the lexicon are mapped onto real number vectors. Basically it includes a mathematical embedding from a space with one dimension for each word to a continuous vector space with a lower dimension [61] [62] [63].

Word and phrase embedding are used as the underlying input representation to boost the performance in NLP tasks such as sentiment analysis and syntactic parsing [64].

2.2.11 Natural Language ToolKit (NLTK)

In this work, we need to split words in a string of text and separate the text into parts of speech by tagging word labels according to their positions and functions in the sentence. The resulting tagged words are then processed to extract the meaning and produce a response as speech or action as required. Different grammar rules are used to categorise the tagged words in the text into groups or phrases relating to their neighbours and positions. This type of grouping is called chunking into phrases, such as noun phrases and verb phrases.

In this thesis, NLP pre-processing operations like tokenization and POS tagging are needed for our extracted text manipulation. These NLP operations are done using the NLTK.

2.2.12 Machine Learning

This is a field of computer science that employs statistical techniques to give the ability to computer systems to learn with data, without being clearly programmed [65]. The name machine learning evolved from the study of computational learning theory and pattern recognition in artificial intelligence [65]. It explores the study and the building of algorithms that can learn from and make predictions on data [66].

Machine learning is related to, or overlaps with, computational statistics that also focus on prediction-making via the use of computers. It also has relations with mathematical optimization, which provides methods, theory and application domains to the field [66].

Machine learning tasks are classified into two main categories, according to their learning "signal" or "feedback" availability to a learning system [67]:

- Supervised learning: the computer is provided with example inputs and their expected outputs, given by a "teacher", and the aim is to learn a general rule that maps inputs to outputs. The input signal can be partially available, or restricted to special feedback. Semi-supervised learning, active learning, reinforcement learning are branches of supervised learning [68].

- Unsupervised learning: in this category of machine learning, no labels are given to the learning algorithm, it is left on its own to structure its input. Unsupervised learning is a goal in itself to discover hidden patterns in data or a way towards an end as in feature learning [69].

Machine learning applications is another categorization for machine learning tasks which emerges when considering the desired *output* of a machine-learned system. Some machine learning applications are [70]:

- Classification.
- Regression.
- Clustering.
- Density estimation.
- Dimensionality reduction.

2.2.13 Artificial Neural Networks (ANNs)

These are also named connectionist systems or neural intelligent systems which are computing systems that are inspired by the biological neural networks that constitute animal brains [71]. These systems learn to do tasks by considering examples, without being programmed with any task-specific rules. For example, in image recognition, they could learn to identify images that contain cats by analysing example images that have been manually labelled as "cat" or "no cat" then using the results to recognise cats in other images [72]. They do this without any previous knowledge about cats, for example that they have cat-like faces, tails, and whiskers. They automatically create identifying characteristics from the learning material that they process [73].

An ANN idea depends on a group of connected units or nodes called artificial neurons which model the neurons in a biological brain. Each connection, like the synapses in a biological brain, transmits a signal from one artificial neuron to another. The artificial neuron that receives a signal, processes it, and then stimulates additional artificial neurons connected to it [73].

The original aim of ANNs was to solve problems in the same way that a human brain does. However, attention has recently moved to performing specific tasks and this lead to deviations from biology. Artificial neural networks are used in a variety of tasks, such as speech recognition, computer vision, machine translation, playing board and video games, social network filtering, and medical diagnosis [72].

2.3 Literature Review

Designing an Online Feedable Chatbot and populating it from the web is a new area of research. Few researchers have investigated the possibility of educating a new Chatbot that embodies an artificial figure. Some authors suggest extracting Chatbot knowledge from the discussion forums available online [74, 75]. Others start database population from the web or plain text depending on a particular object or person [15]. The data extracted from web pages needs significant processing before it is ready for conversational systems, especially if the text is unstructured like that in Wikipedia. Filtering and analysis are required for some text in the database to have meaning. One of the popular forms of data in Chatbot knowledge bases is QAPs. The majority of QAPs are either written manually or acquired from existing online discussion forums [75]. Generating QAPs for a Chatbot requires much processing and filtering of the sentences to generate the corresponding questions. Some kind of hypothesis is necessary to build a framework to derive questions from sentences [76]. The hypothesis then needs to be converted to specific rules and processes to mechanically populate the database. Validation is also applied to the QAPs and compared with other question generation systems. The following categorised literature review demonstrates the details of the development of question generation and question answering in the field of online feedable and tutoring Chatbots and how their databases are populated with information.

2.3.1 Online Feedable Chatbot

Since the first Chatbot, Eliza, was presented, Chatbot knowledge has been manually scripted or learning knowledge has been added using user responses [77]. Different Chatbots or dialogue systems have been developed utilising speech or text communication and applied to a variety of domains such as language education, linguistic research, website help, customer service, and fun. However, the majority of the Chatbots have been restricted to knowledge that is hand coded in their files, and to specific natural languages that are written or spoken [77, 78].

Some agents employ user responses to reply to other users; authors explore the notion of self-agency in developing agent-based systems that support human-to-human communication. They refer to the fact that a challenge in developing an agent-based system is the transfer of conversational experiences that agents gain to their users. Self-agency is presented as a key to address this challenge. An experimental system is also presented to inspect the sense of self-agency impact on the successful conversational experiences transferred from agents to their users [79]. However, this is a form of managing conversations rather than creating them. Other type of Chatbots are used to gather information from chatting to users. A Chatbot is used as a question answer interface. To automatically retrain the Chatbot knowledge-base, the TRE09 QA track is used. The aim is to improve the algorithm for building the database [80]. However, this keeps the Chatbot knowledge restricted to user's knowledge.

Chatbots use databases of responses often chosen from a corpus of text created for a different purpose; for instance, film scripts or interviews [81]. Existing corpora are converted to Chatbot style as a trial to extend Chatbot knowledge. A software is presented to convert a corpus to a specific Chatbot format, which is used after that to re-train a Chatbot and create a chat closer to human language. Different corpora are used such as dialogue corpora, like the British National Corpus of English (BNC), the holy Quran, which is a monologue corpus, and the FAQ where questions and answers are pairs. The aim of this automation is to generate different Chatbot prototypes that speak multiple languages depending on the corpus [77]. In addition, an approach is presented to use crowdsourced data to increase the response database from an existing corpus focusing on responses which people judge to be inappropriate. The goal is to create a data set of more suitable chat responses. Researchers found that the version with the expanded database was rated better with regard to response level appropriateness and the ability to engage users [81]. Updating information in this case will remain frozen until an updating process is created for the existing corpora as this depends on experts' work.

The internet gives more updatable, extendable, and at any time accessible information [82] that is more useful in Chatbot knowledge enrichment. A number of works used the internet as a source to acquire information for the Chatbot knowledge base. Using Wikipedia, an idea is presented to identify significant facts in the text representing the life of a historical figure to build a Chatbot. This Chatbot should be able to learn from previous experiences in order to act more realistically. A generic form of sentence is proposed to solve the problem of learning to enable the Chatbot to acquire as much information as possible relating to the personality and life of the person that the Chatbot teaches about. The source of information to feed this Chatbot is websites such as Wikipedia for unstructured data and DBpedia for structured data. NLP techniques are used to convert plain text to structured text and then restructure them into a generic form of a sentence [15]. Using a child-care forum, an automatic Chatbot knowledge acquisition method from online forums is presented. It contains a combination of a classification model based on rough set theory, and the theory of ensemble learning to make a decision. Multiple rough set classifiers are built and trained first. After that all replies are classified with those classifiers. The final results are drawn by selecting the output of these classifiers. The corresponding replies are selected as Chatbot knowledge. As the authors state, relevant experiments on a child-care forum prove that the method based on rough set theory has high recognition efficiency of related replies and the combination of ensemble learning improves the results [75]. Also, using discussion forums, extracting thread-title reply pairs from online discussion forums is presented to populate Chatbot knowledge base. These pairs are extracted in a cascade framework and then ranked using a ranking depending on their content quality. The top N ranked pairs are selected as part of Chatbot knowledge [74].

Moreover, multi-purpose algorithms are introduced to design a Chatbot that can hold a conversation on any topic. This Chatbot employs snippets of internet search results to continue within a context. This Chatbot also detects a possibility of replying with a pun via analysing the input sentence and generating a new one when the timing is adequate [83]. However, none of the current studies have investigated reformulation of` the extracted information to QAPs.

The idea in [15] motivated us to extract our chatbot information from the web. The authors in [15] bring the information from Wikipedia and put it in Chatbot knowledge format but as chapters under certain topics. Our idea is to retrieve the information of our chatbot from Wikipedia and to convert the extracted text into QA pairs.

2.3.2 Tutor Chatbots

There is an extensive body of studies and researches in the field of online tutoring [84]. Chatbots can play a good role for educational purposes since they have an interactive mechanism in comparison to traditional learning systems. Students can continuously interact with the Chatbot by asking questions regarding a certain field. In spite of the presence of Chatbots since the middle 1960's, only a few of them are used for educational purposes [85] . The studies mostly focus on dialogue systems or Chatbots that depend mainly on the subject instructor to place and modify the database.

In recent years, approaches have used conversational systems to deal with the learners. Some studies focused on Chatbot engagement in the pedagogical field.

Some authors discuss the development of conversational agents and Intelligent Tutoring Systems, especially Open Learner Modelling. They describe a Wizard-of-Oz experiment so as to examine the feasibility of using a Chatbot to uphold negotiation. They conclude that a combination of the two fields (Conversational agents and Intelligent Tutoring Systems) can be a reason for developing Chatbot negotiation techniques and Open Learner Model enhancement. This technology could have applications in schools, universities and other training scenarios [84].

A software platform named Chatbot is presented [3] and designed to foster engagement while teaching basic Computer Science concepts. Two experiences are continued using Chatbot and the known Chatbot Alice. These experiences are an online nationwide competition and an in-class 15-lesson pilot course in high schools. Moreover, a Chatbot to teach an undergraduate class in Sweden is provided [85] and the possibility of using learning analytics is explored to predict the design of an intelligent language tutor, Chatbot Lucy. This Chatbot focuses on using student-created data to understand the design of Lucy such as to assist English language learning designers to improve second language acquisition. The chatting log architecture is also presented in addition to students' learning journey and data trails [86]. Also an architecture that facilitates building pedagogical Chatbots is proposed [87] to interact with students in natural language. The proposal provides a modular and scalable framework to develop such systems efficiently. Geranium, another system is also presented to help children to appreciate and protect their environment with an interactive Chatbot developed following their scheme.

The role that an educational Chatbot could provide in improving the learning process of a student is explored [88]. A virtual agent is presented, as part of an e-Learning platform capable of guiding and answering questions related to a studied subject. An automated question answering system is introduced for a Chatbot system [89]. This proposed system answers the queries posted by the student in a more interactive way, like a virtual teacher (Chatbot system). The QA knowledge base can be accessed and modified by the instructor. That instructor could also know the areas where the students are more prone to doubts, thus helping the student as well as the instructor.

A prototype of MOOCs conversational agent is developed in [90] and integrated into MOOCs website to respond to the learner's enquiries using text or speech input. MOOC-bot is using the commonly used AIML to develop its knowledge base. The system architecture of MOOC-bot consists of the knowledge base of AIML interpreter, chat interface, MOOCs website and Web Speech API to provide speech synthesis capability. The initial MOOC-bot prototype has the general knowledge from the Alice Chatbot, a course's frequently asked questions, and a course's content offered by the University Technical Malaysia Melaka. The advantages of MOOC-bot lie in the fact that it has the ability to provide a 24-hour service and have knowledge in different domains, and can be shared by multiple sites simultaneously. In [6], a Chatbot is especially built to provide the FAQBot system with the object of being an undergraduate student advisor. The Chatbot accepts natural language input, navigates through the knowledge base, then responds with student information in natural language. The design semantics contain an AIML specification language for authoring the information repository. An intelligent web-based computer aided instruction system is also presented in [91] for learning a foreign language. The system is Computer Simulator in Educational Communication (CSIEC). Such a system can grammatically understand the sentences in English given by the users through the internet in addition to speaking with the users. Important notations of English grammar are used.

However, all the current studies rely completely on instructors in preparing the taught material. The instructor has supervision and control over the Chatbot and any change in material is basically made by the instructor. The following work tries to reduce the restriction of material modification by making the Chatbot learn from the student's feedback: The authors in [92] developed a Chatbot that focuses on engineering education. The purpose of this engineering Chatbot is to build an online artificial intelligence called "Anne G. Neering" which is useful to students in their courses. Students are encouraged to pop some personality into the responses remembering that other students learn from the responses. The Chatbot gives students the chance to explore course content in their own words and from the student's point of view [92].

In the next scenario, learning takes a completely new dimension. Internet usage as a source of learning material has emerged. The work in [93] focuses on the main advantages of an open and modular e-learning software platform to give a boost to cognitive tasks completed by the main actors of the learning process. The authors in [93] present the integration inside the platform of two conversational agents devoted to chatting to the student and acquiring new information sources on the web. The process is sent off as a reply to the system's perception that the student feels disaffected with the presented contents [93]. However, this study uses two agents, one to manage the conversation and one for the content. The content agent updates its knowledge from the web but not in a form of QAPs, in order to simplify the material understanding and to prevent visiting the internet for every information piece needed. In addition, it allows teachers and tutors to make the educational interventions explicit and to customise the learning process.

The idea in [93] motivated us to propose a tutor dialogue system whose database can be populated and updated automatically from the web and the user can decide the information they need. Our tutor Chatbot works without any interventions from an instructor and learns new information as the user requests a new learning subject.

2.3.3 Question Answer Systems

Answer selection is the most complicated phase of a question answering (QA) system [94] because the answer determines the success of a QA system. The common approach is to acquire the candidate answers from an information source and then select the most frequent answers as the best answers [95]. To solve this task, many approaches have been presented using different models to find the best match between a question and an item among a number of items. Some researchers presented surveys like the survey in [95] that describes the state of the art for a QA answering task in three different lines of research: a number of works that focus on candidate answers are presented; then, the idea of a cooperative answer that is correct, useful, and non-misleading is recovered; after that, attempts to address cooperative answering are presented [95].

The issue of answer selection is investigated in information retrieval systems that have typically been concerned with retrieving a set of documents that are relevant to a user's query [96]; for example, in [94] the authors supervised discriminative models are studied to learn to rank answers using examples of QAPs. The pair representation is provided implicitly by kernel combinations. Exploiting the application of structural kernels to syntactic/semantic structures, the authors represent QAPs by generalization methods in order to reduce the burden of large amounts of manual annotation [94]. In addition, the use of linguistic features is investigated to improve the search for ranking answers to non-factoid questions. The possibility of utilising existing large collections of QAPs from online social Question Answering sites is shown so as to extract those features and then to train ranking models which combine them effectively. A range of feature types using natural language processing are investigated, such as named-entity, coarse word sense disambiguation, identification, semantic role labelling, and syntactic parsing [97]. Additionally, a system is described to retrieve a smaller section of text as a direct answer to a user question. The SMART IR system is employed to extract a ranked set of relevant passages to the query. Entities are extracted from the targeted passages as possible answers to the question then ranked for plausibility according to their match score to the query, and according to their position and frequency in the passages [96]. With their system QALC (Question Answering program of the Language and Cognition), the authors in [50] contributed to the Question Answering track of the evaluations for TREC8, TREC9 and TREC10. QALC analyses documents using multi-word term search and linguistic variation to minimize the number of selected documents and to provide additional clues during question comparison and sentence representation. This comparison also uses the results of syntactic parsing for the questions and functionalities of Named Entity Recognition. Answer extraction depends on syntactic patterns application according to the kind of information that is searched for, and is categorised based on the question syntactic form [50]. In the same area, a Turkish factoid, which is a shallow type of questions QA system is presented to utilise surface level patterns, named answer patterns, to extract the answers from documents retrieved from the web. The approaches of named entity tagged answer patterns extraction, and factor assignment, increased the performance of the presented QA system. A new query expansion technique is also described to enhance the performance [98]. Maximum Entropy methods are exploited to combine various lexical, syntactic and semantic features extracted from the text. The focus is on the relation extraction component of the Automatic Content Extraction (ACE) system [99]; however, no application is mentioned in the conversational area.

The evaluation of a number of machine learning techniques is presented for a ranking answers task to *Why* questions. TF-IDF and a set of 36 linguistic features are used to characterise questions and answers. A number of machine learning techniques are experimented with for the purpose of finding out how the different machine learning methods can adapt with their highly imbalanced binary relevance data regardless of hyper parameter tuning [100]. Distributional Semantic Models' (DSMs) role in Question Answering (QA) systems is investigated. The aim is to employ DSMs for re-ranking of answers in Question Cube which is a framework for QA system building. DSMs shape words to look like points in a geometric space which is known as semantic space. If words are close in that space, they are similar. The idea is that DSMs models can help to compute the relation between user questions and candidate answers using paradigmatic relations among words, which leads to providing better re-ranking for answers [101].

To improve existing search engines, an architecture was developed to enhance existing search engines in order to make them boost natural language question answering. The task involves, in sequence: query modulation, document retrieval, passage extraction, phrase extraction and answer ranking. A number of probabilistic approaches for the last three of the five stages is investigated. The proposed technique is applied to a number of existing search engines. The proposed algorithm, Probabilistic Phrase Re-ranking (PPR), employs proximity and question type features to achieve a total document reciprocal rank of 20 on the TREC8 corpus [102].

To improve the existing datasets, researchers discuss the WIKIQA dataset which has a publicly available QAPs set, collected for research on open domain question answering. WIKIQA is built using a natural process and is of an order of size larger than the previous datasets. The WIKIQA dataset includes questions with no correct sentences, so as to enable researchers to put an effort into answer triggering [103]. Also, an end-to-end neural architecture for a task is proposed to solve the problem of the large set and variable lengths of candidate answers in the Stanford QA dataset (SQuAD) [104].

Depending on the information on the web, researchers evaluate two entity normalization methods in [105] which depend on Wikipedia in the context of passage and document extraction for question answering. It is found that a simple normalisation method causes improvements of early precision for both document and passage retrieval. Other researchers introduce Aranea (named after the Brazilian armed spider Phoneutria nigriventer) [106] which is a question answer system that uses knowledge annotation and knowledge mining techniques to extract answers from the World Wide Web. Knowledge annotation utilises semi-structured database techniques and knowledge mining utilises statistical techniques. Aranea combines these two different question answering models into a single framework [106]. An idea is also proposed to find QAPs from the web by detecting the question in a thread of an extracted forum. A method of graph-based propagation is used to detect the answer from the same thread. An Open Instructor based on Wikipedia is presented to extract unstructured text and transfer it to corresponding sentences without mentioning the Chatbot as an application for their idea [74, 75, 107, 108]. Another idea of an automatic generating mechanism is proposed by the authors in [109] to give expressive opinion sentences from numerous reviews extracted from the web. The authors in [109] base the analysis on the frequency of adjectives, sentence length, and contextual relevance to rank the reviews.

For pedagogical purposes, gated self-matching networks are presented for question answering of reading comprehension style that is used to test the reading of English language learners. The question is first matched with the passage with gated attention based on recurrent networks to produce question-aware passage representation. Then a self-matching attention technique is proposed to process the representation by matching the passage with itself. The pointer networks are also used to assign the answers' positions from the passages [5], and the advantage of surface text patterns is explored in open-domain question answer systems. In order to obtain an optimal set of patterns, a method to learn the targeted patterns automatically is developed. A tagged corpus is constructed from the internet in a bootstrapping procedure by supplying a few handcrafted examples for each question type to AltaVista. Then, patterns are automatically acquired and standardized from the returned documents [110].

For conversational purposes, the relation between question answering and constraint relaxation in dialogue systems is explored in [111] for developing dialogue strategies for selecting and concisely presenting information. Methods are described to deal with database query results in information seeking dialogues. The aim is to build the dialogue in a way that the user does not become confused. Using existing conversational material, the authors in [112] propose a system that finds good answers in a community forum for SemEval-2016, Task 3 on Community Question Answering system. The approach used is based on semantic similarity features that rely on fine-tuned word embedding and topics similarities [112]. Then, [113] presents SemEval2017 Task 3 for Community Question Answering by rerunning the four subtasks of SemEval2016. The authors added a new subtask to the approach in [112] to enable experimentation with Detection of Multi-domain Question Duplication in a larger-scale scenario, utilising Stack Exchange sub forums [113]. To add more interaction, researchers report design and implementation of YourQA, the open-domain, interactive QA system. YourQA depends on a Web search engine to find answers to fact-based and complex questions like descriptions and definitions. They describe the discourse moves and management model to make YourQA interactive. They discuss the new model's chat-based dialogue interface architecture, implementation and evaluation [114]. However, the studies in the conversational area did not use multiple features to find the best answer to a question.

The idea in [99] motivated us to design a QA system that exploits multiple Term, Syntactic, and Semantic features extraction to find the similarity scores between a query and a set of candidate answers extracted from the web and filtered using part of speech tagging. The similarity scores of the sentences are re-ranked so as to select the best answer for a Chatbot query. This QA system is planned to be the search engine in our proposed Chatbot to find the best response for a user query in the Chatbot SQL database or even online.

2.3.4 Automatic Question Generation System

Question Generation (QG) is one of the key challenges facing interactional systems with natural languages. The potential advantages of using automated question generation help shrink the dependency on humans to generate questions [115]. In the educational area, the instructors frequently involve accompanying questions when they prepare learning materials for students in order to guide learning [13]. Researchers have discussed different strategies for generating questions for general and pedagogical purposes. Prior question generation methods focused fundamentally on generating factoid questions that are not often pedagogically important questions for the learners. [116].

Some researchers presented multiple choice question answering for the purpose of teaching. An unsupervised method is investigated to apply Relation Extraction to automatically generate multiple-choice questions (MCQs). Semantic relations in a document are identified without allocating explicit labels to relations to ensure broad coverage predefined relations. Three types of surface pattern are investigated, each implements different hypotheses about linguistic expression of semantic relations among named entities. The application for this method is in e-Learning systems and other NLP scenarios [117]. However, no internet is used in this approach. Also results of a study that is seeking to find similarity measures are reported to generate better quality multiple-choice test distractors. The similarity measures utilised in the procedure of distractors selection are collocation patterns. Four different WordNet based semantic similarity methods are used in [118] and a computer aided approach to generate multiple choice test questions from electronic documents is described. The system employs language resources such as corpora and ontologies in addition to using different NLP techniques, such as computing of semantic distance, shallow parsing, sentence transformation, and automatic term extraction to analyse the sentences that are used to generate questions [119]. An existing corpus is also used as well to describe a computer-aided procedure to generate multiple choice tests from electronic instructional documents. The program uses language resources such as a corpus and WordNet, in addition to the use of various NLP techniques like term extraction and shallow parsing. The approach generates test questions and distractors, allowing the user to edit the test items later [120].

Using Wikipedia, automatically built multiple choice test generators that have two main components are proposed. The first is for the generation of QAPs, the second for the generation of distractors. Two approaches are followed: the first is using word features to return QAPs and distractors respectively; the second approach uses the web for automatic generation of syntactic patterns. A set of syntactic patterns are generated and used to create multiple choice tests. An interface for the automated system was developed and both approaches were evaluated in a newspapers corpus and Wikipedia texts [121]. However, multiple choice questions are not very suitable for conversational formats like Chatbots. Attempts resulted in short or shallow questions to present an approach to automatically generate short answer questions for reading comprehension assessment. Lexical Functional Grammar (LFG) is introduced as a linguistic framework for question generation, which enables systematic utilisation of semantic and syntactic information. The approach can generate questions of better quality and uses paraphrasing and sentence selection in order to improve the cognitive complexity and effectiveness of questions [122]. However, this system generates short answer questions and there is no evidence whether it works with factual questions or not. Two algorithms are also developed in order to supply biology instructors with questions for students in introductory classes of biology. One of the algorithms generates questions from photosynthesis knowledge. The other retrieves biology questions from the web and human students validate questions. The exact pattern of results shows a little improvement in the pedagogical benefits of each class. This suggests that the generated questions may work well helping students to learn [123]. However, the authors in [123] stated that the questions generated may be shallower than questions written by professionals.

Generating different types of questions automatically has been investigated. The work in [13] introduces a sophisticated template based approach which incorporates semantic role labels into a question generation system. This system generates natural language questions automatically to support online learning. The authors state that they have not yet integrated a learning context completely into their approach [13]. Also, in this approach, questions are not answerable from the sentences they are generated from because a question is generated from a part of a sentence, so the rest of the sentence is not related to the generated question. Another template-based approach is introduced to generate questions that incorporate semantic roles with a method that generates general and domain-specific questions. The evaluation shows that the approach is effective in generating pedagogical questions [116]. An automatic question generator is described to utilise semantic pattern recognition to create questions of various depths and kinds for self-study. Source sentences' semantic role labels are employed in a domain independent way to generate questions and answers in relation to the source sentence [124]. Considering that each topic is related to a piece of text containing useful information about that topic, questions are generated using named entity information and the predicate argument formats of the sentences in the body of texts. Syntactic tree kernels are also used to automatically judge the syntactic correctness of the questions. The questions are ranked by their importance and syntactic correctness [125].

Taking all possibilities into consideration, a system is presented to automate the generation of all possible questions from a sentence which contains these questions and answers. The system generates elementary sentences from the input complex sentences using a syntactic parser. Depending on subject, verb, object, and prepositional phrase, the sentence is classified to determine possible question types that can be generated from a sentence [115]. To generate even deeper and subjective questions, an extension is presented to a state-of-the-art question generation system that makes it possible to produce deep and subjective questions appropriate for group discussion. Generating questions from paragraphs and sentences provides the First Shared Task Evaluation Challenge detailed account on Question Generation which took place in 2010. The operation included two tasks that take text as input and produce questions as output: Task A, which generates Questions from Paragraphs, and Task B for Question Generation from Sentences. [126].

Two automatic Question Generation Systems are described in [127] to generate questions of various types and scopes for the user from natural language text input. The aim is to generate assessment questions for the content knowledge that a student has acquired through reading a text. The systems are not concerned with grammar or vocabulary assessment or language learning. Both these systems factor the QG process into several stages, enabling more or less independent development of particular stages [127]. However, these systems use user input text to generate the questions, which restricts the questions to limited knowledge.

Although all the works above use documents or sentences or both to generate one or different types or levels of questions, none of them generated questions from automatically extracted information from online sources. Furthermore, we have not found a study that automatically generates questions from information extracted from the web and for a Chatbot knowledge base. We generate factual or definition questions in order to support the pedagogical aspect of our proposed Chatbot. Factual and definition questions can be classified as cognitive pedagogical questions [128].

Based on the discussion above we propose a QG system for generating factual and definition questions for pedagogical purposes. These questions are generated from the sentences that are extracted from the plain text, which is acquired from Wikipedia. The resultant QAPs are part of our tutor Chatbot database.

2.3.5 Extracting Imperative Sentences

Few researchers have discussed extracting sentences or phrases that involve commanding meanings. Actionable or advice revealing sentences have been investigated; for instance, an approach to automatically extract human activity knowledge from web articles is presented to describe performing task methods in different domains. The corresponding knowledge base consists of ingredients, activity goals, and actions, which are extracted using syntactic pattern-based and probabilistic machine learning based methods [129]. In addition, a new approach is presented for automatically detecting actionable clauses in how-to instructions. The focus is on processing non-imperative clauses to extract implicit commands or instructions. Depending on some predominant linguistic pattern in how-to instructions, actionable clauses detection is formulated using linguistic features, such as syntactic, and modal characteristics. The presented algorithm makes it possible to acquire complete sequences of action and convert them to a structural form for problem solving tasks [130]. To detect advice regarding travel in web forums, a methodology for advice-revealing sentences extraction from the web forums is provided, especially in the travel domain. It is defined as a sequence-labelling problem using various features instead of viewing the problem as a simple classification. Three types of features are identified: syntactic, context features, and sentence informativeness. A new method using the Hidden Markov Model (HMM) is also proposed for sequential sentence labelling [131].

For the searching purpose in the database, a system is proposed to translate natural Arabic Quran DBs users' requests like questions or imperative sentences into SQL commands in order to retrieve answers from a Quran DB. The proposed system in [132] performs parsing and morphological processes depending on a subset of Arabic context-free grammar rules so as to act as an interface layer between Quran DB and its users [132]. And for solving problems in object oriented programming, [133] it addresses the concern of not understanding innovative programming techniques like object-oriented programming in the context of reverse engineering. It discusses the development of a method to recognize objects in an imperative code, specifically in

FORTRAN-77. The proposed algorithm that uses an approach to object extraction is presented. The imperative code is analysed using the concepts of graph theory [133].

The studies from [129] to [133] have not used the information from the web to extract imperative sentences for conversational agent purposes. We propose an approach that uses verb tense and POS tags to extract imperative sentences from a set of candidate sentences that are extracted from plain text acquired from Wikipedia. The resultant imperative sentences are planned to be used for more actionable activities in our Chatbot.

2.3.6 Implementation of the OFC

In this section we focus on the works that focus on implementing Chatbots as we needed to base some of Chatbot evaluation metrics used in these studies. Below is a review of these studies.

The authors in [6] explain the design of a Chatbot that was especially built to provide a FAQBot system with the object of being an undergraduate student advisor. The Chatbot accepts natural language input, navigates through the knowledge base, then responds with student information in natural language. The authors in [6] model the knowledge base using a connected graph where nodes containing information and links interrelate the information nodes. The design semantics contain the AIML specification language for authoring the information repository [6].

The authors in [40] present a literature review of quality issues and attributes related to the current issues of Chatbot development and implementation. The focus in [40] is on the text-based conversational agents available online and on the Internet of Things (IoT) devices. This contrasts with voice-activated conversational agents such as Cortana, Siri, Samsung S Voice, and Google Now that are not considered Chatbots [40].

The work in [134] presents a hybrid method where conversational trees are developed for particular types of conversations. Then, using a bespoke scripting language called OwlLang, domain knowledge is acquired from semantic web ontologies. New knowledge that is obtained from the conversations is also stored in the ontologies creating a developed knowledge base. The evaluation involves using a learning management system experience and the experience of students with an intelligent tutor system [134].

The authors in [2] report on the Sinhala Chatbot System design and implementation. Sinhala Chatbot can communicate with a user using Sinhala language and it is the first ever Sinhala Chatbot. Sinhala Chatbot can be asked about operating system related concepts such as date and time, identifies individuals, and greets appropriately. Sinhala Chatbot has been built as an extension of a Sinhala parser and it is an to catch verbal syntax and semantics in Sinhala language to a machine translation [2].

The focus of [135] is to evaluate the use of a neural network-based approach to create an end-to-end trainable Chatbot, which has the capability to automate a restaurant booking service. A sequence-to-sequence prototype is implemented and trained on dialogue data [135].

The authors in [37] investigate methods to adapt and train a Chatbot to a particular user language or application, using a user supplied training corpus. They utilise open-ended trials via real users. Examples like Afrikaans Chatbot for Afrikaans speaking researchers and students in South Africa are used. [37].

Chapter Three

Automatic Web-based Question Answer Database Generation for Online Feedable Chatbot

3.1 Introduction

The idea proposed in this chapter carries through the ideas presented in [15]. The main purpose of this chapter is to generate factual questions from existing factual sentences, i.e. reverse engineering. These sentences are extracted from plain text retrieved from the web and pre-processed using a part of speech tag. Entity Name (proper name) recognition and verb tense recognition are used to determine factual sentence type. Then, specific rules are used to classify the factual sentences and then generate and categorise the questions. The second purpose of this chapter is to generate the OFC database by putting the resultant QAPs into an SQLite database that is built in a number of tables to contain different question and answer categories. A Chatbot database is automatically populated with QAPs from the web pages associated with a desired figure or object. This enables the user to chat with a Chatbot that emulates the behaviour of the object they would like. The example figure used is the footballer David Beckham and his Wikipedia page was used to retrieve the associated data and to produce our data set. We present results for our unranked QAPs produced by our rule-based QA generator. The approach in [13] has been adapted to our system and the produced data set is used to generate QAPs which are compared with our system. Subjective assessment is used to evaluate both our and the comparative systems, and conclusions are then drawn.

3.2 Sentence Hypothesis

There are several kinds of sentences in the English language. The simplest form of sentences have been chosen for question generation in order to simplify the procedure of acquisition. The sentence intended is a factual or definition sentence. The hypothesis of active factual and definition sentences is formulated to generic forms as follows:

Simple past sentence is supposed to have the format:

$$\begin{aligned} &\mathit{Subject} + \mathit{The\ first\ verb\ \ in\ simple\ \ past} + \\ &\qquad \mathit{object\ or\ sentence\ completion} \end{aligned} \tag{3.1}$$

For example:

Example 1

David Beckham played football.

Auxiliary verbs are excluded from the form above since their rules are different from the simple past tense. Auxiliary verbs *was* and *were* as the main verb sentence hypothesis is:

$$\begin{aligned} &\mathit{Subject} + \mathit{the\ verbs\ was\ or\ were} + \\ &\qquad \mathit{sentence\ completion} \end{aligned} \tag{3. 2}$$

For example:

Example 2

David Beckham was a footballer.

The simple present sentence hypothesis is similar to the simple past form except the main verb:

$$\begin{aligned} &\mathit{Subject} + \mathit{The\ first\ verb\ in\ simple\ present} + \\ &\qquad \mathit{object\ or\ sentence\ completion} \end{aligned} \tag{3. 3}$$

For example:

Example 3

David announces his retirement.

and the hypothesis of the verbs *is* and *are* is similar to the one for *was* and *were*:

$$Subject + the\ verbs\ is\ or\ are + sentence\ completion \qquad 3.4$$

For example:

Example 4

England is in Europe.

The subject type chosen here is just the proper name according to the connection between the fact and definition associated with these types of nouns.

The hypothesis of the verb tense for the main verb of the sentence is:

i. Should be the first verb after the subject.

ii. The auxiliary verb should not be followed by a past participle because this makes it passive which is not included in the proposal in this paper.

iii. The auxiliary verb should not be followed by a present participle since this make it present continuous which is not needed in our hypothesis.

3.2.1 Syntactic Analysis for the Sentence

The normal steps performed in QA systems start with syntactic analysis for the question based on the hypothesis set for the question, followed by extracting the information that enables the system to detect the best answer. The procedure proposed in this chapter is to analyse the sentence (answer) according to the built rules, then generate the question depending on the question hypothesis set in advance. Part of speech tags and NER are used to analyse the sentences to filter them and acquire the desired sentences according to the hypothesis stated above. Verb tense recognition exists and is quite straightforward to perform in NLP, but differentiation between the verbs to filter them needs hard coding. The POS tags are mainly used to recognise the main verb tense and then processing needs to separate the required verbs from the eliminated ones. The diagram in fig.3.1 shows sentence analysis with regard to the three hypotheses built above and NLP analysis concepts.

3.3 Question Hypothesis

Question type '*Wh*' is generated from each hypothesis based on selected answers. Factual or definition questions are generated according to verb tense and Entity Name class. So, the hypothesis is as illustrated below:

The generic form of the question in the case of simple past tense is:

$$What + did + NE + the\ verb\ in\ simple\ present\ form + '\ ?' \qquad 3.5$$

The sentence in example 1 can be used to generate the following question:

Example 5

What did David Beckham play?

The question format of *was* and *were* auxiliary verb tense is different from the simple past form:

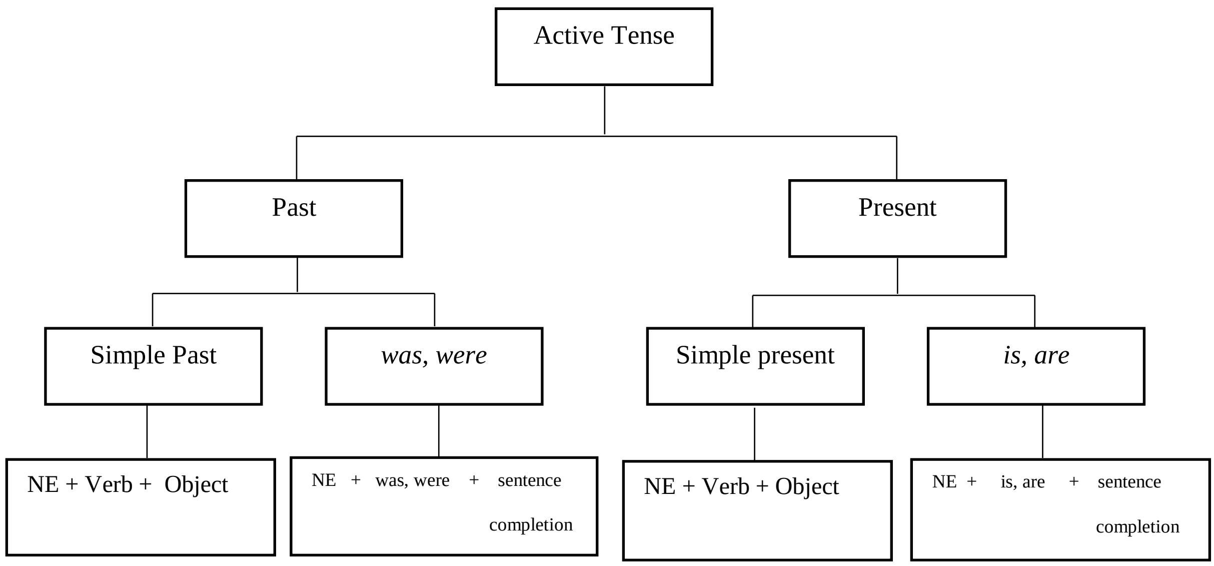


Fig. 3. 1: Factual sentence analysis relating to verb tense.

$$\begin{pmatrix} Who \\ What \\ Where \end{pmatrix} + \begin{pmatrix} was \\ were \end{pmatrix} + (NE) + '?' \qquad 3.\ 6$$

where the question word ‘who’ is used if the Entity Name (EN) type is PERSON or ORGANIZATION, ‘What’ is used in the case of EN type is FACILITY, and the word ‘Where’ is used for asking about LOCATION or GPE.

The question that can be generated from example 2 is:

Example 6

Who was David Beckham?

The question hypothesis for the simple present tense is similar to the one for simple past except the auxiliary verb *did*; it is *does* here:

$$What + does + NE + the\ verb\ in\ simple\ present\ form + '\ ?' \qquad 3.\ 7$$

The sentence in example 3 can be used to generate the following question:

Example 7

What does David announce?

Is and *are* questions are the same format as *was* and *were*:

$$\begin{pmatrix} Who \\ What \\ Where \end{pmatrix} + \begin{pmatrix} Is \\ Are \end{pmatrix} + (NE) + '?' \qquad 3.\ 8$$

Example 8

Where is England?

The answer for each type of question must be the same type considered in the sentence hypothesis stated above. The detailed diagram of question analysis according to verb tense is shown in fig.3.2.

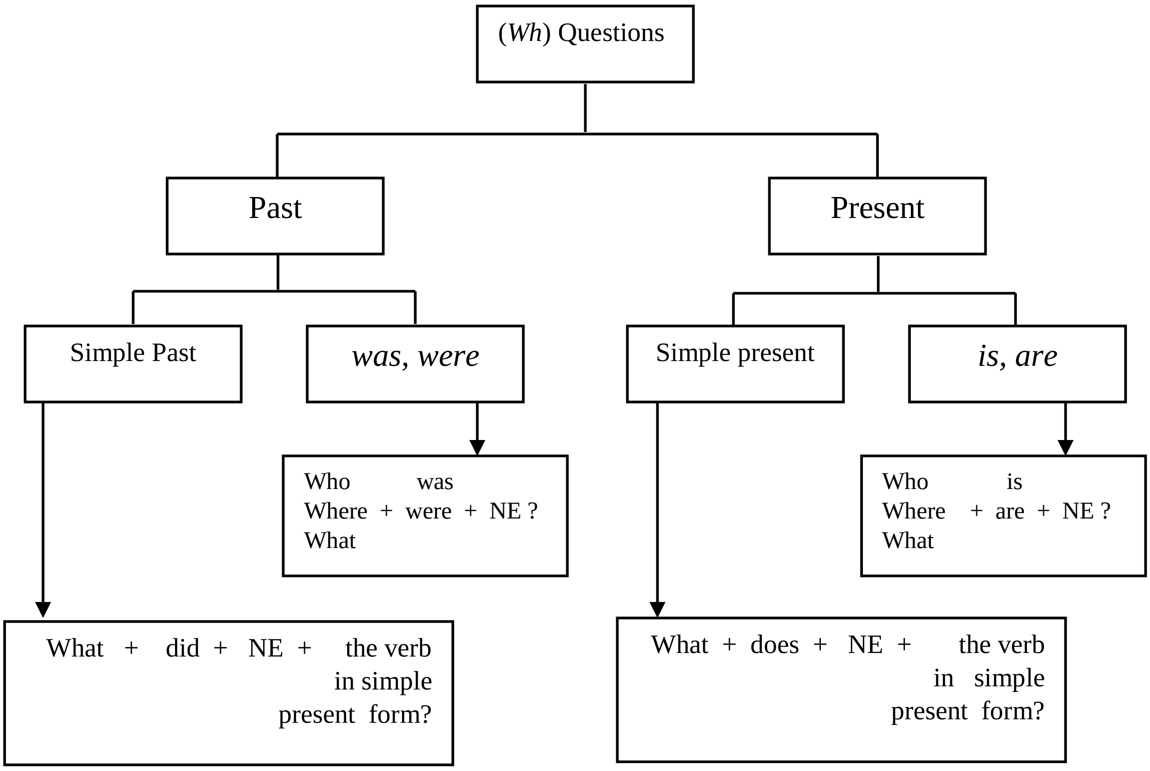


Fig. 3. 2: The analysis of 'Wh' factual questions with regard to verb tense and named entity type.

3.4 Source of Error

The assumptions and hypothesis considered in the theoretical part are not overly complicated. The rules are simplified to minimize the errors that may result from over complexity. Errors have been found in the results and the sources of these errors are not the theoretical propositions. The errors are mainly due to the implementation where tools are used to analyse the text or the sentences. The modules of NLTK and NLTK-NE in the Python library are very useful tools to analyse the sentences syntactically and to identify the NEs, but both have errors in their library; for instance, NLTK-NE recognizes the word ‘please’ in the beginning of the sentence as a named entity and the NLTK library considers the verb ‘saw’ as the present and the past of the verb ‘see’. These types of defects in the tools cause many of the observed errors in the results.

3.5 The Proposed System

The proposed system begins with a web crawler which is capable of accessing web pages and it retrieves plain text from the web starting with a desired URL called the start/seed URL. Buffering has been used in order to avoid storage limitation problems. The buffer enables the crawler to keep the number of pages within the limits of computer memory by controlling the generation of new pages. Pre-processing is applied to the HTML code to extract the plain text. Then, further manipulation is applied to the resultant text to filter the redundant symbols such as stop words, non-English letters and words, and punctuation.

NLP operations are applied to split (tokenize) the text into sentences in separate lines and each sentence is then tokenized into a group of words. The split words are then tagged by speech parts. Named Entity Recognition (NER) is used to identify the sentences with proper name subjects and the verb tense is identified in order to set the verb form. The QAPs are produced and then the results are loaded into an SQLite database. The diagram of the proposed system is shown in fig.3.3. The system is implemented in the Python programming language and the implementation details are presented below.

The web crawler for the proposed system begins with a seed URL of a page associated with a desired figure or the object. The seed URL is used to make a request to the associated web page and then to store the HTML document with the page data in a file. URLs are extracted to be saved in a (*To Visit*) file by parsing the HTML document. A *Try* and *Catch* programming mechanism is used to track the saved URLs in order to check the availability of each of them and then to visit the associated new web pages.

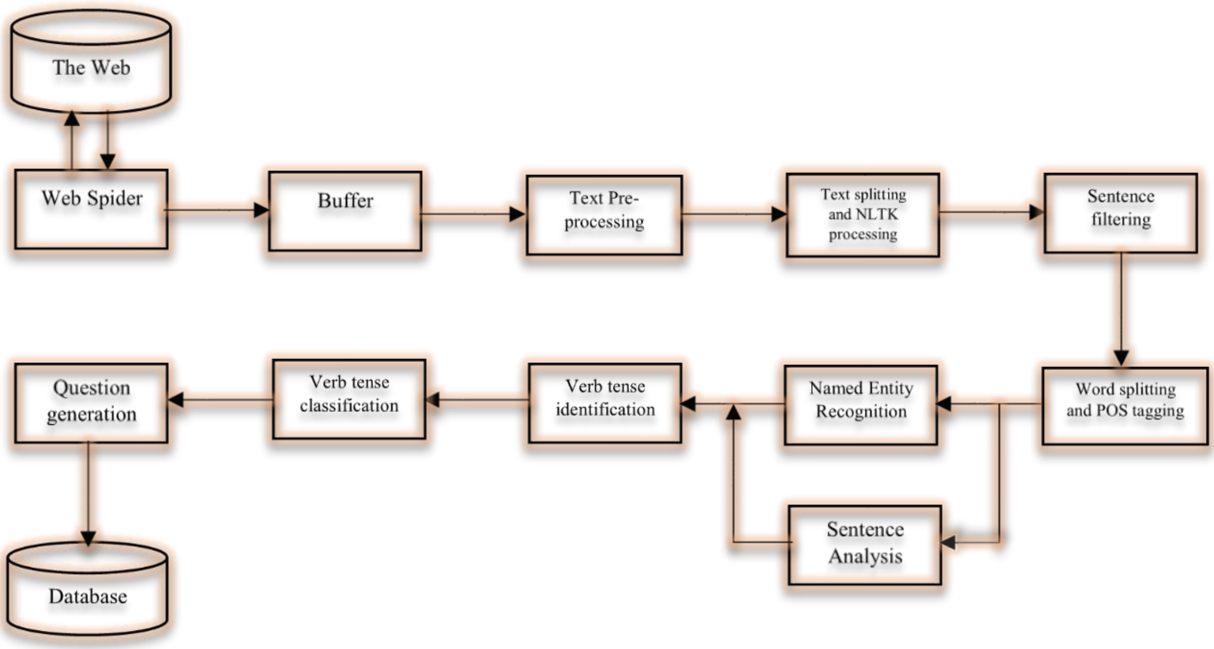


Fig. 3. 3: The main diagram of proposed QA production system.

Plain text is extracted from the web pages of existing URLs and saved into a text file which is processed in the next stage. The process carries on up to the last URL in the (*To Visit*) file. The diagram in fig.3.4 shows the flow of the web crawler operation.

The text extracted from the web is read from the (*To Visit*) file. The plain text could contain different undesired codes after filtering from the HTML. For example: *u* appears before each word in the text and it is called UNICODE. Therefore, the text is encoded to ASCII so as to make it easier to deal with. The text after that is broken down into sentences using the NLTK sentence tokenizing operation. The resultant sentences are then filtered to remove redundant English symbols, and punctuation, in addition to non-English letters and symbols. The filtered sentences are split into words by the word tokenizing NLTK operation and each word is POS tagged with a part of speech label (POS). Sentences with fewer than three words are not useful in this system because two words do not make a complete sentence. Therefore, too short (fewer than two or three words) and too long sentences (more than 20 words) are filtered after POS tagging. The main idea we rely on to identify the subject is detecting the named entity at the beginning of the sentence as the subject. Depending on this concept, the sentences that begin with a named entity are only used to extract question forms. Hence, the proper name named entity is determined for each sentence using NLTK named entity recognition (NLTK-NE). NLTK-NE normally identifies only one word named entity and does not recognize multiple word named entities as a single entity; to deal with this problem, a function has been written to detect and obtain a continuous chain named entity from the multiple entities that NLTK-NE nominated. To generate the proposed form of the questions, named entity (subject) and the verb tense should be known. The verb tense is determined in the same stage of finding the named entity subject to prepare both together for the stage of question generation. Based on the verb tense, the sentence is manipulated in one of the four question manufacturing functions in the implemented software. If the verb tense is past and *was* or *were*, it goes to the function that generates the specified category questions. If it is not *was* and *were*, it is simple past and goes to the function that produces simple past category questions. The present tense is also divided into two categories: *is* and *are* and simple present. Each present tense category also has a function that extracts the question from the sentence. The resultant QAPs are then placed in an SQLite database that is designed to maintain the database of a Chatbot. Cosine similarity is calculated for each QAP so as to know the similarity distance between the generated question and the original sentence, which is the answer.

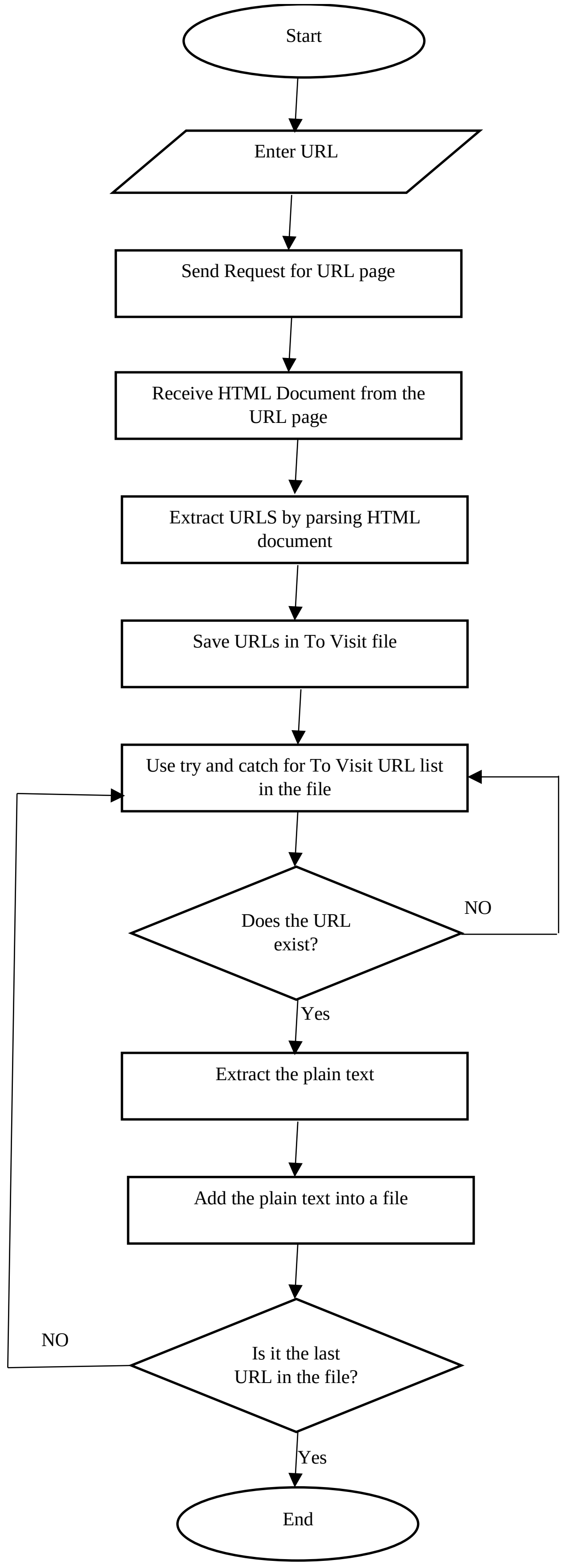


Fig. 3. 4: The implemented web crawler.

The Python modules used in the implemented program are: re (regular expressions), urllib (for web URLs), BeautifulSoup (for HTML filtering), ngram (for term similarity), sqlite3 (database) in addition to NLTK. The flow diagram in fig.3.5 demonstrates the sequence of operations to treat plain text and generate questions from sentences and to produce QAPs.

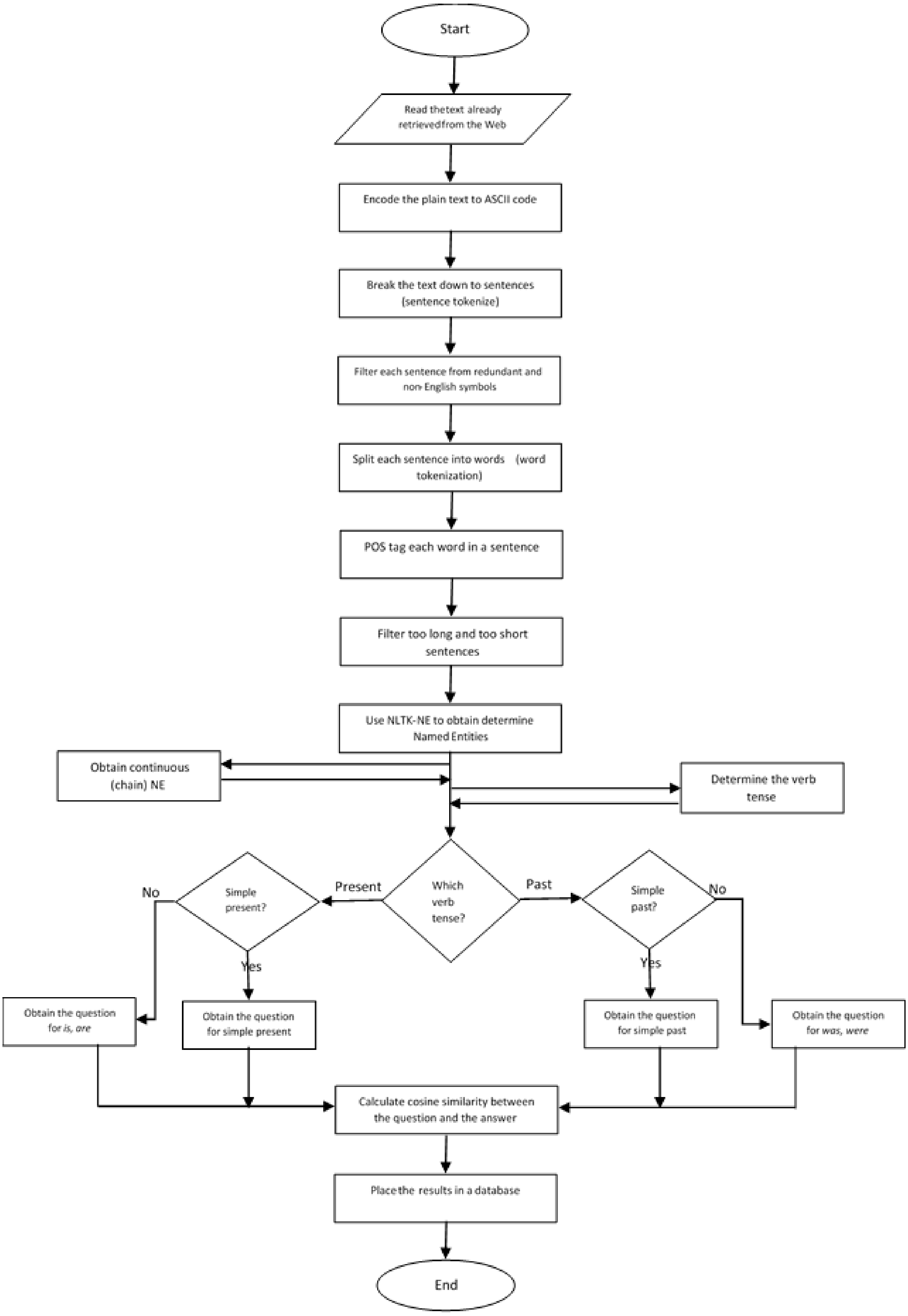


Fig. 3. 5: The implemented steps to treat plain text to generate questions from sentences and produce QA pairs.

3.6 SQLite Database

The data resulting from the processing needs to be saved into a database for storage and for later evaluation. An SQLite database has been built to contain the resultant QAPs.

3.7 Evaluation

Each designed system needs an evaluation stage to assess whether the work is successful and competitive or not. An experiment of two parts has been done to measure the validity of our questions and answers and to compare our work with other systems. The experiment was divided into two parts: the first part was to design and build our automatic question generation system and then evaluate it by human participants, and the second part was to adapt a comparative question generation system and use the same evaluation method to assess it in order to compare the two systems to each other. A subjective assessment was used in order to evaluate the validity of the questions, the answers, and both of them as a pair. Thirty-four participants were included in the experiment and divided into two groups: one to assess our system and the other to evaluate the comparative system.

A questionnaire was prepared to ask the user to assess the relevance of the questions, the answers, and the QAPs according to the subject area (football) or the figure (David Beckham). The questionnaire included four tables of the four main QA categories. Each table contains a number of rows equal to the number of QAPs and the columns represent Question, Answer, Question Relevance, Answer Relevance, and QA pair matching relevance. The example figure in this experiment was the footballer David Beckham and we used his page on Wikipedia as a source to prepare the data set. The questionnaire scores range between 1, which means totally unacceptable, and 4, which means fully acceptable. The questionnaire was applied to both our and the comparative system's experimental results.

3.7.1 Evaluation Metrics

There are no specific evaluation metrics for question generation and “There remains no standard set of evaluation metrics for assessing the quality of question generation output. Some present no evaluation at all” [13]. Also “Among those who do perform an evaluation, there does appear to be a consensus that some form of human evaluation is necessary” [13]. Therefore, we used a subjective assessment that depends on human participants’ opinions.

After searching in references [14] and comparative studies [13] we found that Precision is the best measurement metric for unranked retrieved data or information, as our work is. In addition, Precision is used as an evaluation metric in the subjective assessment measurements that we used to produce our evaluation results. Thus, Precision was applied to the subjective assessment results of both parts of the experiment in order to assess the accuracy we achieved and compare it with the comparative system. The results of applying the former evaluation metrics are discussed in the following sections. Precision is calculated according to the following relation [136]:

$$Precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{\{retrieved\ documents\}} \qquad 3.9$$

The same evaluation metrics are used for the subjective assessment results for the two parts of the experiment in order to measure the enhancement that could be achieved by our QG system. In order to justify the subjective evaluation results, average score, standard deviation, t-test, and p value were calculated for both our and comparative systems.

3.7.2 Experiment1

The experiment starts with using the rules and the hypotheses to select particular sentences extracted from the text retrieved from Wikipedia and pre-processed, then to generate questions from these carefully selected sentences. After that, the resultant

QAPs are compared with the other QG system's after adapting this system to our system. The results of both systems were assessed by a subjective assessment. The experiment has been conducted with a set of participants who are PhD students in different research areas to assess the performance of our system.

The evaluation data was collected depending by the following steps:

1. Meeting each participant, explaining the questionnaire to them, and giving them the questionnaire.
2. Collecting the completed questionnaire from the participants.
3. Calculating the aggregate scores provided by the participants in the questionnaire.
4. Inputting the aggregate scores to a program we prepared to calculate and classify the results.

3.7.3 Aim of Experiment

The aim of the experiment was to assess our Question Generation System that was generated for the OFC in comparison with the other approach in [13] when applying the hypotheses we put for generating QAPs. In addition, we determined the precision level of improvement added to the question generation area by relying on human assessment as the eventual user for any designed conversational system.

3.7.4 Experiment Participants

This experiment was conducted with PhD students from the University of Essex. These students are specialised in different fields of PhD study, such as computer science, electronic engineering, linguistics, and mathematics. Approximately 35% of the judges were native English speakers and the rest (65%) were non-native English speakers.

The study included 34 participants of both genders (male and female) distributed into two equal groups. The participants were chosen and split into groups using the theory of *within and in between* [137] which resulted in dividing the judgers into two groups, one to evaluate the performance of our system and the other to evaluate the comparative system's work.. The questionnaire was completed by each participant to allow us to measure the level of accuracy that was achieved by our Question Generation system. The results of this questionnaire were calculated by aggregating the participants' responses.

3.7.5 Experiment Steps

The experiment was run through the following steps:

1. A programming language was used to implement our Automatic Question Generation System and the program worked in different stages for one single execution. These stages are:
    - i. Information collection from the web: this information was collected as text extracted using a web crawler (fig.3.4).
    - ii. After collecting the plain text from Wikipedia, the text was pre-processed.
    - iii. In the third stage the desired sentences were selected from the set of extracted sentences according to the hypothesis set described in section 3.3 above. The selection operation involved Named Entity Recognition as the first word or words in the sentence, and verb tense identification to classify the sentences to groups as shown in fig.3.1 above.
    - iv. Questions were generated from the selected sentences from the previous stage and the resultant QAPs were stored in a database pre-prepared specially for this purpose.
2. The QAPs stored in the database were collected and sorted into tables to put them in an evaluation form.
3. A questionnaire was prepared for a subjective evaluation that included an explanation about the idea and the experiment and the tables of experimental results.
4. Human participants were chosen to do the subjective evaluation. The participants were kind of experts selected carefully and the vast majority of

them were PhD research students from the University of Essex in different research areas. The participants were aware of the famous footballer David Beckham, Football, and Sports.

5. The questionnaire was distributed to the participants and the idea of the experiment was explained briefly to each of them, then the completed questionnaire was collected from the participants.
6. Precision was calculated, prepared, and stored for comparison with the comparative system.
7. The same steps above were applied to the comparative system in [13] and the results were for comparison with our system.
8. A program was written to draw the bar graphs of precision values for our and comparative systems. The graphs are shown and discussed in the following sections.
9. Average score, standard deviation, t-test, and p value for both our and comparative systems were calculated using a program in order to justify the subjective evaluation results.

3.7.6 Outputs of Our System

Results of our system (AWQDG) are presented in four tables in a database. The tables present QAPs for four categories of factual or definition sentence classes. The text extracted from a 100 web pages or URLs was treated to produce 12 QAPs in the *is* and *are* category (examples are shown in Table 3.1).

Table 3. 1: *Is* and *are* QA group examples of AWQDG.

	Question	Answer
1.	Who is Elton John?	Elton John is godfather to Brooklyn and Romeo Beckham their godmother is Elizabeth Hurley.
2.	Where is Beckham?	Beckham is currently playing Major League Soccer for LA Galaxy.

In the simple past category, 36 QAPs were generated and examples are shown in Table 3.2.

Table 3. 2: Simple past QA group examples AWQDG.

	Question	Answer
1.	What did Beckham choose?	Beckham chose to wear number.
2.	What did Beckham become?	Beckham became only the fifth Englishman to win caps.
3.	What did David help?	David helped launch our Philippines Typhoon children’s appeal which raised in the UK alone.

Only two QAPs appear to belong to the category of simple present as demonstrated in Table 3.3.

Table 3. 3: Simple present QA group examples AWQDG.

	Question	Answer
1.	What does Greatest Britons award?	Greatest Britons awards The Celebrity number.
2.	What does Man Utd play?	Man Utd play down Arsenal rift.

Thirteen pairs of QA are created within the *was* and *were* category and examples of the results are shown in Table 3.4.

Table 3. 4: *Was* and *were* group examples AWQDG.

	Question	Answer
1.	Who was Beckham?	Beckham was a Manchester United mascot for a match against West Ham United in.
2.	Who was Tottenham Hotspur?	Tottenham Hotspur was the first club he played for.
3.	Who was Ryan Giggs?	Ryan Giggs The 39-year-old was the first of Fergie s Fledglings.

The example QAPs include the mistaken ones. The resultant QAPs that are produced by our QAPs generator are put into an SQL database in order to use them as part of the Chatbot OFC.

3.8 Comparative System

We will give the comparative system in [13] the abbreviation GNLQ (Generating Natural Language Questions) and our system the abbreviation AWQDG. These abbreviations are extracted from the titles of the comparative work and the work in this chapter. The approach in [13] has been selected to be considered as a comparative system to assess our hypothesis for the following reasons:

i. The approach in GNLQ uses single sentences to generate questions from sentences, which is quite similar to our system.
ii. GNLQ considers target identification by determining specific words or phrases to ask about, which is similar to our narrowing for the selection of a specific subject type (True Noun Named Entity) and specific verb tense.
iii. GNLQ generates template-based questions and to some extent uses syntactic or/and semantic information to select the sentences or generate questions. Our approach uses semantic features and verb tense types to select the sentences and generate questions in a form similar to the template-based category.
iv. GNLQ generates questions to support learning online and our system generates questions for a conversational agent that can be a tutor about a figure or an object it contains information about. This gives us another justification for the comparison.
v. GNLQ does not simplify the selected sentences to generate the questions from; i.e. it does not cut words or phrases from the selected sentence. It uses predicates of a sentence to generate a question.
vi. GNLQ focuses on generating specific kinds of questions and it selects only the sentence targets appropriate for those kinds of questions. Although the kinds of questions generated in our approach are fundamentally different, similar comparisons can still be made.

GNLQ is briefly described as follows:

- It extracts sentences from online documents and filters the sentences for redundancy. GNLQ does not simplify the sentences or cut words or phrases from the selected sentences to generate questions because the authors believe that simplification of a sentence will eliminate useful semantic content.

- The authors of GNLQ produced templates for each predicate type to generate one or more questions from that predicate.
- GNLQ could generate more than one question from a sentence depending on the number of predicates.
- GNLQ templates depend on assuming copula auxiliary verb in a predicate to generate a question and the authors of GNLQ think that non-copula predicates are not meaningful. GNLQ also filters by auxiliary copulas. We do not filter by auxiliary verbs in our system and we kept them when applying the GNLQ approach to see what the results would be.
- GNLQ does not extract the questions from the whole sentence but uses predicate frames rather than the whole sentence.
- It develops a template based QG framework. It combines the benefit of semantic and syntactic categories of QG with a template-based QG.
- The questions in GNLQ are not answerable from the sentences they are generated from, whereas they are answerable from the documents the sentences are extracted from [13].

We adapted the comparative system to our system by selecting specific templates designed by the comparative system's authors that match our sentence filtering, subject type, and verb tense then we programmed the selected templates. Our data set was used to produce QAPs from the comparative templates in GNLQ and the results are shown in the following sections. A block diagram of GNLQ templates adapted to our system for comparison is shown in fig.3.6. The results are evaluated using a subjective questionnaire given to human judges for assessment and then compared to our system.

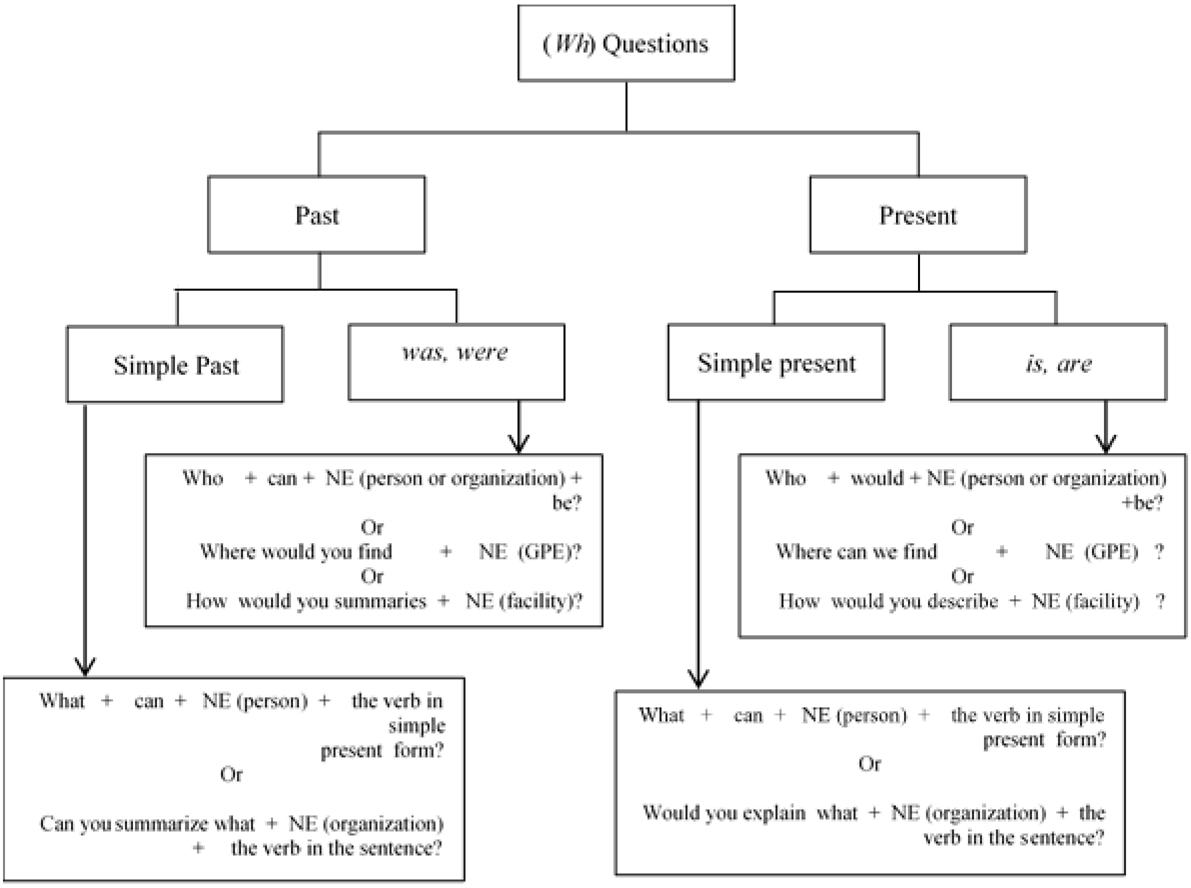


Fig. 3. 6: The analysis of GNLQ question templates adapted to our system for comparison.

3.8.1 Comparative System's Experimental Results

Experimental results of GNLQ are also presented in four tables in an SQLite database. The tables present QAPs of four categories. The text extracted from a 100 web pages or URLs was treated to produce 13 QAPs in the *is* and *are* category (examples are shown in Table 3.5).

Table 3. 5: *Is* and *are* QA group examples of GNLQ.

	Question	Answer
1.	Where can we find Messi?	Messi is very authentic to his sport.
2.	Who would Roy Keane be?	Roy Keane is among those who like Ferguson disapproves of Beckham 's extravagant lifestyle.

In the simple past category, 33 QAPs were generated and examples are shown in Table 3.6.

Table 3. 6: Simple past QA group examples of GNLQ.

	Question	Answer
1.	Can you summarize what Brooklyn Beckham set?	Brooklyn Beckham set to be released by Arsenal already.
2.	What can Beckham make?	Beckham made you want to believe that right feet could be described as educated too.

Four QAPs appear in the category of simple present, as demonstrated in Table 3.7.

Table 3. 7: Simple present QA group examples of GNLQ.

	Question	Answer
1.	Would you explain what Shell say?	Shell say you were rubbish today Ryan.
2.	Would you explain what Greatest Britons awards?	Greatest Britons awards The Celebrity number.

Seven QAPs were created within the *was* and *were* categories and examples of the results are shown in Table 3.8.

Table 3. 8: *Was* and *were* group of GNLQ.

	Question	Answer
1.	Would you summarize what Tottenham Hotspur be?	Tottenham Hotspur was the first club he played for.
2.	What can Ronaldo be?	Ronaldo was concerned but not afraid about coming back.

3.9 Evaluation Results

As discussed in the previous sections, subjective assessment was used to evaluate the experimental results of our system AWQDG and the comparative system GNLQ.

After finishing the calculation of data classes for both AWQDG and GNLQ, a Python program was used to calculate the precision value for each part in each group of the two compared systems. Precision was calculated for Question, Answer, and QA pair match for each of the groups *is* and *are*, simple past, simple present, and *was* and *were* in both systems AWQDG and GNLQ. Precision calculation results have been recorded and then entered into a MATLAB program to produce comparative bar graphs between AWQDG and GNLQ. The graphs are drawn as follows:

The bar graph demonstrated in fig.3.7 is for precision levels of Question, Answer, and QA pairs matching in group *is* and *are* for both AWQDG and GNLQ. The graph shows proximity in precision levels between AWQDG and GNLQ. However, AWQDG surpasses GNLQ by 3 points in Questions with 0.96 for the former and 0.93 for the latter and also QA match by 3 points with 0.98 for the former and 0.95 for the latter. In contrast, in Answers GNLQ precision value surpasses AWQDG by 10 points with 0.67 for the former and 0.57 for the latter.

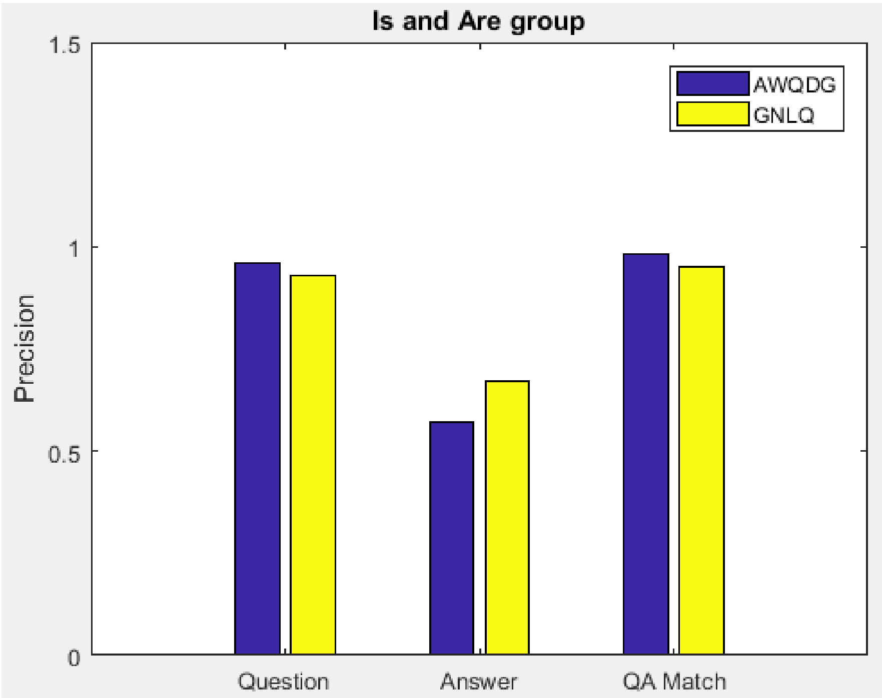


Fig. 3.7: Precision comparison between AWQDG and GNLQ (is and are group).

The bar graph in fig.3.8 illustrates the precision levels of Question, Answer, and QA pair match in the simple past group for both AWQDG and GNLQ. The graph shows another proximity between AWQDG and GNLQ with an increase for AWQDG over

GNLQ in Answers, and QA pairs match with 0.96, 0.95 respectively for the former and 0.89, 0.92 respectively for the latter. The results show better performance for AWQDG by 7 points in Answers over GNLQ and 3 points in QA pairs match. By contrast, in Questions GNLQ surpasses by 1 point with 0.89 for AWQDG and 0.90 for GNLQ.

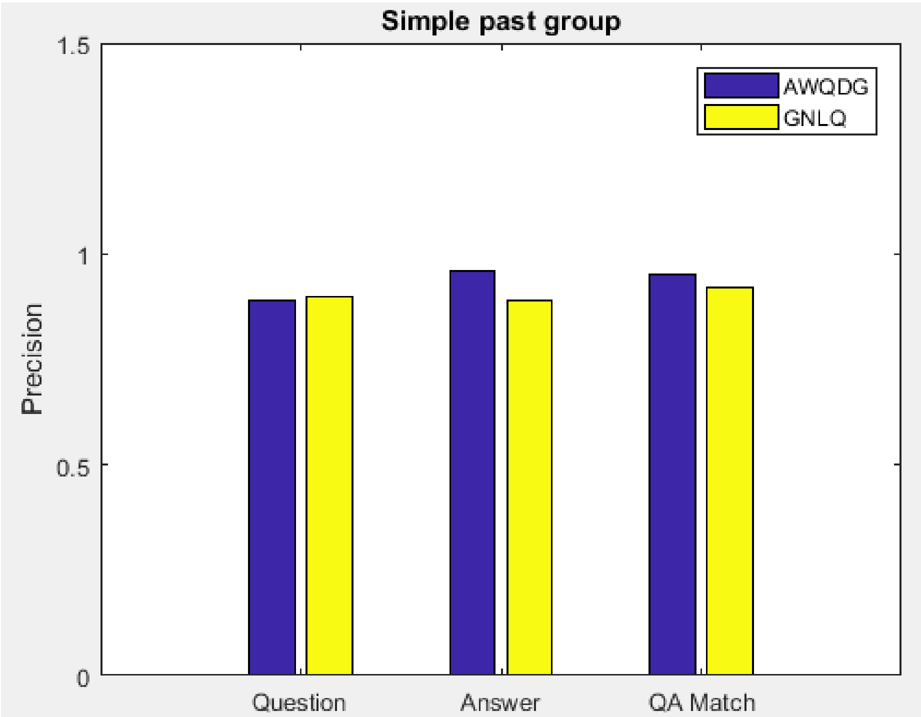


Fig. 3. 8: Precision comparison between AWQDG and GNLQ (simple past group).

The levels of precision for Question, Answer, and QA pairs match for simple present groups of both AWQDG over GNLQ are shown as a bar graph in fig.3.9. The graph shows a significant excess by 84 points for AWQDG over GNLQ in the Questions part where the values were 0.95 for the former and 0.11 for the latter. Equality is shown for both in Answers where the values are 0.11 for both. AWQDG also exceeded GNLQ in QA pair match by 3 points as 0.97 for the former and 0.94 for the latter

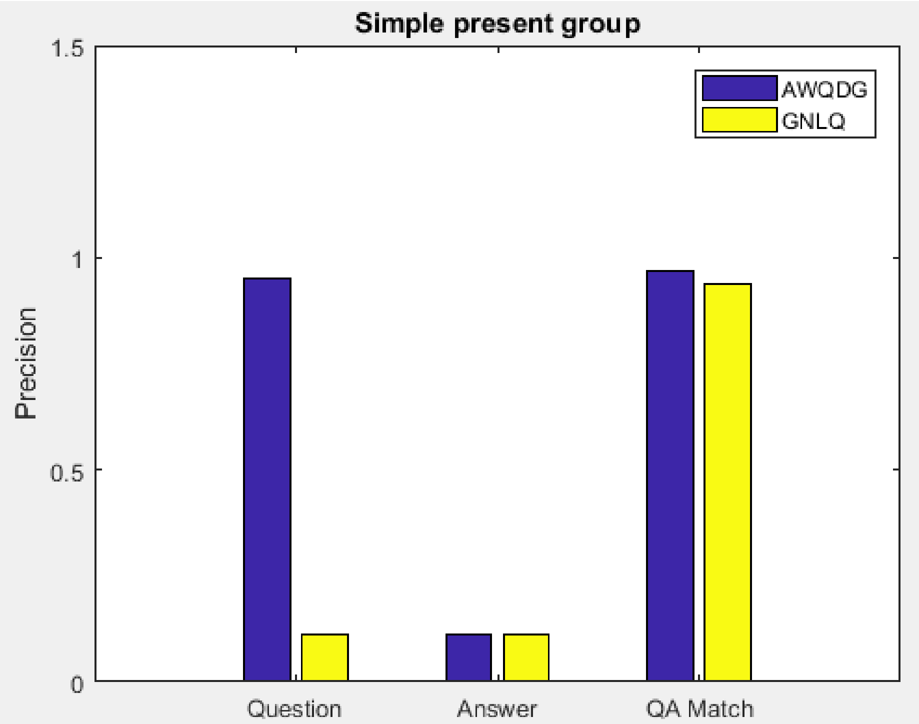


Fig. 3. 9: Precision comparison between AWQDG and GNLQ (simple present group).

Fig.3.10 presents the bar graph for precision levels of Question, Answer, and QA pair match for *was* and *were* groups in AWQDG against GNLQ. The bar graph shows a considerable increase in the precision level of AWQDG over GNLQ by 68 points with 0.93 for the former and 0.25 for the latter. AWQDG also beats GNLQ in QA pair match by 27 points and the numbers are 0.94 and 0.67, respectively, for the former and the latter whereas GNLQ exceeds AWQDG in Answer part by 24 points with values of 0.91 and 0.67, respectively.

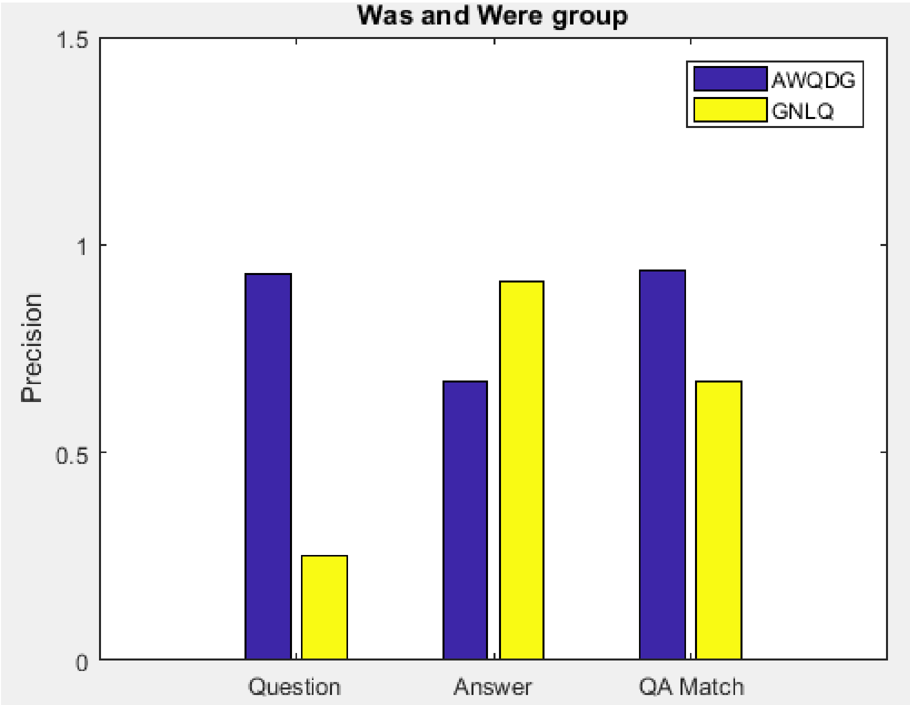


Fig. 3. 10: Precision comparison between AWQDG and GNLQ (was and were group).

The comparison of overall precision levels for both AWQDG over GNLQ is shown in the bar graph of fig.3.11. The graph shows a remarkable rise in favouring AWQDG over GNLQ in the *was* and *were* group by 33 points with 0.89 in AWQDG and 0.56 in GNLQ and in the simple present group by 17 points with 0.67 for the former and 0.50 for the latter. Even so, GNLQ overtakes AWQDG in the *is* and *are* and simple past groups by 5 points in the *is* and *are* groups with 0.88 for the former and 0.83 for the latter and 2 points in the simple past group with 0.96 for the former and 0.94 for the latter.

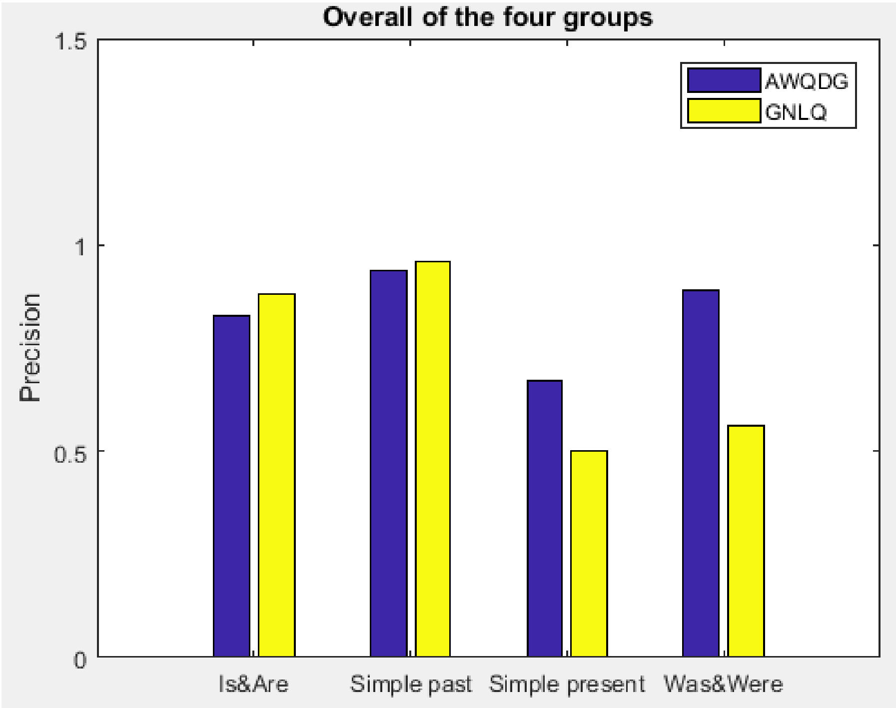


Fig. 3. 11: Precision comparison between AWQDG and GNLQ (overall of the four groups).

It is noticeable from the bar graph shown in fig.3.12 that AWQDG outperforms GNLQ in overall Questions and QA pair match. The graph shows an increase by 12 points in Questions for AWQDG over GNLQ with 0.93 for the former and 0.81 for the latter. Also, the graph shows that the QA pair match part rises in AWQDG over GNLQ by 7 points and the numbers are 0.96 and 0.89 for them, respectively. However, GNLQ increases in the Answers part over AWQDG by 8 points with 0.9 for the former and 0.82 for the latter.

Overall, the recorded precision value for our question generation system was 0.91 relating to the subjective assessment results we implemented for our system evaluation, whereas an overall precision value of 0.86 has been obtained for the comparative system that has been adapted to our system and our produced data set using the same evaluation method for the experimental results. The overall values present a clear success for our system over the comparative system by 5 points. An improvement is also shown in Questions and QA pairs match, which means that our system AWQDG generates more answerable questions than the comparative system GNLQ.

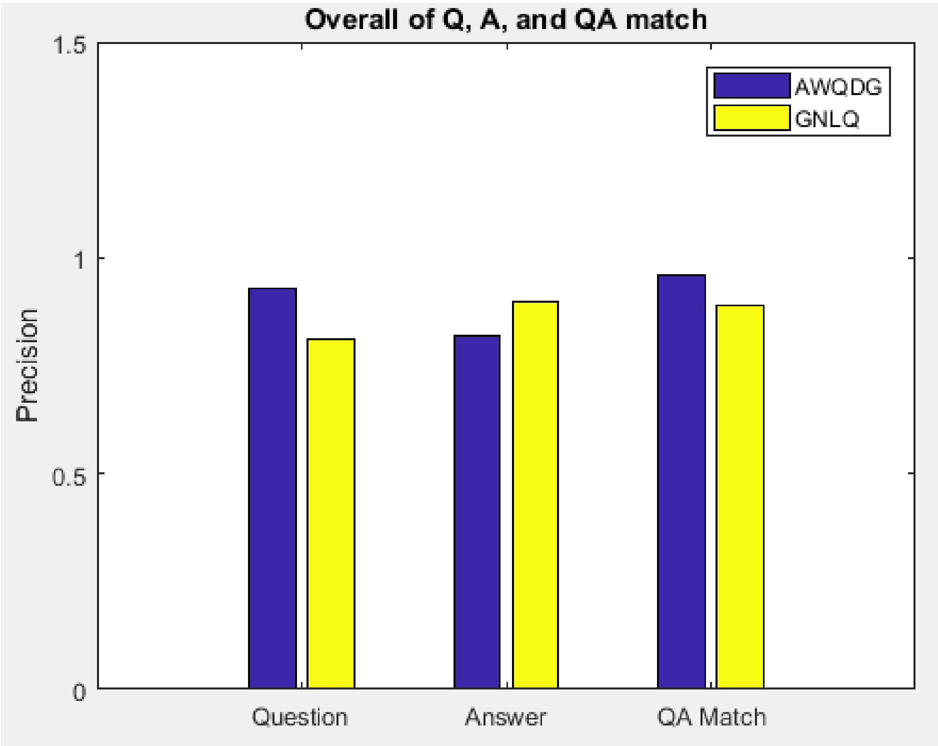


Fig. 3.12: Precision comparison between AWQDG and GNLQ (overall of Questions, Answers, QA match).

To support the subjective evaluation results, average score, standard deviation, T-test, and p value for both our and comparative systems were calculated using a Python program and the results are shown in Table 3.9. The t-test was calculated using the following relation for independent samples [138] [139]:

$$t = \frac{\overline{x}_1 - \overline{x}_2}{S_p\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}} \tag{3. 10}$$

$$S_p = \sqrt{\frac{(n_1-1)S_1^2 + (n_2-1)S_2^2}{n_1+n_2-2}} \tag{3. 11}$$

where

$\overline{x}_1$ = Mean of first sample

$\overline{x}_2$ = Mean of second sample

n_1 = Sample size (i.e., number of observations) of first sample

n_2 = Sample size (i.e., number of observations) of second sample

S_1 = Standard deviation of first sample

S_2 = Standard deviation of first sample

The statistical values mentioned above were calculated for the groups Question, Answer, and QA match. The results show significance in the t-test values for our system in the groups Question and QA Match and a value close to significance in the Answer group for the comparative system.

Table 3. 9: The results of statistical calculations for subjective assessment evaluation for both AWQDG and GNLQ.

Statistics	**Question**		**Answer**		**QA Match**	
	AWQDG	GNLQ	AWQDG	GNLQ	AWQDG	GNLQ
Average	3.2009	2.9492	3.0768	2.8861	3.0666	2.8646
Standard deviation	0.7718	0.6311	0.6037	0.5010	0.6177	0.5217
T-test	1.9724		-1.8841		1.9507	
P value	0.0509		0.0619		0.0535	

3.10 Conclusions

In this chapter, two main contributions are presented. The first contribution is generating factual questions from existing factual sentences. Plain text was extracted from the 100 URLs from the Wikipedia page of the footballer David Beckham. Factual sentences were extracted from the plain text after pre-processing. Named Entity (proper name) Recognition (NER) and verb tense recognition were used to identify the factual sentence category. Specific rules were built to categorize the sentences and then to generate questions and categorize them. The new built SQL database was used as knowledge for the Online Feedable Chatbot that can answer questions about the personality of a desired figure or the behaviour of an object and improve over time. Four categories of QAPs were produced and examples of these categories are presented. A comparative system was incorporated into our system using our produced dataset and compared with our system. A subjective assessment to validate the QAPs was performed and the evaluation stage was implemented after the subjective assessment was made for the two systems. The overall precision levels obtained for the subjective assessment show an enhancement by 5 percentage points for our system over the comparative system. Also, the results show a clear improvement for our system in the Question and QA pair match categories, which means that our system produces more answerable questions from a sentence than the comparative system. The resultant QAPs produced by our question generation system was put into an SQLite database to be used as part of the knowledge base for our Online Feedable Chatbot.

Chapter Four

Question Answer System for Online Feedable Chatbot

(Part of this chapter is published in [17])

4.1 Introduction

Populating a Chatbot database from the web is considered a new research area. Few researchers have examined educating a new Chatbot in order to incorporate an artificial character. Some authors start database population from web pages or plain text based upon a certain genre or person [15, 140]. Using data from web pages needs numerous operations, such as pre-processing, filtering, mining, and quantification before classification and rank ordering.

Feature extraction methods are used after text mining for both query and response sentences. Quantification uses content analysis involving occurrences, tabulation and statistical semantics for content units [141]. Particular features are used as score measurements for quantification according to statistical calculations.

The aim of this chapter is to present a new method that employs various feature extraction methods to quantify text responses for Chatbot queries. The well-known footballer David Beckham is considered as the 'personality' for this Chatbot experiment. The results show that the highest scored sentences match well with subjective analysis and the objective evaluation results show that cosine similarity is more accurate than Jaccard's coefficient to find term match between the query and the response sentences.

The study in [112] has been adapted to our system and our dataset for the purpose of comparison. The contribution presented in this chapter is using a new combination of multiple feature extraction methods to find a best response to a Chatbot query.

4.2 Query and Features

4.2.1 Query

The processed data is tested using a sentence or a question that is associated to the extracted material. The sentence or question, which is used to test the processed, is called the query. The query sentence, in this work, is used to test the relevance of the extracted data in order to measure the performance of the system. The query and input data are analysed to form their basic components and then the features, which are the measurement metrics, are extracted to quantify the output data [142]. The query here in the experimental part is a set of TREC8 type factual questions about the personal and the career lives of the footballer David Beckham as in the examples below:

- Who is David Beckham?
- Who is David Beckham's wife?
- Which club did David Beckham play for?
- Where was David Beckham born?

Query formulation is the operation of creating a list of keywords from the question. This list of keywords forms a query sent to an information retrieval system. The type of query to form, basically, depends on the application. If question answering is applied to the web, a keyword can simply be created from every word in the question, letting the web search engine remove any stop words automatically. Often, the question word, such as where, when, and so on, is left out. In addition, keywords can be formed from the terms found in the noun phrases in the question, applying stop word lists to remove function words and high frequency, low content verbs [143].

4.2.2 Feature Extraction

In order to be able to quantify the training text according to the query, particular features are extracted from both the query and the sentence. Then comparison is made between these features. Feature extraction enables discrimination between text classes [144] and it is considered an important step in improving the performance of the designed system [145]. A system that uses a combination of feature extraction methods together such as term, syntactic, and semantic at once is called a hybrid feature system.

4.2.3 Feature Selection

The examination of features that are extracted is of a primary concern. Arbitrary selection of features will decrease the accuracy of quantification and thus affect the performance of the system. Relevant features should be selected in order to reduce general data, save storage space in memory, improve the system performance, and simplify the extracted data. In this chapter, the features have been selected based on previous experimental work as in [99], which selects term, syntactic, and semantic features to quantify the similarity between two sentences.

4.3 Term Match

Term level analysis means converting a piece of text into a sequence of strings (words) called tokens. Each token has an identified meaning. In this method, the number of matching words between the query and the response sentence is determined [146]. The overlapping words are counted up to the number of words in the query and if the number of matching words is equivalent to the number of union words, there is a 100% match.

4.3.1 Similarity Measurement Methods

The association between two words, phrases, or sentences is determined using different comparison metrics, such as similarity, dissimilarity, and distance. As the issue in this chapter is matching, then similarity metrics are of this chapter's interest. There is a number of term similarity measurements [147]. In this chapter, Jaccard's coefficient and cosine similarity methods are used to measure term similarity between the query sentence and each response sentence.

4.3.2 Jaccard's Coefficient

Jaccard's coefficient measures the similarity between two data sets through dividing the number of common properties between the compared sets by the total number of features [148, 149]. For example, if X and Y are two sets, then Jaccard's coefficient between them is:

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} \qquad 4.\ 1$$

The technique used in this method is n-grams which is a method that counts the number of words in a sentence or a text and considers each word as a gram. There is a feature in the N-grams, which allows finding the overlapping words between two lists of words. This feature is employed in this paper to find the overlapping words between the query and the response sentence according to the following relation:

$$m1 = r_N \cap q_n \qquad 4.\ 2$$

And the union words can alternatively be obtained by using the following relation:

$$m2 = r_N \cup q_n \qquad 4.\ 3$$

where:

m1 is the intersectional words between the query and the response sentences.

m2 is the union words between the query and the response sentences.

r_N is the response sentence.

q_n is the query sentence.

N is the number of words in the response sentence.

n is the number of words in the query sentence.

The length of the resultant intersectional list m1 indicates the number of overlapping words (i.e. the number of matches), and the rate of match according to Jaccard's coefficient comes from dividing the overlapping words number by union the words number:

L 1= Len (m1)

L 2= Len (m2)

$$M(q,r)= \frac{L1}{L2} \quad 4.\,4$$

where:

M is the rate of term match.

$L1$ is the number of intersection words.

$L2$ is the number of union words.

4.3.3 Cosine Similarity

One of the popular similarity metrics used in text document processing techniques is cosine similarity. Cosine similarity is used when the text documents are considered as vectors. The similarity between two documents are here considered as the correlation of two vectors [150]. For example, if two document vectors t_i and t_j are given, cosine similarity between them is:

$$SIM_{cos}(\vec{t_i}, \vec{t_j}) = \frac{\vec{t_i} \cdot \vec{t_j}}{\vec{t_i} \times \vec{t_j}} \quad 4.5$$

where t_i and t_j are m dimensional vectors of the term set $T = \{t_1, \dots, m\}$. Each term with its weight is represented by a non-negative dimension. The cosine similarity is bounded between 0 and 1 and it is document length independent [151].

The query can be described as a vector $q_n = (w_{q0}, w_{q1}, w_{q2}, \dots, w_{qn})$ and the same for the response sentence $r_N = (w_{r1}, w_{r2}, w_{r3}, \dots, w_{rN})$. To apply cosine similarity on the query and the response sentences, the relation will be as follows:

$$SIM_{cos}(\overrightarrow{q_n}, \overrightarrow{r_N}) = \frac{\sum_{i=1}^{i=n} \sum_{j=1}^{j=N} q_n(w_{qi})\, r_N(w_{rj})}{\sqrt{\sum_{i=1}^{n} q_n^2(w_{qi})}\sqrt{\sum_{j=1}^{N} r_N^2(w_{rj})}} \quad 4.6$$

Where q_n and r_n are term vectors of the query and the response sentences respectively and $n = N$.

In the implementation part, one of the term similarity metrics (Jaccard's coefficient or cosine similarity) is used every run to find term similarity between the query and the response sentences.

4.4 NLP Match (Syntactic Analysis)

Syntactic or POS tag analysis is a technique used to analyse a sentence into chunks or phrases depending on POS tags performed according to language and grammar rules. Tokenizing the sentence into separate words is required before applying POS tagging. Syntactic predicate match can be used to obtain the extent of concordance between two compared sentences. The matching can be obtained by comparing the POS tags of the query and the response answer [152]. The function of comparison is shown in the relation below:

$$P = \sum_{i=0}^{i=n} \sum_{j=0}^{j=N} PT \quad where\ PT = \begin{cases} 1 \ when\ POS(i) = POS(j) \\ 0\ when\ POS(i) \neq POS(j) \end{cases} \quad 4.7$$

where:

P is the number of POS tag matches.

PT is the POS tag.

i refers to the position of a POS tag in the query sentence words.

j refers to the position of a POS tag in the response sentence words.

By dividing by the total number of POS tags, a syntactic or POS tag match is obtained as shown below:

$$PTM = \frac{P}{LEN(PT_q \cup PT_r)} \quad 4.8$$

where:

PT_q is the POS tag list of the query sentence.

PT_r is the POS tag list of the response sentence.

LEN is the number of union POS tags in the query and the response sentences.

PTM is POS tag match between the query and the response sentence words.

4.5 Semantic Similarity

Semantic relationship is one of the ways to measure similarity between two sentences. It is used in NLP to compare units of language, such as words, sentences, paragraphs, and documents [153]. "Semantic measures are mathematical tools used to estimate quantitatively and qualitatively the strength of the semantic relationship between units of language, concepts, or instances" using symbolic or numerical description gained according to formal or implicit information comparison for the meaning of these units [154]. Semantic similarity is used as a metric to measure the similarity in sense between two documents regardless of the size of these documents.

There are different semantic measurement metrics; two of them are used in this work. Named entity and semantic cosine similarity are used in this chapter.

4.5.1 Named Entity

In the previous chapter, named entity was used as a feature extracted after analysing a sentence to identify the subject of the sentence, which is the target to derive a question from this sentence. The concept of named entity is explained in Chapter 2 (Section 2.2.9)

In this chapter, the entity type semantic method is used as a feature extracted from the query and the extracted sentence in order to examine the similarity between them. The summation of similar named entities between the query and the response sentence is calculated as follows:

$$\mathrm{ENE} = \sum_{k=0}^{k=K}\sum_{l=0}^{l=L} \mathrm{NE} \text{ where } \mathrm{NE} = \begin{cases} 1 \text{ when } k = l \\ 0 \text{ when } k \neq l \end{cases} \quad 4.9$$

where:

ENE is the number of named entity matches between the query and the response.

NE is a named entity match.

K indicates the number of named entities in the query sentence.

L indicates the number of named entities in the response sentence.

k indicates a named entity in the query sentence.

l indicates a named entity in the response sentence.

Then the named entity match rate *NEM* is:

$$NEM = \frac{ENE}{LEN((NE_q \cup NE_r)+(NE_q \cap NE_r))} \quad 4.10$$

where:

NE_q is the named entity list of the query.

NE_r is the named entity list of the response sentence.

LEN is the number of unity words between NE_q and NE_r.

4.5.2 Semantic Cosine Similarity

Cosine similarity is discussed in Section 4.3.3 above. The meaning of cosine similarity is different from the meaning of semantic cosine similarity. At term level, cosine similarity is when compared sentences are converted into term vectors. Cosine similarity is obtained by using the magnitudes of similar words in term vectors to calculate the overall similarity using the relation of cosine similarity. Semantic cosine similarity deals with semantic vectors of sentences rather than term vectors. WordNet is used to obtain semantic vectors of the query and the response sentences. WordNet is a lexical source that is used to model lexical knowledge of the English language. The smallest term in WordNet is a logical group of synonym set (synset). The synset represents the particular meaning of a word and the synsets have semantic relations explicitly with each other [155]. Hence, the query can be described as a semantic vector $q_{sn} = (w_{qs0}, w_{qs1}, w_{qs2}, \dots, w_{qsn})$ and the same for the response sentence $r_{sN} = (w_{rs1}, w_{rs2}, w_{rs3}, \dots, w_{rsN})$. To apply cosine similarity to the query and the response sentences, the relation will be as follows:

$$SIM_{Semcos}(\overrightarrow{q_{sn}}, \overrightarrow{r_{sN}}) = \frac{\sum_{i=1}^{i=n} \sum_{j=1}^{j=N} q_{sn\,(w_{qsi})} \cdot r_{sN\,(w_{rsj})}}{\sqrt{\sum_{i=1}^{n} q_{sn}^{2}(w_{qsi})}\sqrt{\sum_{j=1}^{N} r_{sN}^{2}(w_{rsj})}} \qquad 4.\ 11$$

where q_{sn} and r_{sn} are semantic vectors of the query and the response sentences respectively and $n = N$.

4.6 Maximum Percentage of Match

Total match is calculated by adding the term, syntactic, and semantic features together in the combination using Jaccard’s coefficient and named entity as shown below:

$$M_{total} = M(q,r) + PTM(PT) + NEM(NE) \qquad 4.\ 12$$

In the combination using cosine similarity and named entity, the total match equation is:

$$M_{total} = SIM_{cos}(\overrightarrow{q_n}, \overrightarrow{r_N}) + PTM(PT) + NEM(NE) \qquad 4.13$$

The combination using Jaccard's coefficient and semantic cosine similarity equation is:

$$M_{total} = M(q,r) + PTM(PT) + SIM_{Semcos}(\overrightarrow{q_{sn}}, \overrightarrow{r_{sN}}) \qquad 4.14$$

Finally, the formula of the combination using cosine similarity and semantic cosine similarity is:

$$M_{total} = SIM_{cos}(\overrightarrow{q_n}, \overrightarrow{r_N}) + PTM(PT) + SIM_{Semcos}(\overrightarrow{q_{sn}}, \overrightarrow{r_{sN}}) \qquad 4.15$$

The percentage match is then obtained as in equation 4.16 below:

$$Percentage\ match = M_{total} \times 100 \qquad 4.16$$

The maximum match percentage is obtained by re-ranking the percentage matches calculated above and the highest score can indicate the best match according to the assumptions considered for the output of the proposed system.

4.7 The Proposed System

The proposed system begins with a web crawler with the ability to obtain plain text from the web using a desired URL as the start/seed. The block diagram of the proposed system is shown in fig.4.1.

The diagram in fig.4.2 shows the flow of the implemented system. The text retrieved from the web is read from the (To Visit) file. The plain text may contain different undesired code after being acquired from the HTML. One example is 'u' appearing before each word in the text and this is called UNICODE. So, the text is encoded to ASCII in order to make it easier to deal with. The text is then split into sentences using the NLP. The resulting sentences are filtered after that to remove redundant English symbols, punctuation, and non-English letters and symbols. The filtered sentences are broken down into words by the word tokenizing NLP process and each word is tagged by a part of speech label (POS). Named Entity Recogniser is then used to identify entity names for both the query and the extracted sentences. Features are extracted and compared to calculate the matching score.

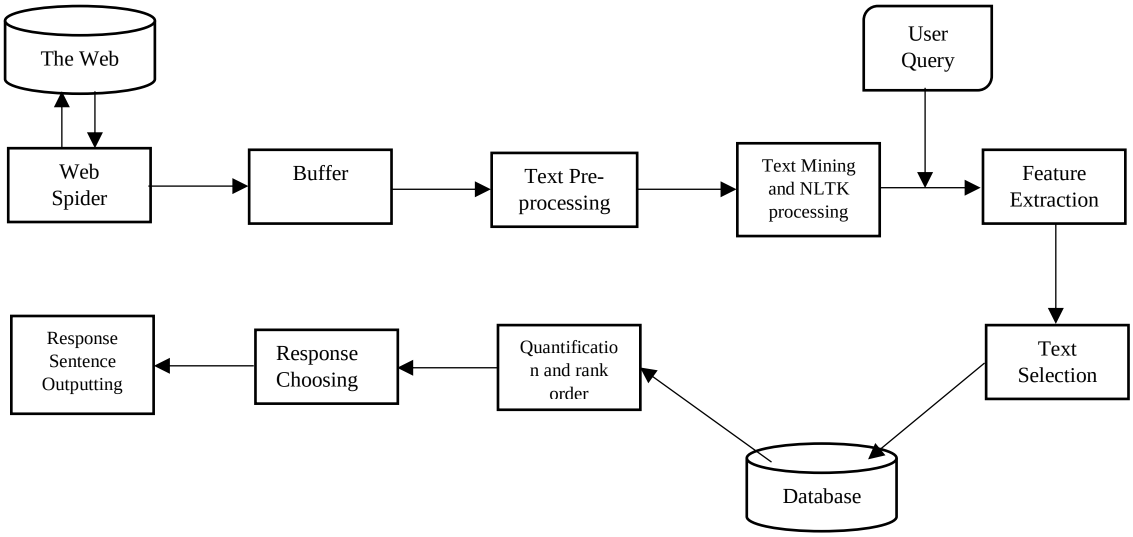


Fig. 4. 1: A block diagram for the proposed system.

The scores of syntactic, semantic, and term match are calculated then summed to obtain a total match score then a percentage match score. The sentences and the calculation results are all placed in an SQLite database prepared for that purpose. The sentences are then descending re-ranked according to their matching scores to evaluate relevance of the highest scores.

The system is implemented in Python and the modules used in the implemented program are: NLTK, re, ngram, urllib, sqlite3, in addition to BeautifulSoup and WordNet.

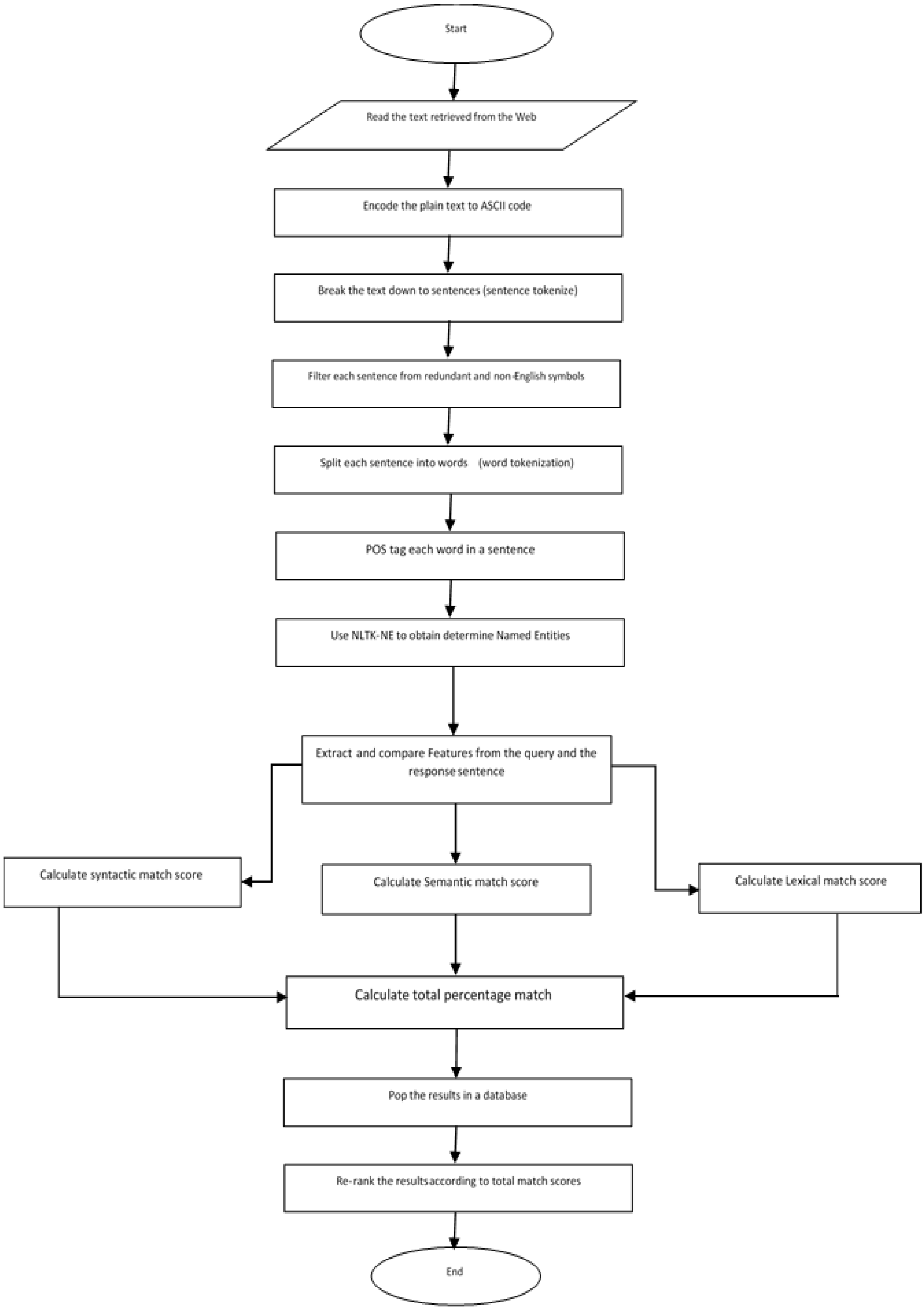


Fig. 4. 2: A flow diagram of the implemented system.

4.8 Evaluation

An experiment was prepared to test the proposed system's efficiency. The first stage of this experiment was preparing the questions that were the dataset of the test. We used our own dataset that was extracted from the web from the Wikipedia page of the English footballer David Beckham. The second data set used to evaluate the proposed system was the Stanford QA dataset (SQuAD). The second stage of the experiment was writing the rules for answer selection. The third stage was writing the algorithm of the program then coding the program. The Python programming language was used to code the implementation programs. The fourth stage was running the code and using the QA data sets as inputs to produce outputs. The fifth stage was storing the output data in an SQLite database for the purpose of evaluation. The final stage was transferring the evaluation results into graph forms using the Python code.

4.8.1 Experiment's Evaluation Metrics

In this experiment evaluation metrics were applied to see the enhancement our system added and also to compare it with other comparative systems. Precision@N and mean average Precision (MAP) were used to evaluate ranked information retrieval [14]. Thus, Precision@10, Precision@20 and MAP were used to evaluate our output answers to queries after rank-ordering the answers.

The results were also evaluated using the mean reciprocal rank (MRR) method, which is suitable for measuring the performance of the implemented system. MRR is calculated relating to the following relation [156]:

$$MRR = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{r_i} \tag{4. 17}$$

where:

n is the number of questions.

i is the individual question number.

r_i is the reciprocal rank of the correct answer.

4.8.2 Experiment2

The experiment starts with inputting the formulas and rules that score response sentences and rank them according to the scores obtained. The input text was extracted from the Wikipedia page of the English footballer David Beckham. Python programming codes were then produced to implement the equations that are derived in the theoretical part. Different Python modules are used to extract and filter the plain text then to sentence and word tokenize the resultant filtered text; for example, NLTK, NLTK-NE, WordNet, NumPy, ngrams, urllib, and BeautifulSoup. The resultant sentences were popped into a text file for retrieval in the run time of the calculation programs. Sixty-four questions (not the questions generated in Chapter 3) and their predicted answers were prepared to use with the extracted and prepared text.

Also a SQuAD data set was prepared by inputting 54 source text paragraphs into source files and preparing 200 questions to be used with these text sources. Our four invented formulas in addition to the comparative system's in [112] were coded into five Python programs, one for each of our four formulas and one for the comparative system; and we ran them all together using the two data sets; one run for each of the five programs per each query. The experimental results of the 264 runs for each program were stored in SQLite databases. The evaluation part was started when all runs for all the corresponding experimental programs were completed. Precision@10, Precision@20, MRR, and MAP were calculated and graphs for the evaluation results are presented.

4.8.3 Experiment Goal

The aim of the experiment was to implement our QA system that is proposed for the OFC. The experiment was also conducted in order to evaluate the performance of the system by applying two QA data sets and storing the results in an SQLite database prepared for this purpose. Moreover, Precision@10, Precision@20, MAP, and MRR were used for the purpose of evaluation.

4.8.4 Experiment Requirements

The experiment requirements can be listed as below:

1. The theoretical rules and hypotheses that were needed to be implemented in order to find a best answer to a query from a set of sentences extracted from a source text.
2. A programming language to implement the proposed rules and hypothesis, and to calculate the evaluation results then to draw the evaluation graphs. The programming language we used is Python. Python was chosen because it is a reliable language in text processing and array operations. The MATLAB programming language was also required to draw the bar graphs of the results.
3. Data set(s) of QA as input to the program coded in Python in order to test and evaluate whether the system is successful and comparative.
4. Python modules, such as NumPy, urllib, NLTK, NLTK-NE, and BeautifulSoup. These modules help in retrieving information from the web using a web crawler (the same web crawler in Chapter 3 fig. 3.4) and in extracting the required plain text which is then filtered and tokenized into sentences and saved in a source file.
5. Other Python modules such as WordNet, NumPy, ngrams, and sqlite3 are required in the stage of calculations and storing the results.
6. A database browser for SQLite database is needed to create tables required for experimental results storage and to run short SQLite programs to rank order the stored answer sentences according to their percentage scores.

4.8.5 Experiment Steps

The steps that the experiment runs through are as follows:

1. Two data sets were prepared to evaluate our system and to compare it with other comparative systems.
    - The first data set was our QA data set prepared according to the adaptation of TREC data set types to our subject, which is the footballer

David Beckham. We prepared 64 questions about the career and the personal life of that person. The source text used with this QA set was our extracted text from David Beckham's Wikipedia page. The QA set used in this chapter are not the ones generated in Chapter 3.

- The second data set was the Stanford QA dataset (SQuAD). We saved 54 source text paragraphs of the data set in source files and we selected 200 QA sets that are related to these text sources.

2. A Python programming code was written to implement the proposed QA system. The code is divided into five programs; one for each of the four formulas in equations (4.12) to (4.15), and one for the comparative system in [112]. The five programming codes are doing the following tasks:

   i. Accessing the Wikipedia page of David Beckham using the URL of this page using the web crawler that is explained in Chapter 3 and illustrated in fig.3.4. The purpose of this access is to extract the information in the corresponding URL. We then extract all the URLs embedded in the main Wikipedia page of David Beckham. The extracted URLs were used to extract more information from the pages related to these URLs. Information from 100 URLs was extracted afterwards.

   ii. Filtering the HTML code after extracting the information from the 100 web pages to acquire the plain text. This plain text was filtered to remove extra punctuation, non-English symbols or letters, or any redundant information. The filtered text was then converted to the right ASCII code format.

   iii. Sentence tokenizing the resultant text then saving it in a text source file for the purpose of processing.

   iv. Retrieving the resultant filtered sentences from the source file, then processing each sentence to extract features from it. The processing starts from word tokenizing the sentence, then POS tagging, followed by some more filtering for more symbols like redundant brackets, and then using feature extraction and score calculation.

v. Syntactic information was extracted using POS tag and then the similarity score between the query and the response sentence was calculated.
vi. Term similarity was obtained by calculating Jaccard's coefficient and cosine similarity values.
vii. Semantic features were extracted and semantic similarity values were calculated using named entity and semantic cosine similarity.
viii. The results of syntactic, semantic and term similarity scores were added together using equations (4.12) to (4.15).
ix. The percentage of the results was calculated and the values of syntactic, semantic, and term similarity scores were stored in an SQLite database together with the corresponding response sentence, overall scores and the percentage scores. The results of 264 queries per each of the five programs were stored in 10 databases, one database for each program per each data set.

3. The experimental results were examined table by table after ranking the answers in descending order according to the overall percentage score of each sentence to find the relevant answers.
4. After recording the rank order of each relevant answer, the evaluation stage was started by calculating the precision@10, precision@20, and recall values for each query per five tables. Then MAP and MRR were calculated for each of the two data sets per each of the five programs (the Python programs one program per each of our four formulas and one for the comparative system).
5. Precision-recall graphs were produced for precision@10 and precision@20 for each of the two data sets: our QA dataset and SQuAD.
6. Bar graphs of MAP and MRR values for each five data groups for each dataset were plotted using the MATLAB programming language.

4.9 Comparative System

The comparative system in [112] is named by its authors as SemEval-2016 Task 3. The SemEval-2016 Task 3 is described as follows [112]:

1. SemEval-2016 Task 3 aims to solve a QA problem in community forums, which is caused by users having to post questions that have already been asked and answered. This problem annoys the users because it makes them refer to those previously asked questions.
2. The main subtask (C) asks a question to find an answer which already exists in the community forum and it is suitable as a response to a newly posted question.
3. SemEval-2016 Task 3 uses semantic vector similarity by using semantic word embedding obtained from Word2Vec.
4. In semantic vector similarity, SemEval-2016 uses a number of similarity features calculated by using the centroid word vectors of the question.
5. The centroid word vectors of the question are constructed according to the following relation:

$$Centroid\ (w_{1\ldots n}) = \frac{\sum_{i=1}^{n} w_i}{n} \qquad 4.18$$

   Where:

   w_i represents a word in a sentence.

   n is number of words in the sentence.

6. The authors in [112] assume that the relevant answer should have the closest centroid vector to the centroid vector of the question.
7. SemEval-2016 Task 3 orders each word in the answer to the question body centroid vector according to their similarity and takes the average similarity of the top N words. The authors of SemEval-2016 Task 3 assume that if the average similarity of the top N most similar words is high, the answer should be relevant.
8. In addition to semantic features, the authors of SemEval-2016 Task 3 consider some metadata common meaning features, such as answer length, question length, and question to comment length.

According to the points we have described above, SemEval-2016 Task 3 uses semantic features to extract the highest scored answers in addition to some metadata features. We programmed the system in SemEval-2016 Task 3 with our work and we used the same data set to test it and compare it with our system.

4.10 Experimental Results

Let us give the combination in equation (4.12) the abbreviation Jaccard POS tag Named Entity (JPNE) and the combination in equation (4.13) the abbreviation Cosine POS tag Named Entity (CPNE). Also, let us give the combination in equation (4.14) the abbreviation Jaccard POS tag Semantic Cosine (JPSC) and the combination in equation (4.15) the abbreviation Cosine POS tag Semantic Cosine (CPSC). The name of the comparative system is already explained in the previous section.

The input data to the system was the unstructured text retrieved from David Beckham's page on Wikipedia (https://en.wikipedia.org/wiki/David_Beckham). Over a hundred URLs associated with Beckham's page on Wikipedia were accessed to retrieve their unstructured data in order to extract the plain text needed. The output was a set of rank ordered sentences according to a query given after filtering and structuring the unstructured plain text. The resultant output was put into a table of an SQLite database for evaluation purposes. General and personal queries were used to verify the proposed system operation. The resultant sentences have been filtered to obtain typical length (i.e. not too long) sentences of no more than 21 words in order to elicit clear and concise answers. Random examples of the queries, the closest match sentences, and the closest match scores for combinations in equations (4.12), (4.13), (4.14), and (4.15) are tabulated in Table 4.1. The experimental results demonstrated in Table 4.1 give the highest scored and the closest matches out of over 2000 records.

The number of records means the total number of sentences in the database table which the highest and the lowest score sentences are part of. The sentences with 0 scores have been excluded and are not recorded in the database.

The experimental results were stored in 10 SQLite databases; two for each combination formula: one of these two is for our OFC data set, which contains 64 tables of unranked answers and 64 reviews for the answers in the 64 tables after ranking order. The second of the two databases was for the SQuAD data set and it contains 200 tables for unranked answers and 200 reviews for the answers in the 200 tables after ranking order.

Table 4. 1: Examples of Experimental results.

No.	Query	Nearest match Sentence	JPNE %	Record order in JPNE	CPNE %	Record order CPNE	JPSC %	Record order in JPSC	CPSC %	Record order in CPSC	SemEval-2016 Task 3	Record order in SemEval-2016 Task 3	No. of records
1.	Where was David Robert Joseph Beckham born?	David Robert Joseph Beckham OBE 4 b k m/ born 2 May 1975 is an English former professional footballer.	32.97	2	33.33	2	57.27	1	57.63	1	47.43	2	2165
2.	When did he announce his retirement?	He announced his retirement in May 2013 after a 20-year career during which he won 19 major trophies.	20.93	2	22.73	1	34.3	3	36.1	1	33.1	2	2313
3.	With which team did Beckham's professional club career begin in 1992?	Beckham's professional club career began with Manchester United where he made his first-team debut in 1992 aged 17.	33.5	2	33.8	2	54.57	1	55	1	41.53	1	2743
4.	Which local youth team did he play for?	He played for a local youth team called Ridgeway Rovers.	37.03	1	37.13	1	54.13	1	54.2	1	45.23	1	2178
5.	Which trials did Young Beckham have and which school of excellence did he attend?	Young Beckham had trials with his local club Leyton Orient Norwich City and attended Tottenham Hotspur's school of excellence.	22.87	8	22.3	9	38.6	1	38.03	1	32.71	3	2717
6.	How many years did he spend playing football?	Beckham played in all of England 's matches at Euro 2004.	44.25	2	45.17	2	65.23	1	65.23	1	52.18	2	1827

4.11 Evaluation Results

After collecting the experimental data and putting it into the SQLite databases, the evaluation stage began. The orders of the correct answers were recorded for each query in the two data sets for the five experimental groups of data. Precision at the two cut off regions 10 and 20, precision@10 and precision@20 were calculated for each experimental data group for the two data set queries. Recall was also calculated for the same items that precision was calculated for. Precision and recall values were calculated using a Python programming code and the graphs of Precision@10-Recall and Precision@20-Recall were drawn. Precision@10-Recall and Precision@20-Recall graphs are reported as follows:

Fig.4.3 illustrates the Precision@10-Recall graph of the five experimental data groups for our OFC data set. The graph shows that the highest accuracy is given by CPSC and the lowest by JPNE. The comparative system is in the middle of the five comparative models. The plotted curves are leaning slightly more to the left because we consider only one specific correct answer for each query and this affects the values of precision and recall in the middle and makes them lower.

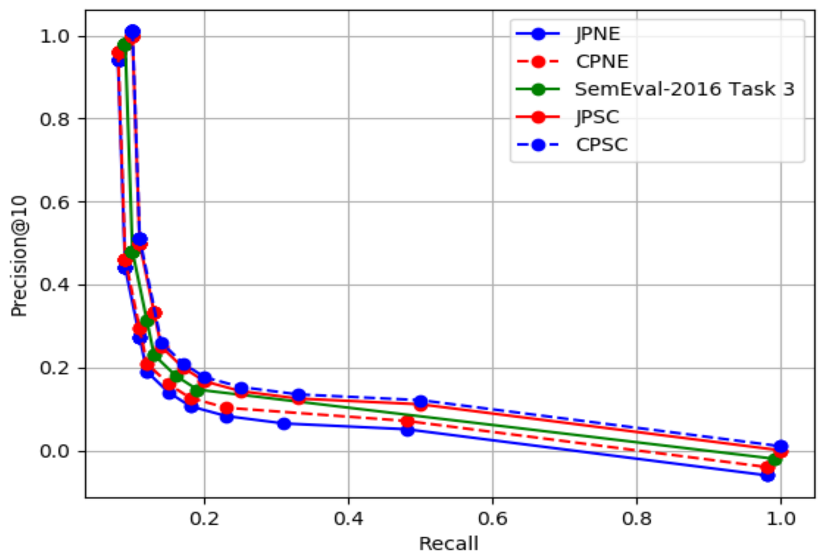


Fig. 4. 3: Precision@10-Recall graph for evaluation of the four experimental data groups and the comparative system (using our OFC data set).

Fig.4.4 below demonstrates the Precision@20-Recall graph of five experimental data groups for our OFC data set. The graph reports that CPSC, JPSC, and SemEval-2016 Task 3 are becoming closer to each other and leading while CPNE and JPNE are lagging.

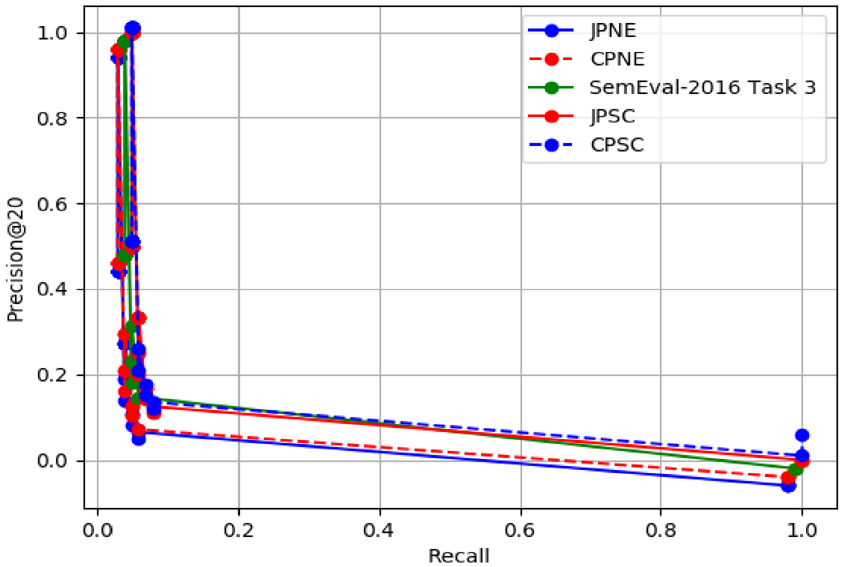


Fig. 4. 4: Precision@20-Recall graph for evaluation of the four experimental data groups and the comparative system (using our OFC data set).

Precision@10 graph of the five groups of data using the SQuAD data set is shown in fig.4.5. The graph shows that CPSC and JPSC are closer to each other and leading. The comparative system is slightly closer to the lagging ones, CPNE and JPNE.

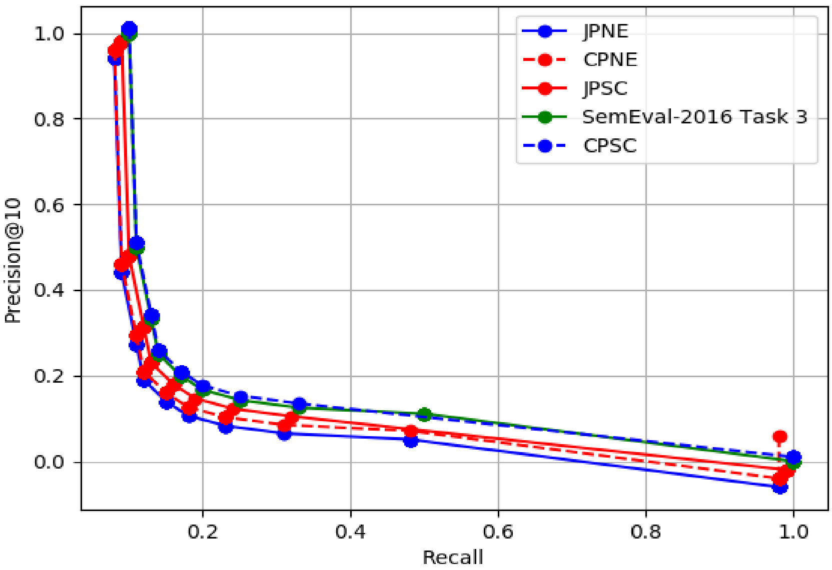


Fig. 4.5: Precision@10-Recall graph for evaluation of the four experimental data groups and the comparative system (using SQuAD data set).

Fig.4.6 below shows the Precision@20-Recall plot for five experimental data groups including the comparative system using the SQuAD data set. The graph illustrates that CPSC and SemEval-2016 Task 3 occupy the top of the curves' positions, while JPNE and CPNE are at the bottom, and JPSC occupies the middle.

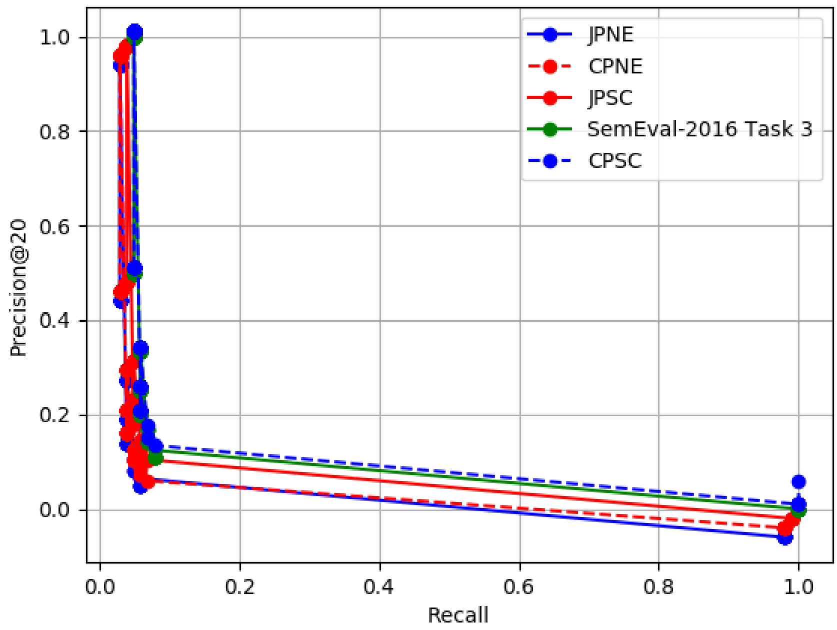


Fig. 4. 6: Precision@20-Recall graph for evaluation of the four experimental data groups and the comparative system (using SQuAD data set).

MAP and MRR curves were also plotted over the five experimental groups including the comparative system as other evaluation metrics and comparison criteria between our proposed four formulas and the comparative system. The graphs were drawn for the two data sets used and are explained as follows:

Fig.4.7 shows the bar graph of MAP values of the five experimental data groups using our OFC data set. The bar graph reports that CPSC has the highest MAP with a value 0.85 and JPNE has the lowest with the number 0.71. SemEval-2016 Task 3 is in the middle having the value 0.76 between CPNE and JPSC.

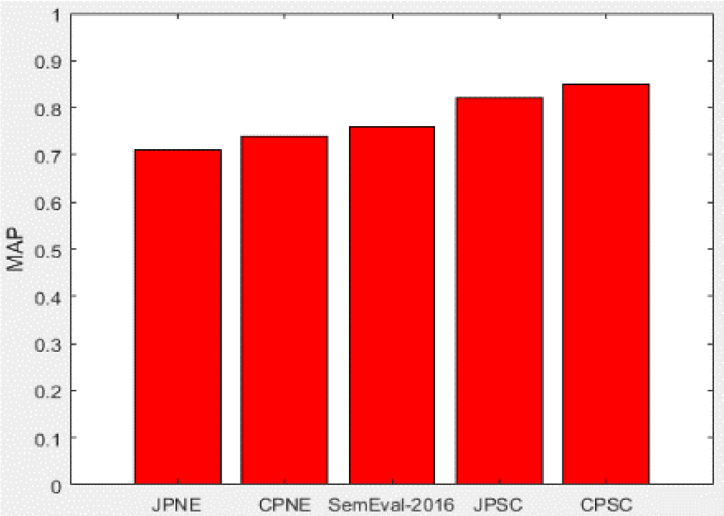


Fig. 4. 7: MAP values graph for evaluation of the four experimental data groups and the comparative system (using our OFC data set).

Fig.4.8 demonstrates the bar graph of MAP for the five experimental groups of data using the SQuAD data set. The graph shows that SemEval-2016 Task 3 occupies the second position after CPSC with the values 0.78 and 0.80, respectively. JPNE is the lowest with 0.64 then CPNE with the value 0.67 and JPSC with 0.72.

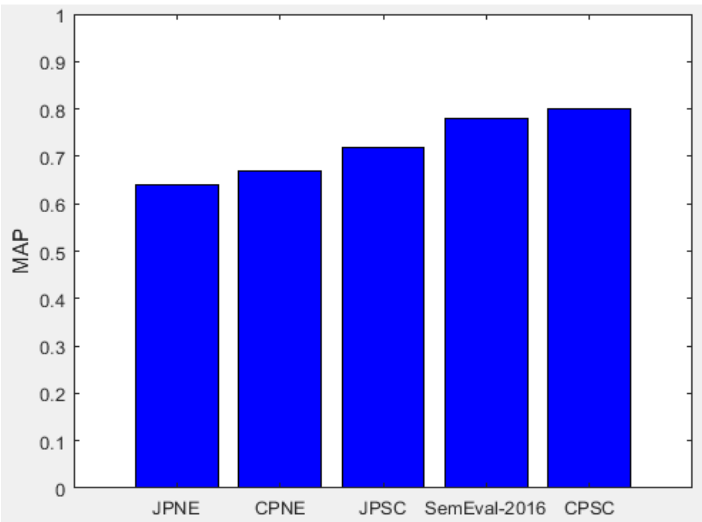


Fig. 4. 8: MAP values graph for evaluation of the four experimental data groups and the comparative system (using SQuAD data set).

The bar graph of MRR values for the five experimental data groups using our OFC data set is shown in fig.4.9. The bar graph illustrates that the highest MRR is achieved by CPSC and the lowest by JPNE with the values 0.85 and 0.71, respectively. SemEval-2016 Task 3 is in the middle with the value 0.77 between CPNE and JPSC with the values 0.74 and 0.82, respectively.

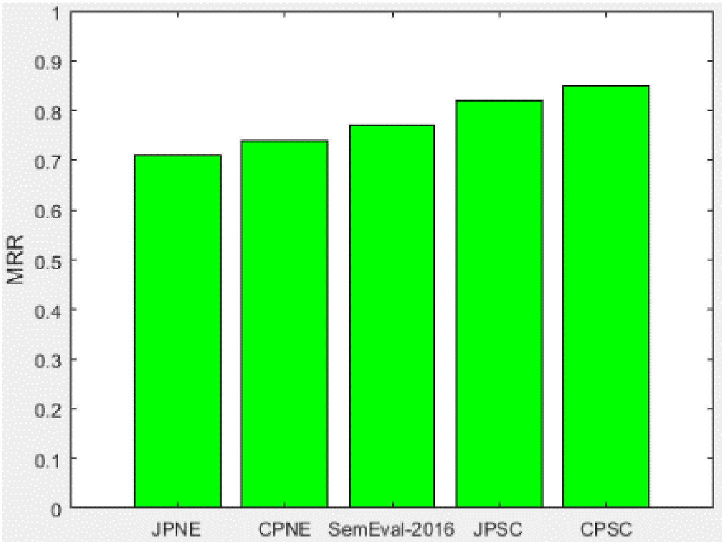


Fig. 4. 9: MRR values graph for evaluation of the four experimental data groups and the comparative system (using our OFC data set).

The bar graph in fig.4.10 presents the values of MRR for the five experimental data groups using SQuAD data set. The graph shows leading again for CPSC over the other four groups with the value 0.80 followed by SemEval-2016 Task 3 with the value 0.78. JPNE is again the lowest one with the value 0.64 after CPNE and JPSC with the values 0.67 and 0.72, respectively.

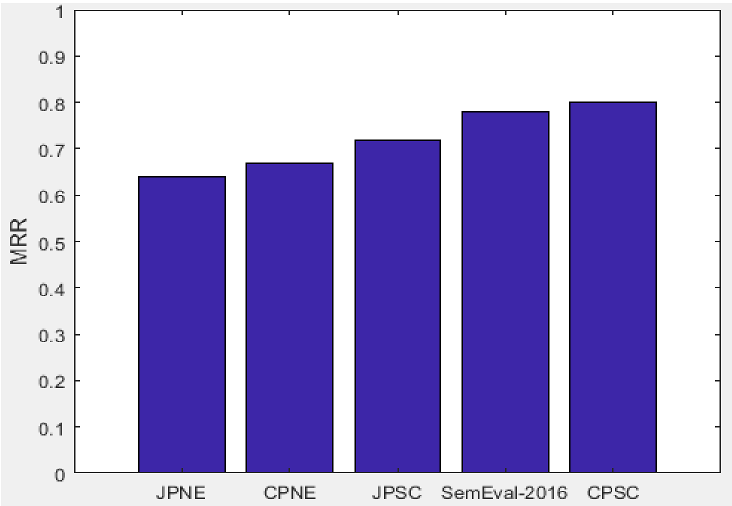


Fig. 4. 10: MRR values graph for evaluation of the four experimental data groups and the comparative system (using SQuAD data set)

The overall MAP values of the five experimental data groups are shown as a bar graph in fig.4.11. The graph reports a clear contribution for CPSC with the value 0.825 followed by SemEval-2016 Task 3 and JPSC with the values 0.77 and 0.765, respectively. Then CPNE and JPNE come with the values 0.705 and 0.675, respectively.

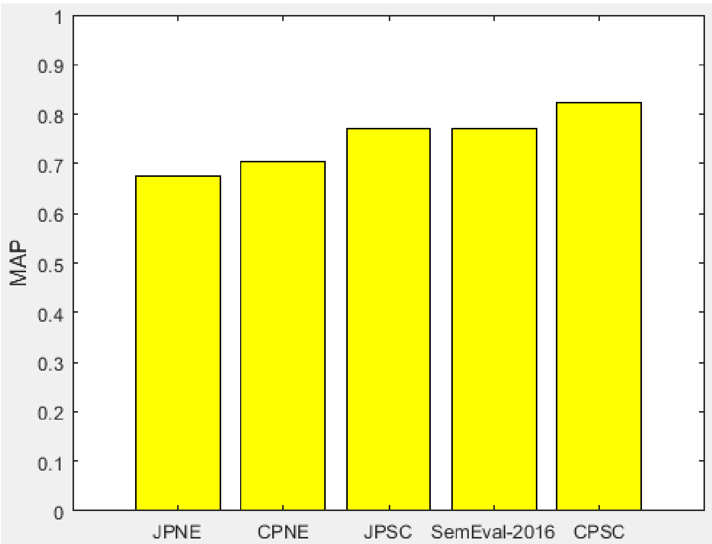


Fig. 4. 11: Overall MAP values graph for evaluation of the four experimental data groups and the comparative system.

The overall MRR values of the five experimental data groups are shown in the bar graph of fig.4.12. The bar graph illustrates an achievement for the CPSC group with a value 0.825 followed by SemEval-2016 Task 3 with 0.775 and JPSC with 0.77. CPNE and JPNE values are 0.705 and 0.675, respectively.

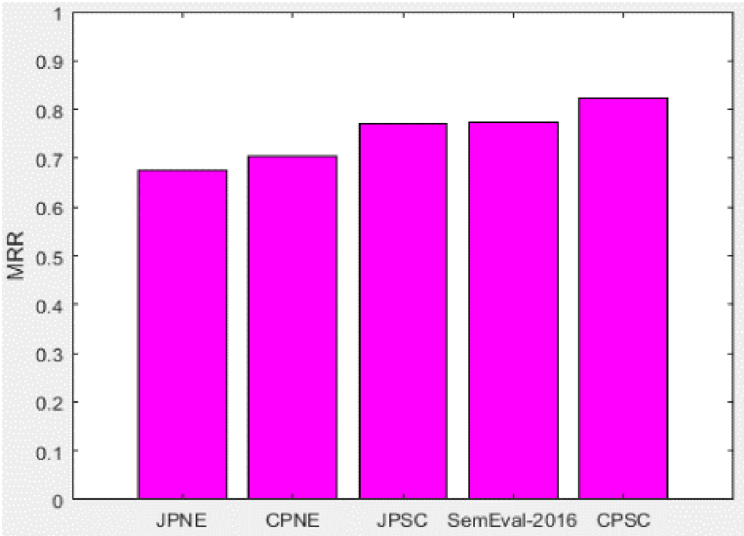


Fig. 4. 12: Overall MRR values graph for evaluation of the four experimental data groups and the comparative system.

Overall, the results indicate that the formula in equation (4.15), which is the combination of cosine similarity, POS tag, and semantic cosine (CPSC), outperforms the comparative system SemEval-2016 Task 3 1.55 percentage points in MAP and 5 points in MRR. The other three combinations are just behind the former two. This means that CPSC is the best among the five evaluated systems.

4.12 Conclusion

In this chapter, a new method that employs multiple feature extraction has been presented to quantify text responses for a learning Chatbot. More than one measurement metric has been examined at the same time to find the best match to a Chatbot query. Four combinations of extracted features were formulated and compared with a comparative model. Re-ranking the scores of extracted features for text responses gave the most semantically meaningful sentences. The experimental results show that the highest scored sentences are the nearest to a query. Evaluation results show that the system performance rises significantly by using the cosine similarity metric for term match and semantic cosine similarity for semantic match. The combination of cosine similarity, POS tag, and semantic cosine similarity achieved the highest values in evaluation metrics and outperformed our other three combinations and the comparative system.

Chapter Five

Automatic Extraction of Imperative Sentences from the Web for Online Feedable Chatbot

5.1 Introduction

In daily discourse, different forms of speech are needed, such as greetings, questions and answers, requests, obligations, explanations, permissions, and commands. Imperative sentences are a style of speech needed in humans' every day activities in different places and situations. They are also needed when using computer programming [133], searching in a huge database [132] or searching for ideas in articles [129].

In order to make conversation more natural with dialogue systems, phrases or sentences from daily life need to be inserted into their knowledge bases like imperative sentences. An imperative sentence is simply a command telling someone to do something [157]. This type of command helps the user to direct the dialogue system to do a needed action, such as searching in the web, opening a new window, starting a program, or even shutting the computer down or putting it in sleep mode.

This chapter is concerned with designing a system for automatic acquisition of imperative sentences from the web for the knowledge base of the OFC. Thousands of sentences are extracted from 200 Web pages associated to the Wikipedia page of the famous footballer David Beckham and pre-processed then filtered in order to be the data set of the proposed system. NLTK-POS tagger and verb tense type are used to identify and select the imperative sentences from the extracted set of sentences. The resultant sentences are stored in an SQL database to be used as part of the knowledge base of the OFC. The aim is to add more actionable activities to the Chatbot in the future using the extracted imperative sentences like controlling applications on the computer. The results are evaluated using a human assessment subjective test and compared with other comparative systems. The evaluation results show that our system's performance outperforms the comparative system's.

5.2 Imperative Sentence

Sentences in the English language normally consist of the main sentence parts starting with the *subject*. The sentence formulation is usually as follows:

$$Subject + Verb + Object((or)sentenceCompletion) \qquad 5.1$$

Examples like:

1. The dog sat on the mat.
2. She is very beautiful.
3. Layan goes to school.

are representative of the rule presented in equation 5.1.

The subject could be a true noun or a pronoun and there is a variety of verb tenses in the English language, such as simple present, perfect present, simple past, present participle, auxiliary verbs, etc.

In imperative sentences of the examples below:

4. Please open the door.
5. Shut down the computer.
6. Can you reach the salt?
7. I encourage you to exercise every day;

there is no obvious subject seen and the subject here is (you). A person in this kind of sentence orders or commands another person to do something. According to the main rule in equation 5.1, the sentences in examples 4 and 5 should be as follows:

8. You please open the door.
9. You shut down the computer.

The subject (you) here is considered as *understood* in order to formulate the imperative sentences in examples 4 and 5.

As noticed, the imperative sentences come in different styles: indirect like in examples 6 and 7, and direct like in examples 4 and 5 [158-160].

In this chapter, we concentrate on one type of direct imperative sentences which begin with the imperative verb.

5.3 Imperative Sentence Identification

As mentioned in the previous section, the type of imperative sentences extracted in this chapter is the direct one starting with a verb. Our hypothesis of extracting direct imperative sentences that begin with verb is as follows:

i. The sentence should begin with a verb.
ii. The sentence should not begin with a noun, pronoun, or any words other than verbs.
iii. The verb at the beginning of the sentence should be simple present.
iv. The sentence is considered even if it is only one word (a verb).

According to the hypothesis stated above the rule of an extracted imperative sentence is as in the following equation:

$$Verb + Object(\text{a noun or noun phrase, or propositional phrase}) \qquad 5.\ 2$$

5.3.1 Syntactic Analysis for the Sentence

To analyse a sentence using NLP, word tokenising is needed for the sentences in order to split the sentence into individual words. Then each word should be POS tagged with a part of speech to identify the type of each word in the sentence. NLTK-POS tagger tags the first word in the sentence as NNP, which means a proper name, or PRP which means a pronoun, assuming that the first word in a sentence must be a subject according to basic English language rules. For example:

10. Beckham is joining the Los Angeles Galaxy after years with Real Madrid.

is POS tagged as follows:

[('Beckham', 'NNP'), ('is', 'VBZ'), ('joining', 'VBG'), ('the', 'DT'), ('Los', 'NNP'),

('Angeles', 'NNP'), ('Galaxy', 'NNP'), ('after', 'IN'), ('years', 'NNS'), ('with', 'IN'), 'Real', 'JJ'), ('Madrid', 'NNS')]

11. It is not that easy.

is POS tagged as follows:

[('It', 'PRP'), ("is", 'VBZ'), ('not', 'RB'), ('that', 'IN'), ('easy', 'JJ')]

NLTK-POS tagger does the same thing with the sentences that begin with a verb. It tags the verb which sits in the beginning of the sentence as NNP, which makes it indistinguishable from the nouns or pronouns. For example:

12. Give me a football.

[('Give', 'NNP'), ('me', 'NNP'), ('a', ' DT '), ('Football', 'NNP')]

We hard coded in Python programming language to tag the verb in the beginning of a sentence as a verb. The sentence in example 12 above can be POS tagged as follows:

13. [('Give', 'VB'), ('Me', 'NNP'), ('a', ' DT '), ('Football', 'NNP')]

5.4 The Proposed System

In the proposed system, the first process done is crawling the web pages using a web crawler in order to extract the plain text needed. The same web crawler in Chapter 3 fig. 3.4 is used in the proposed system.

After extracting the plain text from the web pages associated to the given main URL, the text is filtered from the HTML code and UNICODE and the text is turned into ASCII code. The resultant text is sentence tokenised then each sentence is word tokenised. Then the resultant sentences are filtered to remove undesired (redundant) information, such as non-English letters or symbols, and English redundant symbols and punctuation. POS tagging for each word in the resultant sentences is done in order to tag the words with the parts of speech after the long sentences with more than 15 words are eliminated because they are too long to be evaluated. Fig.5.1 shows the main block diagram of our proposed system of imperative sentence extraction.

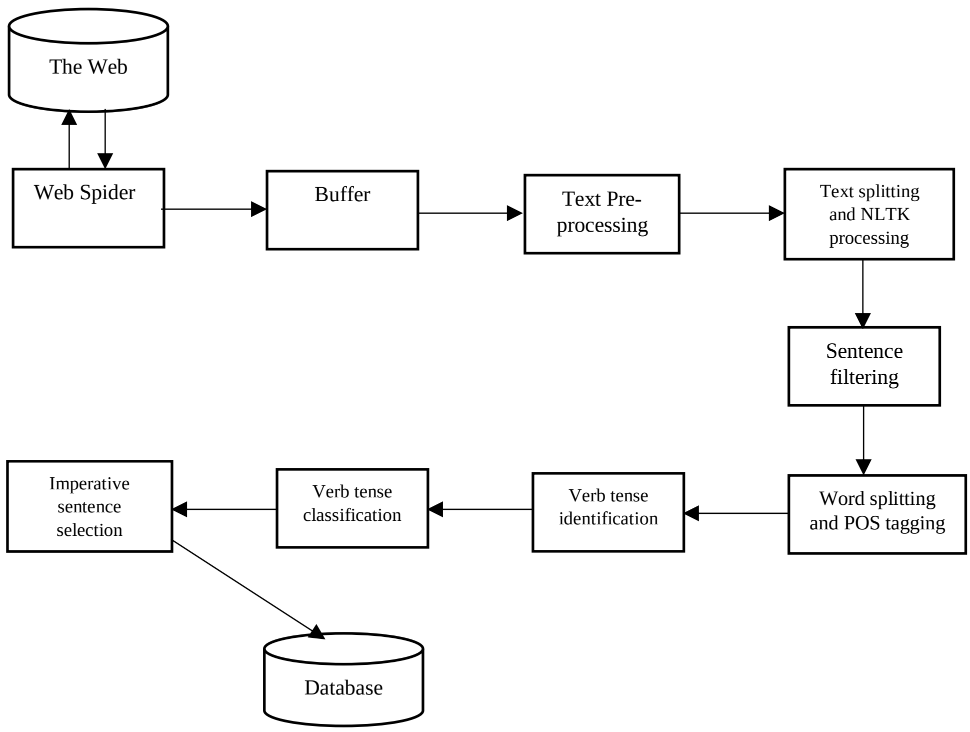


Fig. 5. 1: The main block diagram of Imperative sentence extraction from the web.

The main idea that we decided to identify and select the imperative sentences is detecting the main verb at the beginning of the sentence. This is implemented by detecting the POS tag of the first word in each sentence. If the POS tag of the first word denotes a verb and the verb is simple present, then the sentence is selected as an imperative. The resultant imperative sentences are placed in an SQL database in order to validate them and also as a part of the knowledge base of the conversational agent OFC. The implemented steps of imperative sentence extraction are shown in the flow diagram in fig.5.2.

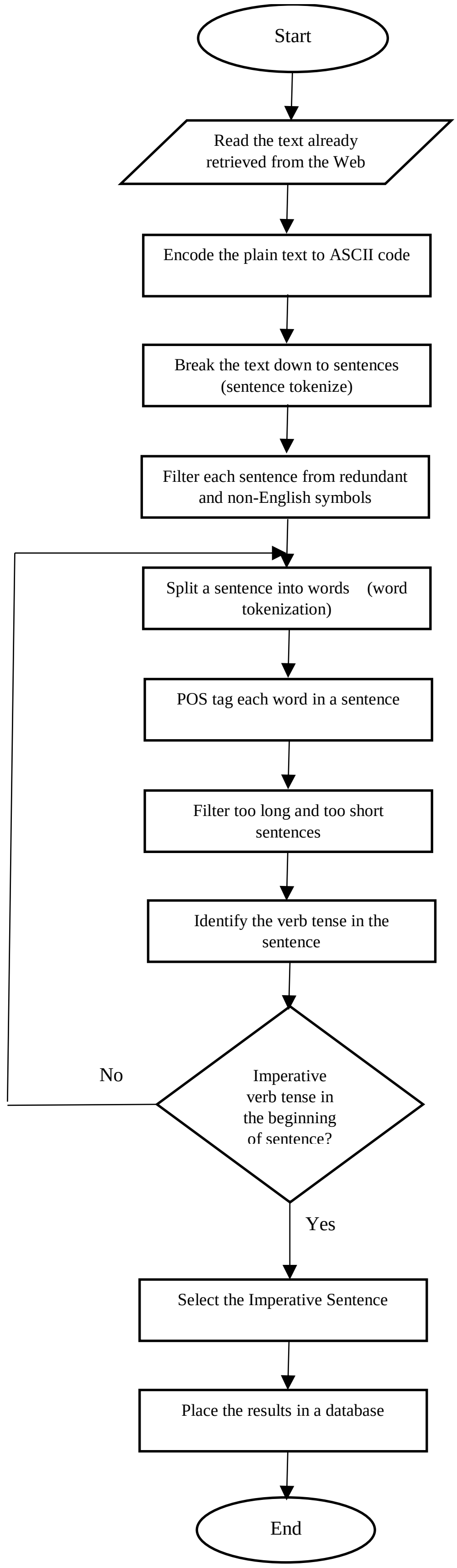


Fig. 5. 2: Implementation steps to process plain text and extract imperative sentences.

5.5 Evaluation

In order to evaluate our proposed system, an experiment was needed. For this purpose, a subjective test experiment of two parts using human assessment was conducted. The first part of the experiment was to design and implement our proposed system of automatic extraction of imperative sentences from the web and then to evaluate it by human assessors. The second part was to adapt the system in [132] to our data set and implement it then evaluate it using human participants. The subjective assessment was used in order to validate the resultant imperative sentences from our proposed system and then compare our system with the comparative system in [132]. The number of the participants who joined the subjective test was 30 and they were divided into two groups: one to evaluate our system's output and the other to evaluate the comparative system's.

Implementing the subjective test required us to prepare a subjective questionnaire that asked the users to score the relevance of the imperative sentences between 1 and 4 as: 1. Totally unacceptable. 2. Unacceptable. 3. Acceptable 4. Strongly acceptable. The assessment is in terms of grammar, meaning, relation to the subject (David Beckham or football), and being imperative or not. This questionnaire form was used for both our and the comparative system.

5.5.1 Experiment's Evaluation Metrics

After using the subjective test, the measurement metric used was the same as that used in unranked retrieved systems, which is Precision [14]. Precision was used to evaluate the results of the subjective assessment of the experiment in order to assess the accuracy of our system and to compare it with the comparative system. The same evaluation metric was used for both the proposed and the comparative system and the results are discussed in the following sections. Also, to justify the subjective evaluation results, average score, standard deviation, t-test, and p value were calculated for both our and comparative system's using a Python program.

5.6 Experiment3

Like the experiments in the previous chapters, this experiment began with thinking of the rules and the hypotheses needed to extract imperative sentences from a piece of text extracted from web pages (Wikipedia) using a web crawler. Then the resultant sentences were evaluated by subjective assessment and compared to a comparative system that was adapted to our dataset and evaluated using the same subjective test. The experiment was conducted with a group of participants who are PhD students in different research areas at the University of Essex. Then the evaluation data was collected using the following steps:

1. Meeting each participant personally to explain the questionnaire to them and give them the questionnaire.
2. Meeting the participants again to collect the completed questionnaire from them.
3. Calculating the aggregate scores provided by them from the questionnaires.
4. Using aggregate scores in a Python program prepared to calculate, classify, and plot the graphs of the results.

5.6.1 Aim of Experiment

The goal of this experiment is to evaluate our automatic imperative sentence extraction system for the OFC and compare it with the system in [132] when implementing the hypothesis proposed to design this system. Precision level assessment was also applied in order to measure the improvement that can be added by our system

5.6.2 Experiment Participants

This experiment, as in Chapter 3, was conducted with PhD students. These PhD students are from University of Essex and are specialised in different fields, such as computer science, electronic engineering, linguistics, and mathematical sciences. Around 40% of the judgers were native English speakers and the rest (60%) were non-native English speakers.

The study included 30 participants (male, and female) distributed into two equal groups. The set of participants were chosen and then divided into groups depending on the theory of *within and in between* [137]. One of the two groups evaluated our system's output and the other evaluated the output of the comparative system. Each participant filled in the questionnaire to allow us to measure the accuracy accomplished by our proposed system. The results were the aggregate scores of the participants' responses across the questionnaire.

5.6.3 Experiment Steps

The experiment was executed through the following steps:

1. The Python programming language was used to implement our proposed automatic extraction for imperative sentences. The program runs through stages for a single execution. These stages are:
    a) Collecting the needed plain text from the web and this plain text was extracted using a web crawler (Chapter 3 fig.3.4). The web crawler crawled the Wikipedia page of the famous football player David Beckham and acquired a list of URLs within this page. Then, the web crawler accessed 200 other pages and extracted the plain text from these pages using the extracted URLs from the main page.
    b) The extracted plain text was pre-processed in the second stage. The pre-processing starts with filtering out undesired information such as extra punctuation, and non-English letters, non-English words, and non-English symbols. Then the filtered text is split into individual sentences and then the sentences are split into single words.
    c) In this stage the imperative sentences were selected from the group of extracted sentences according to the hypothesis explained in Section 5.3 above. The selection operation depends on detecting the verb tense in the beginning of the sentences. The verb detected in the beginning of the sentence should be simple present.

d) The resultant imperative sentences were saved in an SQLite database pre-prepared for the purpose of evaluation and to be used as part of the database of the OFC.

2. Copying the imperative sentences saved in the SQLite database and putting them in a table of an evaluation form.
3. Preparing the questionnaire for a subjective test including a table to clarify the idea and present the experiment results.
4. Searching for the participants to do the subjective assessment and choosing them. The participants should be kind of experts and carefully selected. They are relatively familiar with the famous footballer David Beckham, football, sports, and the English language.
5. Handing the questionnaire to the participants and explaining the experiment to every participant. After that, the completed questionnaire was collected from the participants.
6. Precision was then calculated and saved to be compared with the comparative systems'.
7. The same steps above were applied to the comparative system in [132] and the results are saved for the purpose of comparison with our system.
8. Bar graphs were drawn for the precision values of our and comparative systems. The graphs are discussed in the following sections.
9. For the purpose of validation of the subjective evaluation results, average score, standard deviation, t-test, and p value are calculated for both our and comparative systems' using a Python program.

5.7 Comparative System

We named the comparative system in [132] ANLIS from Arabic Natural Language Interface System in the title of the paper and our system as AEIS from Automatic Extraction of Imperative Sentences. The system in ANLIS is described briefly as follows:

1. ANLIS parses and interprets Arabic natural language inputs, such as imperative sentences and questions.
2. ANLIS applies context free grammar of Arabic language and morphology to analyse the input entries.
3. ANLIS detects imperative sentences or clauses in user entries depending on the analysis tools used in 2 above.
4. ANLIS extracts direct and indirect imperative sentences from user entries.
5. ANLIS produces an SQL command according to the information extracted from the imperative sentences.
6. ANLIS uses SQL commands to retrieve more suitable information from the Quran database.
7. The approach in ANLIS allows the users to use natural language to search for the information they need.

The system in ANLIS extracts the imperative sentences from computer inputs or user entries and uses free-context grammar to analyse the input sentences, whereas our system extracts imperative sentences from the web and uses the parts of speech tag approach to analyse the extracted sentences. These two points gave us the idea to compare ANLIS to our system. Although both systems detect imperative sentences, each uses a different approach to analysis, extraction, and application.

The system in [132] has been adapted to the requirements of our system by making it extract direct imperative sentences only and by using the English language instead of an Arabic parser then implemented and evaluated with the same evaluation method used for our system, then a comparison was made between them. The results of the comparison are shown in the sections below.

5.8 Experimental Results

The input to our system AEIS is the text extracted from the Web pages associated with the main web page of the English footballer David Beckham and the output is a number of imperative sentences depending on the number of URLs used. We ran the program of AEIS 8 times in order to test the number of sentences obtained in the output in relation to the number of the URLs used in the input to extract the plain text, and in order to find the optimum number of URLs that gives the optimum number of sentences. We used 1, 10, 50, 100, 200, 300, 400, and 500 URLs one value for each execution and we obtained the results as a graph shown in Fig.5.3. The graph shows that the number of output sentences increases rapidly as the number of URLs increases up to 400 then it decays at 500. We think that the pages start to repeat themselves after 400 so the resultant sentences number no longer increases after 400 URLs. The results represented by fig.5.3 include repeated items and redundancy. The redundancy increases as the URLs number increases because some of the pages are repeated as the URLs number becomes higher.

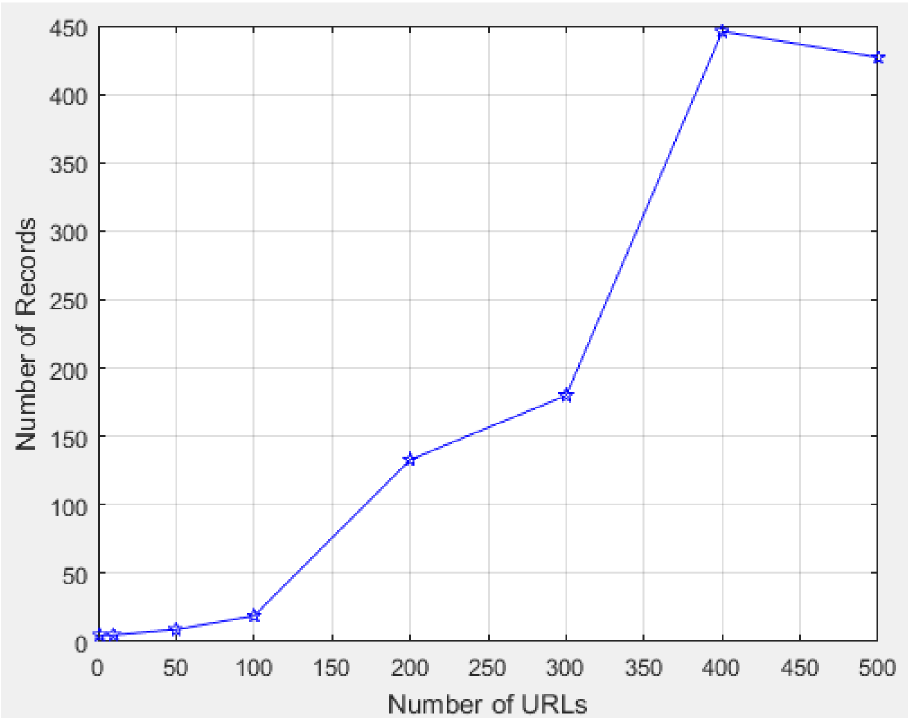


Fig. 5. 3: Extracted data against number of web pages before filtering.

The filtered results graph is shown in fig.5.4 after removing the redundant and repeated outputs. The aim of testing AEIS with different numbers of input URLs is to find the optimum number in the output after filtering in order to prepare it for the subjective test. As a result, 200 URLs was found to be the optimum number to produce the optimum output for the evaluation purpose because it gave us the minimum repetition in the results.

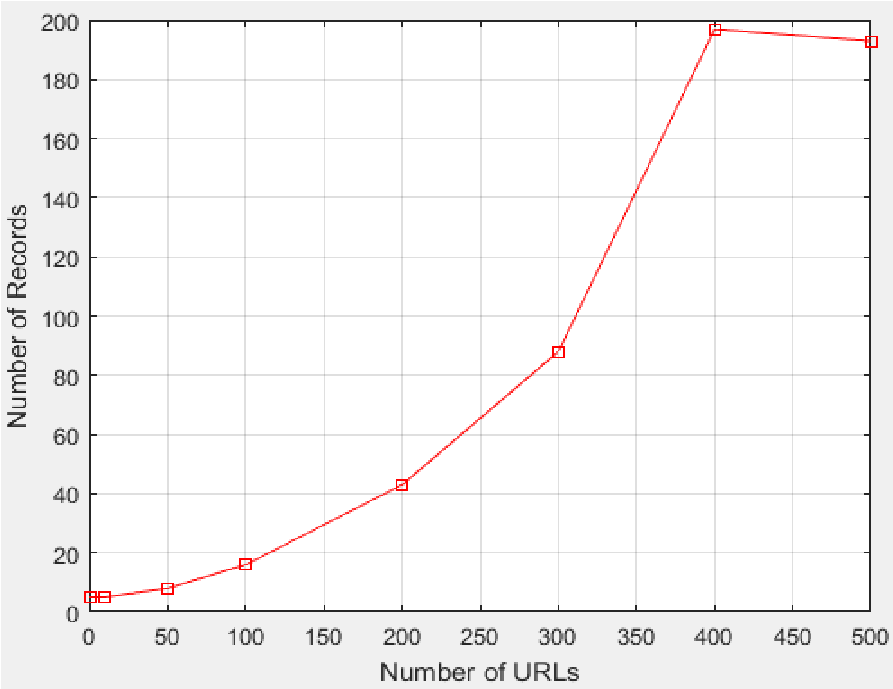


Fig. 5. 4: Extracted data against number of web pages after filtering.

So, the imperative sentences obtained from the plain text of 200 Web pages associated with David Beckham's Wikipedia page were stored in an SQL database for evaluation. Forty output sentences were extracted from 200 URLs in AEIS and thirteen output sentences were extracted from the same number of URLs in ANLIS. Samples of the experimental results are shown in Table 5.1.

Table 5. 1: Experimental results.

No.	**Imperative Sentence**
1.	Give that man a Knighthood.
2.	Give Me Football.
3.	Find out more about page archiving.
4.	Try our site map.
5.	Try again.
6.	Choose your language.

5.9 Evaluation Results

A subjective test was used to evaluate the experiment results of our system AEIS and the experimental results of the comparative system ANLIS. A questionnaire was prepared for the purpose of evaluation of both AEIS and ANLIS in order to use them for human participants. Expert human participants were needed to evaluate our system. The participants should have knowledge about the footballer David Beckham and football as well as English language. We chose 30 participants to join our subjective assessment and the vast majority of these participants were PhD students from University of Essex in different research areas. The evaluation questionnaire was given to the participants after explaining to them what to do and how to select scores. The questionnaire was then collected from the participants and the aggregate scores were calculated.

After finishing calculation of the data classes for both AEIS and ANLIS, we used a Python program to calculate the precision value for each part in each group of the two systems. Precision values were calculated for Grammar, Meaning, Relation to Subject, Imperative or not, and overall data groups in both AEIS and ANLIS. Precision calculation results were collected and saved then entered into a MATLAB program to produce comparative bar graphs for AEIS and ANLIS. The graphs are shown as follows:

The bar graph in fig.5.5 illustrates the precision levels of Grammar and Meaning groups for both AEIS and ANLIS. The graph shows proximity between AEIS and ANLIS in Grammar with AEIS exceeding by 1 percentage point with 0.81 for AEIS and 0.80 for ANLIS. AEIS outperforms ANLIS in the Meaning group by 3 percentage points with 0.85 for AEIS and 0.82 for ANLIS.

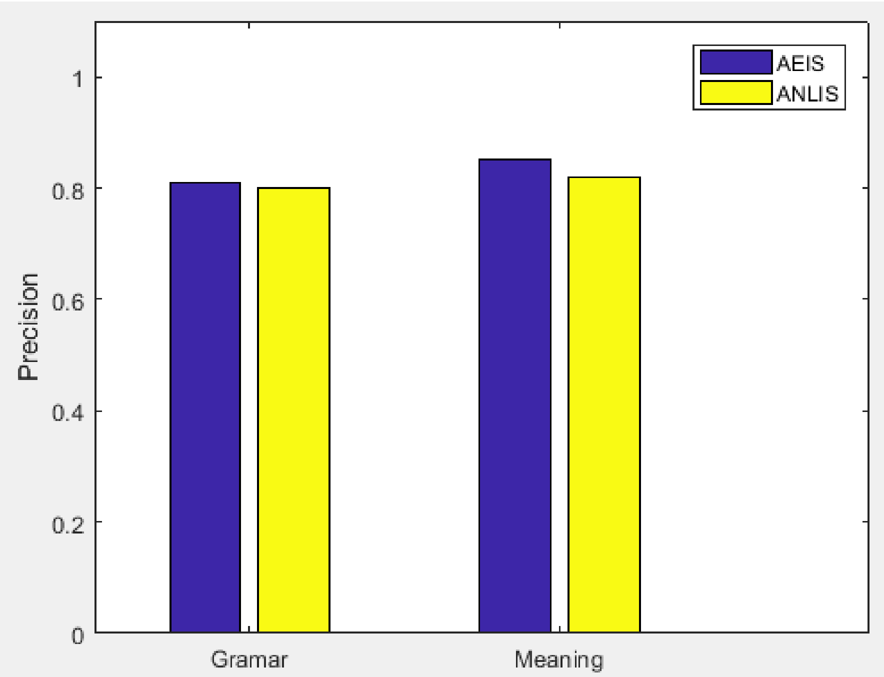


Fig. 5. 5: Precision comparison between AEIS and ANLIS (Grammar and Meaning)

The bar graph demonstrated in fig.5.6 is for precision levels of Relation to Subject, Imperative or not, and Overall groups for both AEIS and ANLIS. The graph reports exceeding by10 percentage points in the Relation to Subject group for ANLIS over AEIS with values of 0.67 for the former and 0.57 for the latter. AEIS beats ANLIS in the Imperative portion by 14 percentage points with 0.87 for the former and 0.73 for the latter. AEIS outperforms ANLIS in Overall precision by 5 percentage points with 0.81 for AEIS and 0.76 for ANLIS.

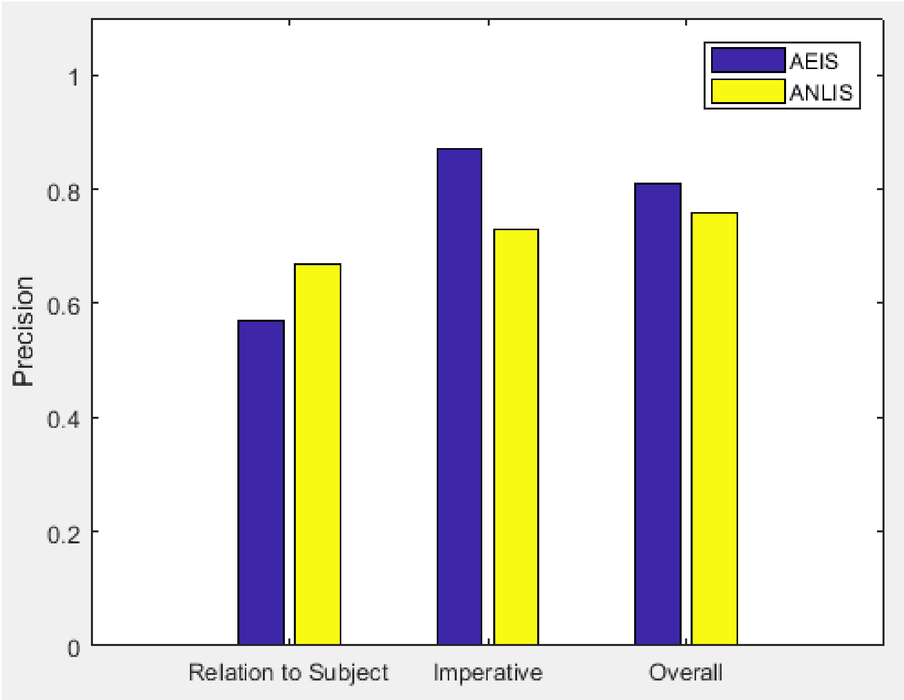


Fig. 5. 6: Precision comparison between AEIS and ANLIS (Relation to Subject, Imperative, and Overall).

Overall, recorded values of precision show that our system outperforms the comparative system by 5 percentage points. Also, according to the results, the portion of Imperative sentences in the output data in our proposed system is larger than the portion of Imperative sentences in the output data of the comparative system.

To validate the subjective evaluation results, average score, standard deviation, t-test, and p value for both our and comparative systems' were calculated using a Python program and the results are shown in Table 5.2. The t-test was calculated using the same relation for independent samples in Chapter 3 Section 3.10 [138] [139]:

The statistical values mentioned above were calculated for the groups Grammar, Meaning, Relation to Subject, and Imperative or not. The results show significance in the t-test for our system from the value of p in the Grammar, and Imperative or not groups and a value near significance in the Meaning group. Also, the comparative system obtained a value near significance in Relation to Subject.

Table 5. 2: The results of statistical calculations for subjective assessment evaluation for AEIS and ANLIS.

Statistics	**Grammar**		**Meaning**		**Relation to Subject**		**Imperative or not**	
	AEIS	ANLIS	AEIS	ANLIS	AEIS	ANLIS	AEIS	ANLIS
Average	3.4598	3.1778	3.5280	3.2893	2.9682	2.7298	3.0990	3.3900
Standard deviation	0.6210	0.6557	0.4945	0.6142	0.5042	0.6200	0.5927	0.6649
T-test	1.9626		1.8907		-1.8637		2.0401	
P value	0.0532		0.0625		0.0663		0.0447	

5.10 Conclusion

In this chapter, a system for automatic extraction of imperative sentences from the web pages was described. Thousands of sentences were extracted from 200 Web pages related to the Wikipedia page of the well-known footballer David Beckham. Pr-processing and filtering was done for the extracted text to be the data set of the proposed system. NLTK-POS tagger and the type of verb tenses were used to identify and select the imperative sentences from the extracted set of sentences. The resultant sentences were saved in an SQL database to be used as part of the knowledge base of the OFC Chatbot and for the evaluation purpose. The results were evaluated using a subjective test and compared with a comparative system. The evaluation results show that our system's performance outperforms the comparative system. In addition, our system's Imperative sentence percentage to overall output data is higher than the comparative system's. Two main contributions were obtained in this chapter. First, we enriched our Chatbot OFC knowledge base by extracting more useful information from the web. The second main contribution is the automatic extraction of imperative sentences from the web using verb tense type and POS tag.

Chapter Six

Implementation of Online Feedable Chatbot

6.1 Introduction

The rapid growth in commercial conversational agents' prevalence has led to an upsurge in research in terms of natural language understanding and machine learning for conversational systems. Chatbot development has been fairly well studied since Turing proposed his Imitation Game (TIG) [33, 161]. The Chatbot idea originated with the first Chatbot named ELIZA, which was built to demonstrate natural language conversation between human and computer [40]; then ALICE was another milestone [162]. The Loebner Prize and The Chatbot Challenge are annual competitions that have their roots in TIG [6]. Conversation is a special form of interaction that follows social conventions, and the purpose of building a Chatbot system is to simulate a human conversation [6, 163]. The Chatbot architecture combines computational algorithms and a language model to emulate chat communication between a computer and a human user using natural language [6, 164].

There are many challenges in the context of a conversation and these challenges are getting bigger when people try to create conversations with machines. Some of these challenges are associated with the ways of chatting (Dialogue manager) and others are related to the information (knowledge base) [165]. Trying to work on improvements in both directions is the challenge of our work.

In this chapter, we present the implementation of our proposed Chatbot. We built the platform of the OFC in order to test the consistency in work between the Chatbot manager and the SQL database prepared and populated using the approaches explained in the previous chapters. The implemented Chatbot is evaluated using humans in an experiment. We recruited 26 participants to implement our assessment experiment as a conversational session per each participant. The evaluation results reveal reasonable accuracy and acceptability in accordance with human assessment.

6.2 Chatbot Architecture

Most of the first Chatbot toolkits' design and development in the initial period of chat implicitly consider that an utterance from the user is followed by an utterance from the Chatbot [163].

There are basically three types of architecture for building conversational systems:

1. A totally rule-oriented architecture provides a manually coded reply for each utterance as in classical examples of rule-based Chatbots, such as Eliza, and Parry [134]. Eliza can also extract a number of words from sentences to create another sentence using these words according to their syntactic functions. Eliza was a rule-based idea with no reasoning.
2. A totally data-oriented architecture, in contrast to rule-oriented architectures, depends on learning patterns from samples of dialogues to recreate the behaviour of the interaction that is observed in the data. This kind of learning can be implemented using a machine learning approach, or by extracting rules from data instead of coding them manually. Different technologies can be used to build this kind of architecture, such as classical information retrieval algorithms, Hidden Markov Models (HMM), and neural networks. Example Chatbots are: Tay, which was a Chatbot developed by Microsoft to interact with teenagers on Twitter, Xiaoice2 in China, and Rinna3 in Japan [134].
3. A mix of a rules and a data-oriented architecture as in the model of learning in the current ALICE [163] [166] , which is an incremental or/and interactive learning since the developers monitor the robot's conversations and create new AIML content to make the responses more accurate, believable, or human [163] [167].

The architecture of the Chatbot proposed in our work depends on simple rules to build the chat. The novel idea of developing the chat and expanding or updating the information depends on its ability to access the information on the web. Our Chatbot also has the ability to update the information whenever the user desires that without any human control or interference.

A general block diagram of a spoken language Chatbot architecture is shown in fig.6.1 below. The diagram shows the stages that the user's voice passes through in order to obtain the eventual response to a query. The input speech query passes through the Automatic Speech Recognition (ASR) tool in order to convert voice speech to text, which is processed using Natural Language Processing (NLP). The dialogue manager analyses the query according to the analysis system of a Chatbot and then accesses the knowledge base of the Chatbot to prepare the matching response answer depending on the query analysis. Then, the target response answer is generated and converted from text to speech to be delivered to the user.

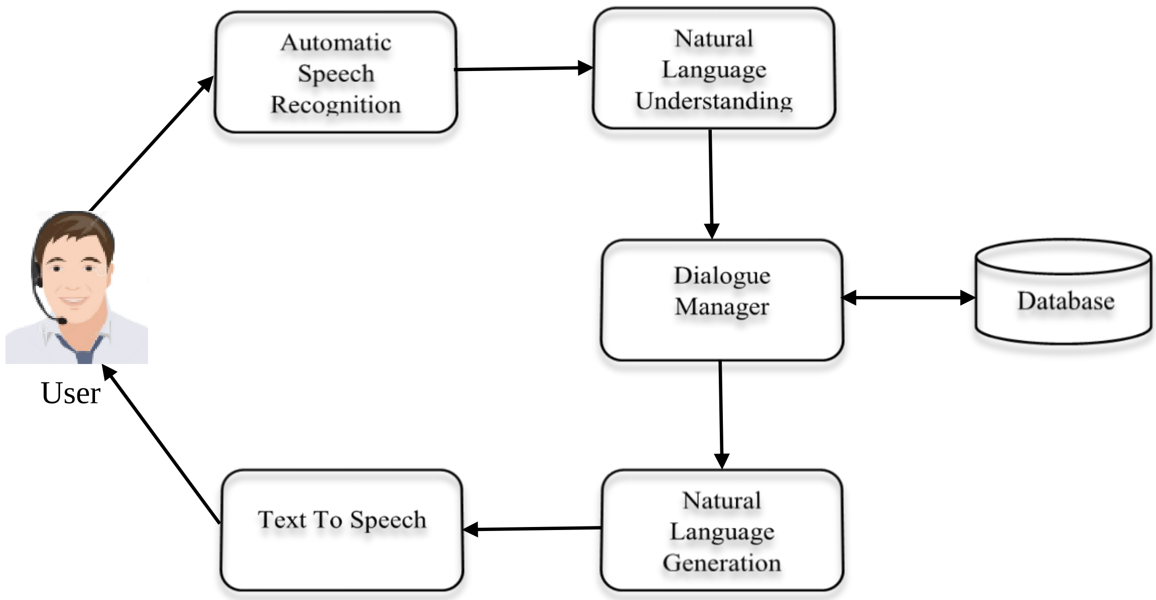


Fig. 6. 1: A general block diagram of a Chatbot architecture

6.3 Our Chatbot (OFC)

The basic idea of our Chatbot, OFC is that the Chatbot knowledge base is empty until the user decides or chooses the figure or the object they need information about. After that, the OFC starts to access the Web pages corresponding to the figure or the object chosen. The OFC uses a web crawler to extract the text in the Wikipedia page associated with the chosen figure or object and a number of pages related to that page. The OFC then uses the techniques proposed to process the extracted plain text to automatically generate QAPs from the set of sentences extracted from the plain text. The database of the OFC is populated with tens of concentrated QAPs about the desired figure of object in a few hours. Then, a Chatbot with a totally new subject is initialised. The database of this Chatbot is extendable and accumulative according to the demand to learn from the user. The core of our Chatbot architecture is the link between the internet and the Chatbot database through the dialogue manager because this is the path of feeding the Chatbot with the chatting knowledge and this is the meaning of “Online Feedable”. The OFC also contains another source of information which is the Canned Responses database that is relatively static and connected to the dialogue manager. The Canned Responses database is used by the OFC for greetings and other simple conversational queries and also when no answers are found for a query. In the future, we will extract all the information in the canned responses from the web to make the Chatbot training fully automated. The block diagram of our Chatbot is shown in fig.6.2 below.

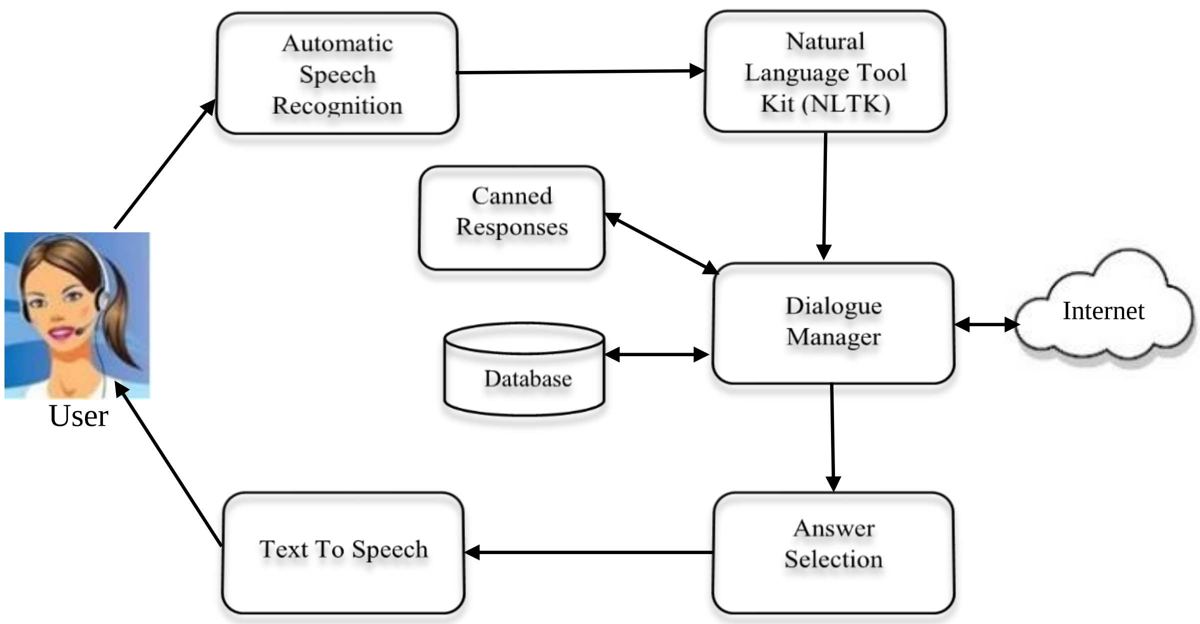


Fig. 6. 2: A block diagram of our OFC chatbot.

6.4 Implementing Our Chatbot

We implemented the first stage of our Chatbot using the Python programming language. The main parts of chatting, such as ASR, natural language understanding, dialogue management, and the answer selection stage, are coded using Python. Natural language processing for the user query is implemented using NLTK, and Microsoft Speech recogniser is used for Speech To Text (STT) and Text To Speech (TTS) operations. SQLite is used to build, update and populate the database of our Chatbot. The chatting process starts when the user pops a query through the microphone or texts a message into the Chatbot input. If the query is greetings or any conversational (not informative or pedagogical) speech, the dialogue manager goes to the canned responses database. If the query is not a greeting or conversational, the dialogue manager goes to the SQLite database to search for the response. If the dialogue manager does not find the response in the SQLite database, it responds to the query from the Canned Responses database of *No Answers*. The Chatbot keeps chatting until the user says any quitting query, such as goodbye, see you later, quit, or exit. The flow diagram of our implemented Chatbot OFC is shown in fig.6.3.

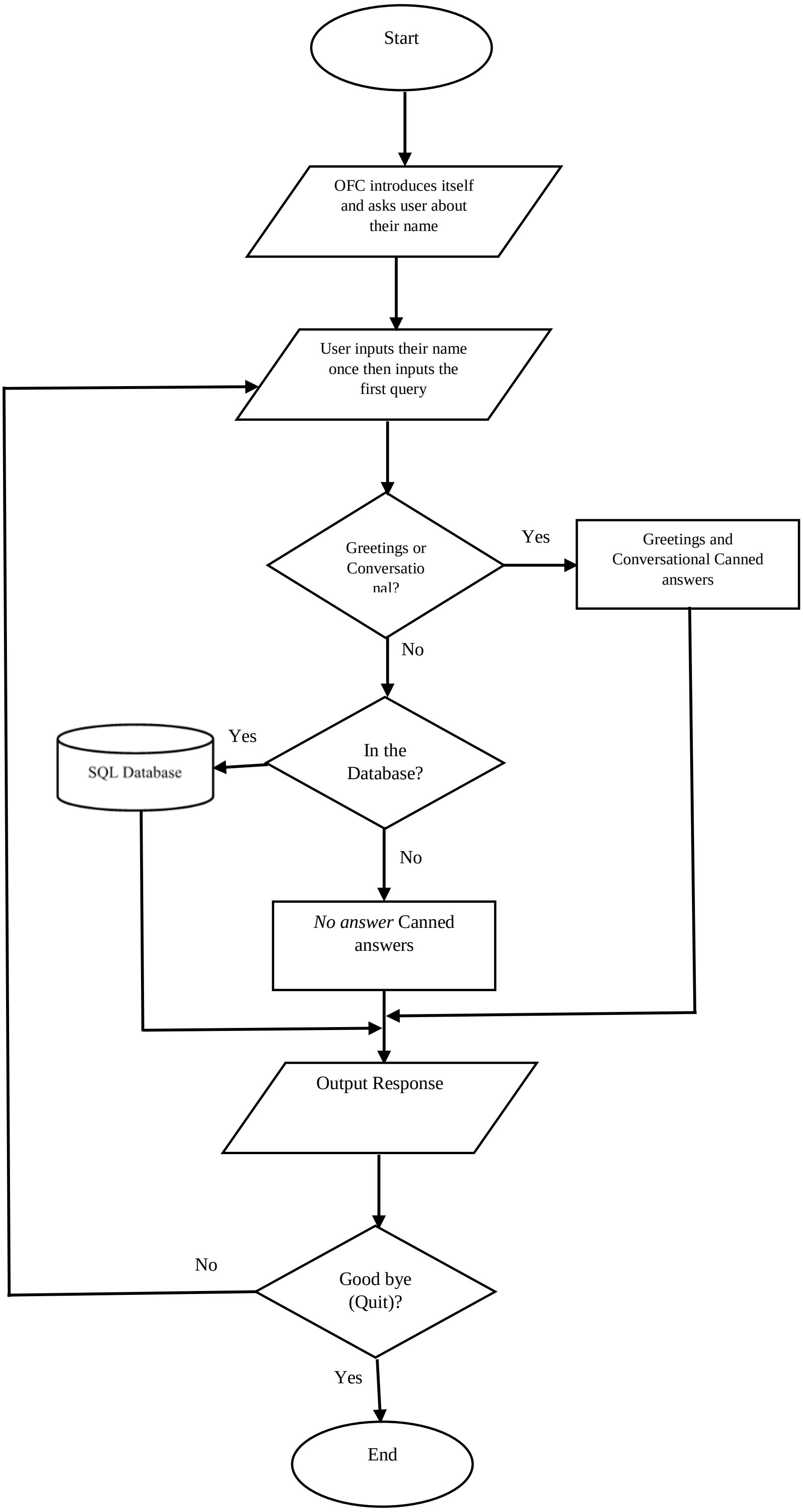


Fig. 6. 3: The flow diagram of our implemented Chatbot OFC.

An example of discourse analysis in our Chatbot OFC for speech act classes is shown in Table 6.1.

Table 6. 1: OFC Specific Speech Acts Classes.

Speech Act	**Example**
Query Greetings	Hello
Chatbot Greetings	Hi
Query Conversational	How are you doing?
Chatbot Conversational	I'm fine.
Query Definition	Who is David Beckham?
Chatbot Inform Definition	He is the first English player to win league titles in four countries England Spain the United States and France.
Query News	What did Beckham say?
Chatbot Inform News	Beckham said I am honoured and privileged to receive this recognition.
Query information	How many years did David spend playing football?
Chatbot Inform Information	David spent at least 20 years playing football.
Query information	Can you tell me the weather?
Chatbot No Answer	Sorry I didn't understand what you said
Query bye	See you later.
Chatbot bye	See you soon.

6.5 Evaluation

Our proposed Chatbot, like any other Chatbots, needs implementation in order to apply the rules that have been placed to design it. The Chatbot performance also needs evaluation after implementation. As the Chatbot's job is to hold conversations with people, the most appropriate evaluator for it could be the human. An experiment has been conducted to implement and evaluate the first stage of our proposed Chatbot. The experiment was in two stages: the first stage was to implement the Chatbot platform using the Python language, and the second stage was the evaluation stage. To evaluate our Chatbot, we used subjective assessment by using human participants. The subjective assessment was conducted in order to measure the performance of our system from the users' point of view. The number of participants was 26 and a conversational session was held with each participant in order to allow them to assess the output responses to their conversational queries. An experimental sheet with 7 queries was prepared with 4 evaluation questions directed to the participant for each query and the answer should be either yes or no. At the end of the query, there was another question of Chatbot humanness and the answer should be one of three options: yes, kind of, and no.

6.5.1 Evaluation Metrics

Evaluation metrics of a Chatbot depend on the challenges that the evaluated Chatbot is built for. The challenges that are promised to be solved by our Chatbot are: producing reasonably accurate responses from the information extracted from the internet, presenting useful educational or pedagogical information about the subject the user needs to learn, and providing acceptable conversational responses.

In order to measure the quality of each response, we tried to classify responses according to an independent human evaluation of accuracy, effect, and whether the information provided matched the question and was correct, acceptable, and useful [37] [6]. The answer to each of the evaluation class questions is either yes or no. We also measure the humanness of the Chatbot by placing a question about that at the end of the query sheet [40]. The answer to the question of humanness classification class is one of the three answers: yes, no, or kind of.

6.6 Experiment4

Experiment4 started with configuring the algorithm and the flow diagram of implementing our Chatbot OFC program. After that the code of the OFC was written in the Python programming language. The experimental sheet was prepared for the purpose of subjective evaluation. The experiment was conducted with a set of participants who are mostly students in the University of Essex. The experimental evaluation data were collected by doing the following.

1. Meeting each participant and asking them about the possibility of participating in our experiment.
2. Meeting each participant again in a lab to do the experiment.
3. Handing each participant the experimental sheet and explaining to them how to evaluate the chatting process.
4. Starting the conversational session for the participants one by one and the filling in of the experimental sheet by the participant.
5. Ending the conversational session and collecting the sheet from the participant.
6. Calculating the aggregate scores given by the participants after completing all the assigned sessions.

6.6.1 Aim of Experiment

The aim of this experiment was to implement the first stage and evaluate the performance of our proposed OFC and the quality of its responses. The evaluation was made in the users' point of view and the aggregate scores of the users' assessment were calculated in percentage values.

6.6.2 Experiment Participants

The experiment was conducted with students from the University of Essex. These participants are undergraduate and postgraduate students in different disciplines.

The study included 26 participants of both genders (male and female). The participants were around 47% native speakers and the rest (53%) were non-native. Each participant did the experiment by holding a conversational session with the OFC and then assessing the response answers according to an experimental sheet prepared for this purpose.

6.6.3 Experiment Steps

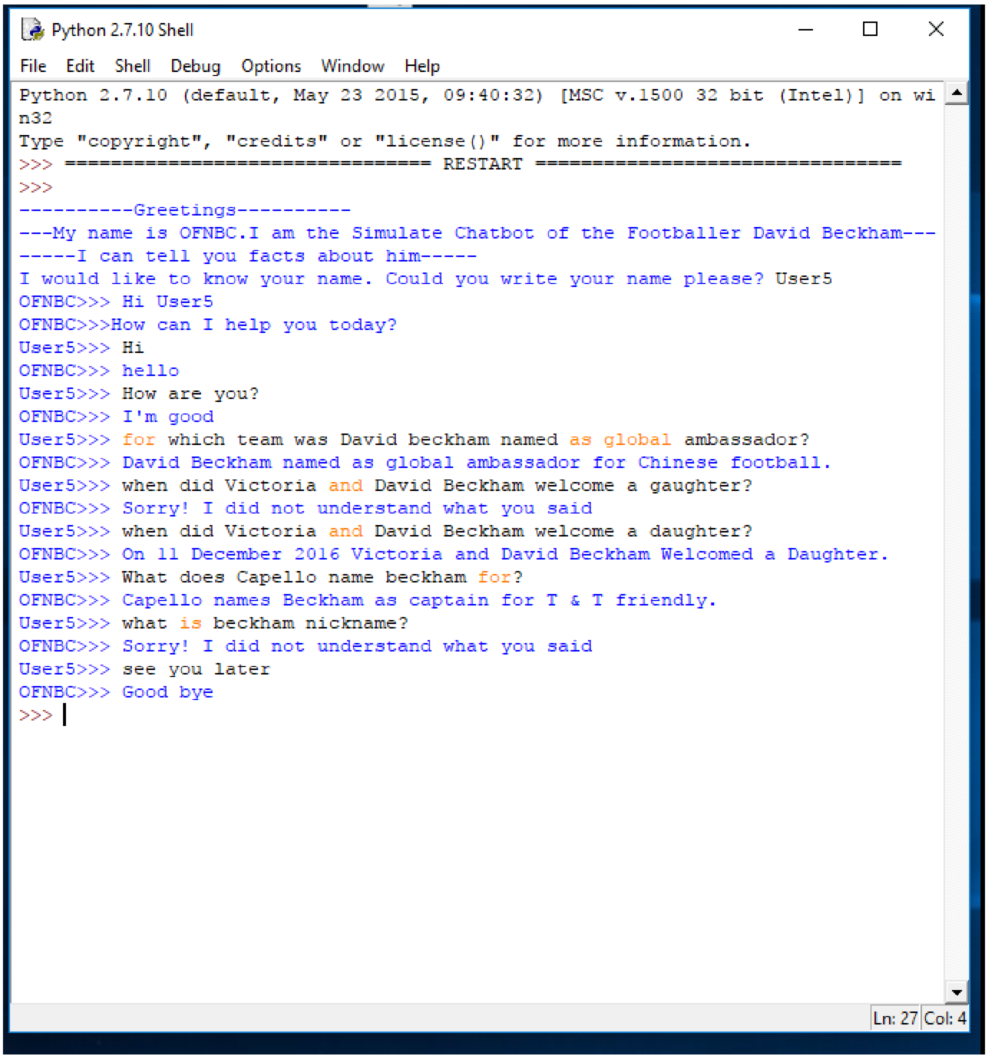
The experiment was run through the following steps:

1. The Python programming language was used to write the code of our proposed Chatbot OFC. The program runs to do the following activities.
    - Posting a greeting at the beginning and then introducing itself as a Chatbot for answering questions about the footballer David Beckham and as a tutor Chatbot, it can tell you facts about the person it gives information about.
    - Asking the user their name, and when the user gives their name, OFC greets the user by their name.
    - Offering help by asking how it can help the user.
    - Replying to greetings.
    - Answering the questions to which it has the answers.
    - Apologising if the answer is not available.
    - Quitting when the user wants; otherwise it continues chatting.
2. Preparing the experimental sheet for the evaluation stage. The experimental sheet includes 7 user queries, 6 of them are put by the authors and one by the user. The user is requested to put a question or more on the experimental sheet. The evaluation fields include evaluation metrics from the users' point of view in terms of matching the question, correctness, acceptability, and usefulness of the Chatbot responses. The users assess either by replying yes or no to each of these four attributes. A fifth evaluation attribute was also written at the bottom of the experimental sheet which is about the assessed Chatbots' humanness, and the answer should be one of the options: yes, no, and kind of.
3. The experimental participants were selected and invited to do an experiment for the purpose of evaluation. Twenty six undergraduate and postgraduate students from the University of Essex were recruited as experimental participants.
4. The experiment was carried out by holding conversational sessions between the participants and the Chatbot. The participants used the experimental sheets to ask the OFC queries and assess the Chatbot responses.

5. The experimental conversational sessions were completed and the experimental sheets were collected.
6. The evaluation scores were calculated manually by aggregating the scores recorded in all the experimental sheets and then the results were tabulated and shown in the experimental results section.

6.6.4 Experimental Results

Samples of experimental conversational sessions are shown in fig.6.4 and fig.6.5. These samples are selected randomly from the snap pictures taken for the conversational sessions on the Python shell during the experiment run. The picture of Fig.6.4 shows the conversational session of the OFC with User5, which was the name given on the experimental sheet to the participant.



```
Python 2.7.10 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 2.7.10 (default, May 23 2015, 09:40:32) [MSC v.1500 32 bit (Intel)] on wi
n32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
----------Greetings----------
---My name is OFNBC.I am the Simulate Chatbot of the Footballer David Beckham---
-----I can tell you facts about him-----
I would like to know your name. Could you write your name please? User5
OFNBC>>> Hi User5
OFNBC>>>How can I help you today?
User5>>> Hi
OFNBC>>> hello
User5>>> How are you?
OFNBC>>> I'm good
User5>>> for which team was David beckham named as global ambassador?
OFNBC>>> David Beckham named as global ambassador for Chinese football.
User5>>> when did Victoria and David Beckham welcome a gaughter?
OFNBC>>> Sorry! I did not understand what you said
User5>>> when did Victoria and David Beckham welcome a daughter?
OFNBC>>> On 11 December 2016 Victoria and David Beckham Welcomed a Daughter.
User5>>> What does Capello name beckham for?
OFNBC>>> Capello names Beckham as captain for T & T friendly.
User5>>> what is beckham nickname?
OFNBC>>> Sorry! I did not understand what you said
User5>>> see you later
OFNBC>>> Good bye
>>> |
Ln: 27 Col: 4
```

Fig. 6. 4: A user session in our experiment on Python 2.7.10 execution shell.

The picture shown in fig.6.5 shows the conversational session on Python shell between User23 and the OFC.

```
>>> ================================ RESTART ================================
>>>
----------Greetings----------
---My name is OFNBC.I am the Simulate Chatbot of the Footballer David Beckham---
-----I can tell you facts about him-----
I would like to know your name. Could you write your name please? User23
OFNBC>>> Hi User23
OFNBC>>>How can I help you today?
User23>>> Greetings
OFNBC>>> hey
User23>>> How is it going on?
OFNBC>>> I'm good
User23>>> What did Beckham make?
OFNBC>>> Beckham made a short speech revealing it was a dream to join the Spanis
h giants.
User23>>> What did David say?
OFNBC>>> David is looking forward to going to the States Oliveira said.
User23>>> What did Brooklyn Beckham set?
OFNBC>>> Brooklyn Beckham set to be released by Arsenal already.
User23>>> Who is David Beckham wife?
OFNBC>>> Sorry! I did not understand what you said
User23>>> see you later
OFNBC>>> See you later
>>> |
```

Fig. 6. 5: A sample of an experimental user session for one participant.

The experimental queries were categorised into five categories. Each category included a number of queries distributed all over the experimental sheets. The query category and the number of queries for each category are shown in Table 6.2 below.

Table 6. 2: Query categories used in the experimental sheet.

Query Category	**Number of Queries**
Greetings	26
Friendly conversational	26
Informative, Definition, and Pedagogical	104
Quit	26

In the evaluation results calculation, we merged the categories Greetings, Friendly conversational queries, and Quit queries under Conversational queries and Informative, Definition, and Pedagogical queries under Informative and Pedagogical. After collecting the subjective assessment results of the experiment, they are tabulated in table 6.3 below.

Table 6. 3: Experimental evaluation results according to human assessment.

Category	Quality Attribute	Metric	Conversational	Informative and Pedagogical
Accuracy	Response answer match the query.	% of Queries Success.	75%	64%
	Response answer correct.	% of Queries Success.	75%	51%
Affect	Response Acceptable.	% of Queries Success.	74.7%	67%
Information Provided	Response answer Useful.	% of Queries Success.	32%	49%
Humanness	Does it respond like human?	% of users who classify.	88%	

The evaluation results in Table 6.3 show that our OFC has an overall 69.5% match between queries and answers, overall 73.5% correct response answers, overall 70.85% acceptable responses, overall 40.5% useful response answers, and 88% humanness, all according to user assessment.

6.7 Conclusion

In this chapter, we presented the implementation of our proposed OFC. We built the platform of our Chatbot in order to test the accuracy, acceptability, and usefulness of our Chatbot’s responses to queries. The humanness of the proposed Chatbot was also tested. The implemented Chatbot was evaluated using a subjective assessment experiment. We used 26 participants to implement our assessment experiment as a conversational session per each participant. The evaluation results show reasonable accuracy, and acceptability and 88% of humanness according to human assessment.

Chapter Seven

Conclusions

7.1 Summary

This thesis focuses on the design and implementation of a tutor Chatbot that has information of an entity that it can answer questions about. This Chatbot is capable of retrieving information about the entity from the web to populate its SQL database. Our Online Feedable Chatbot (OFC) can hold a conversation with a user and answer questions about the information it extracted from the web. Our Chatbot can update or amend its accumulative database without any interference from any instructor or administrator. New approaches are presented to generate questions and to extract imperative sentences from sentences that are extracted from Wikipedia and filtered from the HTML code and redundant information. Named entity and verb tense type are used to carefully select the factual sentences and then generate the QAPs using the targeted sentences. Moreover, verb tense and POS tag techniques are used to select and extract imperative sentences from the same dataset as the QAPs generated. A type of QA system was developed to find the best response for a Chatbot query among a set of sentences using hybrid term, syntactic, and semantic extracted features. Jaccard's coefficient and cosine similarity are used for term features, POS tags are used for syntactic ones, and named entity and semantic cosine similarity are used for semantic features extraction. The sentences dataset is acquired from the same source as for the QAP and the imperative sentences. This response search method is planned to be used as the search model used by our Chatbot's dialogue manager in order to find the best answers for a query in its database or from online. Comparative systems were adapted to our systems and datasets in order to be compared with our implemented systems. The QG and imperative sentence extraction systems were evaluated using human assessment and compared with the adapted comparative systems. The QA system was also evaluated and compared to a comparative system adapted to our system for this purpose. Our OFC invented QA dataset and Stanford QA data set were used to test our QA system and to produce the output data for evaluation purposes.

The proposed Chatbot was implemented and assessed by the users using a subjective assessment.

The experimental results show that our designed Chatbot OFC can hold a simple conversation with a user and answer the users' questions on the figure or the object it contains information about. In addition, the OFC can populate its database from the Wikipedia and it can update its database every time a user request without any interference from any instructor or controller. Our Chatbot populates its database by generating QAPs from the plain text it extracts from Wikipedia. Moreover, the OFC extracts imperative sentences from the same retrieved text from Wikipedia. The QA system was not implemented with the OFC and left for future work. The results of all the implemented systems are illustrated and show that our systems outperform the comparative systems in all the modelled proposals we presented in chapters 3, 4, and 5.

7.2 Limitations in This Work

One of the limitations in this work is using human assessment, as it depends on the individual's opinion which is related to different circumstances and variables like the participant's mood. That could affect the human's decision and consequently the assessment results.

Using Wikipedia pages as the only source of information from the web is a weakness as well. Moreover, the use of simple heuristics for generating questions constrains the quality and the quantity of the generated questions. The narrow types of questions which depend only on the subject and verb tense also limit the work outputs of chapter 3.

Using only a few number of data sets to evaluate our implemented systems is another limitation in this work since it gives results restricted to these data sets.

Another issue is in the software and the programming languages used in the implementation part that could cause errors in output results (as explained in section 3.5 Source of Error).

7.3 Future Work

Design and implementation of our Chatbot needs a huge amount of work; from building and populating the knowledge base through the process of developing appropriate methods for this purpose; finding the best way to select a response to a query; implementing the Chatbot and using humans to evaluate the system and then using the feedback of the users to enhance the system. The first stage of our Chatbot has been implemented and evaluated. In this section the next stages of our Chatbot design and improvement are explained as follows:

Online Feedable Chatbot. In the next stage of our Chatbot, we plan to add the following enhancements:

1. Using the QA system presented in Chapter 4 as an answer selection method in our Chatbot's dialogue manager. We also plan to use the same response search system to search for responses online after reducing the processing time.
2. Expanding the QAPs database of our Chatbot via adding more categorising to the database by dividing the information into genres of subjects about the entity or entities it answers questions about.
3. Converting the tenses in the QAPs from third person to first person in order to satisfy the simulation proposal in this thesis.
4. Increasing the OFC's chatting abilities by automatically extracting more conversational sentences from the web to expand its conversational database and by adding more methods to make the dialogue more natural.
5. Expanding the conversational database by extracting sentiment sentences from the web which could help in adding more naturalism to the conversations.

Automatic Question Generation System.

1. New methods for factual question generation from a sentence are planned to use in order to expand the QAPs database. Object related factual, subject

related are extracted in this work, questions are intended to be generated using novel methods.

2. Generating new types of questions like interpretive or evaluative questions to support the tutoring idea.

QA System. We are intending to enhance the search strategy of our proposed QA system by trying more feature extraction methods. Variance feature extraction approaches such as Euclidean, or Manhattan distance features may be used in the future to extract term features.

Imperative Sentence Extraction.

1. The extracted imperative sentences are planned to be used in the future to generate actionable activities that help the user to access and use applications slightly more easily by using some conversational utterances such as commands as the conversation is carrying on.
2. We have an idea to add more actions to Chatbot activities by using normal conversation to extract indirect imperative words; then to use these words to control applications and software on the computer. This could help by adding more actions to a Chatbot's work rather than using specific commands to perform specific tasks.

Bibliography

[1] I. V. Serban, C. Sankar, M. Germain, S. Zhang, Z. Lin, S. Subramanian*, et al.*, "A Deep Reinforcement Learning Chatbot," *arXiv preprint arXiv:1709.02349,* 2017.

[2] B. Hettige and A. S. Karunananda, "First Sinhala chatbot in action," *Proceedings of the 3rd Annual Sessions of Sri Lanka Association for Artificial Intelligence (SLAAI), University of Moratuwa,* 2006.

[3] L. Benotti, M. C. Martínez, and F. Schapachnik, "Engaging high school students using chatbots," in *Proceedings of the 2014 conference on Innovation & technology in computer science education*, 2014, pp. 63-68.

[4] C. Xing, W. Wu, Y. Wu, J. Liu, Y. Huang, M. Zhou*, et al.*, "Topic Aware Neural Response Generation," in *AAAI*, 2017, pp. 3351-3357.

[5] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, "Gated self-matching networks for reading comprehension and question answering," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 189-198.

[6] S. Ghose and J. J. Barua, "Toward the implementation of a topic specific dialogue based natural language chatbot as an undergraduate advisor," in *2013 International Conference on Informatics, Electronics & Vision (ICIEV)*, 2013, pp. 1-5.

[7] I. Papaioannou and O. Lemon, "Combining Chat and Task-Based Multimodal Dialogue for More Engaging HRI: A Scalable Method Using Reinforcement Learning," in *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 2017, pp. 365-366.

[8] A. S. Lokman and J. M. Zain, "Extension and prerequisite: An algorithm to enable relations between responses in chatbot technology," *Journal of Computer Science,* vol. 6, p. 1212, 2010.

[9] M. Dahiya, "A Tool of Conversation: Chatbot," *International Journal of Computer Sciences and Engineering,* 2017.

[10] I. V. Serban, R. Lowe, P. Henderson, L. Charlin, and J. Pineau, "A survey of available corpora for building data-driven dialogue systems," *arXiv preprint rXiv:1512.05742,* 2015.

[11] T. Hiraoka, G. Neubig, K. Yoshino, T. Toda, and S. Nakamura, "Active learning for example-based dialog systems," in *Dialogues with Social Robots*, ed: Springer, 2017, pp. 67-78.

[12] G. Skantze, "Exploring human error recovery strategies: Implications for spoken dialogue systems," *Speech Communication,* vol. 45, pp. 325-341, 2005.

[13] D. L. F. Popowich and J. N. P. Winne, "Generating Natural Language Questions to Support Learning On-Line," *ENLG 2013,* p. 105, 2013.

[14] D. M. Christopher, R. Prabhakar, and S. Hinrich, "Introduction to information retrieval," *An Introduction To Information Retrieval,* vol. 151, p. 177, 2008.

[15] E. Haller and T. Rebedea, "Designing a Chat-bot that Simulates an Historical Figure," in *19th International Conference on Control Systems and Computer Science (CSCS)* 2013, pp. 582-589.

[16] S. A. Abdul-Kader, and John Woods, "Automatic Web-based Question Answer Generation System for Online Feedable New-Born Chatbot," *IEEE Computing Conference, London, UK,* 2018.

[17] S. A. Abdul-Kader, and John Woods, "Question Answer System for an Online Feedable New Born Chatbot " *Intelligent Systems Conference, London, UK,* 2017.

[18] S. A. Abdul-Kader and J. Woods, "Survey on Chatbot Design Techniques in Speech Conversation Systems," *IJACSA,* vol. 6, pp. 72-80, July 2015.
[19] C. I. Nass and S. Brave, *Wired for speech: How voice activates and advances the human-computer relationship*: MIT Press Cambridge, 2005.
[20] Y.-P. Yang, "An Innovative Distributed Speech Recognition Platform for Portable, Personalized and Humanized Wireless Devices," *Computational Linguistics and Chinese Language Processing,* vol. 9, pp. 77-94, 2004.
[21] J. Lazar, J. H. Feng, and H. Hochheiser, *Research methods in human-computer interaction*: Morgan Kaufmann, 2017.
[22] J. P. Campbell Jr, "Speaker recognition: a tutorial," *Proceedings of the IEEE,* vol. 85, pp. 1437-1462, 1997.
[23] C.-H. Lee, "From knowledge-ignorant to knowledge-rich modeling: a new speech research paradigm for next generation automatic speech recognition," in *Proc. ICSLP*, 2004.
[24] S. O. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang*, et al.*, "Deep Voice: Real-time neural text-to-speech," *arXiv preprint arXiv:1702.07825,* 2017.
[25] V. Bhargava and N. Maheshwari, "An Intelligent Speech Recognition System for Education System," *nikhilmaheshwari.com,* 2009.
[26] L. Guerrouj, M. Di Penta, G. Antoniol, and Y. G. Guéhéneuc, "Tidier: an identifier splitting approach using speech recognition techniques," *Journal of Software: Evolution and Process,* vol. 25, pp. 575-599, 2013.
[27] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech & Language,* vol. 46, pp. 535-557, 2017.
[28] P. Roy and P. J. Lagassey, "Method for processing the output of a speech recognizer," ed: Google Patents, 2017.
[29] A. M. Galvao, F. A. Barros, A. M. Neves, and G. L. Ramalho, "Persona-aiml: An architecture developing chatterbots with personality," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, 2004, pp. 1266-1267.
[30] J. Ratkiewicz, "Evolutionary Sentence Combination for Chatterbots Dana Vrajitoru " *Computer and Information Sciences Indiana University South Bend, 1700 Mishawaka Ave,* 2004.
[31] M. J. Pereira and L. Coheur, "Just. Chat-a platform for processing information to be used in chatbots," *[Online] Available : https://fenix.tecnico.ulisboa.pt/downloadFile/395145485809/ExtendedAbstract.pdf ,* 2013.
[32] D. J. Stoner, L. Ford, and M. Ricci, "Simulating Military Radio Communications Using Speech Recognition and Chat-Bot Technology," *The Titan Corporation, Orlando,* 2004.
[33] A. M. Turing, "Computing machinery and intelligence," *Mind,* pp. 433-460, 1950.
[34] B. Kirkpatrick and B. Klingner, "Turing's Imitation Game: a discussion with the benefit of hind-sight," *Berkeley Computer Science course. See http://www. cs. berkeley. edu/~ christos/classics/ttest. pdf Accessed,* vol. 1, p. 13, 2009.
[35] S. M. Shieber, "Inverting the Turing Test [review of The Most Human Human by Brian Christian," *American Scientist* vol. (99) 6., 2011.
[36] P. Hingston, "A turing test for computer game bots," *IEEE Transactions on Computational Intelligence and AI in Games,* vol. 1, pp. 169-186, 2009.

[37] B. A. Shawar and E. Atwell, "Different measurements metrics to evaluate a chatbot system," in *Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, 2007, pp. 89-96.
[38] M. McTear, Z. Callejas, and D. Griol, "Conversational Interfaces: Past and Present," in *The Conversational Interface*, ed: Springer, 2016, pp. 51-72.
[39] I. O'Neill, P. Hanna, X. Liu, D. Greer, and M. McTear, "Implementing advanced spoken dialogue management in Java," *Science of Computer Programming,* vol. 54, pp. 99-124, 2005.
[40] N. M. Radziwill and M. C. Benton, "Evaluating Quality of Chatbots and Intelligent Conversational Agents," *arXiv preprint arXiv:1704.04579,* 2017.
[41] K. Chandrasekaran, "A Survey of Design Techniques for Conversational Agents," in *Information, Communication and Computing Technology: Second International Conference, ICICCT 2017, New Delhi, India, May 13, 2017, Revised Selected Papers*, 2017, p. 336.
[42] K. Meffert, "Supporting design patterns with annotations," in *13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems ECBS* 2006, pp. 8-445.
[43] M. L. McNeal and D. Newyear, "Chatbot creation options," *Library Technology Reports,* vol. 49, pp. 11-17, 2013.
[44] D. Mladenić and L. Bradeško, "A survey of chabot system through a Loebner prize competition," *Proceedings of Slovenian Artificial Intelligence Laboratory, Jozef Stefan Institute in Ljubljana,* 2012.
[45] W. Kim, "On optimizing an SQL-like nested query," *ACM Transactions on Database Systems (TODS),* vol. 7, pp. 443-469, 1982.
[46] D. J. Mocek, K. Li, and J. M. Levine, "Natural language translation of an SQL query," ed: Google Patents, 1999.
[47] N. Chinchor and P. Robinson, "MUC-7 named entity task definition," in *Proceedings of the 7th Conference on Message Understanding*, 1997, p. 29.
[48] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes,* vol. 30, pp. 3-26, 2007.
[49] A. Peñas, A. Rodrigo, V. Sama, and F. Verdejo, "Testing the reasoning for question answering validation," *Journal of Logic and Computation,* vol. 18, pp. 459-474, 2008.
[50] O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, C. Jacquemin, L. Monceaux*, et al.*, "How NLP can improve question answering," *Knowledge Organization,* vol. 29, pp. 135-155, 2002.
[51] H. Yu and V. Hatzivassiloglou, "Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 2003, pp. 129-136.
[52] "Types of Questions," *[Online] Available : http://bowmanatbrooks.weebly.com/uploads/8/3/8/3/8383240/levelsofquestions.doc - literal - interpretive - universal.pdf* 2017.
[53] C. Perrault, "Short Kid Stories," *[Online] Available : http://www.shortkidstories.com/story/cinderella-2/*, 2017.
[54] J. Sarma, "The acquisition of wh-questions in English," *Doctoral Dissertations. AAI9215447. ,* 1991.

[55] R. Hawkins and H. Hattori, "Interpretation of English multiple wh-questions by Japanese speakers: A missing uninterpretable feature account," *Second Language Research,* vol. 22, pp. 269-301, 2006.
[56] T. Brants, "TnT: a statistical part-of-speech tagger," in *Proceedings of the sixth conference on Applied natural language processing*, 2000, pp. 224-231.
[57] E. Brill, "A simple rule-based part of speech tagger," in *Proceedings of the third conference on Applied natural language processing*, 1992, pp. 152-155.
[58] H. Schmid, "Probabilistic part-of-speech tagging using decision trees," in *New methods in language processing*, 2013, p. 154.
[59] T. Cohn and P. Blunsom, "Semantic role labelling with tree conditional random fields," in *Proceedings of the Ninth Conference on Computational Natural Language Learning*, 2005, pp. 169-172.
[60] S. P. Ponzetto and M. Strube, "Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution," in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 2006, pp. 192-199.
[61] R. Lebret and R. Collobert, "Word emdeddings through hellinger PCA," *arXiv preprint arXiv:1312.5542,* 2013.
[62] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111-3119.
[63] M. A. Qureshi and D. Greene, "EVE: explainable vector based embedding technique using Wikipedia," *Journal of Intelligent Information Systems,* pp. 1-29, 2017.
[64] R. Socher, J. Bauer, and C. D. Manning, "Parsing with compositional vector grammars," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2013, pp. 455-465.
[65] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of research and development,* vol. 3, pp. 210-229, 1959.
[66] H. Mannila, "Data mining: machine learning, statistics, and databases," in *ssdbm*, 1996, p. 2.
[67] D. Freitag, "Machine learning for information extraction in informal domains," *Machine learning,* vol. 39, pp. 169-202, 2000.
[68] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering,* vol. 160, pp. 3-24, 2007.
[69] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *Journal of machine learning research,* vol. 5, pp. 845-889, 2004.
[70] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing,* vol. 70, pp. 489-501, 2006.
[71] D. Klerfors and T. L. Huston, "Artificial neural networks," *Sant Louis University School of Business & Administration,* 1998.
[72] I. A. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals, computing, design, and application," *Journal of microbiological methods,* vol. 43, pp. 3-31, 2000.
[73] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research," *Journal of pharmaceutical and biomedical analysis,* vol. 22, pp. 717-727, 2000.
[74] J. Huang, M. Zhou, and D. Yang, "Extracting Chatbot Knowledge from Online Discussion Forums," in *IJCAI*, 2007, pp. 423-428.

[75] Y. Wu, G. Wang, W. Li, and Z. Li, "Automatic chatbot knowledge acquisition from online forum via rough set and ensemble learning," in *Network and Parallel Computing, 2008. NPC 2008. IFIP International Conference on*, 2008, pp. 242-246.
[76] N.-T. Le, N.-P. Nguyen, K. Seta, and N. Pinkwart, "Automatic question generation for supporting argumentation," *Vietnam Journal of Computer Science,* vol. 1, pp. 117-127, 2014.
[77] B. AbuShawar and E. Atwell, "Automatic Extraction of Chatbot Training Data from Natural Dialogue Corpora," in *RE-WOCHAT: Workshop on Collecting and Generating Resources for Chatbots and Conversational Agents-Development and Evaluation Workshop Programme (May 28 th, 2016)*, p. 29.
[78] J. Jia, "CSIEC: A computer assisted English learning chatbot based on textual knowledge and reasoning," *Knowledge-Based Systems,* vol. 22, pp. 249-255, 2009.
[79] J. Nakanishi, Y. Sakatani, M. Okubo, S. Fujii, K. Matsuo, T. Yamada*, et al.*, "Sense of Self-Agency in Sharing Conversational Experiences between Agents and Human Users," in *IEEE Tenth International Conference on Semantic Computing (ICSC)*, 2016, pp. 270-271.
[80] B. Abu Shawar and E. Atwell, "Chatbots: can they serve as natural language interfaces to QA corpus?," in *Proceeding (689) Advances in Computer Science and Engineering/690: Internet and Multimedia Systems and Applications-2010*, 2010.
[81] Z. Yu, Z. Xu, A. W. Black, and A. Rudnicky, "Chatbot evaluation and database expansion via crowdsourcing," in *Proceedings of the RE-WOCHAT workshop of LREC, Portoroz, Slovenia*, 2016.
[82] Y. Kim and D. Chung, "Method and apparatus for information delivery on the internet," ed: Google Patents, 2001.
[83] R. Rzepka, W. Shi, M. Ptaszynski, P. Dybala, S. Higuchi, and K. Araki, "Serious processing for frivolous purpose: a chatbot using web-mining supported affect analysis and pun generation," in *Proceedings of the 14th international conference on Intelligent user interfaces*, 2009, pp. 487-488.
[84] A. Kerly, P. Hall, and S. Bull, "Bringing chatbots into education: Towards natural language negotiation of open learner models," *Knowledge-Based Systems,* vol. 20, pp. 177-185, 2007.
[85] R. Hoffmann, S. Kowalski, R. Jain, and M. Mumtaz, "E_Universities Services in the New Social Eco-systems: Security Risk Analysis: Using Conversational Agents to Help Teach Information Security Risk Analysis," *International Conference on Social Psychology and Language, 1st. University of Bristol,* vol. , 2011.
[86] Y. F. Wang and S. Petrina, "Using learning analytics to understand the design of an intelligent language tutor–Chatbot lucy," *Editorial Preface,* vol. 4, 2013.
[87] D. Griol and Z. Callejas, "An architecture to develop multimodal educative applications with chatbots," *International Journal of Advanced Robotic Systems,* vol. 10, 2013.
[88] C. G. Rospide, C. Puente, and J. A. Olivas, "Improving Educational Methods by Means of an Intelligent Virtual Agent," in *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, 2013, p. 1.
[89] M. Niranjan, M. Saipreethy, and T. G. Kumar, "An intelligent question answering conversational agent using Naïve Bayesian classifier," in *IEEE International Conference onTechnology Enhanced Education (ICTEE)*, 2012, pp. 1-5.
[90] S. L. Lim and O. S. Goh, "Intelligent Conversational Bot for Massive Online Open Courses (MOOCs)," *arXiv preprint arXiv:1601.07065,* 2016.

[91] J. Jia, "CSIEC (Computer simulator in educational communication): an intelligent web-based teaching system for foreign language learning," *arXiv preprint cs/0312030,* 2003.
[92] S. Crown, A. Fuentes, R. Jones, R. Nambiar, and D. Crown, "Anne G. Neering: Interactive Chatbot to Motivate and Engage Engineering Students," in *American Society for Engineering Education*, 2010.
[93] C. A. Bentivoglio, D. Bonura, V. Cannella, S. Carletti, A. Pipitone, R. Pirrone*, et al.*, "Intelligent Agents supporting user interactions within self regulated learning processes," *Journal of e-Learning and Knowledge Society,* vol. 6, 2010.
[94] A. Moschitti and S. Quarteroni, "Linguistic kernels for answer re-ranking in question answering systems," *Information Processing & Management,* vol. 47, pp. 825-842, 2011.
[95] A. C. Mendes and L. Coheur, "When the answer comes into question in question-answering: survey and open issues," *Natural Language Engineering,* vol. 19, pp. 1-32, 2013.
[96] S. Abney, M. Collins, and A. Singhal, "Answer extraction," in *Proceedings of the sixth conference on Applied natural language processing*, 2000, pp. 296-301.
[97] M. Surdeanu, M. Ciaramita, and H. Zaragoza, "Learning to rank answers to non-factoid questions from web collections," *Computational linguistics,* vol. 37, pp. 351-383, 2011.
[98] N. P. Er and I. Cicekli, "A Factoid Question Answering System Using Answer Pattern Matching," in *IJCNLP*, 2013, pp. 854-858.
[99] N. Kambhatla, "Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations," in *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, 2004, p. 22.
[100] S. Verberne, H. van Halteren, D. Theijssen, S. Raaijmakers, and L. Boves, "Learning to rank for why-question answering," *Information Retrieval,* vol. 14, pp. 107-132, 2011.
[101] P. Molino, P. Basile, A. Caputo, P. Lops, and G. Semeraro, "Distributional Semantics for Answer Re-ranking in Question Answering," in *IIR*, 2013, pp. 100-103.
[102] D. Radev, W. Fan, H. Qi, H. Wu, and A. Grewal, "Probabilistic question answering on the web," *Journal of the Association for Information Science and Technology,* vol. 56, pp. 571-583, 2005.
[103] Y. Yang, W.-t. Yih, and C. Meek, "WikiQA: A Challenge Dataset for Open-Domain Question Answering," in *EMNLP*, 2015, pp. 2013-2018.
[104] S. Wang and J. Jiang, "Machine comprehension using match-lstm and answer pointer," *arXiv preprint arXiv:1608.07905,* 2016.
[105] M. A. Khalid, V. Jijkoun, and M. De Rijke, "The impact of named entity normalization on information retrieval for question answering," in *European Conference on Information Retrieval*, 2008, pp. 705-710.
[106] J. Lin, A. Fernandes, B. Katz, G. Marton, and S. Tellex, "Extracting answers from the web using knowledge annotation and knowledge mining techniques," DTIC Document2006.
[107] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun, "Finding question-answer pairs from online forums," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 467-474.
[108] F. Wu and D. S. Weld, "Open information extraction using Wikipedia," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 2010, pp. 118-127.

[109] Y. Matsuyama, A. Saito, S. Fujie, and T. Kobayashi, "Automatic expressive opinion sentence generation for enjoyable conversational systems," *IEEE/ACM Transactions onAudio, Speech, and Language Processing,* vol. 23, pp. 313-326, 2015.

[110] D. Ravichandran and E. Hovy, "Learning surface text patterns for a question answering system," in *Proceedings of the 40th annual meeting on association for computational linguistics*, 2002, pp. 41-47.

[111] S. Varges, F. Weng, and H. Pon-Barry, "Interactive question answering and constraint relaxation in spoken dialogue systems," *Natural Language Engineering,* vol. 15, pp. 9-30, 2009.

[112] T. Mihaylov and P. Nakov, "SemanticZ at SemEval-2016 Task 3: Ranking Relevant Answers in Community Question Answering Using Semantic Similarity Based on Fine-tuned Word Embeddings," in *SemEval@ NAACL-HLT*, 2016, pp. 879-886.

[113] P. Nakov, D. Hoogeveen, L. Màrquez, A. Moschitti, H. Mubarak, T. Baldwin*, et al.*, "SemEval-2017 task 3: Community question answering," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 27-48.

[114] S. Quarteroni and S. Manandhar, "Designing an interactive open-domain question answering system," *Natural Language Engineering,* vol. 15, pp. 73-95, 2009.

[115] H. Ali, Y. Chali, and S. A. Hasan, "Automation of question generation from sentences," in *Proceedings of QG2010: The Third Workshop on Question Generation*, 2010, pp. 58-67.

[116] D. L. Lindberg, "Automatic question generation from text for self-directed learning," Applied Sciences: School of Computing Science, 2013.

[117] N. Afzal, V. Pekar, and G. C. St, "Unsupervised Relation Extraction for Automatic Generation of Multiple-Choice Questions," in *RANLP*, 2009, pp. 1-5.

[118] R. Mitkov, L. A. Ha, A. Varga, and L. Rello, "Semantic similarity of distractors in multiple-choice tests: extrinsic evaluation," in *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, 2009, pp. 49-56.

[119] R. Mitkov, L. A. Ha, and N. Karamanis, "A computer-aided environment for generating multiple-choice test items," *Natural Language Engineering,* vol. 12, pp. 177-194, 2006.

[120] R. Mitkov and L. A. Ha, "Computer-aided generation of multiple-choice tests," in *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing-Volume 2*, 2003, pp. 17-22.

[121] A. Papasalouros, K. Kanaris, and K. Kotis, "Automatic Generation Of Multiple Choice Questions From Domain Ontologies," in *e-Learning*, 2008, pp. 427-434.

[122] Y. HUANG and L. HE, "Automatic generation of short answer questions for reading comprehension assessment," *Natural Language Engineering,* vol. 22, pp. 457-489, 2016.

[123] L. Zhang and K. VanLehn, "How do machine-generated questions compare to human-generated questions?," *Research and Practice in Technology Enhanced Learning,* vol. 11, p. 1, 2016.

[124] K. Mazidi and R. D. Nielsen, "Linguistic Considerations in Automatic Question Generation," in *ACL (2)*, 2014, pp. 321-326.

[125] Y. Chali and S. A. Hasan, "Towards topic-to-question generation," *Computational Linguistics,* 2015.

[126] V. Rus, B. Wyse, P. Piwek, M. Lintean, S. Stoyanchev, and C. Moldovan, "A detailed account of the first question generation shared task evaluation challenge," *Dialogue and Discourse,* vol. 3, pp. 177-204, 2012.

[127] R. Shah, "Automatic question generation using discourse cues and distractor selection for cloze questions," International Institute of Information Technology Hyderabad, 2012.

[128] M. D. Gall, "The use of questions in teaching," *Review of educational research,* vol. 40, pp. 707-721, 1970.

[129] J. Ryu, Y. Jung, K.-m. Kim, and S. H. Myaeng, "Automatic extraction of human activity knowledge from method-describing web articles," in *Proceedings of the 1st Workshop on Automated Knowledge Base Construction*, 2010, p. 16.

[130] J. Ryu, Y. Jung, and S.-H. Myaeng, "Actionable Clause Detection from Non-imperative Sentences in Howto Instructions: A Step for Actionable Information Extraction," in *TSD*, 2012, pp. 272-281.

[131] A. F. Wicaksono and S.-H. Myaeng, "Automatic extraction of advice-revealing sentences foradvice mining from online forums," in *Proceedings of the seventh international conference on Knowledge capture*, 2013, pp. 97-104.

[132] K. N. El Sayed, "An Arabic natural language Interface system for a database of the holy Quran," *International Journal of Advanced Research in Artificial Intelligence,* vol. 4, pp. 9-14, 2015.

[133] B. Achee and D. L. Carver, "A greedy approach to object identification in imperative code," in *IEEE Third Workshop on Program Comprehension, 1994. Proceedings.,* 1994, pp. 4-11.

[134] K. O. Lundqvist, G. Pursey, and S. Williams, "Design and implementation of conversational agents for harvesting feedback in eLearning systems," in *European Conference on Technology Enhanced Learning*, 2013, pp. 617-618.

[135] A. Strigér, "End-to-End Trainable Chatbot for Restaurant Recommendations," ed, 2017.

[136] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *arXiv preprint cmp-lg/9709008,* 1997.

[137] A. Almuhaimeed, "Enhancing Recommendations in Specialist Search Through Semantic-based Techniques and Multiple Resources," University of Essex, 2016.

[138] J. J. Summers, A. Waigandt, and T. A. Whittaker, "A comparison of student achievement and satisfaction in an online versus a traditional face-to-face statistics class," *Innovative Higher Education,* vol. 29, pp. 233-250, 2005.

[139] S. B. Wegner, K. C. Holloway, and E. M. Garton, "The effects of Internet-based instruction on student learning," *Journal of Asynchronous Learning Networks,* vol. 3, pp. 98-106, 1999.

[140] Z. Yan, N. Duan, J. Bao, P. Chen, M. Zhou, Z. Li*, et al.*, "DocChat: an information retrieval approach for chatbot engines using unstructured documents," in *ACL*, 2016.

[141] R. Franzosi, "Content analysis: Objective, systematic, and quantitative description of content," *Content analysis,* vol. 1, pp. XXI-XLX, 2008.

[142] X. Li, P. Nguyen, G. Zweig, and A. Acero, "Presenting search results according to query domains," ed: Google Patents, 2017.

[143] D. Jurafsky, "Speech and language processing: An introduction to natural language processing," *Computational linguistics, and speech recognition,* 2000.

[144] C. Lee and D. A. Landgrebe, "Feature extraction based on decision boundaries," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 15, pp. 388-400, 1993.

[145] O. D. Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition-a survey," *Pattern recognition,* vol. 29, pp. 641-662, 1996.

[146] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management,* vol. 24, pp. 513-523, 1988.
[147] E. Stamatatos, "A survey of modern authorship attribution methods," *Journal of the Association for Information Science and Technology,* vol. 60, pp. 538-556, 2009.
[148] S. Niwattanakul, J. Singthongchai, E. Naenudorn, and S. Wanapu, "Using of Jaccard coefficient for keywords similarity," in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, 2013, pp. 13-15.
[149] J. Lewis, S. Ossowski, J. Hicks, M. Errami, and H. R. Garner, "Text similarity: an alternative way to search MEDLINE," *Bioinformatics,* vol. 22, pp. 2298-2304, 2006.
[150] A. Huang, "Similarity measures for text document clustering," in *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, 2008, pp. 49-56.
[151] J. L. Fagan, "Automatic P hrase Indexing for Document Retrieval: An Examination of Syntactic and Non-Syntactic Methods," in *ACM SIGIR Forum*, 2017, pp. 51-61.
[152] S. Sanyal, S. Hazra, S. Adhikary, and N. Ghosh, "Resume Parser with Natural Language Processing," *International Journal of Engineering Science,* vol. 4484, 2017.
[153] A. Budanitsky and G. Hirst, "Evaluating wordnet-based measures of lexical semantic relatedness," *Computational Linguistics,* vol. 32, pp. 13-47, 2006.
[154] S. Harispe, S. Ranwez, S. Janaqi, and J. Montmain, "Semantic Measures for the Comparison of Units of Language, Concepts or Instances from Text and Knowledge Base Analysis," *arXiv preprint arXiv:1310.1285,* 2013.
[155] F. Rahutomo, T. Kitasuka, and M. Aritsugi, "Semantic Cosine Similarity," in *The 7th International Student Conference on Advanced Science and Technology ICAST*, 2012.
[156] O. Chapelle, D. Metlzer, Y. Zhang, and P. Grinspan, "Expected reciprocal rank for graded relevance," in *Proceedings of the 18th ACM conference on Information and knowledge management*, 2009, pp. 621-630.
[157] S. He, C. Liu, K. Liu, and J. Zhao, "Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 199-208.
[158] N. Postman and C. Weingartner, "LINGUISTICS, A REVOLUTION IN TEACHING," *ERIC,* 1967.
[159] B. Srinivasan and R. Parthasarathi, "A survey of imperatives and action representation formalisms," *Artificial Intelligence Review,* vol. 48, pp. 263-297, 2017.
[160] S. Crabbe, "Developing a New Controlled Language for Technical Documents," in *Controlling Language in Industry*, ed: Springer, 2017, pp. 89-105.
[161] K. Warwick and H. Shah, "Taking the fifth amendment in Turing's imitation game," *Journal of Experimental & Theoretical Artificial Intelligence,* vol. 29, pp. 287-297, 2017.
[162] R. J. L. John, N. Potti, and J. M. Patel, "Ava: From Data to Insights Through Conversations," in *CIDR*, 2017.
[163] M. G. de Bayser, P. Cavalin, R. Souza, A. Braz, H. Candello, C. Pinhanez*, et al.*, "A Hybrid Architecture for Multi-Party Conversational Systems," *arXiv preprint arXiv:1705.01214,* 2017.
[164] J. Cahn, "CHATBOT: Architecture, Design, & Development," University of Pennsylvania, 2017.
[165] L. Shao, S. Gouws, D. Britz, A. Goldie, B. Strope, and R. Kurzweil, "Generating Long and Diverse Responses with Neural Conversation Models," *arXiv preprint arXiv:1701.03185,* 2017.

[166] R. Wallace, "The elements of AIML style," *Alice AI Foundation,* 2003.
[167] M. d. G. B. Marietto, R. V. de Aguiar, G. d. O. Barbosa, W. T. Botelho, E. Pimentel, R. d. S. França*, et al.*, "Artificial Intelligence MArkup Language: A Brief Tutorial," *arXiv preprint arXiv:1307.3091,* 2013.