A METHOD FOR REPRESENTING KNOWLEDGE AND IMPROVING ANSWER PREDICTION IN QUESTION AND ANSWER DOMAIN

A Thesis Presented to

The School of Computing and Engineering

University of Huddersfield

In Partial Fulfilment

Of the Requirements for the Degree

Doctor of Philosophy

By

Feyiseye Fadeni

September 2019

Abstract

This research offers a new method for knowledge representation and answer retrieval, mainly targeting question and answer systems. The research is based on the study of real-world natural language processing problems, which mostly occur when systems or algorithms are trained to answer questions correctly within a given subject domain. When the same algorithms are faced with a new domain of questions, then new sets of rules or training must be done. Existing systems such as natural language personal assistants, Chatbot's, and query analysers are particularly useful testing tools for this research.

The problem of training systems to answer questions correctly within a set of domains is that, if the question domain changes, then extra time must be allocated again to the retraining of systems so as to adapt to a new set of domain rules. This research offers a generic method to improve the current limitation in this field without having to always retrain algorithms to match a new set of rules.

An Information System (IS) methodology was adopted for the research. A conceptual framework for the problem was developed by combining the understanding from three viewpoints:

1. A study and preliminary experimenting of the real-world language processing systems
2. Reviewing the relevant disciplines for methods and concepts leading to theories for supporting the problem requirements.
3. Investigating popular natural language classifiers to the cognizance of the depth of passage information extraction done with classifiers. Applying the knowledge of Artificial Intelligence (AI) research in the field of knowledge representation and natural language processing, a generic architecture that incorporates language learning for improved answer retrieval was proposed. The Stanford NLP Classifier was used to

categorise, and sort large passages of text given as input to the algorithm. A newly created technique using dependency structure, lemma, parts of speech and name entity recognition was used to filter passages and retrieve the correct answer to a question.

Based on the classification result obtained from Stanford NLP Classifier, a generic knowledge-base was created to represent this knowledge for easier retrieval of answers.

Stories on various topics were shared out to test users to write down question of choice from them. These questions were used to test the algorithm against other language processing systems and Chatbots.

Following this, a confusion matrix was used to analyse and evaluate the algorithms functionality and output against other systems. The algorithm was found to produce more accurate answers for questions in different domains and lesser errors when compared with other systems.

Acknowledgements

All thanks go to God Almighty for the successful completion of this research program. I want to specially appreciate my beautiful wife Dominique for all the support and for standing by me through the course of this program. I also want to appreciate my parents for believing in me and always being there when I needed them to.

A big thank you to my supervisor and co-supervisor, Dr David Wilson and Dr Simon Parkinson for both academic and moral support.

I also appreciate all the people without whom the completion of this research wouldn't have been possible

Hollubee, Folubee, Jaiden, Sapphire, Bola, Raymond, Bros Agbas, Jagaban, Black Spiff et. al

Table of Contents

List of Figures

List of Tables

List of Abbreviations

1. 5W1H – Who, Where, When, What, Why, How

2. AE – Amazon Echo

3. AI - Artificial Intelligence

4. AIML – Artificial Intelligence Modelling Language

5. A.L.I.C.E – Artificial Linguistic Internet Computer Entity

6. API – Application Package Interface

7. CRF – Conditional Random Field

8. FN – False Negative

9. FP – False Positive

10. G3 – Generalized Grounding Graphs

11. GPA – Google Personal Assistant

12. IS – Information System

13. NER – Named Entity Recognition

14. NLP – Natural Language Processing

15. PARK – Planning, Autonomy and Representation of Knowledge

16. POS – Parts of Speech

17. RDF – Resource Description Framework

18. SDC – Spatial Description Clause

19. TN – True Negative

20. TP – True Positive

21. VEIL – Verb-Environment-Instruction Library

Chapter 1. Introduction

As humans, we possess many special abilities, one of which is understanding. The deep cognitive abilities possessed by humans have made it possible for people to provide suitable methods to make challenging task easier over the years. Through understanding, humans have built, produced and even communicated with each other. Communication and understanding are the vital tools for personal development and learning. Natural Language Processing (NLP) is not a new field, as there have been various innovative advances made from the 1950s including information extraction, classification, building syntactic and semantic representations. However, learning of a language that would appear trivial for a child is not trivial for the computer. Therefore, many of these complexities have risen from non-formal forms of languages which have made it extremely difficult for computers to perform straight forward language interpretations (Farghaly & Shaalan, 2009). Solving this problem could take collaborative effort to have a unified language understanding process.

1.1 Research Description

How can computers understand language? This is a question at the core of the area of semantics in natural language processing.

During interaction or question answering, a language processing system is expected to respond to natural language statements, questions, comments etc. Most modern search engines offer some level of functionality for factoid question answering. These systems have high accuracy in pulling out passage sections from popular platforms like Wikipedia, but their understanding could still be significantly improved. From the technical perspective, question answering offers a compromise between challenging semantics problems that are not expected to be solved in the near future, and problems that will be solved in the foreseeable future. The most recent development in question answering is that of the retrieval of answers from open knowledge-bases such as Freebase (a factoid database of various facts without a specific domain tying them all together).

The objective of this research is to explore various methods to improve semantic representations in language, with open questions answering and conversations handling being potentially an important application for testing them. These semantic representations can either be representative (enhanced with a probabilistic interpretation) or they can be predictions in a continuous geometric space.

1.2 Aims and Objective

1.2.1 AIMS

The aim of this research is to improve the accuracy of question and answer systems.

1.2.2 OBJECTIVES

1. Survey research literature to identify NLP state-of-the-art and knowledge gaps.
2. Design and perform preliminary empirical tests to validate limitations in state-of-the-art systems and methods.
3. Investigate algorithmic approaches to improve beyond preliminary results.
4. Construct prototype software tool based on developed algorithm.
5. Perform empirical analysis using the developed software tool to measure performance characteristics.

1.3 RESEARCH QUESTIONS

At the commencement of the thesis, some vital questions were asked. The results obtained for these questions will help narrow or expand the focus where and when necessary.

Below is a list of questions asked; during the start of the research.

1. Is it possible for an algorithm to understand a sentence semantic structure?
2. Is it possible for an algorithm to identify a sentence subject and other sentence attributes?
3. What are the existing NLP methods available used in question and answer systems?
4. What are the limitations of these methods?
5. Is it possible to improve these methods and reduce the knowledge gap?

1.4 CONTRIBUTIONS TO KNOWLEDGE

Throughout this programme of research, the following contributions to knowledge have been established:

1. Novel algorithm that achieves results beyond (8% improvement) current state-of-the-art. The algorithm utilises dependency methods for answer retrieval to get its results.
2. New method of representing knowledge to overcome the limitations of using current approaches.

1.5 Thesis Structure

Listed below is a summary of all the chapters in this thesis.

1.5.1 Background Research

This is the critical stage of the research where existing literature and methods will be studied and documented. This process will involve narrowing down and defining the research's scope. In this stage, experiments will equally be conducted to understand the logic behind language processing and understanding systems and algorithms. Ideas obtained in this phase will be used in creating a new model.

1.5.2 Methodology

The research methodology chapter is a documented process to be used in conducting the thesis life cycle. This chapter will also contain the preliminary tests conducted on existing system to identify their limitation.

1.5.3 Design and Implementation

After identifying alternative theories, a product will be built to test these theories. This chapter is written to explain the full design and development phase that will be carried out during the product development life cycle.

1.5.4 Conducted Experiment

The prototype built will be used to conduct experiments on the system to test the suggested hypothesis. All the results obtained from the experiment will be documented for analysis.

1.5.5 Results Analysis

The results obtained from experimenting will be used to perform critical analysis by statistical methods.

1.5.6 Evaluation and Critical Assessment

When confirmed theories have been established, evaluation of the whole research will be done. The evaluation will cover the method applied, complexity analysis, the research’s contribution to knowledge etc.

1.5.7 Conclusion and Future Work

This chapter discusses future improvements on the methods and algorithm utilised during this research.

CHAPTER 2. BACKGROUND RESEARCH

This chapter focuses on the study of existing works carried out by other researchers and to explore several solutions used in modern day Natural Language Processing (NLP). In addition to studying several methods used in this area, the chapter will also discuss different limitations of certain techniques encountered by other researchers and explore possible improvements. This review will aid in the completion of this research by identifying the extent of work done in this field and help the researcher acquire more knowledge on techniques to meet the aims of the research. Alongside trying to answer the research questions listed in Chapter 1.3, this review will focus on investigating four main points described in (Russell & Norvig, 1995) as the important keys to developing fully intelligent systems.

1. Investigating how machines communicate in a language of choice
2. Investigating methods that help them to effectively represent their existing knowledge of the world in a form which can be retrievable in the future for use.
3. Investigating how machines currently retrieve data from their existing knowledge and its limitations.
4. Investigating how machines learn and adapt to continuous learning.

2.1 Natural Language Commands for Robotics

Since the beginning of Natural Language Processing in Artificial Intelligence, one of the primary aims researchers have explored is the effective delivery of dynamic methods to apply language processing on robots which will enable them to carry out actions by linguistic instructions rather than hardcoding these individual instructions directly into them.

For robots to be fully useful team mates to humans, they must possess the ability to vigorously follow instructions. (Tellex et al., 2011) An example of instructions is the wheelchair user equipped with a robotic arm giving instruction to the arm; “Get me the paper from the kitchen table”. Commands that seem so simple like this can still be very challenging to a machine as this language command consists of events “get”, the object “paper” and a place “the kitchen table”. Scenarios like these prove to be challenging for a robot because such commands are usually mapped to the real world which may be composed in a variety of ways to a countless number of objects in different locations.

Table 2.1 below shows several commands which (Tellex et al., 2011) system understands when applied in a robotic forklift domain and the main processes of how this was achieved will be explained later.

Table 2.1 - Showing commands applicable to robots in a forklift domain (Tellex et al., 2011)

Commands from the corpus
Go to the first crate on the left and pick it up
Pick up the pallet of boxes in the middle and place them on the trailer to the left
Go forward and drop the pallets to the right of the first set of tires
Pick up the tire pallet off the truck and set it down

The problem of following instructions was framed as deducing the most likely robot state sequence from a natural language command (Tellex et al., 2011). Other approaches to solving such problems like (Kollar et al., 2010; Shimizu and Haas, 2009 cited in Tellex et al., 2011) have assumed that; for natural language commands to be understood by robots, they must possess a flat and fixed structure which can then be exploited when deducing actions for the robot. From the experiments carried out by (Tellex et al., 2011) it was found that approaches like the flat and fixed sequential structure will not permit variable arguments or nested clauses. While using phrases like “the pallet beside the truck” during flat and fixed sequential testing, the system was unable to separate the preposition relation “beside” from the rest of the phrase (object) “the truck.” Also, the flat structure approach disregards the structural argument of verbs. For example, the command “get the paper from the kitchen table” has two arguments (“the paper” and “on the kitchen table”), both of which are essential to derive an accurate meaning for the verb “get”. For meaning of unconstrained natural language command to be inferred, the model needs to use compositional and hierarchical linguistic structure at different processing levels e.g. the learning and the interference stage.

As such, a Generalized Grounding Graphs (G3) method was used to address the flat and fixed sequence command structure used in the experiment (Tellex et al., 2011). From instructions given through a natural language command, the G3 model structure is induced using Spatial Description Clauses (SDCs), which was introduced by a semantic structure in (Kollar et al., 2010; Shimizu and Haas, 2009 cited in Tellex et al., 2011). For every SDC in a passage, a relation can be mapped from the linguistic constituent from the SDC command to a piece of the real world referred to as grounding; such as a place, an event or an object.

The approach used in Tellex et al. (2011) is to send an input of natural language command into the system and then, the system tries to identify a plan as an output result for the robot. To derive a correct and applicable plan, the system firstly tries to map different parts of the natural language using the SDC logic to the real world which is also known as grounding in the world (objects, paths, and places). This means that breaking up the commands into smaller chunks should provide more effective scenarios than larger chunks of commands. These mappings derived from the first section is formalised by representing the outputs as a grounding graph, which is a model of probabilistic graphs detailing random variables corresponding to groundings of the real world in a certain environment. Each 'grounding' will be acquired from a semantic map of the immediate environment, consisting of a metric map having the shape, location, the name of each object and the place; it will also consist of the environment’s connectivity in a form of topology. Thereafter, the system uses the individual mappings to sum up a grounding corresponding to the entire command, and later interprets it into a suitable plan of instructions for the robot engine to execute.

$$\underset{\Gamma}{\operatorname{argmax}}\, p(\phi = True|command, \Gamma)$$ (Tellex et al., 2011)

This formula in the equation above explains how the system infers with its surrounding area and corresponds to the entire command to maximize its conditional distribution. This is then deduced as the constructed plan for the robot to execute. By expressing this problem as the formula shown above, the system can perform domain-independent learning. Γ is defined as sets of all the groundings (γi) for any given command. For all uncertainties in grounding candidates, binary correspondence variables Φ was introduced. For each uncertainty (φi), φi ∈ Φ is true if γi ∈ Γ is properly represented as part of the natural language command, or otherwise false.

Table 2.1.2 Showing the experiment result from (Tellex et al., 2011).

SDC Type	**Precision**	**Recall**	**F-score**	**Accuracy**
OBJECT	0.93	0.94	0.94	0.91
PLACE	0.70	0.70	0.70	0.70
PATH	0.86	0.75	0.80	0.81
EVENT	0.84	0.73	0.78	0.80
Overall	0.90	0.88	0.89	0.86

The performance of their method was evaluated through experimenting on trained datasets and statistically calculating the results as shown in Table 2.1.2. To evaluate end-to-end performance, their system was inferred by giving commands for a test set and a starting location for the robot. When compared with a dataset not used for the training, there was a large difference in the result as shown in Table 2.1.3.

Table 2.1.3 Showing the result when experiment was compared on random data (Tellex et al., 2011).

Command	**Precision**
Command with original video	0.91 (0.01)
Command with random video	0.11 (0.02)

The SDC induction approach works best in the G3 Grounding state. Since the G3 state requires physical constraints such as the shape, location, name and place of an object, the model will not suit systems limited by physical capabilities. From the evaluated result showed in Table 2.1.3, the system had very low performance when tested on random data as opposed to a trained data set.

(Misra, Sung, Lee & Saxena, 2015) reviewed the creation of an algorithm capable of learning the grounding maps for robotic instruction output as explained earlier. However, (Misra, Sung, Lee & Saxena, 2015) focused more on the ability to handle missing or incomplete natural language instructions. For example, when given a natural language instruction such as 'heat the pot, the instruction does not clearly say if the pot should be placed on the stove, and such commands need to be deduced from the task constraints and the environment context rather than from the sentence meaning itself. Also, it is not always necessary that one should follow natural language instructions precisely from the instruction meaning, rather, sometimes, one must come up with alternatives that are appropriate for a robot to perform in certain situations. (Misra, Sung, Lee & Saxena, 2015) alternative for handing missing natural language command was to develop a method modelled on variations in natural language considering natural language ambiguities and grounding instructions as well as ambiguities to robotic instructions with task constraints and environment context.

The following steps have been taken in creating this model.

- Firstly, a database of robotic and natural language instruction categorisations executed for several tasks in an online simulated game was collected.
- Using the data obtained, a verb-environment-instruction library (VEIL) was built. A verb-environment-instruction library (VEIL) is; for a certain verb clause (called C_i), training data (D) used to derive sample sets of instruction models $\{I_i^{(s)}\}$.
- During the robotic ground mapping training, a large set of verb clauses $C^{(j)}$ is collected, and the resultant instruction-sequence $I^{(j)}$ and the environment $E^{(j)}$.
- Some of the parameters of the commands may not be available in the new test environment for the training dataset of specific values. Therefore, such parameters are replaced with generalized variables (Z) in the training instructions. For instance, grasp(bottle01)} or moveTo(cup01) would be represented as generalised instruction sequence grasp(Z_1) and moveTo(Z_2) respectively and the original mapping $Z_1 \rightarrow$ bottle01 and $Z_2 \rightarrow$ cup01 is stored in $\xi^{(j)}$, which is where the original mapping is stored.
- After collecting the database of robotic and natural language instructions, a machine learning approach that simulates the relations between the language, robotic instructions and environment states is adopted. The learning model used is isomorphic to a conditional random field (CRF), which encodes properties in the form of possible tasks on the boundaries.

The (Misra, Sung, Lee & Saxena, 2015) experiment was conducted in four stages which are, making coffee, boiling water, making ramen and making affogato. The outcome of their chance performance test was very low, thus reducing the chances of getting the correct sequences. This approach was problematic in working with large search tree indexes and equally fails with vague language.

(Misra, Sung, Lee & Saxena, 2015) built VEIL algorithm which mainly considers the actions clauses (called C_i) in natural language. According to (MORI, OHTA, FUJIHATA & KUMON, 2001) for machines to really understand complex natural language passages, a deeper level of language analysis is needed. This goes beyond the reliance on parts of speech or entity recognition alone.

2.2 Natural Language Semantics and Application

As far back as 1974, (Thompson & Winograd, 1974) identified a way of representing knowledge and language understanding in a usable and flexible manner. Some of these concepts (mostly word classes) discussed immediately below can be seen in the implementation of modern-day classifiers like (de Marneffe & Manning, 2008). The system they proposed takes certain steps in understanding the sentence and can be determined directly by syntactic construction, special knowledge about a word in the passage, and a fact about the world as shown in Figure 2.2.1

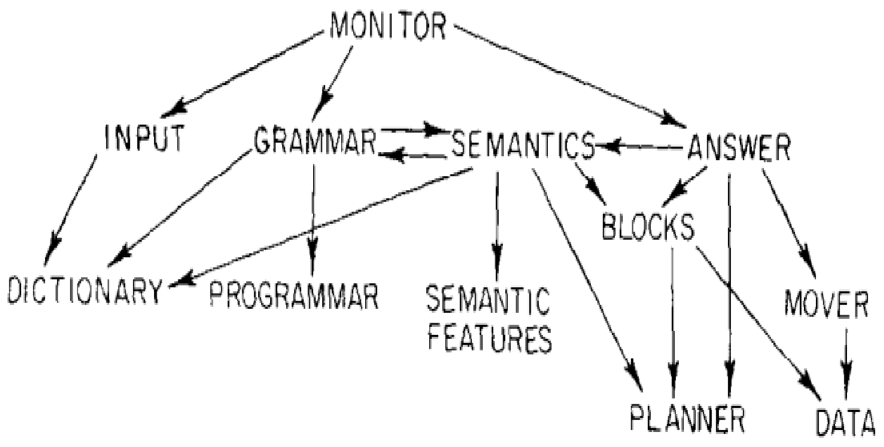


Figure 2.2.1 Organisation of the system, (Thompson & Winograd, 1974)

The arrows in Figure 2.2.1 above show which sectors of the system make use of other sectors for its direct processing.

- The **Monitor** is referred to as the main method or function;
- **Input** accepts typed input, normally in English language orthography and punctuation. Inputs are then extracted to get individual words from the input and searches for them in a dictionary to extract their lemma. The input returns a string of words together with their dictionary meaning back as input.
- **The Grammar** coordinates the total language understanding process.
- **The Semantics** interprets sentence inputs from the grammar.
- **The semantic** phrase then calls the **Planner** to make use of inference in translating sentences from **The Semantics**.
- **Answer** contains several experimental codes for finding the answer to a question.
- **The Programmer** helps in interpreting T**he Grammar**.
- **The Dictionary** consists of topographies related to each word and a semantic meaning for each word.
- **The Semantic** makes special provision for word forms such as "geese" or "slept", so therefore, only the definitions of such root words are stored.
- **The Semantic Features** are used to analyse initial semantics.
- **The Blocks** contains theorems about the knowledge of the system and properties of the physical world. (Thompson & Winograd, 1974)

Word Classes

Similar word classes identified in (Thompson & Winograd, 1974) have been seen in recent natural language classifiers for their word relationship. Each of these classes is divided into subclasses by the feature which the individual words possess. The list of the different word classes is presented in an alphabetical order below. Breaking down parts of speech further into word classes improves natural language classification.

- BINDER: A Binder basically is used to bind 2 clauses together for example, “before we got there, they had left. I’ll go, if you do.
- CLASF: A CLASF can appear in a position as an adjective, but it is usually another noun e.g. “boy scout”.
- NUM: This is a large class of numbers, both countable and uncountable infinite).
- NUMD: This applies to more complex word number specifications, e.g. “at least five”
- ORD: The ordinal class comprises of ordinal numbers “first”, “second”, “third etc. A few other words like “last”, “next” etc. can fit into this category; the superlative adjectives in the Noun group can also be included in this group.
- PRT: This is a combination of the use of a “particle” and a verb together e.g. “stand up” or “knock out”. The second words in the examples are types of PRT.
- QADJ: A type of QUESTION CLAUSE makes use of QADJ such as how, when or where as they are elements of question.
- TPRON: This are the small groups of words that are made up of suffixes and quantifiers e.g. “-thing”.

“Sentences are considered learnable if they are correctly learned by heart” (Hinaut et al., 2014) One of the major challenges with language processing is the ability to make a link between the form of a sentence and its meaning. Word association (associating single words with respect to roles) is not enough; considering that the one sentence can be represented syntactically in different ways and would still make enough sense to its reader. For example, “John was hit by Mary” and “Mary hit John” have the same meaning but are of different constructs; how does a single written algorithm for language processing know that both sentences have the same meaning without having to create different algorithm for each different question type? Or how could an infant determine who is carrying out the action “the agent” and who is the receiver of the action? “the object” (Hinaut et al., 2014). Relying only on semantic words and their order in the sentence, does not successfully distinguish the *agent* from the *object*. (Goldberg, 2003) Research on the notion of mapping sentence form and their meaning, which is called grammatical construction, was taken into consideration (as citied in Hinaut et al., 2014). Grammatical construction can be used to answer questions as "Who did what to whom" by replacing sentences with different forms of entity such as (*agent*, *direct object*, *indirect object*, *recipient*) and finding the appropriate role for each semantic word as shown in Figure 2.2.2 below. In the sentence "Mary hit John", the constituting semantic words have been assigned specific roles.

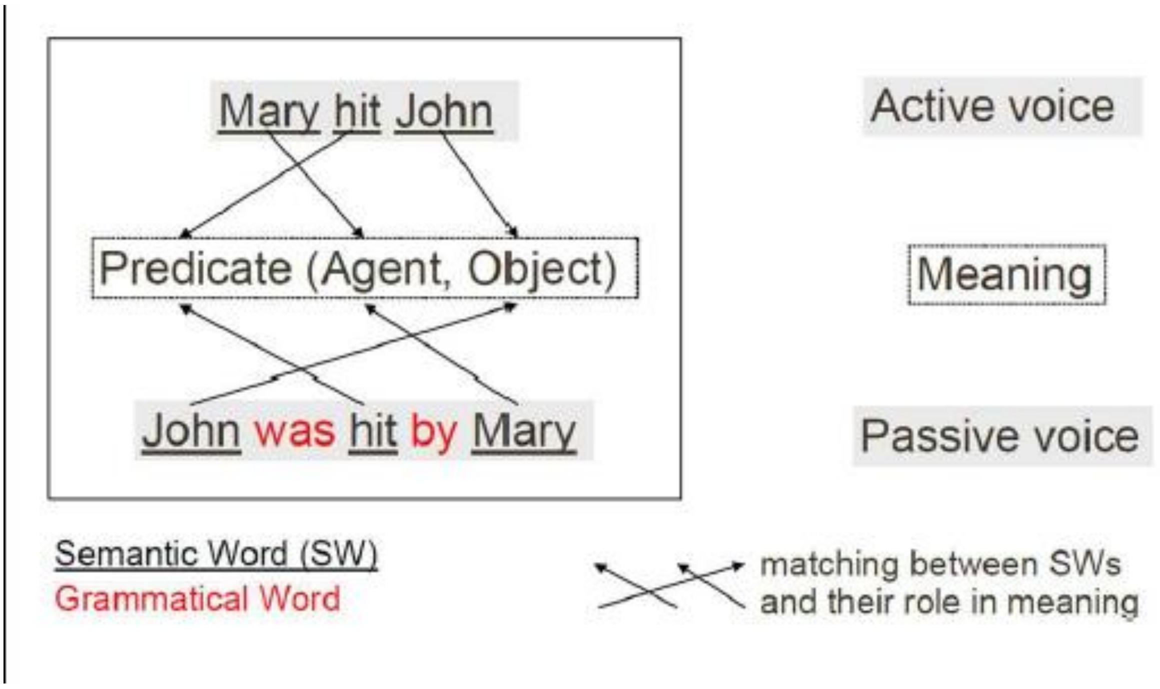


Figure 2.2.2 - Showing characterisation of roles to semantic words (Hinaut et al., 2014)

Figure 2.2.2 exemplifies the thematic role assignment task. Solving this task involved the finding a satisfactory mapping between word or semantic contents and the roles they played in the meaning of a given sentence. Experiments conducted on (Hinaut et al., 2014) asked the iCub robot to *put (violin, left)* and the robot then put the violin on the left. For other trials, the required actions were *point (guitar)* and *put (violin, left)*. The robot could firstly point the guitar and then put the violin left. Following this process, they tested text with the same meaning but different grammatical constructions. Though the iCub used this model to identify the correct entities of the sentences and mapped the actions to the entities, this method may be insufficient for complex sentences in a question and answer domain where named entity recognition is simply not enough to prove proper sentence understanding (Allen, 1987). This paper stated that while some researchers may use keyword or entity matching techniques e.g. lawyer, sue, court, affidavit etc. to interpret natural language, it cannot be said that the system is really understanding the text, rather it is using a simple matching or recognition technique. It would be very unlikely that such systems would be able to handle complex natural language retrieval task such as "find all the articles on property buyouts where the amount is greater than £1,000,000". Google's search engine can produce near accurate results by keyword matching, but at times, it generates unrelated content to questions due to its inability to fully understand the content, word senses and sentence meaning appropriately.

(MORI, OHTA, FUJIHATA & KUMON, 2001) journal was written to discuss the improvement of query algorithms. (MORI, OHTA, FUJIHATA & KUMON, 2001) proposed query algorithm produces generalised answer to questions. (MORI, OHTA, FUJIHATA & KUMON, 2001) paper discusses some useful algorithms and techniques for question answering. Some sections in the paper discussed about patterns for question breakdown and passage classification.

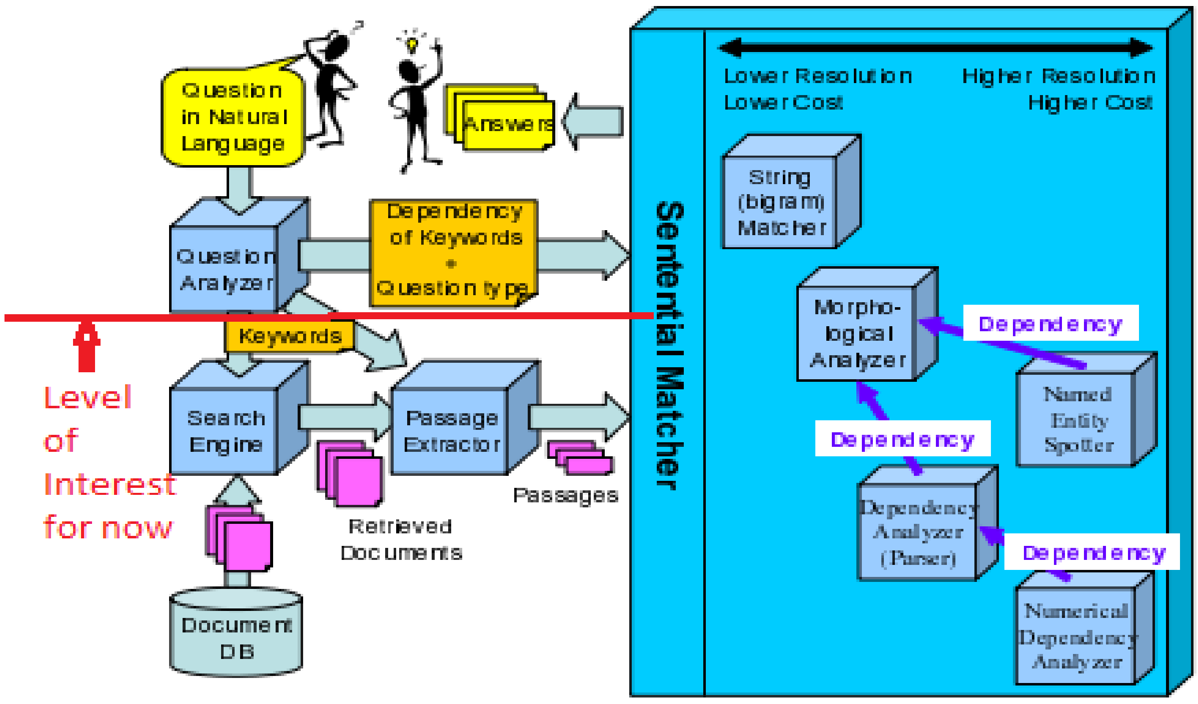
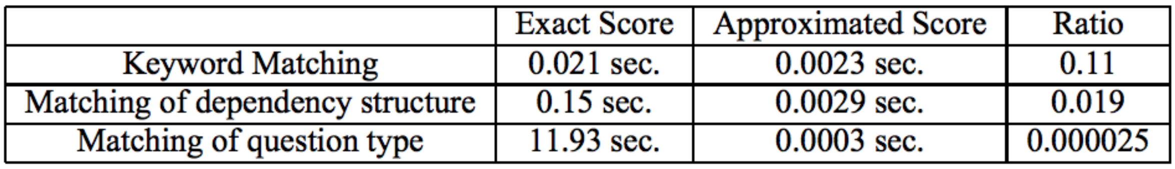


Figure 2.2.3 - Showing the architecture of the (MORI, OHTA, FUJIHATA & KUMON, 2001)

For relevant materials to be queried using the above technique illustrated in Figure 2.2.3, there are some important things to be noted. A typical question and answer system would normally accept factual questions like “who, what, when, where, why and how” (5W1H). Each of the individual 5W’s can consist of a different return attribute value e.g. *How* can return numerical expressions like time, distance etc. *Who* can return a noun or a pronoun which will mostly be the name of a person, and *when* can return the time at which an event is taking place. From Figure 2.2.3, the authors have proposed a system which is broken down into four individual parts; these parts are:

1. Question Analyser
2. A search engine
3. Passage Extractor
4. A sentential matcher

The question analyser is the most relevant section related to this undergoing research; and noting from Figure 2.2.3, above, a red line has been drawn across the image to demarcate the needed portion from the rest of the image. The clues used in the report for finding answers to questions are divided into two. The first one is to extract keywords from the given sentence and these keywords would be highly related to the answers. More specifically, nouns, verbs and adjectives can be regarded as keywords. The second is to analyse the question type asked. Question type is considered as a type of restriction to generated machine answers like Person, Location, and Money.



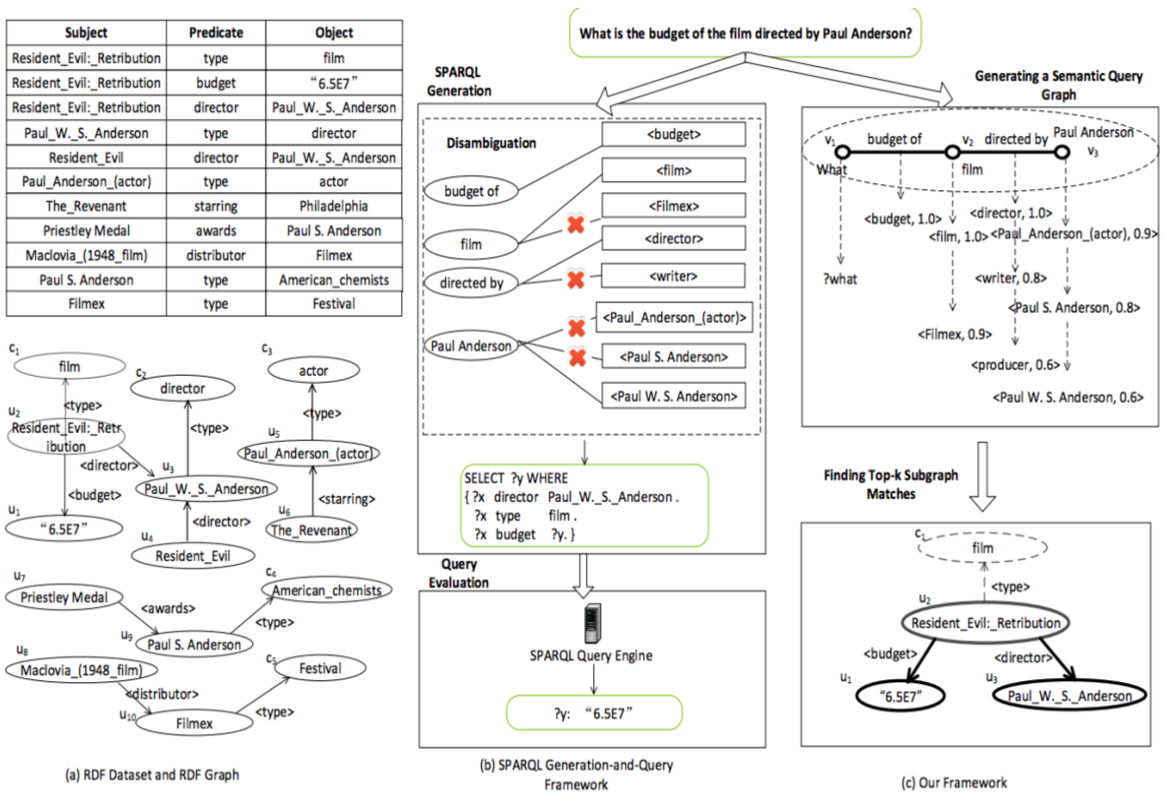
	Exact Score	Approximated Score	Ratio
Keyword Matching	0.021 sec.	0.0023 sec.	0.11
Matching of dependency structure	0.15 sec.	0.0029 sec.	0.019
Matching of question type	11.93 sec.	0.0003 sec.	0.000025

Figure 2.2.4 - Result from (MORI, OHTA, FUJIHATA & KUMON, 2001) experiment

From Figure 2.2.4, (MORI, OHTA, FUJIHATA & KUMON, 2001) experiment focused more on cutting down computational time and improving computational efficiency rather than its ability to retrieve accurate answers.

2.3 Resource Description Framework

RDF (Resource Description Framework) repository is a web standard knowledge-base and its collection is denoted as subject, predicate and object. (Hu, Zou, Yu, Wang & Zhao, 2017) SPARQL is the major way to access or query RDF data, because of the RDF schema and SPARQL syntax complexity. It is difficult for end users to use this framework. An ideal system or modification to the existing system could allow users to benefit from the open and vast power of semantic web standards while providing an easy to use system for these users. According to (Hu, Zou, Yu, Wang & Zhao, 2017), the two main steps to solve the issue of using RDF for question and answering systems is to create a question understanding and query evaluation system. RDF’s major issue is its difficulty in the ambiguity of unstructured natural language question sentences, which is divided into 2 phases, the linking phrase and the composition.



Subject	Predicate	Object
Resident_Evil:_Retribution	type	film
Resident_Evil:_Retribution	budget	“6.5E7”
Resident_Evil:_Retribution	director	Paul_W._S._Anderson
Paul_W._S._Anderson	type	director
Resident_Evil	director	Paul_W._S._Anderson
Paul_Anderson_(actor)	type	actor
The_Revenant	starring	Philadelphia
Priestley Medal	awards	Paul S. Anderson
Maclovia_(1948_film)	distributor	Filmex
Paul S. Anderson	type	American_chemists
Filmex	type	Festival

```
SELECT ?y WHERE
{ ?x  director  Paul_W._S._Anderson .
  ?x  type      film .
  ?x  budget    ?y . }
```

Figure 2.3.1 - Using RDF dataset to answer natural language questions (Hu, Zou, Yu, Wang and Zhao, 2017)

Figure 2.3.1 above shows three stages that have used RDF dataset to answer a natural language questions; “What is the budget of the film directed by Paul Anderson?” Section (a) shows the question being broken down using RDF dataset and RDF Graph. Section (b) interprets the question as a SPARQL query which is evaluated to produce the answers.

The phrase linking issue using RDF dataset occurs because a natural language phrase can have several meanings, and sometimes these phrases should be mapped with the non-atomic structure of the Knowledge graphs. The approach used in (Hu, Zou, Yu, Wang & Zhao, 2017) was not to generate SPARQL at the first stage (question understanding) as other theories have proposed, but rather, present a better solution which is to allow for ambiguity at the understanding stage of the process, and resolve these ambiguities when the matches are found in the second stage (query evaluation) of the research. In a nutshell, the writers used two data-driven-graph frameworks for RDF Q/A solution, combing query evaluation and disambiguation together. The first step addressed the ambiguity of phrase connection at the query evaluation; while in the other framework, the ambiguity of query graph's structure and phrase connection are both resolved. The authors of the paper extracted relationships from the question "What is the budget of the film directed by Paul Anderson?" which is called Relation-first framework approach from (Christopher D. et al., 2014), as shown in Figure 2.3.2, below. This same question was again tested on the (Christopher D. et al., 2014), this time, by the author of this thesis to extract the relationship and to reaffirm this output. The graphical relationship output retrieved was not the same as what was shown in (Hu, Zou, Yu, Wang & Zhao, 2017) and Figure 2.3.3, below shows the difference in relationship between the paper and the test carried out on Stanford's parser.

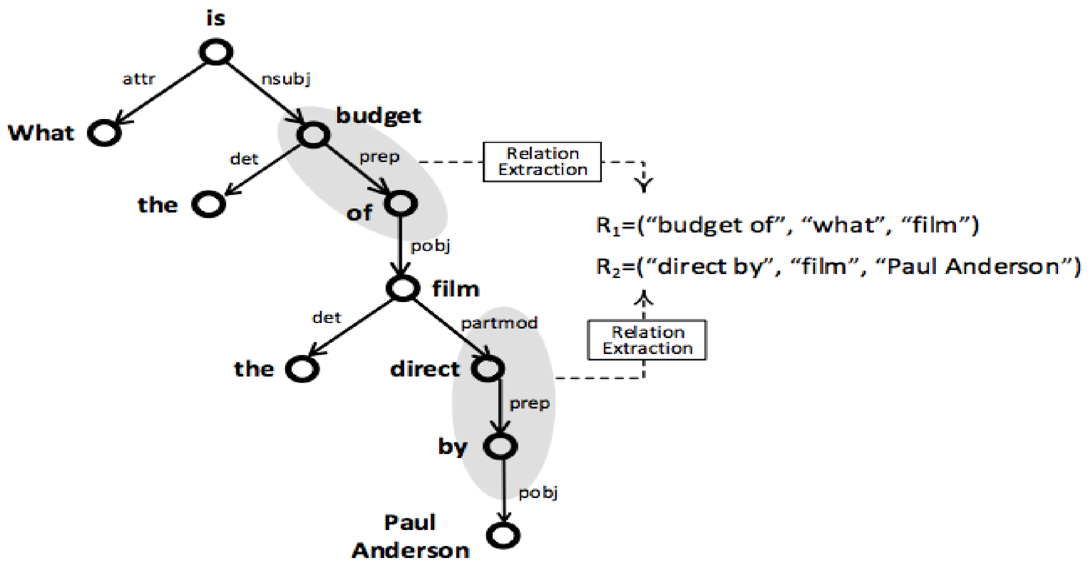


Figure 2.3.2 - Relation used in (Hu, Zou, Yu, Wang & Zhao, 2017)

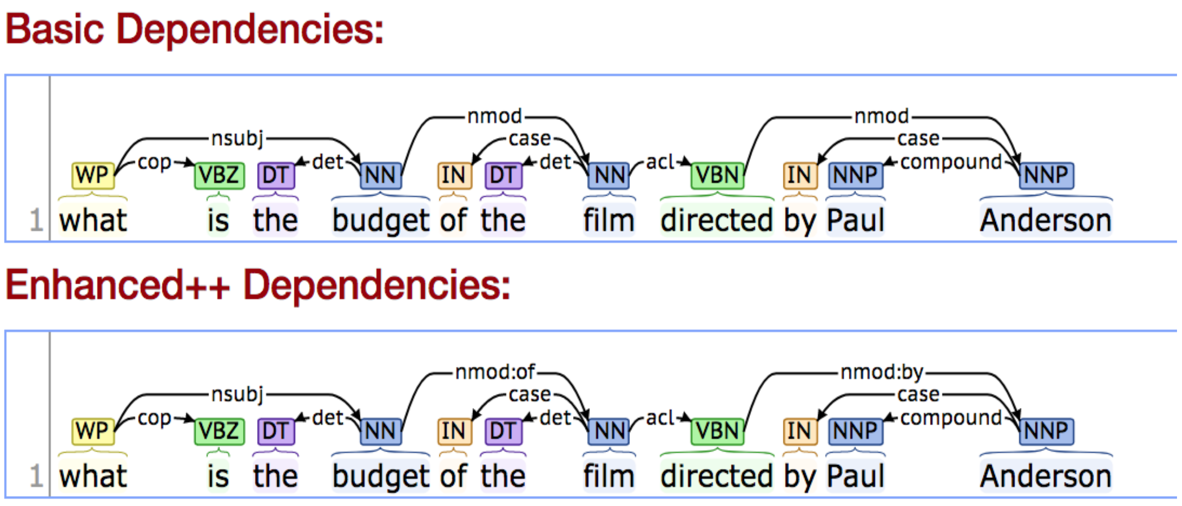

Basic Dependencies:
WP VBZ DT NN IN DT NN VBN IN NNP NNP
nsubj cop det nmod case det acl nmod case compound
1 what is the budget of the film directed by Paul Anderson
Enhanced++ Dependencies:
WP VBZ DT NN IN DT NN VBN IN NNP NNP
nsubj cop det nmod:of case det acl nmod:by case compound
1 what is the budget of the film directed by Paul Anderson

Figure 2.3.3 - Relationship retrieved from (Christopher D. et al., 2014)

By comparing Figure 2.3.2, to Figure 2.3.3, the graphical representations of the word dependency tree do not correspond. For example, in Figure 2.3.2, "budget" is directly connected to the word "of" with the "prep" relationship, while in Figure 2.3.3, "budget" is not directly connected to the word "of"; some of the other dependencies in the Figures equally do not correspond. The major causes of problems using the approach in this paper are:

1. A lot of relations are not captured by the classifiers accounting for 58% of the total error (Hu, Zou, Yu, Wang & Zhao, 2017).
2. Extraction of the wrong nodes. Because classifiers and RDF resources are based on different foundational structures, this error is bound to occur. For example, using the sentence "In which countries do people speak English", the correct semantic relation (subject, predicate, object) expected by the RDF Framework is (Speak, English, Country) the classifier returned (Speak, People, Country) causing a mismatch to occur.

Unger et al. (2017) used the information required from both free text and RDF framework pseudo-queries to show sections contained in the RDF data and free text. The annotations used here are all in XML format. Using QALD (Question Answering Over Linked Data), questions can be marked with these attributes:

- Answer type, which describes the expected answer type to be returned.
- A resource (a URI provided for one or many resources).
- A string value.
- A date.
- A numerical value (decimal, fraction or a whole number).
- A Boolean value (true or false).

If a hybrid annotation exists in the syntax, it means that both RDF and free text data information is required. A sample query derived from the question “who is the front man of the band that wrote Coffee & TV?” was tested on the knowledge base.

```
<question id ="335" answertype="resource" aggregation ="false" onlydbo="true" hybrid="true">
  <string lang="en">Who is the front man of the band that wrote Coffee & TV? </string >
  <pseudoquery>
    PREFIX res: < http :// dbpedia.org / resource / >
    PREFIX dbo: < http :// dbpedia.org / ontology / >
       SELECT DISTINCT? uri
       WHERE {
          res: Coffee_ & _TV dbo:musicalArtist? x .
         ? x dbo:bandMember ? uri .
         ? uri text:" is " text:"frontman" .
       }
  <pseudoquery>
  <answers>
    <answer>http://dbpedia.org/resource/Damon_Albarn</answer>
  </answers>
</questions>
```

The answer obtained enclosed in the above XML answer tag is returned as a URL pointing to the most likely webpage that contains the answer to the question (Unger et al., 2017). The query has been manually edited by adding “*musicalArtist”* RDF type to get the best result. The return values from a URL of "dbpedia" do not give the exact answer to a question but returns a series of text contained in a web page about the suggested answer. From the example denoted in (Unger et al., 2017), this means that the user must be aware of "RDF type" to manually edit queries and this requirement of direct human interference may not be suitable to the current research being undertaken. However, there is still a lot to learn from the operations and model of the RDF Framework.

(Nicula, Ruseti & Rebedea, 2015) suggested a way to query resources automatically using an algorithm. Figure 2.3.4 presents a diagram on how to use automated systems to retrieve answers from knowledge bases.

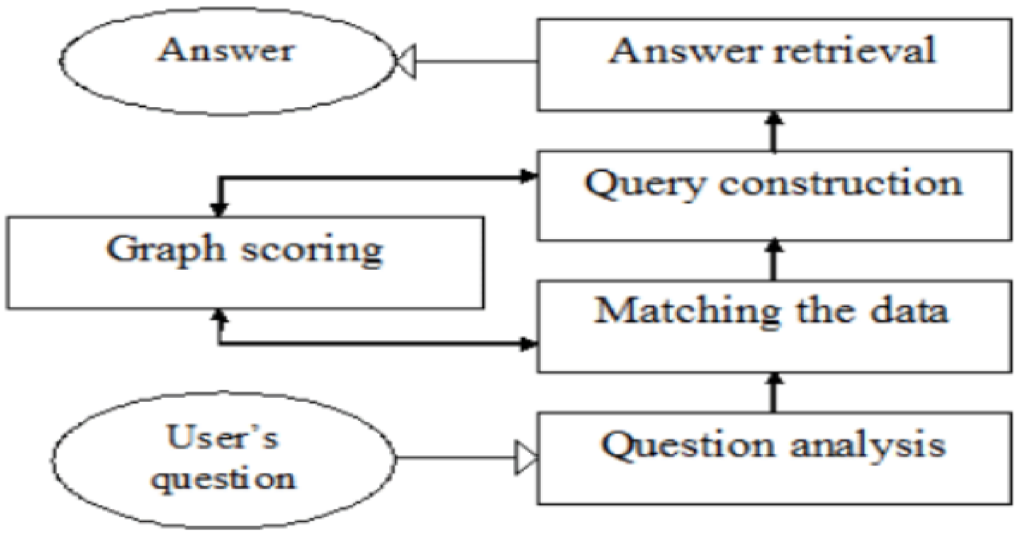


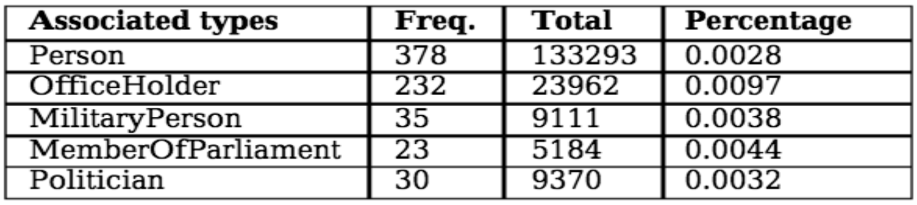
Figure 2.3.4 - RDF steps to retrieve data (Nicula, Ruseti & Rebedea, 2015)

The question analysis in Figure 2.3.4 takes the users question and extracts the suggested data to be used in the query. For type detection, a synonym approach may be appropriate for simple queries like "football player" or "soccer player" but it may prove to be inaccurate for more complex queries like, "offensive midfielder" so the synonym approach was dropped. A knowledge base with several possible RDF expression-types e.g. "MusicalArtist" was developed following two steps. For step one, they extracted the entities and classes in the first line from the DBpedia web page because it was assumed that the first sentence will always contain the answer. So if it does not, then the logic will be wrong. They extracted a list of entities and classes from the sentence using Stanford NLP. Stanford NLP extracts complement preceded by copula verbs and makes small modifications to the entities. For the second step, the parameters extracted from the passage are put together to create a custom expression. Finally, at the end of these steps, the following form of the tuple should look like *<URI, dbpedia_type, List<probable_expressions>>*. For example, the tuple should look like:

<http://dbpedia.org/page/Nat_King_Cole, MusicalArtist, List["singer", "musician"]> (Nicula, Ruseti & Rebedea, 2015)

After getting this expression, next step is the filtering stage of the expression. The expression could return over 900,000 possible matches from the knowledge base and its essential to eliminate noise.

Two processes were taken to filter out noisy data which are: statistical cut and DBpedia type update. **Statistical Cut** eliminates words from the list of expressions until it is certain that the remaining words describe "the RDF type" accurately. Statistical cut corrects parsing errors while **Type Update** corrects errors which have resulted from automatically parsing of information through Wikipedia. Not all DBpedia resources are accurate enough, for example in (Nicula, Ruseti & Rebedea, 2015), Kevin Spacey is classified to be a "Person". Though he is a person, he could have been accurately classified as an Actor. The word "diplomat", which is an expression, is related to five different types as shown in Figure 2.3.5.



Associated types	Freq.	Total	Percentage
Person	378	133293	0.0028
OfficeHolder	232	23962	0.0097
MilitaryPerson	35	9111	0.0038
MemberOfParliament	23	5184	0.0044
Politician	30	9370	0.0032

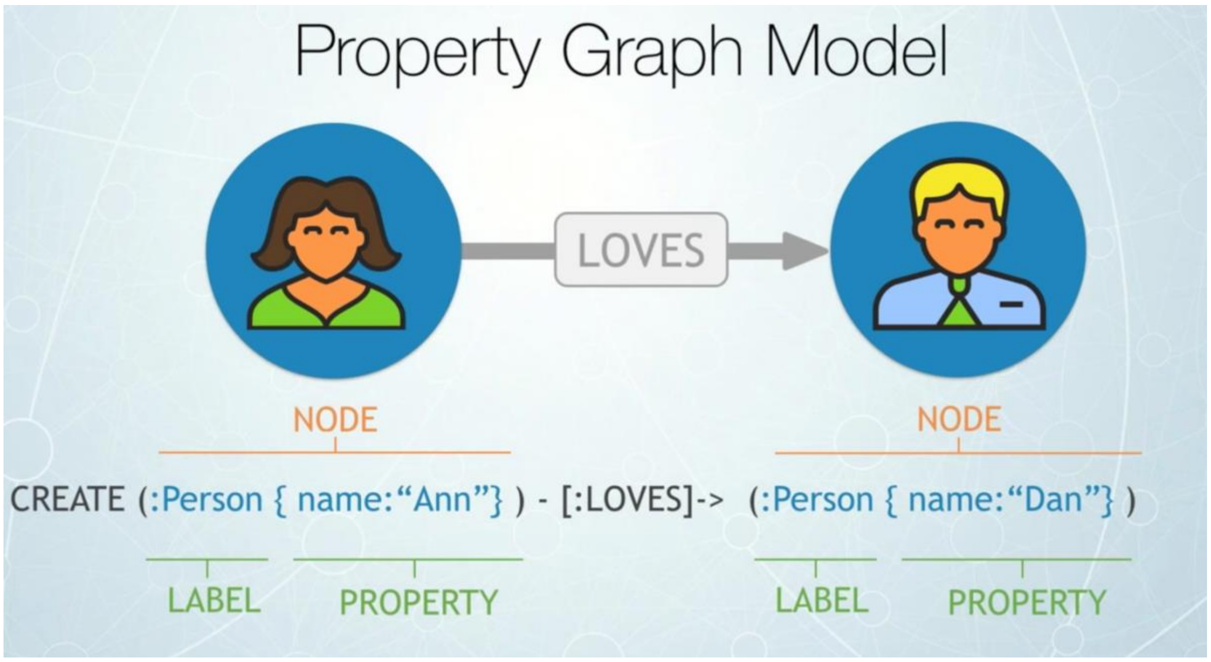
Figure 2.3.5 - RDF association types and frequency (Nicula, Ruseti & Rebedea, 2015)

Type Update process tries to associate the "diplomat" word with the right RDF type to narrow down noise where *Relevance = Freq./Total*

The precision (of types) retrieved from this method was between 0.45 to 0.49 and the recall was between 0.943 and 0.983. The method was only measured to check the closest RDF type but not the closest answer to a question and still, the precision of the method they adopted is quite low.

2.4 Graph Databases

(Negro, 2017) is simply a process of representing data as graphical patterns that enables loosely coupled queries making it easy to understand whilst still maintaining the relationships between data.



```
CREATE (:Person { name:"Ann"} ) - [:LOVES]-> (:Person { name:"Dan"} )
```

Figure 2.4.1 - Showing Neo4J entity relationship model (Neo4j, 2017)

Figure 2.4.1 identifies the connectivity between entities (called nodes) in a graph database. The reference between property one (Ann) and property two (Dan) is named “loves”. The property graph model emphasises more on the kind of relationship that exists between entities and with such relationships, queries can be generated. This relationship in (Neo4j, 2017) is like (Hinaut et al., 2014) paper where the nodes are nouns and the relationships are verbs. The disadvantages of relying on mostly parts of speech tagging for complex language understanding have already been explained earlier in section 2.1 and 2.2. However, the form of knowledge representation is very flexible and graphical and could be considered as a potential knowledge base for modern systems.

2.5 Sentence Analysis

Sentence analysis will be done on English language to understand how sentences are broken down. This section is written to explain how semantic can be broken down to enhance sentence understanding and provide answers to some of the research questions which will facilitate the aims of this research. According to (Bureman, 2012) a sentence must have a subject and predicate. Since all sentences must have a subject and predicate then a complete language processing tool should have the capability to identify the subject and predicate of a sentence. This way, the system will be able to tell if the sentence being analysed is a complete sentence or not. The (de Marneffe & Manning, 2008) Stanford library discussed in section 2.6 is one of the APIs that can identify the subject of a sentence (which it refers to as nsubj). Diagrammatic representations of sentences are useful because it is an easier way for a reader or viewer to know if an individual or a machine has understood the segmented language, by re-representing their understanding of a passage in diagrams. Semantics which was once said to be the future of language processing is now the present. (Ballard, 2001) Semantics is described as the study of meaning and is very important because the whole concept of language is to convey meaning through communication to either listeners or recipients. The only problem (if it is a problem at all) is that meaning cannot be bordered around the same principles of **lexis** (the words of a language) or **grammar** (words combined into sentences). Semantics operates on a totally different level but also work in connection to them all. For more on lexis and sentence structure, refer to Appendix B.

2.5.1 Tips for sentence segmentation and diagramming

- The easiest way to segment and represent passages or sentences as diagrams is to identify the parts of speech in the sentence (Ballard, 2001). However, from (MORI, OHTA, FUJIHATA & KUMON, 2001), using algorithms to breakdown passages into individual parts of speech could be very time ineffective. Also, a single word can represent different parts of speech depending on the way it is used.
- As previously explained, the subject of the sentence must be identified
- Identify the main verb of the sentence and by so doing, the subject's action can be determined.
- Create a list of all adjectives or adverbs, which are acting as a modifier to the subject, object or verb.
- Get the list of any preposition or phrases acting as a modifier or that provide extra information, about something in the sentence (Ballard, 2001).

By meeting the requirements in the above listed points, the segmentation process of proper language understanding should be easier. Normally, the subject and the (main) verb of any sentence can serve as the base or starting point.

2.6 Language Classifiers and Chat Bots and Advanced Tools

One major task in semantic representation is to consider the combinations of individual words and their meaning to form coherent sentences. The (de Marneffe & Manning, 2008) Stanford dependency library is one of the APIs used in semantics for natural language processing. The Stanford typed dependencies were designed for the sole purpose of non-linguistic experts to understand and effectively provide the simplest description of the grammatical associations and relationships in a sentence, so they can extract textual relations with ease. In a simple example to demonstrate how this library works, here is a sentence: *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products*." The sentence is represented with the Stanford Dependencies (SD) as:

- nsubj (makes-8, Bell-1)
- nsubj(distributes-10, Bell-1)
- vmod(Bell-1, based-3)
- nn(Angeles-6, Los-5)
- prep_in(based-3, Angeles-6)
- root(ROOT-0, makes-8)
- conj_and(makes-8, distributes-10)
- amod(products-16, electronic-11)
- conj_and(electronic-11, computer-13)
- amod(products-16, computer-13)
- conj_and(electronic-11, building-15)
- amod(products-16, building-15)
- dobj(makes-8, products-16)
- dobj(distributes-10, products-16)

The numbers written by the right side of the words in the bracket stands for the original position of a word in the sentence. From the first and second bullet point of the dependency tree break down, the nominal subject (nsubj) in point one shows that **Bell,** which is the company and the first word in the sentence (-1) is directly related to **makes,** the eight word (-8) of the sentence. **Bell** (-1), as a company, is also directly related to **distribute** (-10), another nominal subject (nsubj) in the sentence. The (de Marneffe & Manning, 2008) Stanford typed dependencies currently contains 50 grammatical relations, which is being used in grammatical analysis and what they represent individually.

(Chinchor 1998; Sundheim 1995 cited in Soon, Ng & Lim, 2001) explained that co-reference resolution is a very important aspect of Information Extraction systems (IE). Co-reference already supported by Stanford dependency identifies two or more terminologies referring to the same entity in a passage.

Artificial Intelligence Modelling Language (AIML) is a mark-up language based on XML for creating intelligent applications and Chatbot (Wallace, 2019) AIML promotes simple implementation, easy to understand and highly maintainable interfaces. AIML was used to create Alicebot; a free software based on Artificial Linguistic Internet Computer Entity (A.L.I.C.E.). A simple example of how to train an AIML model is shown in the code below.

```
<aiml version = "1.0.1" encoding = "UTF-8"?>
   <category>
      <pattern> HELLO ALICE </pattern>
      <template>
         Hello User!
      </template>
   </category>
</aiml>
```
("AIML Tutorial", 2018)

The *<template>* tag represents the reply of the system when interacted with.

Result from the sample code above

User: Hello Alice

Bot: Hello User

This model of training is very complex and will require a lot of technical skills to expand the knowledge base. It is also very human resource dependent which is one of the areas this research hopes to make improvement; reducing human involvement in language understanding for interactive systems. Wolfram is a large database of facts and numbers which makes the application a better alternative to google if a person is in search for direct and factual answers rather than general related query from different webpages. It can also be said that Wolfram Alpha focuses more on quality than quantity. Wolfram is capable of processing both linguistic sentences and mathematical questions. The thing that really makes Wolfram outstanding is its ability to run sophisticated algorithms for mathematical numeric computations with or without the formulas entered. (Gray, 2017) An example will be to ask Wolfram the distance of the earth to the moon. The distance lying between the earth and moon is not constant and it can change by about a mile in one minute. Wolfram gives not only the conventional time, but also the exact time the current minute. To carry out such computations or precise analysis, Wolfram knows specific coordinates to calculate the exact answer to the question; and that also is the main issue of Wolfram.

For every specific calculation, a unique program algorithm is individually created. Of course, some of these algorithms can be re-used for similar tasks but for a new task, a new algorithm must be derived. Wolfram Alphas solution will generate more accurate results but requires hands on approach for every new challenge. Meaning if there are a thousand new formats, a thousand-new algorithm must be derived to successfully provide answers. Also, for wolfram, because their database contains only facts, the team must singularly insert accurate data for every occurrence into the database taking a very long time to do that.

("IBM Watson", 2017) is another classifier based on unstructured information management which is a standard used for analysing textual contents. ("IBM Watson", 2017) states that when using Watson, “it is possible to go beyond artificial intelligence”. Watson became a very popular artificial intelligence solution when it defeated all contestants (humans) in a game show called “Jeopardy”. (Engadget, 2011). Jeopardy is a televised game show aired in America. It takes the form of a quiz-based competition where all the contestants are presented with clues (general knowledge) that takes the form of an answer and they must phrase their responses to take the form of a question. After Watson’s incredible win in the game show, IBM opened Watsons API to third party developers, and it has since been used in several areas of natural language processing such as businesses analytic classifications and in training chat bots.

Watson is a super computer with a huge number of cores, meaning it is on its own a super computer giving it the ability to process data rapidly. The major capabilities of Watson as an API will be tested in the next chapter to identify if it can be used to satisfy the requirements of this research.

Open information extraction is a tool that extracts possible relations from public passage with a user’s supplied information. (Center, 2016) The user enters 3 parameters into the system:

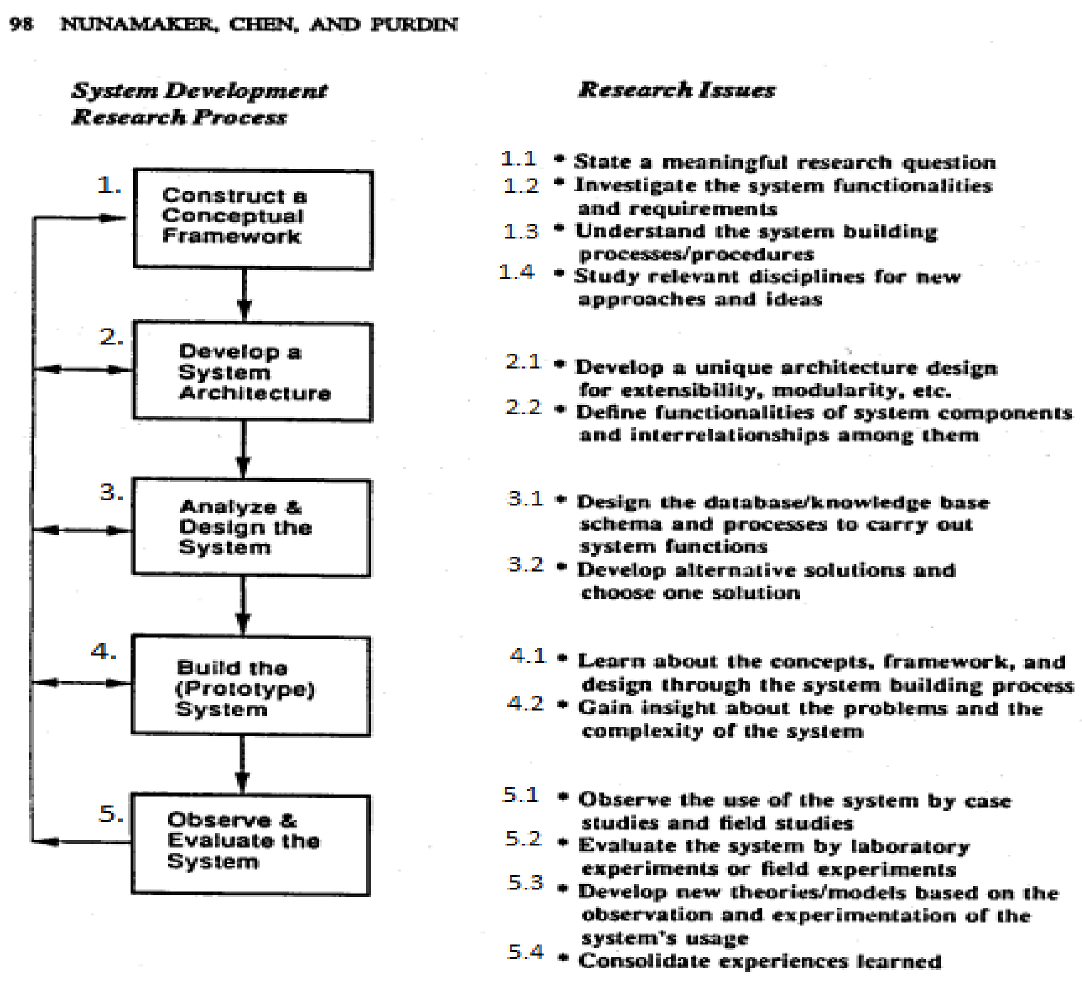
1. Argument 1 – What/Who
2. Argument 2 – Verb phrase
3. Argument 3 – What/Who

With these parameters, matches from various online contents are suggested. This is a very useful tool for streamlining matches to an event on the web.

From the existing literature examined and methods discussed in this chapter, adequate knowledge has been gained with new ideas to approach other aspects of the research. Undertaking this review will positively contribute to the overall output of this work. A lot of alternative methods applied to theories, algorithms and products in language processing have been studied. Most of them have introduced several unique concepts and these ideas will be applied when defining a method.

2.7 Research Methodology

A system development methodology is described as a procedure where a framework will be used in planning, structuring, and taking control of the development of a system (Avison & Fitzgerald, 2003) To successfully study Information System (IS) research methodologies, different approaches must be considered; each with its advantages and disadvantages depending on the focus and application of the research in question. The main reason for choosing IS research methodology is because IS focuses mainly on effective delivery of designs and the use and effects of IT systems in the society and organisations (The Systems Development or Engineering Approach to Research in Information Systems: An Action Research Perspective, 2016). The series of arguments brought up by researchers in past have resulted in the omission of the System Development (SD) methodologies from most classifications of IS research methods. This is mostly due to the assumption that IS research is purely a social science field therefore ignoring its technology aspects. Still, there are a lot of researchers that know that IS methodologies usually involve an unavoidable technical component (Cecez-Kecmanovic 1994, Parker et al 1994; cited by Selecting a development approach, 2016). Thanks to (Nunamaker Jr et al. 1990). The acceptance of (Nunamaker Jr et al. 1990) may have bridged the gap that existed between the social and technological aspects of IS.


98 NUNAMAKER, CHEN, AND PURDIN
System Development Research Process
1. Construct a Conceptual Framework
2. Develop a System Architecture
3. Analyze & Design the System
4. Build the (Prototype) System
5. Observe & Evaluate the System

Research Issues

1.1 • State a meaningful research question
1.2 • Investigate the system functionalities and requirements
1.3 • Understand the system building processes/procedures
1.4 • Study relevant disciplines for new approaches and ideas

2.1 • Develop a unique architecture design for extensibility, modularity, etc.
2.2 • Define functionalities of system components and interrelationships among them

3.1 • Design the database/knowledge base schema and processes to carry out system functions
3.2 • Develop alternative solutions and choose one solution

4.1 • Learn about the concepts, framework, and design through the system building process
4.2 • Gain insight about the problems and the complexity of the system

5.1 • Observe the use of the system by case studies and field studies
5.2 • Evaluate the system by laboratory experiments or field experiments
5.3 • Develop new theories/models based on the observation and experimentation of the system's usage
5.4 • Consolidate experiences learned

Figure 2.7.1 - System Development Research Methodology (Nunamaker Jr et al. 1990)

Due to the general acceptance of this research method, the flow of IS presented in Figure 2.7.1 appears to have all the necessary models to guide this research to the expected end.

2.8 Conclusion

Different natural language literatures, methods, algorithms, APIs have been reviewed. Language processing as a field has had various advances and contribution but from the reviews, there is still a wide gap in the methods used in natural language processing to understand passage and carry out action or answer questions. This review showed models like (Tellex et al., 2011) that produced good result when trained with dataset but very poor result when tested against random data. The issues that surround training systems with specific dataset is that it is time consuming and static. Though in a lot of cases, the accuracy of such trained systems tends to be quite high; the model needs to be retrained on every new domain of questions or instruction types which is time consuming and is unpractical. A more generic approach to enable systems train themselves based on any language dataset and this is one of the problems which this research aims to solve. This approach will not only save time but will equally improve the ability for language processing devices to answer questions correctly.

Preliminary test of existing NLP systems and APIs will be done in the next chapter to practically identify the current progress of applications and their limits as well. Some of the tools that will be tested are Wolfram Alpha, Googles Personal Assistant, Amazon Echo, Siri, IBMs Watson, Cortana etc.

Chapter 3. Research Methodology

The research methodology presents and justifies all steps to be used to meet the aims of this research. Basic language processing tools and APIs will be tested here to identify their limitations or any existing gaps in language processing. This research will be conducted following the Information System (IS) research methodology shown in Figure 2.7.1. Several types of research methodologies have been studied (few of which are listed in Appendix D) to identify the most suitable one for this research.

3.1 Research Method Application

3.1.1 Reviewing Literature

The review of existing literature is necessary to identify existing work already done in this field of study.

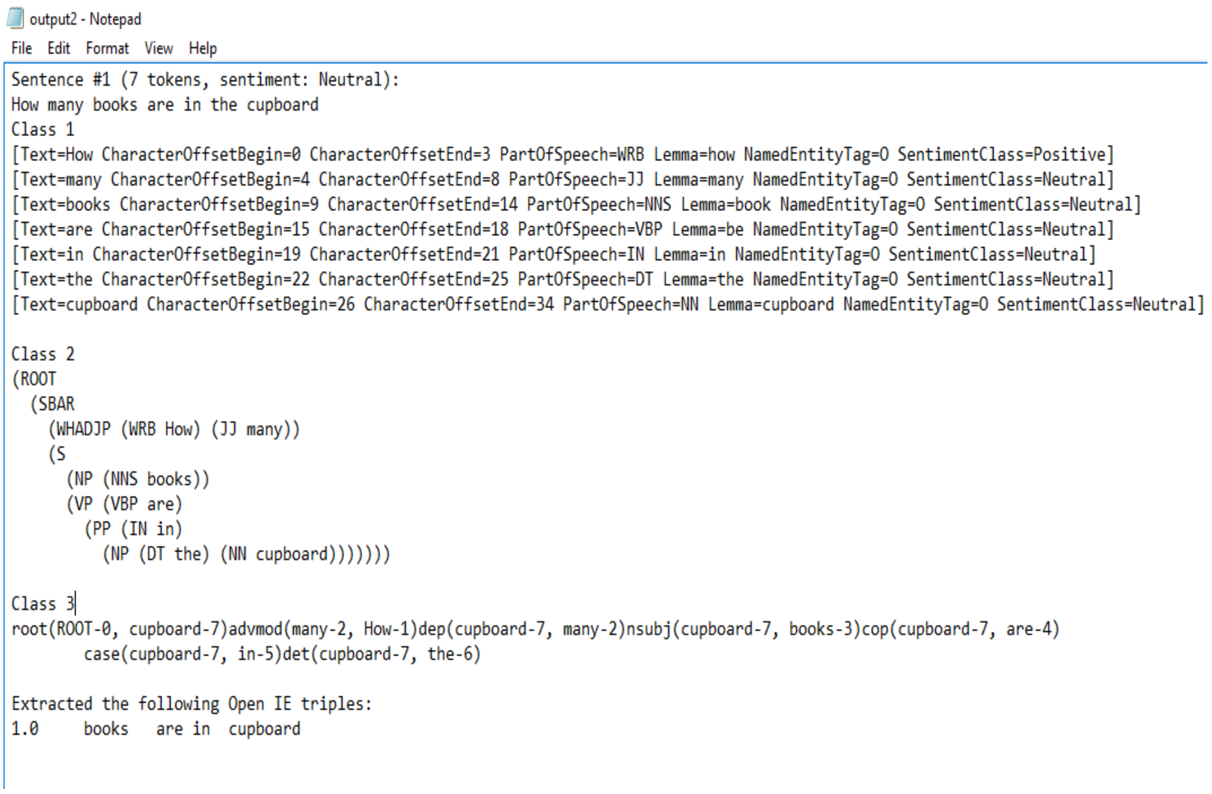
From the literatures reviewed in Chapter 2, several theoretical methods have been studied and from this, the result of the experiments conducted in those reviews have been analysed. From the reviews of the scientific methods done, it shows there is a knowledge gap in the methods currently being used in natural language processing.

3.1.2 Preliminary Test

Conducting preliminary test on existing algorithms and APIs is necessary. This is because; these tests will help to highlight the validity of the concept of theory. The test is aimed at identifying the current level of NLP algorithms and will help in creating a reasonable justification of any recommended solution to meet the aims of the research. The sections below show the types of tests conducted on different systems and APIs.

A. Testing Stanford NLP Tools

Stanford NLP classifier has been mentioned is Chapter 2 and will be tested here to evaluate the classification limitations and strengths. Most of the procedures carried out to test the tool has been done based on trial and error. The test shown in Figure 3.1 displays the result when using the classifier to process a simple sentence "How many Books are in the cupboard".



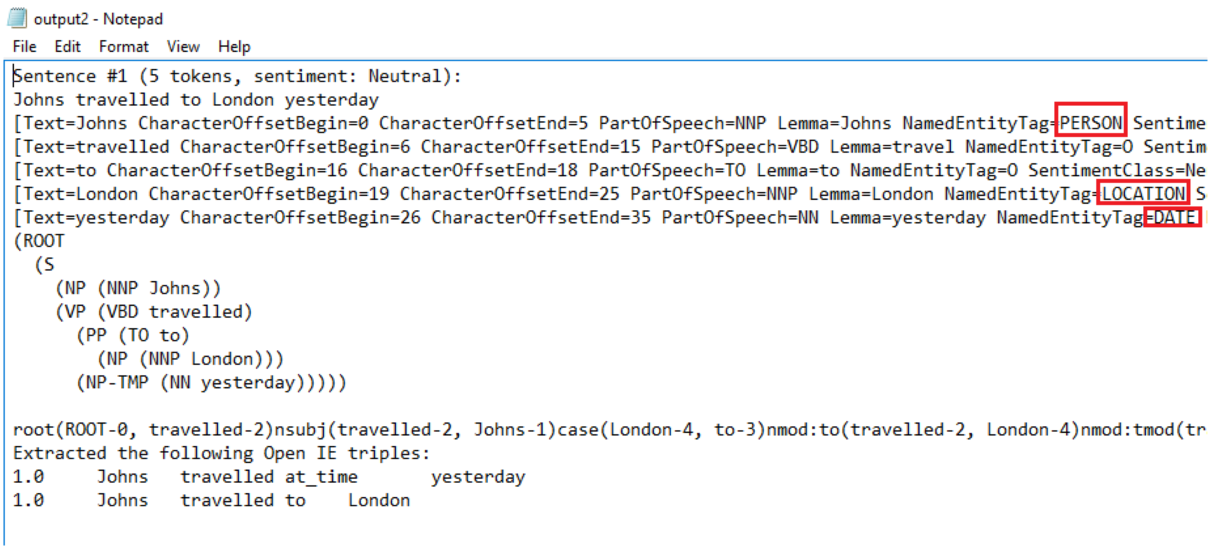
```
Sentence #1 (7 tokens, sentiment: Neutral):
How many books are in the cupboard
Class 1
[Text=How CharacterOffsetBegin=0 CharacterOffsetEnd=3 PartOfSpeech=WRB Lemma=how NamedEntityTag=O SentimentClass=Positive]
[Text=many CharacterOffsetBegin=4 CharacterOffsetEnd=8 PartOfSpeech=JJ Lemma=many NamedEntityTag=O SentimentClass=Neutral]
[Text=books CharacterOffsetBegin=9 CharacterOffsetEnd=14 PartOfSpeech=NNS Lemma=book NamedEntityTag=O SentimentClass=Neutral]
[Text=are CharacterOffsetBegin=15 CharacterOffsetEnd=18 PartOfSpeech=VBP Lemma=be NamedEntityTag=O SentimentClass=Neutral]
[Text=in CharacterOffsetBegin=19 CharacterOffsetEnd=21 PartOfSpeech=IN Lemma=in NamedEntityTag=O SentimentClass=Neutral]
[Text=the CharacterOffsetBegin=22 CharacterOffsetEnd=25 PartOfSpeech=DT Lemma=the NamedEntityTag=O SentimentClass=Neutral]
[Text=cupboard CharacterOffsetBegin=26 CharacterOffsetEnd=34 PartOfSpeech=NN Lemma=cupboard NamedEntityTag=O SentimentClass=Neutral]

Class 2
(ROOT
  (SBAR
    (WHADJP (WRB How) (JJ many))
    (S
      (NP (NNS books))
      (VP (VBP are)
        (PP (IN in)
          (NP (DT the) (NN cupboard)))))))

Class 3
root(ROOT-0, cupboard-7)advmod(many-2, How-1)dep(cupboard-7, many-2)nsubj(cupboard-7, books-3)cop(cupboard-7, are-4)
        case(cupboard-7, in-5)det(cupboard-7, the-6)

Extracted the following Open IE triples:
1.0     books   are in  cupboard
```

Figure 3.1 - Showing the output of the Stanford sentence breakdown

The sentence "How many books are in the cupboard" in Figure 3.1 is broken down into several individual entities. Three classes have been identified and labelled as Class 1, Class 2, Class 3.

Class 1 consists of the properties that were added to the Stanford pipeline of the written code (explained in Appendix C) before compilation. For example, POS = Part of Speech, Lemma = the actual word, NER = Named Entity Tag or Recogniser and the sentiment class (sentiment analysis class is used to deduce people's attitude and feelings towards certain topics like movie, music, games, political outcomes, etc. Sentiment class also called opinion extraction). In class

1, several of these properties have been identified by the library but looking at the Named Entity Tag, it can be observed that is = 0. Named Entity Recogniser (NER) maps the words of a sentence to Names, Places, dates etc. Figure 3.2 below shows how the NER maps John to a "Person" and London to a "Location" and yesterday to a "Date".



```
Sentence #1 (5 tokens, sentiment: Neutral):
Johns travelled to London yesterday
[Text=Johns CharacterOffsetBegin=0 CharacterOffsetEnd=5 PartOfSpeech=NNP Lemma=Johns NamedEntityTag=PERSON Sentime
[Text=travelled CharacterOffsetBegin=6 CharacterOffsetEnd=15 PartOfSpeech=VBD Lemma=travel NamedEntityTag=O Sentim
[Text=to CharacterOffsetBegin=16 CharacterOffsetEnd=18 PartOfSpeech=TO Lemma=to NamedEntityTag=O SentimentClass=Ne
[Text=London CharacterOffsetBegin=19 CharacterOffsetEnd=25 PartOfSpeech=NNP Lemma=London NamedEntityTag=LOCATION S
[Text=yesterday CharacterOffsetBegin=26 CharacterOffsetEnd=35 PartOfSpeech=NN Lemma=yesterday NamedEntityTag=DATE
(ROOT
  (S
    (NP (NNP Johns))
    (VP (VBD travelled)
      (PP (TO to)
        (NP (NNP London)))
      (NP-TMP (NN yesterday)))))

root(ROOT-0, travelled-2)nsubj(travelled-2, Johns-1)case(London-4, to-3)nmod:to(travelled-2, London-4)nmod:tmod(tr
Extracted the following Open IE triples:
1.0     Johns   travelled at_time       yesterday
1.0     Johns   travelled to    London
```

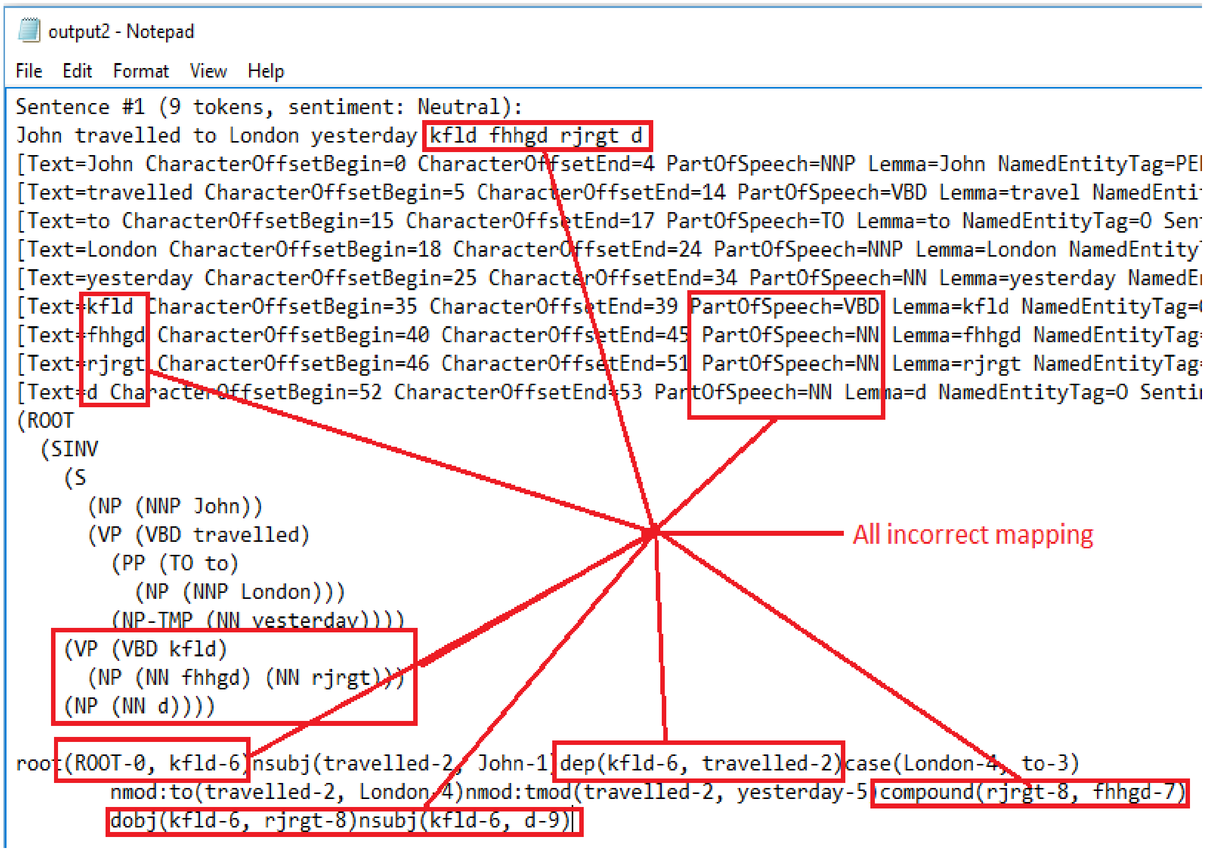
Figure 3.2 - Showing Named Entity Recogniser mapping nouns to Entities

Still in Figure 3.1, class 2 concentrates on the simple grammatical relations (50 grammatical relations explained in section 2.2) of the sentence e.g. John is an NNP. Class 3 shows either the dependencies or relations which the words have with each other and gives a more detailed grammatical relation. Class 3 represents John as an "nsubj" which means, John is the subject of the sentence and it also shows that John is directly related to the word "travelled".

Limitations of CoreNLP library.

1. It takes a lot of time (computationally) for the API to be compiled into a software for use.
2. It was observed that Stanford NLP cannot identify non-existent English words. This is a significant limitation. This classifier assigns different parts of speech to incorrect

sentences and describes a grammatical relation of incorrect word to other semantics of the sentence as shown in Figure 3.3 below



```
Sentence #1 (9 tokens, sentiment: Neutral):
John travelled to London yesterday kfld fhhgd rjrgt d
[Text=John CharacterOffsetBegin=0 CharacterOffsetEnd=4 PartOfSpeech=NNP Lemma=John NamedEntityTag=PE
[Text=travelled CharacterOffsetBegin=5 CharacterOffsetEnd=14 PartOfSpeech=VBD Lemma=travel NamedEnti
[Text=to CharacterOffsetBegin=15 CharacterOffsetEnd=17 PartOfSpeech=TO Lemma=to NamedEntityTag=O Sen
[Text=London CharacterOffsetBegin=18 CharacterOffsetEnd=24 PartOfSpeech=NNP Lemma=London NamedEntity
[Text=yesterday CharacterOffsetBegin=25 CharacterOffsetEnd=34 PartOfSpeech=NN Lemma=yesterday NamedE
[Text=kfld CharacterOffsetBegin=35 CharacterOffsetEnd=39 PartOfSpeech=VBD Lemma=kfld NamedEntityTag=
[Text=fhhgd CharacterOffsetBegin=40 CharacterOffsetEnd=45 PartOfSpeech=NN Lemma=fhhgd NamedEntityTag
[Text=rjrgt CharacterOffsetBegin=46 CharacterOffsetEnd=51 PartOfSpeech=NN Lemma=rjrgt NamedEntityTag
[Text=d CharacterOffsetBegin=52 CharacterOffsetEnd=53 PartOfSpeech=NN Lemma=d NamedEntityTag=O Senti
(ROOT
  (SINV
    (S
      (NP (NNP John))
      (VP (VBD travelled)
        (PP (TO to)
          (NP (NNP London)))
        (NP-TMP (NN yesterday))))
    (VP (VBD kfld)
      (NP (NN fhhgd) (NN rjrgt)))
    (NP (NN d))))

root(ROOT-0, kfld-6)nsubj(travelled-2, John-1)dep(kfld-6, travelled-2)case(London-4, to-3)
        nmod:to(travelled-2, London-4)nmod:tmod(travelled-2, yesterday-5)compound(rjrgt-8, fhhgd-7)
        dobj(kfld-6, rjrgt-8)nsubj(kfld-6, d-9)
```

Figure 3.3 - Showing a major error noticed with CoreNLP.

There is an issue surrounding the definition of words. Is a person's name such as "Feyiseye" or creations made from Cartoons like e.g. "Godzilla"? a word? This may be a good reason why the test results shown in Figure 2.13 have mostly been represented as nouns and one a verb. Possibly the CoreNLP library has assumed these are names of some sort.

A word may appear in several dictionaries making it undoubtedly a word, but the word not appearing in a dictionary cannot necessarily prove that the word is not in existence; language evolves constantly. From all indications documented in 2a and 2b, 2a should be the most appropriate solution but would require suggestions from the stakeholders to finalise.

B. Testing IBM Watsons API

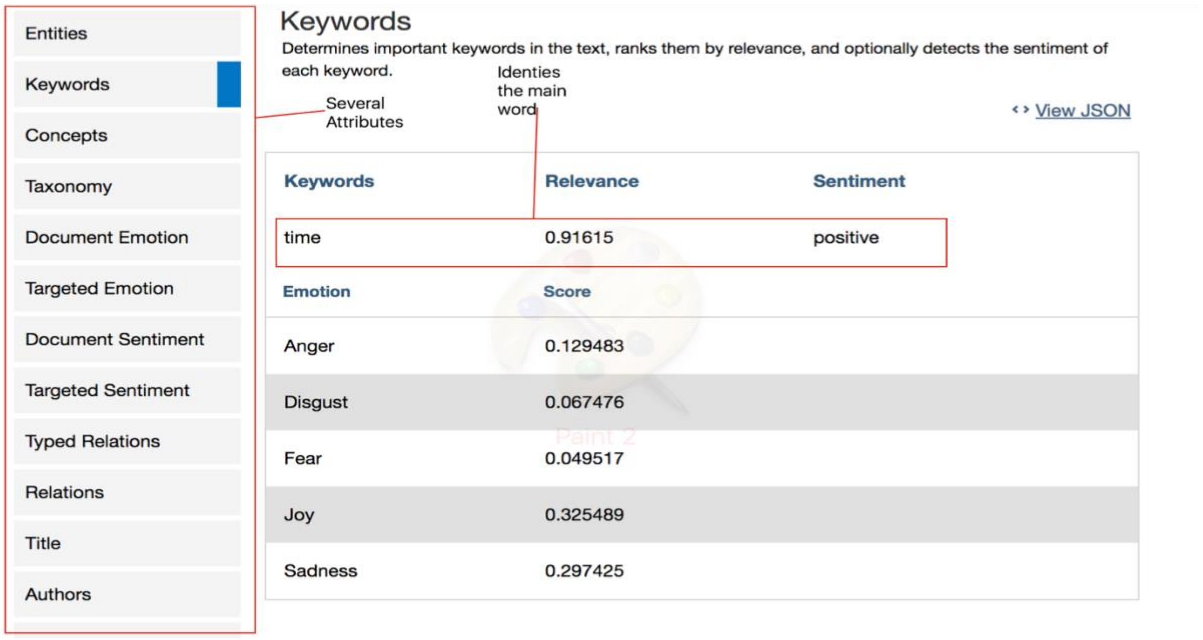
From (Engadget, 2011), Watson demonstrated its ability to provide answers to questions asked thereby winning the Jeopardy competition. So therefore, the demonstration to be carried out now involves asking Watson simple questions. Two simple questions are used in interrogating Watson. The questions have been randomly picked and are simple questions. They are only meant to test the current capacity of the API on its process of answering questions.

1. What is the time?
2. Who is the prime-minister of England?

(AlchemyLanguage demo, 2017) is a web application built on IBM's Watson API, which can be used to directly test Watson. The 2 questions above will be tested on Watson using (AlchemyLanguage demo, 2017) interface.

Question 1: What is the time

In Figure 3.4 below, different attributes relating to the question are listed on the left-hand side of the image, such as entities, keywords, concepts etc. Each of these attributes show different relations of the question. On the other side of the panel the image shows the full elements of each attribute when clicked.



Keywords

Determines important keywords in the text, ranks them by relevance, and optionally detects the sentiment of each keyword.

Keywords	Relevance	Sentiment
time	0.91615	positive

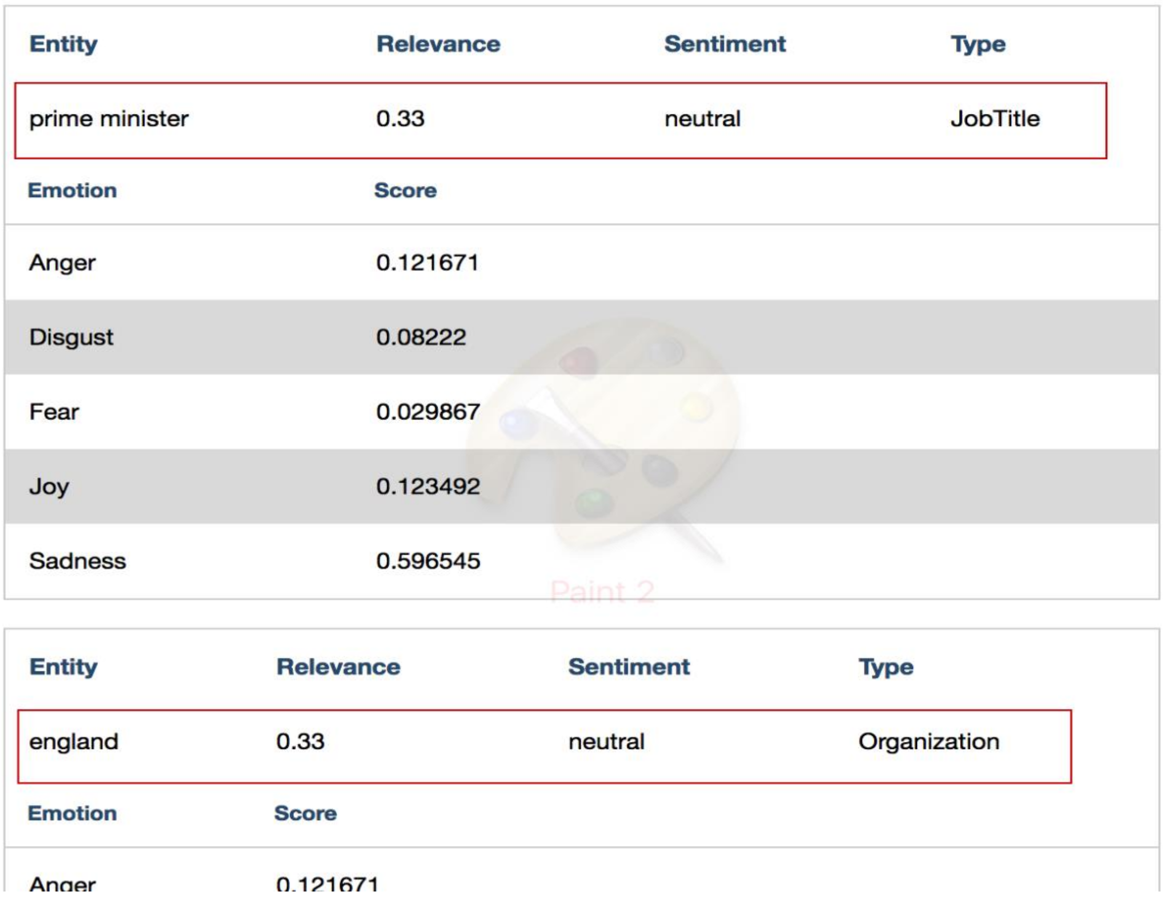
Emotion	Score
Anger	0.129483
Disgust	0.067476
Fear	0.049517
Joy	0.325489
Sadness	0.297425

Figure 3.4 - Showing relevant keywords to the question

Watson correctly identifies what the question is about “The Time” but non-of the attributes listed on the left have an element that provide a direct answer to the question. The attributes only showed different classification and breakdown of the question’s component but no pointer to direct answers or suggestions.

Question 2: Who is the Prime minister of England?

The second question is about the prime minister of England. Watson in Figure 3.5 produced similar answer to the one given in Figure 3.4. This answer correctly identified the two main constructions of this question (prime minster and country). Watson’s construction elements and classifications can be used to simplify the creation of bots or algorithms that can find the right answer to questions. The reason being, the two important components (prime minster and England) have been provided by Watson.



Entity	Relevance	Sentiment	Type
prime minister	0.33	neutral	JobTitle

Emotion	Score
Anger	0.121671
Disgust	0.08222
Fear	0.029867
Joy	0.123492
Sadness	0.596545

Entity	Relevance	Sentiment	Type
england	0.33	neutral	Organization

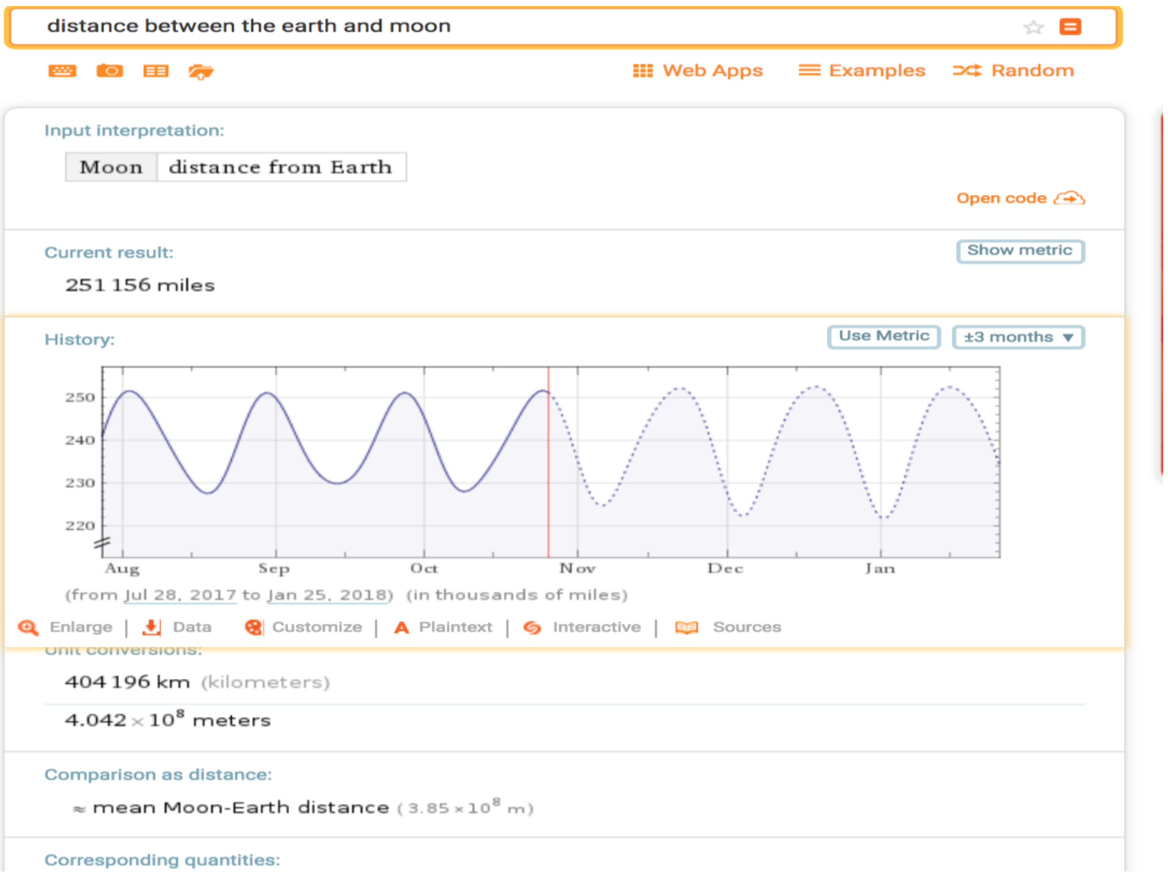
Emotion	Score
Anger	0.121671

Figure 3.5 - Testing the second question on Watson

According to (IBM, 2017) and the tests carried out, Watson is a very intelligent system and can be very helpful in various NLP sections. However anyone using Watson's API still needs to create a third-party interactive tool to make use of Watson's classification results. On its own, Watson can be termed a natural language classifier.

C. Testing Wolfram Alpha

Following the discussion on Wolfram in Section 2.6, Figure 3.6 and 3.7 show examples on using Wolfram Alfa for testing purposes. Figure 3.6 demonstrates Wolfram Alphas computational algorithm skill when tested, while Figure 3.7 tests its language processing skill.


distance between the earth and moon

Web Apps Examples Random

Input interpretation:

Moon | distance from Earth

Open code

Current result:

251 156 miles

Show metric

History:

Use Metric ±3 months

(from Jul 28, 2017 to Jan 25, 2018) (in thousands of miles)

Enlarge | Data | Customize | Plaintext | Interactive | Sources

Unit conversions:

404 196 km (kilometers)

4.042×10^8 meters

Comparison as distance:

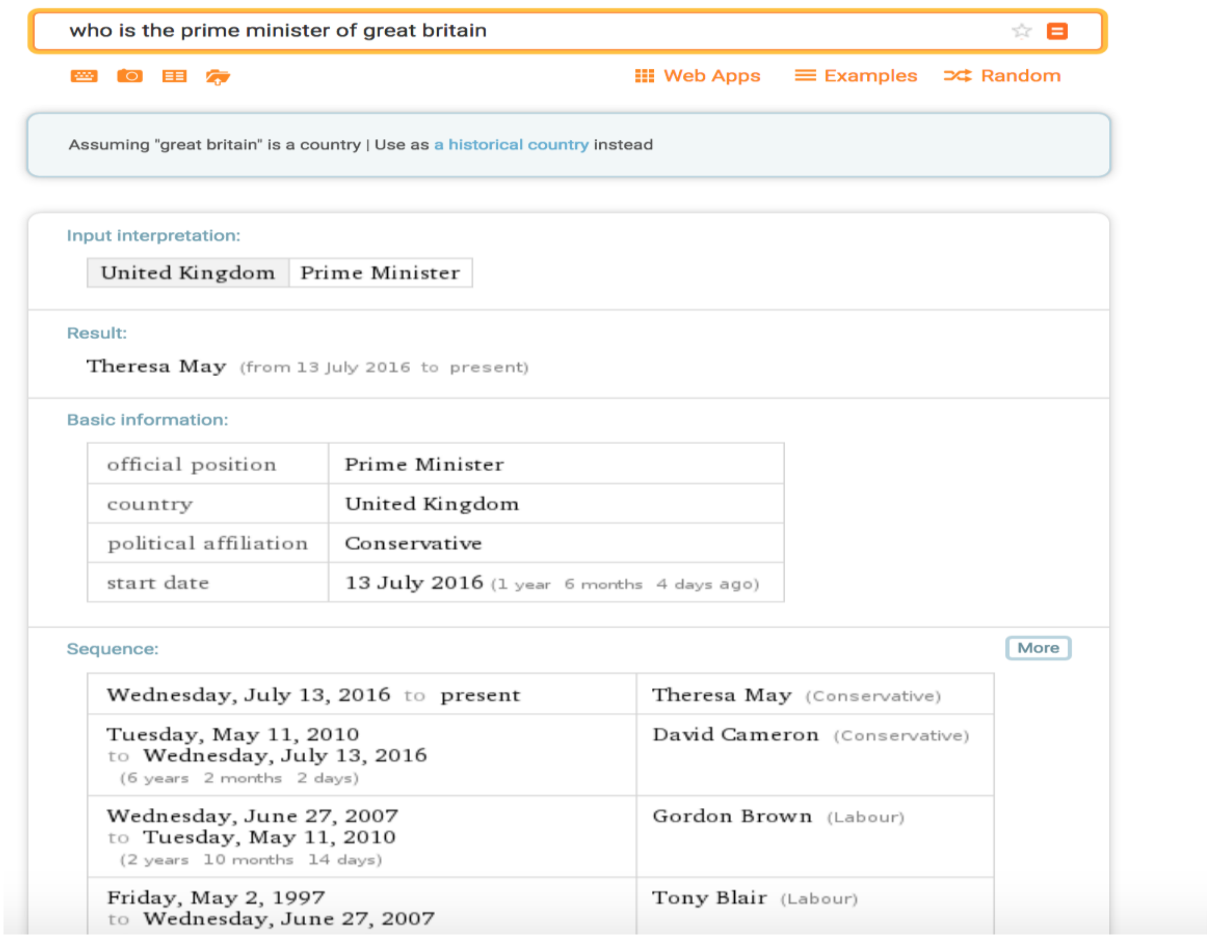
≈ mean Moon-Earth distance (3.85×10^8 m)

Corresponding quantities:

Figure 3.6 - Shows the correct distance between the earth and the moon

Figure 3.6 shows the exact distance between the earth and the moon in a single answer which presents direct, detailed and accurate information on the question asked. For mathematical computation using Wolfram, this is mostly the case as these computations have a fixed formula. Figure 3.6 also shows the main keywords (Moon, distance from the earth) referred to as Input Interpretation which is like the classifications acquired when tests were carried out using IBMs Watson.

Next, Figure 3.7 below shows how Wolfram analyses Natural language sentences.


who is the prime minister of great britain

Web Apps Examples Random

Assuming "great britain" is a country | Use as a historical country instead

Input interpretation:

United Kingdom	Prime Minister

Result:

Theresa May (from 13 July 2016 to present)

Basic information:

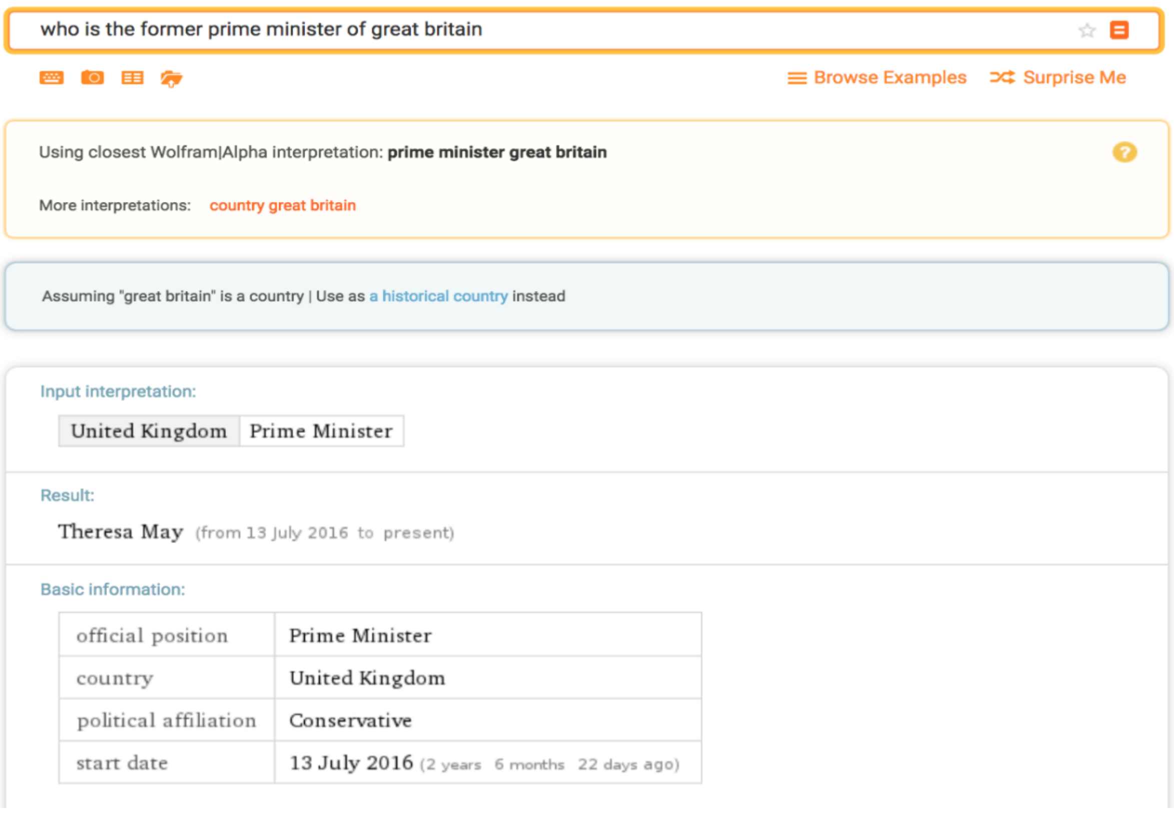
official position	Prime Minister
country	United Kingdom
political affiliation	Conservative
start date	13 July 2016 (1 year 6 months 4 days ago)

Sequence: More

Wednesday, July 13, 2016 to present	Theresa May (Conservative)
Tuesday, May 11, 2010 to Wednesday, July 13, 2016 (6 years 2 months 2 days)	David Cameron (Conservative)
Wednesday, June 27, 2007 to Tuesday, May 11, 2010 (2 years 10 months 14 days)	Gordon Brown (Labour)
Friday, May 2, 1997 to Wednesday, June 27, 2007	Tony Blair (Labour)

Figure 3.7 - Wolfram processing normal sentences

Wolfram Alpha correctly retrieves the answer as shown in Figure 3.7. Just below the correct answer are listed the previous prime ministers and the periods they held office for. Figure 3.8 below shows a different scenario of how adding “former” to the sentence affects the entire result. Adding “former” before prime minister still produces the same result. *Wolfram Alpha* ignored the added input to the sentence.



who is the former prime minister of great britain

Browse Examples Surprise Me

Using closest Wolfram|Alpha interpretation: **prime minister great britain**

More interpretations: country great britain

Assuming "great britain" is a country | Use as a historical country instead

Input interpretation:

United Kingdom	Prime Minister

Result:

Theresa May (from 13 July 2016 to present)

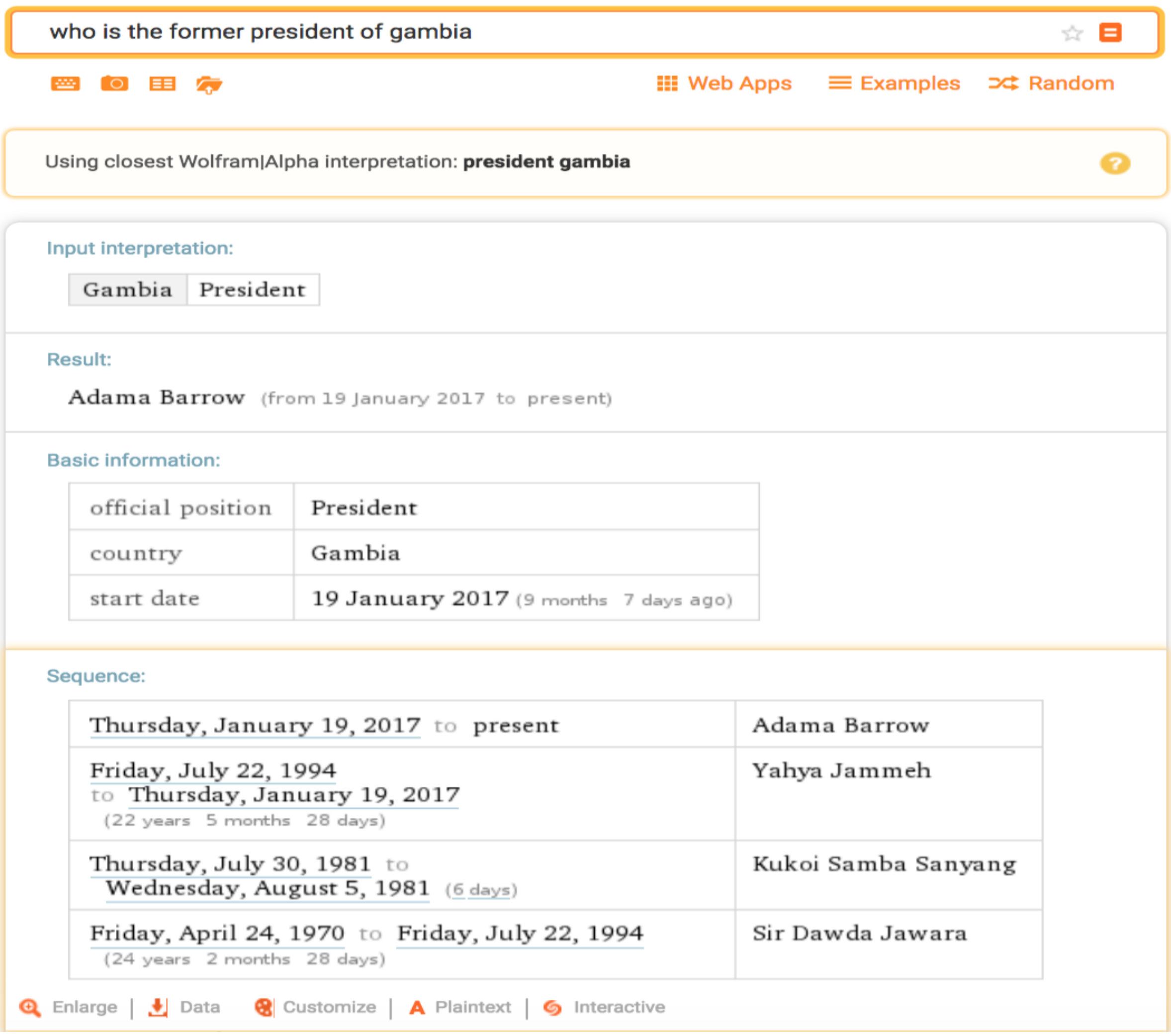
Basic information:

official position	Prime Minister
country	United Kingdom
political affiliation	Conservative
start date	13 July 2016 (2 years 6 months 22 days ago)

Figure 3.8 - Wolfram processing normal sentences 2

Figure 3.8 produced a similar answer as seen in Figure 3.7 due to Wolfram's way of relying on key words such as United Kingdom, Prime minister, and Great Britain.

Figure 3.9 shows another example of Wolfram interpreting a similar passage on a president.


who is the former president of gambia

Using closest Wolfram|Alpha interpretation: **president gambia**

Input interpretation:

Gambia	President

Result:

Adama Barrow (from 19 January 2017 to present)

Basic information:

official position	President
country	Gambia
start date	19 January 2017 (9 months 7 days ago)

Sequence:

Thursday, January 19, 2017 to present	Adama Barrow
Friday, July 22, 1994 to Thursday, January 19, 2017 (22 years 5 months 28 days)	Yahya Jammeh
Thursday, July 30, 1981 to Wednesday, August 5, 1981 (6 days)	Kukoi Samba Sanyang
Friday, April 24, 1970 to Friday, July 22, 1994 (24 years 2 months 28 days)	Sir Dawda Jawara

Figure 3.9 - Interpreting text with Wolfram

The input interpretation (keyword) for this sentence is Gambia and President. This question was asked to test if Wolfram will give any priority to the word "**former** president". If Wolfram does not consider the word "former" or other non-keywords in a sentence, there is a huge chance that the right answer will not be obtained. As expected, Wolfram pulled all result matching the keyword (Gambia and president) from its database thereby showing all presidents of Gambia that exist in its database. This pattern in its result is expected, for it has been observed to be a common occurrence to find such relations in a system output if it relies on keyword matching techniques. (MORI, OHTA, FUJIHATA & KUMON, 2001)

D. Testing Siri, Cortana, Google Voice and Amazon Echo

To carry out successful test on these different advanced intelligent systems, a collection of random sentences have been gathered. The responses they give as output will be noted down for study purposes.

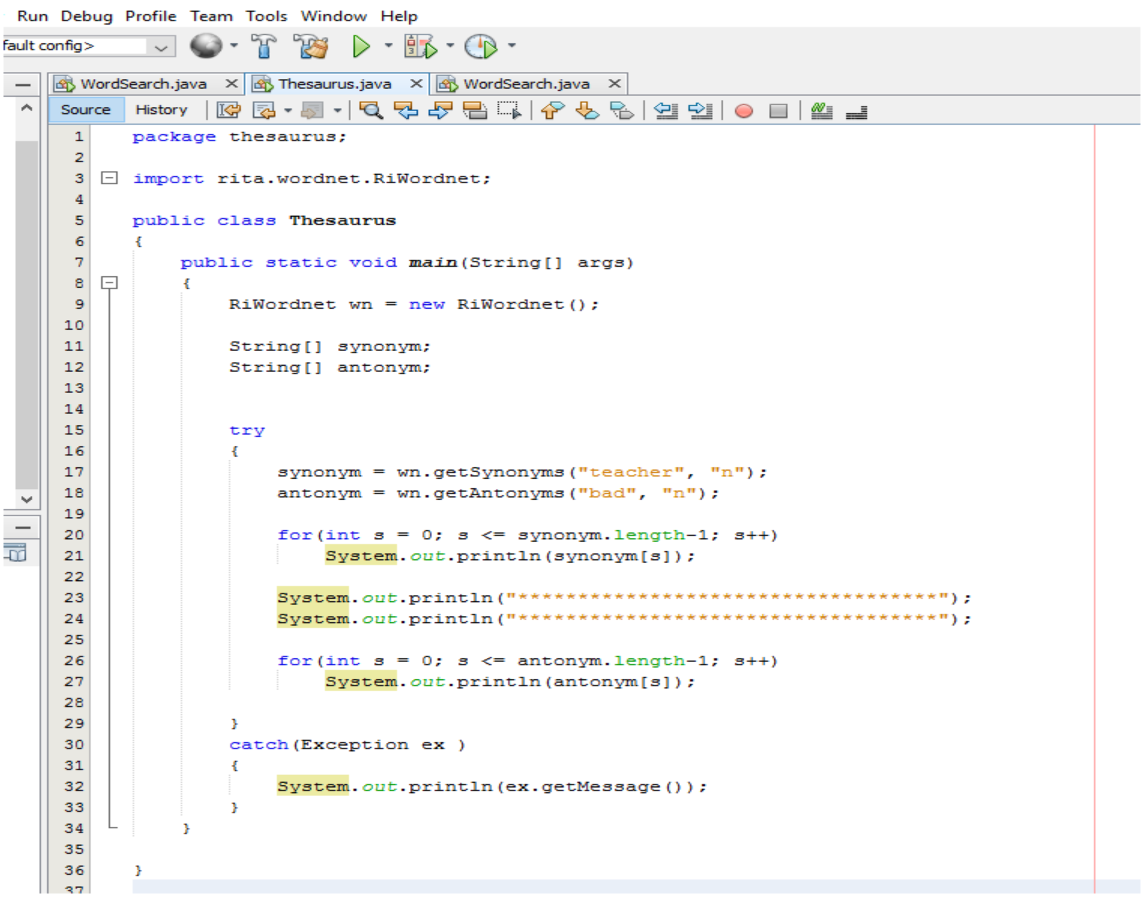
This black box test is carried out to examine what kind of sentence these intelligent systems understand and the responses they give. Random questions are compared in Table F1 in Appendix F.

Table 3.1 - Results for preliminary test on personal assistants

Devices	**Correct Answers**	**Wrong Answers**
Siri	Q1, Q2, Q4	Q3, Q5, Q6
Google Voice	Q1, Q2, Q4	Q3, Q5, Q6
Amazon	Q1, Q2, Q4	Q3, Q5, Q6
Cortana	Q1, Q2, Q4	Q3, Q5, Q6

E. Testing WordNet

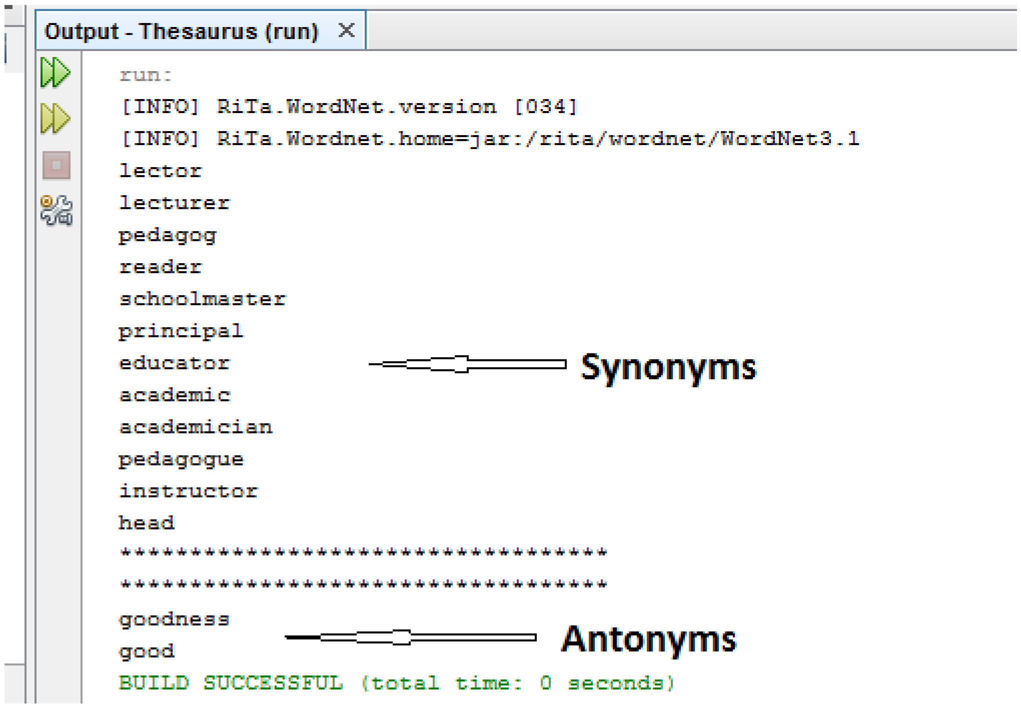
WordNet is a large record of English words. Some parts of speech like nouns, adjectives, adverbs and verbs are grouped into collections of cognitive synonyms. Java WordNet Library is also publicly and freely available for download. JWNL has numerous methods or functions created to aid the research in NLP.



```java
package thesaurus;

import rita.wordnet.RiWordnet;

public class Thesaurus
{
    public static void main(String[] args)
    {
        RiWordnet wn = new RiWordnet();

        String[] synonym;
        String[] antonym;


        try
        {
            synonym = wn.getSynonyms("teacher", "n");
            antonym = wn.getAntonyms("bad", "n");

            for(int s = 0; s <= synonym.length-1; s++)
                System.out.println(synonym[s]);

            System.out.println("************************************");
            System.out.println("************************************");

            for(int s = 0; s <= antonym.length-1; s++)
                System.out.println(antonym[s]);

        }
        catch(Exception ex )
        {
            System.out.println(ex.getMessage());
        }
    }

}
```

Figure 3.10 - Showing WordNet Library for Synonyms and Antonyms

Figure 3.10 shows the WordNet library used to find the Synonyms of the word “Teacher” and the Antonyms of the word “Bad”. The resulting answers from the two variables are passed into an array and then printed out to the screen as shown below in Figure 3.11.



```
run:
[INFO] RiTa.WordNet.version [034]
[INFO] RiTa.Wordnet.home=jar:/rita/wordnet/WordNet3.1
lector
lecturer
pedagog
reader
schoolmaster
principal
educator
academic
academician
pedagogue
instructor
head
***********************************
***********************************
goodness
good
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 3.11 - Answers to Synonyms and Antonyms

As labelled in Figure 3.11, the synonyms and antonyms produced by the software are accurate.

3.1.3 Concept of Theory

From the reviews done and preliminary tests conducted on various systems, the outcome shows that there is the need to improve the algorithm training time and language processing logic for NLP systems. This established the reason to create a new algorithm that will be able to train itself from language datasets saving a considerable amount of time taken when training algorithms in different question domains. The new algorithm will also use a different approach for answering questions.

Hypothesis

The proposed model for this algorithm will explore mainly, the *sentence dependency or relation structure* for training itself on passages and representation of knowledge while combining the same method for answer retrieval. keen consideration will be given to using word dependency and sentence relationships in this algorithm because, it has been observed that adding extra information e.g. former, yesterday etc. to a sentence usually negatively affects the result obtained. The assumption here is, using the same model for training, knowledge representation and answer retrieval, this will produce a higher frequency of correct results in several domains of question. This will also improve the current algorithm training time.

3.1.4 Experimentation and Analysis

After the algorithm is finalised, a prototype will be developed to test the performance of the algorithm. 3 major types of tests will be conducted on the system.

Test 1 (Self-Test)

For the first test, the system will be tested against itself. A passage of the first test will be retrieved from (Bubien, 2017). This passage is completely independent of this research. The passage will be shared to different members of the PARK group and random questions will be generated from them while abiding by the research aims.

Test 2 (Public Domain Test)

Random General Knowledge questions chosen by some of the universities PARK team members from the internet will be used for experimenting to avoid bias. This test will be conducted against the major systems in the public domain (Siri, Cortana, Google Personal Assistant, Amazon Echo) and the prototype. All results obtained during the test will be recorded for analysis.

Test 3 (Closed Domain Test)

Like the second test, the third test type will target systems that exist in closed domains. The target systems here are closed search engines or bots particularly trained for answering direct questions. Test data will be obtained in the same way as test 2. The results obtained from test 2 and test 3 will be collated in the same table to improve readability of this thesis.

General knowledge questions will be obtained from (General Knowledge Quiz, 2018) and (General Knowledge Quizzes, 2018). These two sources of questions are completely independent of this research to avoid any form of bias with the questions.

3.15 Experiment Exemptions

In Chapter 5, Chat bot's will be exempted from the experiment phase due to several inconsistencies observed during the preliminary test phase. Figure 3.12 and 3.13 below show a few examples of the inconsistent responses from A.L.I.C.E Chat bot when asked basic questions.

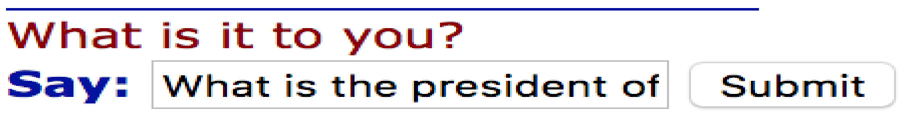

What is it to you?
Say: What is the president of

Figure 3.12 – Inconsistent answer when asked who the president of The United States of America is. (Alice the Chatterbot, 2017).

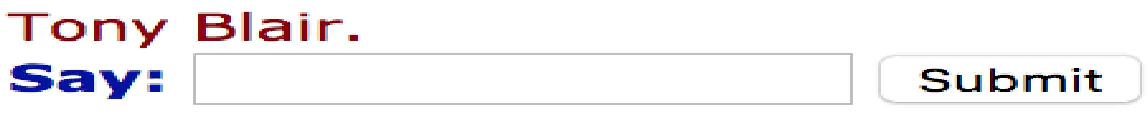

Tony Blair.
Say:

Figure 3.13 – Pandora Bot showing outdated answer when asked who the prime minister of United Kingdom is (Alice the Chatterbot, 2017).

From the study done in Chapter 2, most Chat bots are built for smaller scenarios for example, a taxi Chat bot or a hotel reservation Chat bot. This research targets a wider generic scope where natural language adaptability is key.

The prototype will learn from all the passages that questions are picked from enabling the prototype to have enough information to answer questions. Overly sufficient data will also increase the chances of the prototype making mistakes which is good in this experimental condition. The outcome of the results will be analysed in Chapter 6 (experiment analysis). After experimenting, the accuracy and error of all results will be analysed using statistical confusion matrix (true positive and false positives) and based on the analysis, the outcome will be evaluated accurately. These questions will then be tested on the system and the results recorded. Appendix G will show some of the experiment evidence.

This chapter has explained the process that will be used in conducting this research and given evidence to why a new algorithm that can meet the aims of this research is needed.

The next chapter, Chapter 4 will fully explain the methods and concepts that will be used by the algorithm for training itself, representing its knowledge and answering questions.

Chapter 4. Design and Implementation

Chapter 4 explains how the algorithms built in this research are modelled. The need for this new algorithm has already been covered in chapter 3.

This chapter will also identify the unique contributions of this research while discussing the flow of the algorithm.

4.1 Algorithm Development Process

This section explains how the algorithm development process has been developed and describes the logic used by the algorithm to find correct answers to a question.

When a sentence or passage is passed on to the algorithm for analysis, the first position taken by the system is to understand what the passage is about.

Using the help of Stanford NLP, the passage is classified. The classified data returned from Stanford NLP produces information about the passage: POS, Lemma, NER and the sentence dependencies. This data returned from the classifier on its own is not usable by the algorithm to answer questions from a passage. So, one of the main contributions of this research is to represent the data as useful and retrievable knowledge which helps to make the language querying easier. A knowledge base was thereby created considering the dependencies that existed within the passage itself; which is an important part of question answering as currently hypothesised in this research.

4.1.1 Knowledge Base

This relational database is created to persistently represent the knowledge learnt by the algorithm while in use. This database schema shown in Figure 4.1 below is one of the core contributions of this research. This is a core contribution as the knowledge base model is uniquely derived as a model for this thesis. This exact model has not been used before and was produced as a result of analysis and experimentation done on other models building on their drawbacks as discussed in Chapter 2 and 3. This model uses “word dependency” schema structure which directly integrates with the algorithm.

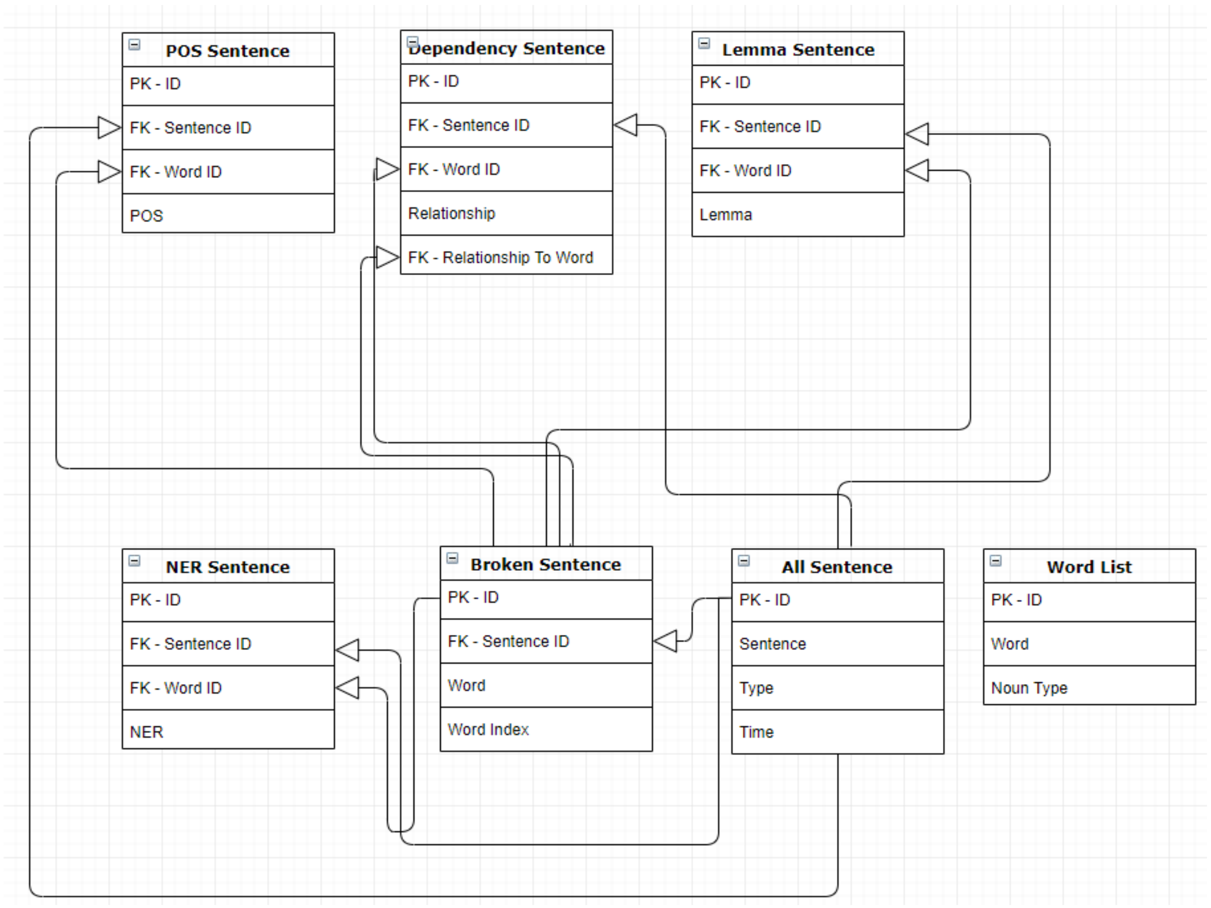


Figure 4.1 - Showing entity relation (ER) diagram

Figure 4.1 shows the relationship each entity has with other entities in the database. This figure also identifies the attributes that relate to other entities and attributes in the system respectively. All the entities and their attributes will be explained fully below starting with the entity that exists on its own called **Word List**.

Describing the Entities and how they are useful to the research

- Word List

Word list contains 3 attributes which are, The ID, Word and Noun Type. This entity contains 235,874 common and not so common English words (singular form) existing in a dictionary and is specifically associating them to their noun type. e.g. THING for CAR, PERSON for HE, ANIMAL for GOAT. This entity will be updated over time per use as more words will be added to it or new noun types defined. This entity is used by the algorithm to filter out answers that do not correlate to an expected noun type. The initial data used for populating the word list table was pulled from (Princeton University, 2019) WordNet library and further fine-tuned with Thesaurus noun type association (Thesaurus, 2016).

- All Sentences

  Each sentence from a passage is saved in this entity. This helps the algorithm to easily retrieve sentences with the correct answer. **The sentence** attribute in this entity holds each sentence of the users passage of text. The primary key of this table is referenced by all the other tables in the schema.

- Broken Sentence

  This holds the individual words contained in a sentence saved in the **All sentences** table. The unique identification number is created for each word of a sentence when broken down and this unique id stands as the entities primary key. The **Broken Sentence** is referenced in several tables such as POS, Lemma, NER and Dependency sentences as a foreign key. It is used by the algorithm to analyse how important a word is in a whole sentence with respect to the question asked.

- NER, POS and Lemma Sentences

  These three entities are combined as one because they have several similarities in their structure. The POS, NER and Lemma is found for every word of a sentence. The algorithm uses all three fields for logic mapping.

- Dependency Sentences

  The **Dependency Sentences** table stores information on how words in a sentence relate with each other in the same and other sentences. The three steps listed below explain how the algorithm saves data into this entity:

  1. A Word ID (according to its index in the broken sentence table) is identified and selected.
  2. The type of relationship that exist; between the selected word and other words in the sentence is identified.
  3. The other words which the first word is related to are saved.

  This entity has a very high usage for logic mapping by the algorithm.

4.1.2 Answer retrieval Algorithm

When a question is asked, the algorithm initially checks if the question matches the research expectations (questions with who or whom as explained in Chapter 1).

1. The question asked by the user is again classified using Stanford NLP to get its individual properties. The breakdown of the question will have different constituents like POS, sentence dependencies, lemmas and NER.
2. The question constituents will be used by the algorithm to individually query the database for related sentences. The Root (main word) of the question which is one of the constituents produced by the classifier is also added amongst the query parameter. All these components used in the query are considered as key components by the algorithm. The lemma is equally important in the knowledge base search to identify words in their base verb form which is likely to have changed depending on how the user asked a question. E.g. For a sentence like "John **eats** rice everyday", both Eats, and its lemma Eat will be considered in the query. If not, for a question that looks like

this, “What does John **eat** every morning”, without using the lemma as part of the query, the algorithm would search for **Eat** only and not **Eats** meaning that it would possibly not return the correct result. Section 4.2 shows the order of priority given to POS in sentence. This priority list has resulted from close study of classifiers as discussed in Chapter 2.

3. From the matches returned from the knowledge base, as possible sentences with the correct answer to a question, the algorithm further uses the PB System to limit the sentences to just one with the highest point. The PBS is explained in section 4.2.1
4. When the PBS returns the sentence with the highest point, the expected return type (a Noun or Pronoun) is used to select the possible list of answers from the sentence; eliminating all results that do not match the expected return type. The expected return type for questions about **Who** or **Whom** is usually going to be a Noun or Pronoun. This does not mean that all nouns or pronouns in the selected sentence will be the answer to the question. A sentence containing the correct answer to a question could have several words which may be nouns or pronouns or even multiple actors. Therefore, further steps are needed to derive the correct answer. If there is only one possible match in a sentence, then that match is returned to the user as the answer.
5. This step will only be triggered when there are two or more possible answer. The algorithm uses a self-generated point-based system (PBS) to finally grade the results and select the right answer using **dependencies** as the main selection criteria for this final step. Section 4.2.2 explains the criteria the PBS uses to pick the final answer.
6. From point 2 mentioned above, if no data is returned during the first query search, extra steps will be taken which are: When no result is produced after the first query search, the algorithm uses the JWNL library to alter the POS and Lemma to get their popular

synonyms. These synonyms will then be used to repeat the same step as explain from step 2 to 5.

7. In a case where the algorithm cannot match any data from the knowledge base when querying, it will inform the user that an answer could not be generated. Also, if the PBS finds more than one possible answer, then, there is a tie and the algorithm will show all possible answers to the user.

4.2 Point Based System.

The POS priority list is part of the PBS and this list is shown in numeric points below. This priority list is one of the processes the algorithm follows to order the data as priorities.

1. Noun
2. Pronoun
3. Verb
4. Adverb
5. Adjective
6. Preposition
7. Conjunction
8. Determinant
9. Interjection

The ordering of this list has been uniquely created following careful study of the (de Marneffe & Manning, 2008). Roles played by different parts of speech in sentences were studied and this order of priority was thought to be sufficient to meet the research aims.

4.2.1 Picking the right sentence

The query data is run considering the POS matching system listed above where the data output is graded in priority according to its position in this list from 1-9; 1 being the highest priority (9 points) and 9 being the lowest priority (1 point). This list shown above is just for the POS matching and it was listed in this order after several reviews done in Chapter 2 and examining different keywords that may relate to questions about Who.

The other parts of the PBS used by the algorithm are the root, NER and direct quote grading. Amongst several options to filter appropriate sentences from the returned queried data, the algorithm checks which sentence has the questions root word in them and assigns 30 point to that question. If there is a “” (direct quote) in the question which is equally found in any of the returned sentences, then that sentence is given 20 points. If there is a named entity (NER) in the question and any possible match is also found in the sentence, then that sentence is also assigned 20 points. 20 points is also assigned for any match in the first five POS listed above.

After the sentence grading is complete, collation of scores is done. The question with the highest score is selected for further processing.

4.2.2 Picking the right answer

The algorithm relies totally on dependencies to pick the right answer from the possible list of answers. The algorithm firstly picks all nouns or pronouns from the sentence. If only one noun or pronoun exist, then that is presented as the correct answer. In the situation where more than one now exists, the algorithm checks the dependencies of all possibilities. If a single possibility is directly related to the root of a question, that sentence is chosen as the answer. The POS list order is equally used for the dependency grading as well. When there is a tie, all ties are presented as suggestion. If no reasonable match is identified, then the user is informed about this.

4.2.3 Conflict Resolution

When conflicts are detected by the algorithm, after all possible cleaning processes as explained above have been applied; answers that have close ties are presented as equal or likely possibilities to the user.

4.3 Pseudo Code

4.3.1 Algorithm Assimilation

```
-request passage from user
-classifies passage with NLP classifier
-if passage is successfully classified and database is connected
    -break passages into sentence
    -break sentence into words
    -save sentence into knowledge base
    -save words into knowledge base
    -identify word dependencies from classification and save into knowledge base
    - identify word POS and save into knowledge base
    -identify lemma and save into knowledge base
    -identify NER and save into knowledge base
    -returns success status to the user.
-else
    -show user an error message
```

4.3.2 Pseudo Code for Answer Retrieval Assimilation

```
-request question from user
-breaks down question with Stanford NLP
-query knowledge base with major POS and root word/root lemma to find sentences that contain possible answer to the question.
-if query returns related rows as sentences with possible answer
    -match POS (20 points), NER (20 points,) Root (30 points), and direct quotes (20 points) in questions to sentences.
    -points are collated and highest sentence is chosen.
```

```
        -if sentence options have a tie for the highest point then all the sentence in that category are used.
        -eliminate none noun or pronoun words from returned sentence
        -if more than one person exists
                -check dependency of the nouns or pronounce.
                -is any noun or pronoun directly related to the root
                        -if yes, present as correct answer.
                        -else
                                -use part of speech list to grade dependencies of the pronoun or noun.
                                -select the word(s) with the high dependency point
                                -find the co-reference of the word(s) if any
                                -present word(s) as answer to the user
        -else
                -present word as the answer
-else
        -use the Lemma from each word and the synonyms from the available POS to retry the entire process for
        new matches
```

4.3.3 Working Examples

The working examples shown below are used to illustrate the process followed by an algorithm to pick the correct answer to a question following the pseudo code presented above. Just for illustration purposes, a sentence will be given to the algorithm, and a question will be asked from this sentence to illustrate the algorithm's process to answering the question.

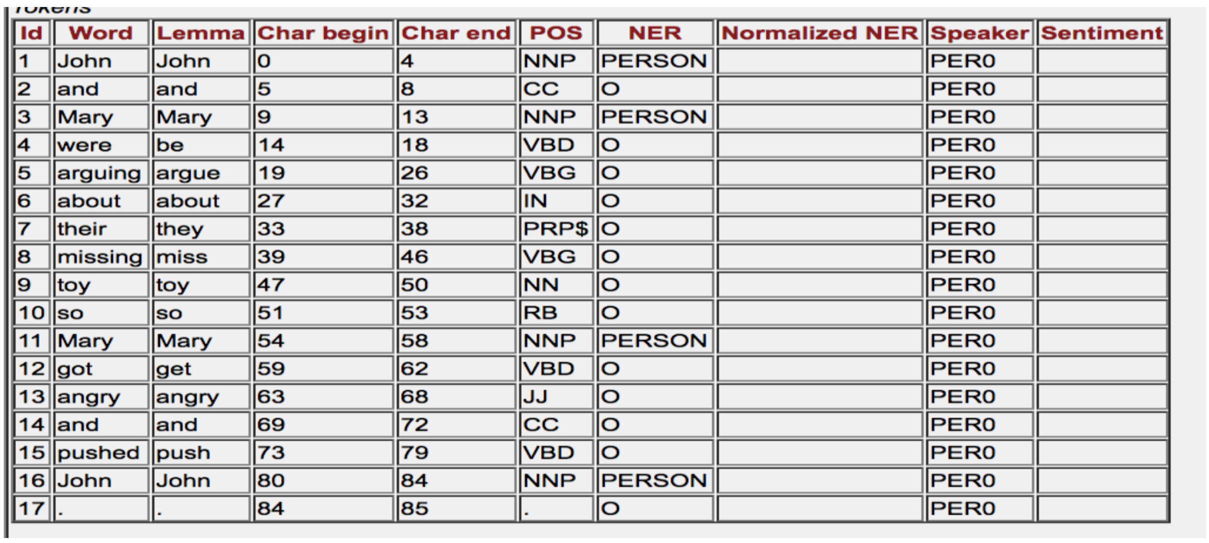
Sentence 1

John and Mary were arguing about their missing toy, so Mary got angry and pushed John.

Algorithm Walkthrough for knowledge representation

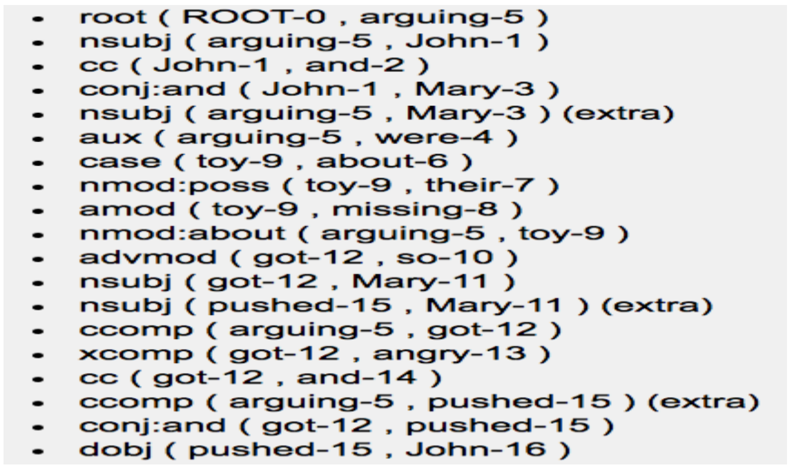
This walkthrough shows how the algorithm cherry picks possible useful information from a large classified dataset.

1. The algorithm breaks down the passage into individual sentences and saves each sentence into the knowledge base.
2. The sentence is broken down using Stanford NLP classifier. Figure 4.2 shows the individual words of each sentence and this is saved into the knowledge base.
3. Also, Figure 4.2 shows the POS, Lemma and NER of each word in a sentence. This are appropriately saved in the knowledge base
4. Figure 4.3 shows the dependencies of words in sentences which are saved in the knowledge base accordingly.
5. Process terminates.



Id	Word	Lemma	Char begin	Char end	POS	NER	Normalized NER	Speaker	Sentiment
1	John	John	0	4	NNP	PERSON		PER0	
2	and	and	5	8	CC	O		PER0	
3	Mary	Mary	9	13	NNP	PERSON		PER0	
4	were	be	14	18	VBD	O		PER0	
5	arguing	argue	19	26	VBG	O		PER0	
6	about	about	27	32	IN	O		PER0	
7	their	they	33	38	PRP$	O		PER0	
8	missing	miss	39	46	VBG	O		PER0	
9	toy	toy	47	50	NN	O		PER0	
10	so	so	51	53	RB	O		PER0	
11	Mary	Mary	54	58	NNP	PERSON		PER0	
12	got	get	59	62	VBD	O		PER0	
13	angry	angry	63	68	JJ	O		PER0	
14	and	and	69	72	CC	O		PER0	
15	pushed	push	73	79	VBD	O		PER0	
16	John	John	80	84	NNP	PERSON		PER0	
17	.	.	84	85	.	O		PER0	

Figure 4.2 - Classified data from the sentence (Stanford CoreNLP, 2015)



- root (ROOT-0 , arguing-5)
- nsubj (arguing-5 , John-1)
- cc (John-1 , and-2)
- conj:and (John-1 , Mary-3)
- nsubj (arguing-5 , Mary-3) (extra)
- aux (arguing-5 , were-4)
- case (toy-9 , about-6)
- nmod:poss (toy-9 , their-7)
- amod (toy-9 , missing-8)
- nmod:about (arguing-5 , toy-9)
- advmod (got-12 , so-10)
- nsubj (got-12 , Mary-11)
- nsubj (pushed-15 , Mary-11) (extra)
- ccomp (arguing-5 , got-12)
- xcomp (got-12 , angry-13)
- cc (got-12 , and-14)
- ccomp (arguing-5 , pushed-15) (extra)
- conj:and (got-12 , pushed-15)
- dobj (pushed-15 , John-16)

Figure 4.3 - Dependency data structure from the sentence (Stanford CoreNLP, 2015)

Question 1

Who pushed John?

Algorithm Walk Through for the Question 1

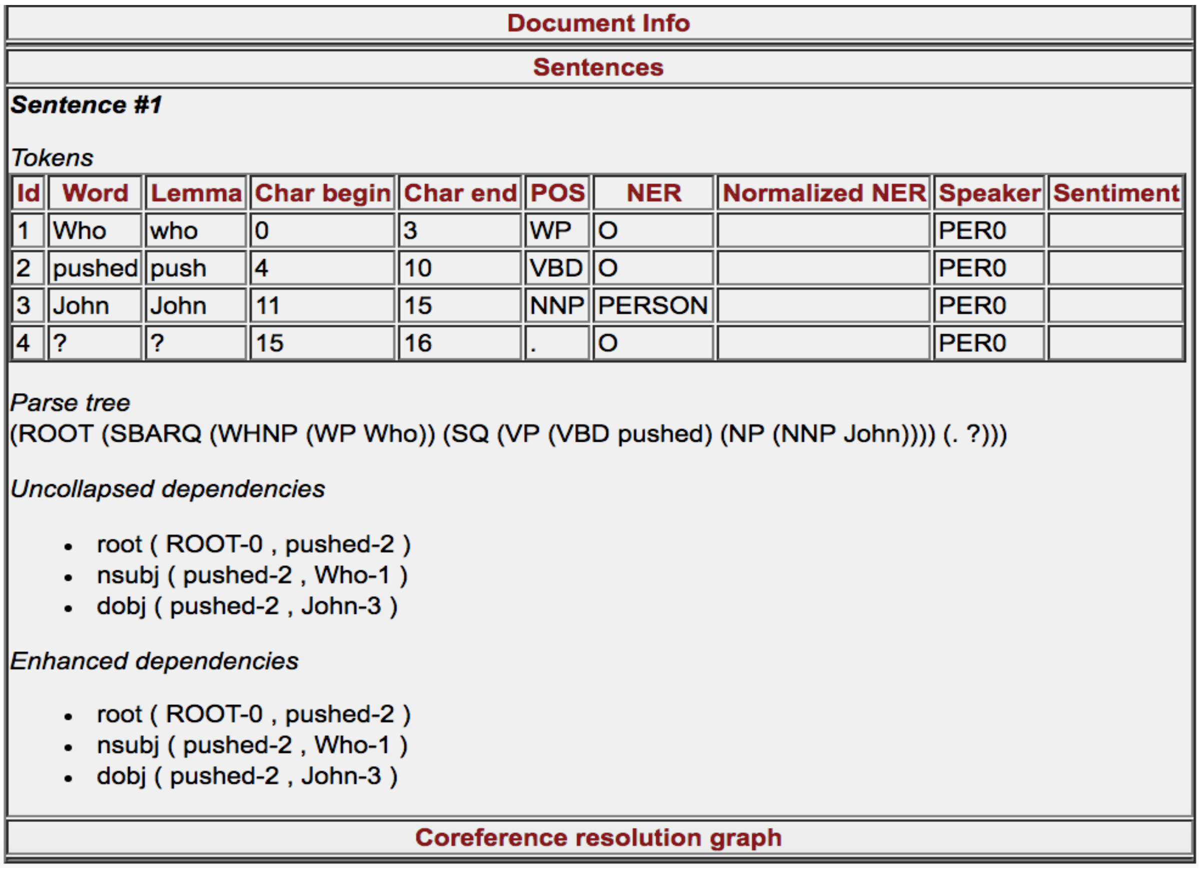
Figure 4.4 shows the classification of the question and the algorithm tries to determine the answer to the question by following the pattern below.

1. The noun (John), verb (Pushed), root word (Pushed) or root lemma (Push) is all used to find a match for the correct sentence with the answer. The noun (John) will be assigned 9 points as explained in the PBS, the verb (Pushed) will be assigned 7 points because of its position in the POS grading list shown in section 4.2. The root word in the question can also be found in the selected sentence so this will be given 30 points.
2. The total points for this sentence match will be 46 after the algorithm has completed the calculation which is a quite high considering that the questions only have a few words. The sentence "John and Mary were arguing about their missing toy, so Mary got angry and pushed John" will be selected from the list of sentences in the database.
3. From the sentence, there are two actors, Mary and John, so the algorithm must determine who pushed the other person.

   The algorithm identifies which of the actors is the object of the question DOBJ (John). The classifier identified who the action was carried on (John as a Direct Object) but it doesn't tell who carried out the action. The algorithm determines who carried out the action by picking the questions root (Pushed) and identify how Mary and John are related to this dependence in original sentence.

   The relationship between Pushed and Mary Shown in Figure 4.3 is NSUBJ while the relationship between Pushed and John is DOBJ as it was in the sentence. Now the algorithm reasons like this *"So, If john was pushed in the question, and he was also pushed in the original sentence, so who pushed him?"*. Also, the algorithm thinks like this *"I know Pushed is the action done to John the receiver so who did it?"*. From this analysis, the algorithm goes to the correct sentence to identify who carried out the action. Mary has a direct NSUBJ connection to the word pushed which automatically makes Mary the carrier of the action.

4. Algorithm presents Mary has the correct answer by considering all the dependencies in that sentence.



Document Info

Sentences

Sentence #1

Tokens

Id	Word	Lemma	Char begin	Char end	POS	NER	Normalized NER	Speaker	Sentiment
1	Who	who	0	3	WP	O		PER0	
2	pushed	push	4	10	VBD	O		PER0	
3	John	John	11	15	NNP	PERSON		PER0	
4	?	?	15	16	.	O		PER0	

Parse tree

(ROOT (SBARQ (WHNP (WP Who)) (SQ (VP (VBD pushed) (NP (NNP John)))) (. ?)))

Uncollapsed dependencies

- root (ROOT-0 , pushed-2)
- nsubj (pushed-2 , Who-1)
- dobj (pushed-2 , John-3)

Enhanced dependencies

- root (ROOT-0 , pushed-2)
- nsubj (pushed-2 , Who-1)
- dobj (pushed-2 , John-3)

Coreference resolution graph

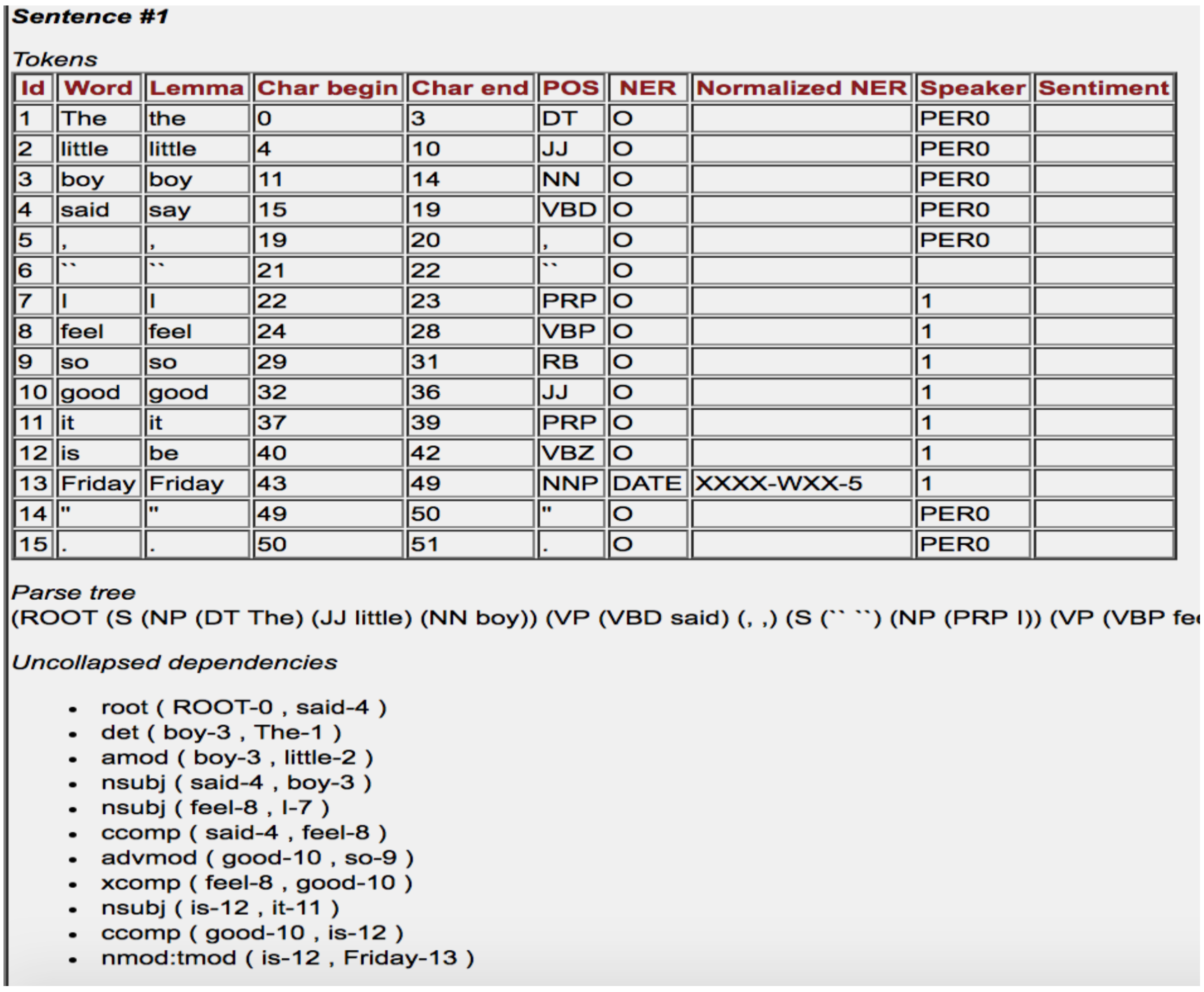
Figure 4.4 - Classified data from the question (Stanford CoreNLP, 2015)

Sentence 2

This is an example on a short passage of a little boy randomly generated for this process walk through.

The little boy said, "I feel so good it is Friday". I can get to play all day tomorrow. When I get home tonight, I will invite all my friends to go come and play with me tomorrow.

The knowledge representation process is the same as sentence 1 so this step will not be explained here again.



Sentence #1

Tokens

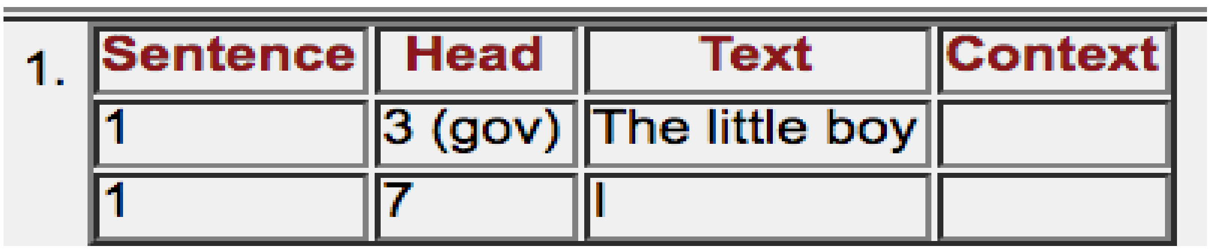
Id	Word	Lemma	Char begin	Char end	POS	NER	Normalized NER	Speaker	Sentiment
1	The	the	0	3	DT	O		PER0	
2	little	little	4	10	JJ	O		PER0	
3	boy	boy	11	14	NN	O		PER0	
4	said	say	15	19	VBD	O		PER0	
5	,	,	19	20	,	O		PER0	
6	``	``	21	22	``	O			
7	I	I	22	23	PRP	O		1	
8	feel	feel	24	28	VBP	O		1	
9	so	so	29	31	RB	O		1	
10	good	good	32	36	JJ	O		1	
11	it	it	37	39	PRP	O		1	
12	is	be	40	42	VBZ	O		1	
13	Friday	Friday	43	49	NNP	DATE	XXXX-WXX-5	1	
14	''	''	49	50	''	O		PER0	
15	.	.	50	51	.	O		PER0	

Parse tree
(ROOT (S (NP (DT The) (JJ little) (NN boy)) (VP (VBD said) (, ,) (S (`` ``) (NP (PRP I)) (VP (VBP fe

Uncollapsed dependencies

- root (ROOT-0 , said-4)
- det (boy-3 , The-1)
- amod (boy-3 , little-2)
- nsubj (said-4 , boy-3)
- nsubj (feel-8 , I-7)
- ccomp (said-4 , feel-8)
- advmod (good-10 , so-9)
- xcomp (feel-8 , good-10)
- nsubj (is-12 , it-11)
- ccomp (good-10 , is-12)
- nmod:tmod (is-12 , Friday-13)

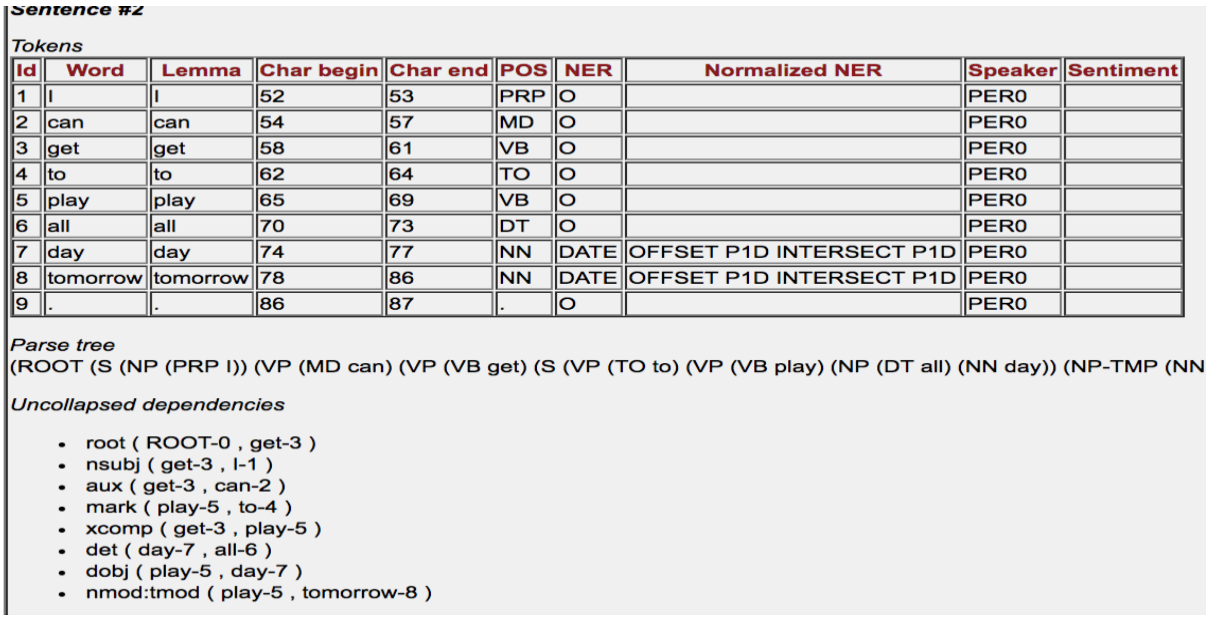
Figure 4.5 - Classification of sentence 2 (the first sentence in the passage) *(Stanford CoreNLP, 2015)*



1.

Sentence	Head	Text	Context
1	3 (gov)	The little boy	
1	7	I	

Figure 4.6 – Co-reference of sentence 2 (the first sentence in the passage) *(Stanford CoreNLP, 2015)*



Sentence #2

Tokens

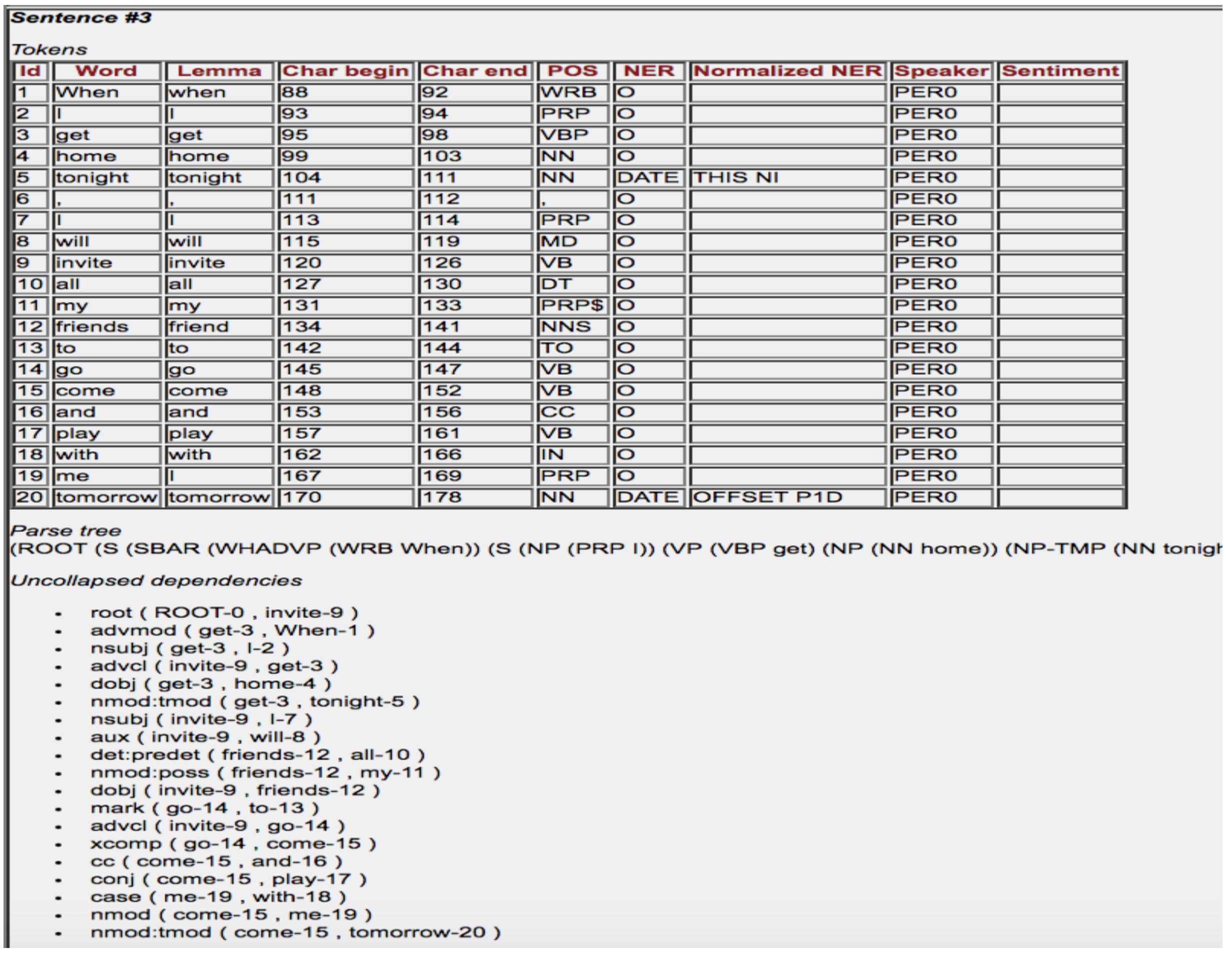
Id	Word	Lemma	Char begin	Char end	POS	NER	Normalized NER	Speaker	Sentiment
1	I	I	52	53	PRP	O		PER0	
2	can	can	54	57	MD	O		PER0	
3	get	get	58	61	VB	O		PER0	
4	to	to	62	64	TO	O		PER0	
5	play	play	65	69	VB	O		PER0	
6	all	all	70	73	DT	O		PER0	
7	day	day	74	77	NN	DATE	OFFSET P1D INTERSECT P1D	PER0	
8	tomorrow	tomorrow	78	86	NN	DATE	OFFSET P1D INTERSECT P1D	PER0	
9	.	.	86	87	.	O		PER0	

Parse tree
(ROOT (S (NP (PRP I)) (VP (MD can) (VP (VB get) (S (VP (TO to) (VP (VB play) (NP (DT all) (NN day)) (NP-TMP (NN

Uncollapsed dependencies

- root (ROOT-0 , get-3)
- nsubj (get-3 , I-1)
- aux (get-3 , can-2)
- mark (play-5 , to-4)
- xcomp (get-3 , play-5)
- det (day-7 , all-6)
- dobj (play-5 , day-7)
- nmod:tmod (play-5 , tomorrow-8)

Figure 4.7 - Classification of sentence 2 (the second sentence in the passage) (Stanford CoreNLP, 2015)



Sentence #3

Tokens

Id	Word	Lemma	Char begin	Char end	POS	NER	Normalized NER	Speaker	Sentiment
1	When	when	88	92	WRB	O		PER0	
2	I	I	93	94	PRP	O		PER0	
3	get	get	95	98	VBP	O		PER0	
4	home	home	99	103	NN	O		PER0	
5	tonight	tonight	104	111	NN	DATE	THIS NI	PER0	
6	,	,	111	112	,	O		PER0	
7	I	I	113	114	PRP	O		PER0	
8	will	will	115	119	MD	O		PER0	
9	invite	invite	120	126	VB	O		PER0	
10	all	all	127	130	DT	O		PER0	
11	my	my	131	133	PRP$	O		PER0	
12	friends	friend	134	141	NNS	O		PER0	
13	to	to	142	144	TO	O		PER0	
14	go	go	145	147	VB	O		PER0	
15	come	come	148	152	VB	O		PER0	
16	and	and	153	156	CC	O		PER0	
17	play	play	157	161	VB	O		PER0	
18	with	with	162	166	IN	O		PER0	
19	me	I	167	169	PRP	O		PER0	
20	tomorrow	tomorrow	170	178	NN	DATE	OFFSET P1D	PER0	

Parse tree
(ROOT (S (SBAR (WHADVP (WRB When)) (S (NP (PRP I)) (VP (VBP get) (NP (NN home)) (NP-TMP (NN tonigh

Uncollapsed dependencies

- root (ROOT-0 , invite-9)
- advmod (get-3 , When-1)
- nsubj (get-3 , I-2)
- advcl (invite-9 , get-3)
- dobj (get-3 , home-4)
- nmod:tmod (get-3 , tonight-5)
- nsubj (invite-9 , I-7)
- aux (invite-9 , will-8)
- det:predet (friends-12 , all-10)
- nmod:poss (friends-12 , my-11)
- dobj (invite-9 , friends-12)
- mark (go-14 , to-13)
- advcl (invite-9 , go-14)
- xcomp (go-14 , come-15)
- cc (come-15 , and-16)
- conj (come-15 , play-17)
- case (me-19 , with-18)
- nmod (come-15 , me-19)
- nmod:tmod (come-15 , tomorrow-20)

Figure 4.8 - Classification of sentence 2 (the third sentence in the passage*) (Stanford CoreNLP, 2015)*

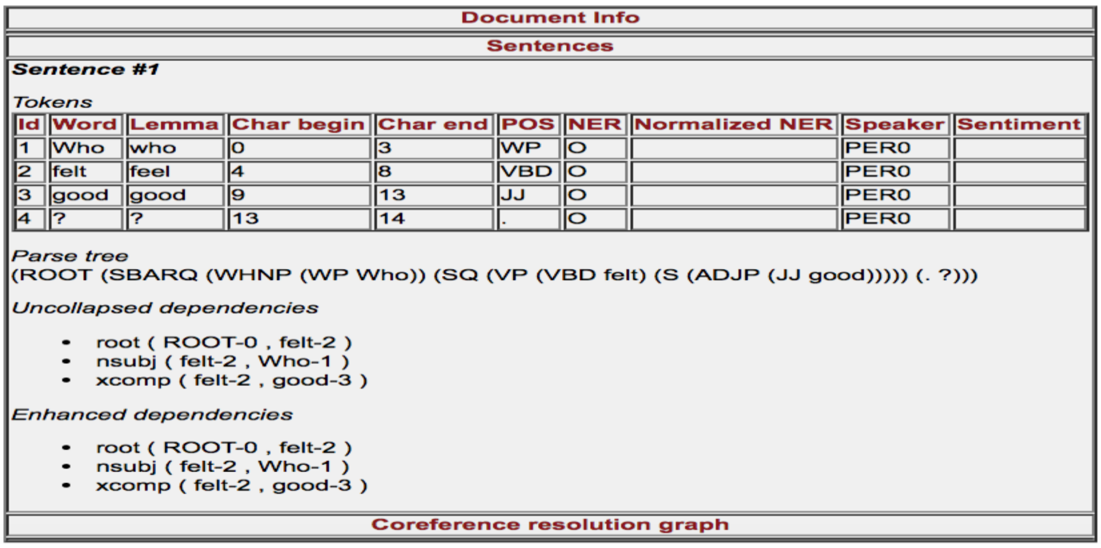
Question 2

Who felt good?

Algorithm Walk Through for Question 2

Though the classification structures in the questions are different as shown in Figure 4.9 compared to Figure 4.4, the algorithm follows a similar pattern to what has been used in question 1.

1. The clausal complement (Good), root word (Felt) or root lemma (Feel) is all used to find a match for the correct sentence with the answer. The root lemma (Feel) is assigned 30 points as explained in the PBS while the clausal complement is assigned 5 points.
2. The total point will be 35 after the algorithm completes the calculation. The little boy said, "I feel so good it is Friday" will be selected from the list of sentences in the database.
3. From the sentence, the algorithm picks the little boy as the answer to the question because there is only one actor as shown in Figure 4.5. The words "I" and "The little boy" are co-referenced. So the algorithm knows they are the same.



Document Info

Sentences

Sentence #1

Tokens

Id	Word	Lemma	Char begin	Char end	POS	NER	Normalized NER	Speaker	Sentiment
1	Who	who	0	3	WP	O		PER0	
2	felt	feel	4	8	VBD	O		PER0	
3	good	good	9	13	JJ	O		PER0	
4	?	?	13	14	.	O		PER0	

Parse tree

```
(ROOT (SBARQ (WHNP (WP Who)) (SQ (VP (VBD felt) (S (ADJP (JJ good))))) (. ?)))
```

Uncollapsed dependencies

- root (ROOT-0 , felt-2)
- nsubj (felt-2 , Who-1)
- xcomp (felt-2 , good-3)

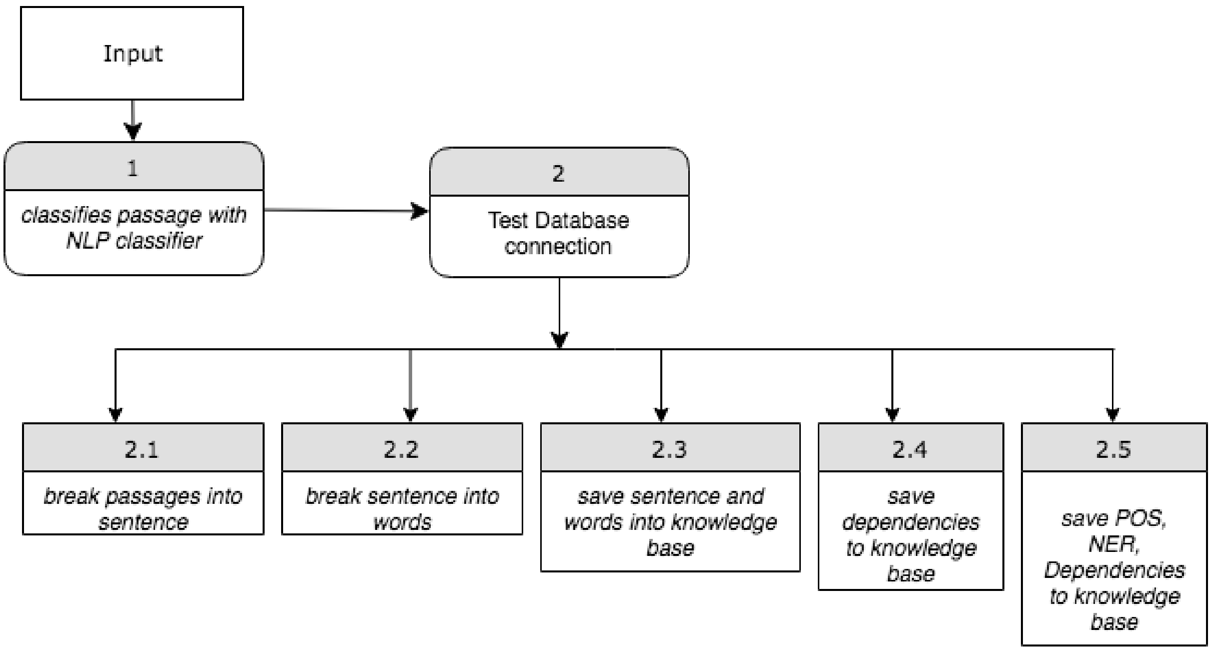
Enhanced dependencies

- root (ROOT-0 , felt-2)
- nsubj (felt-2 , Who-1)
- xcomp (felt-2 , good-3)

Coreference resolution graph

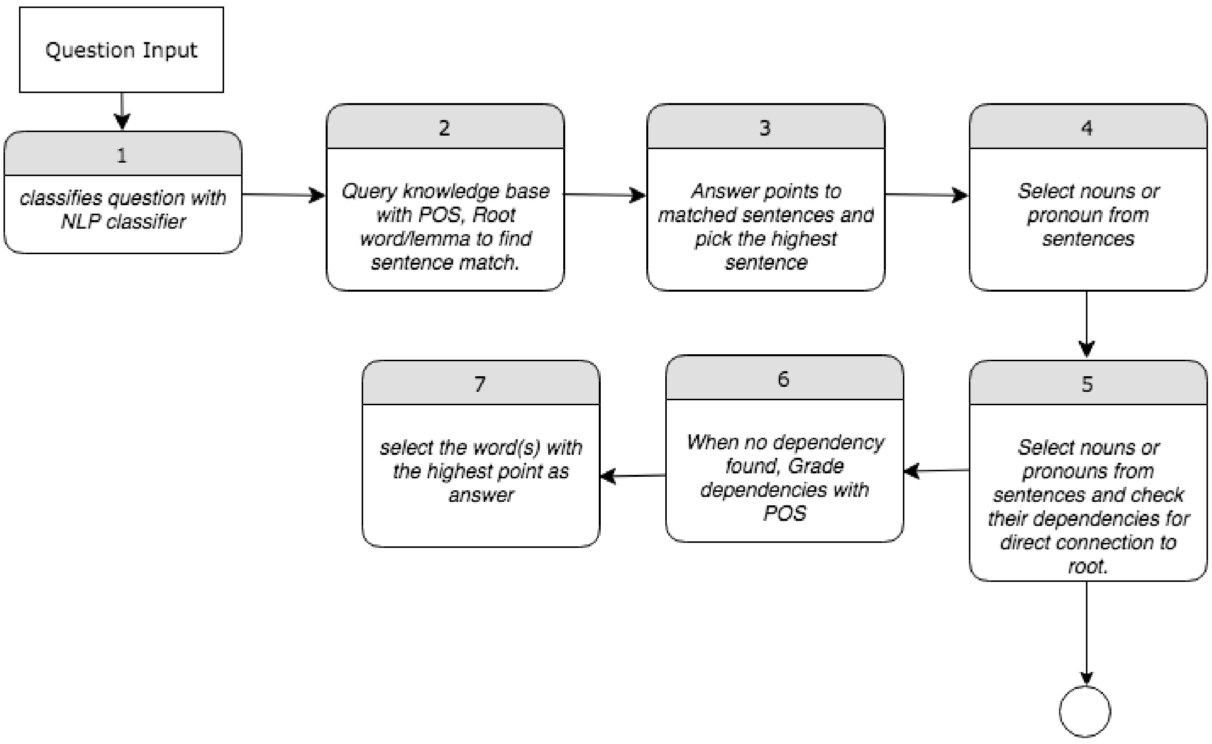
Figure 4.9 - Classification of question 2 (Stanford CoreNLP, 2015)

4.4 Algorithm Architectural



Input
1
classifies passage with NLP classifier
2
Test Database connection
2.1
break passages into sentence
2.2
break sentence into words
2.3
save sentence and words into knowledge base
2.4
save dependencies to knowledge base
2.5
save POS, NER, Dependencies to knowledge base

Figure 4.10 – Algorithm assimilation blueprint



Question Input
1
classifies question with NLP classifier
2
Query knowledge base with POS, Root word/lemma to find sentence match.
3
Answer points to matched sentences and pick the highest sentence
4
Select nouns or pronoun from sentences
5
Select nouns or pronouns from sentences and check their dependencies for direct connection to root.
6
When no dependency found, Grade dependencies with POS
7
select the word(s) with the highest point as answer

Figure 4.11 – Algorithm answer retrieval blueprint

Figure 4.10 and 4.11 above shows the algorithms architecture. The figure visually shows the flow of the pseudo code which explains how the various components of the algorithm are interlinked together to produce a complete architecture. The architecture is divided into 2 main sections, which are:

1. The knowledge base section
2. The answer retrieval section.

From Figure 4.10, labels 2.1 to 2.5 are all unique contributions of this research. Figure 4.11 shows the answer retrieval process of the algorithm previously explained in the pseudo code. Labels 2 to 7 in Figure 4.11 are also unique contributions to this research.

This chapter describes and illustrates how the algorithm works with examples of how it assimilates passages and retrieves answers from them. The chapter has also pointed out the unique contributions of this research which have all been derived from this study and the exact model used do not current exist anywhere else. In the next chapter, live experiments will be conducted.

Chapter 5. Conducted Experiment

As previously shown in Chapter 3, initial tests were carried out on several real-time applications. These tests were conducted to examine how modern NLP systems like Siri, Cortana, Google voice, and Amazon Echo respond to natural language questions.

In this chapter, two types of experiments will be conducted which have already been explained in section 3.1.4. Analysis on the results obtained from experimentation in this chapter will be done in the next chapter.

The first experiment to be conducted on the prototype is a **sole experiment**. This will be conducted solely on a developed prototype (which is a product of this research). This experiment is conducted to test the accuracy of the method the algorithm uses for NLP analysis using this prototype.

As briefly explained in Section 3.1.4, a team consisting of 5 PARK members chose 5 random short stories from the internet. The most complex of the stories (Bubien, 2017) was selected as a good test bed because of the numerous occurrences of punctuations, exclamations and hyphens etc. contained within the passage. Each member of the team asked 5 questions each from the passage which made a total of 25 questions. Duplicate questions were removed and only 15 unique questions were selected. Appendix H shows two real samples of questions asked by members.

Similar to the question retrieval process for the first test as discussed above, 2 different PARK team members used random online websites to select random general knowledge questions from (General Knowledge Quiz, 2018) and (General Knowledge Quizzes, 2018). All the questions from these websites that matched the test scenario were used in conducting the experiment to measure the performance of the prototype against other systems. Section 5.1 and 5.2 discuss the conducted experiments.

5.1 Experiment One (Self-Test)

Table 5.1 shows the results obtained from the algorithm during the experiment on random 15 questions chosen by the PARK members.

Table 5.1 - Showing the sample questions tested on the prototype system

Result Information Correct: The algorithm found the exact answer to the question. Failed: The algorithm presented a wrong answer as the answer to the question. Not Found: The algorithm could not find an answer to the question.		
S/N	**Question**	**Result**
1	Who was in the hospital elevator?	Correct
2	Who giggled mischievously?	Correct
3	Who had a list?	Correct
4	Who said we are half way there?	Not found
5	Who stepped into the hospital room?	Correct
6	Who whispered into the blankets?	Correct
7	Who drummed the fingers on the list?	Correct
8	Whose hand was in mine?	Failed
9	Who is shaking their head?	Correct
10	Whose eyes danced?	Correct
11	Who asked the question, “did you get an answer”?	Correct
12	Who asked, “what are you looking for”?	Not found
13	Who said, "but I don’t like any of them"?	Correct
14	Who held a list of her own?	Correct
15	Who was asked to “say hi to your big sister”?	Not found

Appendix G shows images of the prototype when answering some of the above listed question.

Table 5.2 - Showing the cumulative result of experiment 1

Correct	Failed	Not Found
11	1	3

Table 5.2 above presents a summary of the results in Table 5.1 obtained when a self-test was conducted on the prototype. Analysis of the result will be done in the next chapter.

5.2 Experiment Two (General knowledge questions on closed domain and open domain systems)

50 questions from the general knowledge online questions complying with the aims of this thesis will be tested on the various systems listed within the table below.

Table 5.3 – General knowledge experiments conducted

Result Information C: The system found the exact answer to the question. CS: Correct Suggestions that can easily lead to the answer e.g. web pages with detail F: The system presented a wrong answer as the answer to the question. NA: The system could not find an answer to the question GP: The system opened a Google Page or Bing Page The experiment will be conducted on the following system and they will be represented on the table using their initials 1. Google Personal Assistant (GPA) 2. Siri (SR) 3. Cortana (CT) 4. Amazon Echo (AE) 5. Wolfram Alpha (WA) 6. Prototype (PT)							
S/N	Question	GPA	SR	CT	AE	WA	PT
1	Who played the title character in the BBC series 'Jonathan Creek'?	F	F	F	F	F	C
2	Who succeeded Henry II as King of England?	C	F	F	F	F	C
3	Who is older, Theresa May or Jeremy Corbyn?	F	GP	GP	C	C	C

4	Who was the shortest-serving UK Prime Minister of the 20th century?	F	F	GP	F	F	C
5	Who wrote the books on which the series ‘Game of Thrones’ is based?	C	GP	CS	NA	F	NA
6	Who stood in as host on the first episode of Have I Got News For You following Angus Deayton’s departure in 2002?	CS	NA	NA	NA	F	C
7	Who is the alter-ego of Sir Percy Blakeney?	C	F	F	F	F	F
8	Who was the sixth wife of Henry VIII?	C	GP	C	C	CS	CS
9	Who held the post of US president at the time of the coronation of Queen Elizabeth II?	F	F	F	NA	F	C
10	Who is attributed with the famous quote “Be the change that you wish to see in the world”?	C	GP	F	NA	F	C
11	Who was the only British Prime Minister to be assassinated?	C	GP	C	C	F	C
12	Who was the 2018 Wimbledon Women’s Singles champion?	F	C	GP	C	F	F
13	Who was the first ever winner of ‘Britain’s Got Talent’?	C	GP	F	C	F	C
14	Who wrote the controversial 1955 novel ‘Lolita’?	C	GP	CS	C	C	C
15	Who is the actress that played Phoebe Buffay in the sitcom ‘Friends’?	F	NA	F	C	F	C
16	Who is the former director of the FBI who was sacked by President Trump in May 2017 over his handling of the inquiry into Hillary Clinton’s emails?	GP	GP	GP	NA	F	C
17	Who was the mother of Henry VIII?	C	F	C	C	C	C
18	Who wrote the award-winning novel ‘The Curious Incident of the Dog in the Night-Time’,	GP	GP	GP	NA	F	C

	which has now been adapted into both a film and a play?						
19	Who has been the Channel 4 show Countdown's resident lexicographer and etymologist in 'Dictionary Corner' since 1992?	GP	F	GP	C	F	F
20	Who held the post of Chancellor of Germany between October 1998 and November 2005?	F	F	CS	NA	F	C
21	Who was the fictional woodcarver who created the puppet 'Pinocchio' in Carlo Collodi's story?	GP	GP	C	NA	F	C
22	Who was the Labrador retriever who featured in the Australian soap 'Neighbours' between 1987 and 1993?	C	F	C	NA	F	F
23	Who was the original UK host of the ITV game show 'The Price is Right'?	C	F	F	NA	F	C
24	Who had a UK number one hit for four weeks in 1960 with 'My Old Man's A Dustman'?	GP	GP	GP	NA	F	C
25	Who was the eldest son of Jacob according to the book of Genesis?	F	F	C	NA	F	C
26	Who wrote the 1886 gothic novella 'Strange Case of Dr Jekyll and Mr Hyde'?	C	GP	C	C	F	C
27	Who was the mouse who was Dumbo's only friend in the 1941 Disney animated film?	C	NA	F	C	F	F
28	Who was the first Tudor monarch?	C	GP	C	C	F	CS
29	Who does the Beast fall in love with?	F	F	F	NA	F	F
30	Who performed the theme music to the 2015 James Bond film 'Spectre'?	C	NA	F	NA	F	F
31	Who is the lead singer of Irish rock band 'The Script'?	C	NA	C	C	F	C
32	Who is Reg Dwight better known as?	C	F	GP	NA	CS	F

33	Who provided Nick Wilde’s voice in the 2016 movie of “Zootopia”?	GP	NA	GP	C	F	F
34	Who is the patron saint of music?	C	NA	C	C	F	C
35	Who earned the nickname “Slow-hand”?	C	GP	C	NA	F	C
36	Who was Liverpool’s skipper in the 2005 European Champions League triumph?	F	F	GP	F	F	F
37	Who wrote “The Pit and the Pendulum”?	C	C	C	C	C	C
38	Who made the album “Confessions on a Dance Floor”?	C	C	C	NA	C	C
39	Who was known as the lady with the lamp?	C	NA	C	F	C	F
40	Who was the female lead in Shakespeare’s “Othello”?	CS	NA	GP	NA	F	C
41	Who sang with the Dakotas?	C	NA	GP	C	F	C
42	Whose ship was the first to sail around the world?	C	F	C	NA	F	F
43	Who was the “Prime Minister of Mirth” in the music hall?	GP	F	F	NA	F	C
44	Who was the first peer to disclaim his title following the 1963 Peerage Act?	F	F	GP	NA	F	F
45	Who is the lead female actress in “Orange is the New Black”?	CS	GP	GP	F	F	F
46	Who wrote, “Help Me Make It Through the Night”?	C	C	C	C	F	C
47	Who created the detective, Paul Temple?	C	GP	C	C	F	C
48	Who pricked her finger on a spinning wheel and slept for 100 years?	F	GP	GP	NA	F	F
49	Who became Earl of Stockton on his 90th birthday?	C	GP	GP	NA	F	C
50	Who was the first British monarch to visit New Zealand?	F	GP	C	F	F	CS

Table 5.4 - Showing the cumulative result of experiment 2

System	**C**	**CS**	**F**	**GP**	**NA**
GPA	27	3	13	7	0
SR	4	0	17	19	10
CT	18	3	12	16	1
AE	19	0	8	0	23
WA	6	2	42	0	0
PT	31	3	15	0	1

Table 5.4 above shows a summary of the results obtained when conducting general knowledge test against several systems in Table 5.3.

Having concluded these two experiments, analysis of the results shown in Table 5.2 and Table 5.3 will be carried out in the next chapter.

Chapter 6. Results Analysis

The analysis of the results obtained from experiment one and experiment two in Chapter 5 above will be completed using a confusion matrix. Confusion matrix is used to get accurate evaluation of the results generated from the experiments. Accuracy will mainly be used to correctly determine the efficiency of the Algorithm in providing correct answers to question when compared to other systems in which computational accuracy doesn’t say much about.

True positives (TP), False Positive (FP) and False Negatives (FN) are the three determinants which will be used in this matrix. TP will be assigned to any correct answer which was represented as (C) or (CS) in Table 5.1 or 5.3, and FP will account for every other type where the correct answer was not given e.g. (NA) or (GR) and FN will be recorded as (F) when a system gives a different answer from the correct answer.

Accuracy Calculation

The formula below will be used to calculate the accuracy of all systems that was experimented upon. This will show the percentage of the correct results compared to the wrong results. This formula will also show how good the Algorithm is compared to other systems used in the experiment or other methods that have been described in Chapter 2.

Formula = $\frac{TP}{TP+FP} * 100$ = Correct Predictions / All predictions (this will be returned in percentage)

Error Calculation

The formula below will be used to calculate all the results that have not been correctly answered in the experiment. Asides from (C) and (CS) and (F) as described in Table 5.3, all other results fall in this category.

Formula = $\frac{FP}{TP+FP}$* 100 = Incorrect Predictions / All Predictions

Recall Calculation

The formula below will be used to calculate the results where a wrong answer has been given in place of the correct answer. All (F) as shown in Table 5.3 fall in this category. The recall will be used to validate the accuracy to prove that the accuracy of the prototype is not due to chance.

Formula = $\frac{TP}{TP+FN}$* 100

6.1 Confusion Matrix for the Results of Experiment One

Table 6.1 shows the collated confusion matrix obtained from the experiment conducted on the algorithm against itself.

Table 6.1 - Confusion Matrix for the systems self-test

	p' (Predicted)	**n' (Predicted)**
p (Actual)	TP = 11	FN = 1
n (Actual)	FP = 3	TN = 0

Table 6.2 - Using statistical analysis to determine algorithms accuracy, error and recall

Accuracy	78.57%
Error	21.42%
Recall	91.66%

Table 6.2 shows the accuracy, error and recall calculated with the matrix using the formula previously described above. The question and answer self-test from a randomly chosen passage proves the algorithms ability to comprehend with a scoring average of 78.57% out of 15

questions. This puts the algorithm performance (accuracy) in terms of question answering above average (which will be around 50%). A high recall of 91.66% was also obtained from the calculations above giving more significance to the accuracy. The algorithm failed to produce correct answer for only 4 questions amounting to 21.42% error of the total questions asked. For question 4, in Table 5.1, the algorithm was unable to use its current dependency model to identify the correct answer in the most relevant sentence. This was because the most relevant sentence obtained from the 1st round query on the knowledge base returned a vague sentence “Well then, we're half-way there!" my wife's eyes danced”. This sentence didn’t contain any possible answer to the question, so the algorithm was unable to go further.

The relevant sentence returned for Question 8 in the self-test was a sentence with complex set of dependencies. “I don't remember exactly when that was, but now, with our first daughter's hand in mine as we ascended in the hospital elevator, it really didn't matter.” The root word for this sentence was “matter”. The classifier produced 5 direct dependencies to the word “hand”, 2 dependencies for “mine” and 4 for “daughter”. This reduced the chances of the algorithm finding the right answer.

6.2 Confusion Matrix for General Knowledge Experiment

To calculate the confusion matrix for the general knowledge test, the TP will consist of both correct predictions (C) and Correct Suggestions (CS) as section 6.1(F) will be assigned to FN while all other responses asides will be assigned to FP. Table 6.3 shows the confusion matrix for the results obtained in the experiment.

Table 6.3 – Shows the confusion matrix for the second experiment

Matrix	**GPA**	**SR**	**CT**	**AE**	**WA**	**PT**
TP	30	4	21	19	8	34
FP	7	29	17	23	0	7
FN	13	17	12	8	42	9

Table 6.4 - Final accuracy, error and recall matrix obtained from the experiment

Matrix	**GPA**	**SR**	**CT**	**AE**	**WA**	**PT**
Accuracy	60%	8%	42%	38%	100%	68%
Error	18.91%	87.87%	44.73%	54.76	0%	17%
Recall	69.76%	19.04%	63.63%	70.37%	16%	79%

The percentages shown in Table 6.4 is the final cumulative mathematical matrix derived from the calculations done on the conducted experiment in Chapter 5. This table shows that the model used on the algorithm developed for this research had the highest accuracy rate of 68% with relation to the recall at 79% in the general knowledge questions. This accuracy was followed by Google's personal assistant at 60% accuracy (which referenced Wikipedia's source for most of its correct answers) and 69.76% recall. WA had the most inconsistent accuracy with respect to its recall being very low. This was because most of the answers it produced were (FN) values which meant the accuracy in this case didn't justify its correctness. WA and SR had the poorest result amongst all the systems involved in the experiment as shown in the table above. The algorithms (PT) accuracy, combined with its recall places it above average in comprehension and its ability to represent knowledge in a way that it can produce correct answers. This result makes it the best performing system in the overall comparison. All other applications tested in this experiment (asides GPA) produced answers below the average threshold 50%. Comparing the recall of PT against GPA, it justifies a significance of difference in the accuracy. While it is arguable that accuracy alone may be insufficient to prove significance of a system as seen in the case of WA, a good accuracy score combined with a good recall can prove this significance.

Analysing questions 1, 4, 9, 20 and 25 in Table 5.3, it can be noted that only the algorithm (PT) produced the correct answers to these questions while every other system did not. Questions like, "Who was the eldest son of Jacob according to the book of Genesis?", the most important word in the sentence that will differentiate suggestions from the correct answer is "eldest".

Jacob may have different children, but only the name of the eldest son is required. This was one of the reasons why systems like Wolfram Alpha (WA) were observed to have produced lower accuracy than most of the other test systems. According to observations, WA focuses more on keyword matching e.g. using a name “Jacob”, or a major entity like “Genesis”. After getting the related articles on Genesis and Jacob, any system must consider “eldest” as a dependency to produce the correct answer to that question. Without this consideration, there is a high possibility that the system will not produce the correct answer to the question.

46% of questions failed by the algorithm was equally failed by Google. So, some of these failed questions will be examined. An example of such question was question 12, “*Who was the 2018 Wimbledon Women’s Singles champion?*”. These demonstrate a very deep four stage level dependency mapping. These dependencies are 2018, Wimbledon, Women’s and Singles. Question 19, “*Who has been the Channel 4 show Countdown’s resident lexicographer and etymologist in ‘Dictionary Corner’ since 1992?*”, also shows several interwoven lexicon dependencies which is likely to have resulted in the prototypes inability to find the exact dependencies to map a meaningful relationship. However, the algorithm was still able to find the correct answer to some complex dependency questions like question 1 and 6 which was failed by most of the other systems. The algorithm produced higher accuracy for questions with less complexity dependencies e.g. question 2, “*Who succeeded Henry II as King of England?*” which had mainly 3 dependencies, “succeeded” and “King” and “England”.

6.3 Conclusion

This chapter has analysed the results obtained from the experiment conducted in Chapter 5. Experimenting with multiple general questions from different sources in the second experiment helped in eliminating the possibility of chance; asking a single question would have resulted in a 50/50 possibility. Using the confusion matrix as a process of analysing experimental results, the algorithm demonstrated satisfactory comprehension ability when tested with questions from a random passage. In the second experiment, amongst all the test cases experimented on, the algorithm’s accuracy and recall were the highest. It was observed that of all the incorrect answers returned by the algorithm, the most inconsistent ones were returned when it was asked questions with deeper dependency links (4 dependencies and above). Most of the systems got the correct answer for questions with 1 to 2 dependency levels. The algorithm got more answers from question with 3 dependency levels than any other test system.

Chapter 7. Evaluation

7.1 Introduction

Studies have shown that Natural Language Processing technology is fast becoming an everyday tool in our lives with the likes of personal assistants like amazon echo etc. However, as studied in chapter 2 and tested in section 3.1.2, the current NLP methods or tool do not perform accurately when asked complex questions. Although there have been a lot of attempts to improve the outcome of using natural language for question and answering as discussed in (Tellex et al., 2011), (Misra, Sung, Lee & Saxena, 2015) but the outcome of these methods can significantly still be improved. This chapter completes the work undertaken in this thesis to improve natural language understanding; focusing on question and answer retrieval. Currently, alternative methods have been used to query natural language sentences such as the RDF semantic web standard or key words matching technique as discussed in Chapter 2. But the method applied in this research has recorded an improvement in the results for question and answering systems. Using dependency structure as a primary way of querying sentences and answer retrieval is not as simple as just finding the dependencies in a sentence and using it to obtain the correct answer to a question asked. A systematic study of natural language processing systems and design patterns was done to find out how existing algorithms work in this domain; knowing their limitations and introducing new ways of improving on their current limitations.

In Chapter 6, the algorithms and systems that have been considered in the experiments were analysed for their accuracy, error rate and recall in a general knowledge test. The accuracy and error rate were determined using matrix solutions. Firstly, the algorithm developed in this research was evaluated using a short passage comprehension scenario. Secondly, a more complex scenario was used to evaluate the algorithms performance where it was tested against other systems as shown in Chapter 5. The two scenarios were analysed using metrics to calculate the percentage of its accuracy and error.

7.2 Evaluation Strategy

A strategy for this evaluation will be to conduct experiments on case-studies. The experimental phase in chapter 5 was majorly determined by questions chosen by others, so in this case, the author of the new algorithm will develop a case-study, conduct the experiments and evaluate the results. The issue with this approach is that the author could intentionally or unintentionally create a bias ensuring the experiment results are in favour of the algorithm. To avoid such outcome, the code and experimental results can be independently reviewed, and / or the results and code can be made available for public scrutiny. Time is the main problem factor to consider in securing independent evaluation. Self-evaluation which is not the most ideal, from a practical view point, is the best that can be achieved given the current time limitation.

7.3 Use Cases

Use cases will be used here to evaluate how the algorithm has achieved the goals outlined in the aims and objectives section (1.2) of this thesis. The use cases to be applied are made up of a set of likely sequences of interactions existing between a user and the algorithm. This use cases will contain all system activities that have significance to the user and existing algorithm.

Performance comparison will be made on the results obtained from the use cases when tested on the algorithm. The three main evaluation metrics used in Chapter 6 are **Accuracy, Error and Recall**. The same matrix will be used to evaluate the use cases.

7.3.1 Case 1 (Question Comprehension)

The case analysis will determine how accurately the algorithm can identify the what a question is about using the model explained in section 4.3.2. For this use case, 12 random questions obtained from (Bambuto, 2019) and (General Knowledge Quizzes, 2018) will be run on the algorithm. The Accuracy, Error and Recall will be calculated on question comprehension. The aim of this use case is to further evaluate the performance of the algorithm in question comprehension. Table 7.1 shows the 12 questions asked to the algorithm and it has been able to identify what the question is about.

Table 7.1 - Showing question comprehension key words

S/N	Question	Main Selected keywords and dependencies
1	**Who wrote Hamlet?**	**root(wrote), dobj(wrote)**
2	Who's the Fresh Prince of Bel Air?	root(who), amod(Prince, Fresh), nsubj(Prince)
3	**Who is officially credited with the invention of the light bulb?**	**root(credited), nmod(invention), amod(bulb, light), nmod(bulb)**
4	**Who painted the Mona Lisa**	**root(painted), dobj(Lisa), compound(Lisa, Mona)**
5	**In Greek mythology, who is the Goddess of Agriculture**	**amod (mythology, Greek), nsubj(Goddess), nmod(Agriculture)**
6	**Who invented electricity**	**root(invented), dobj(electricity)**
7	**Who had a horse called Bucephalus**	**root(called), nsubj(horse), dobj(Bucephalus)**
8	Who won the Olympic Gold for the Men’s 5000m at the 2012 London Olympics?	root(won), dobj(Gold), nmod:poss(Men), nmod(5000m), nmod(Olympics)
9	**Who recorded the million-selling album “Life for Rent”?**	**root (recorded), amod(selling), dep (album), xcomp (Life), nmod(Rent)**
10	Who came first, King Edward VIII or George VI?	root(came), advmod (first)
11	**Who was the tallest of Robin Hood’s Men?**	**nsubj(tallest), compound(Hood, Robin), nmod:poss(Hood), nmod(Men)**
12	**Who rejoined Take That in 2013?**	**root(rejoined), xcomp(Take), nmod(2013)**

Having classified the question using Stanford NLP, the algorithm has further sorted the classified data using the method discussed in section 4.2 to comprehend the question. This sorting process used by the algorithm is one of the contributions to this research as explained in Chapter 4. The sorting process helps the algorithm figure out what the question is about. All the rows Bolden in Table 7.1 show the questions that the algorithm fully understands. With the correctly comprehended question needs, the algorithm will be able to pick the correct answer to the question in most cases as seen in the experiments conducted. Taking question 8 as an example, the algorithm did not capture “London” and “2012” in its comprehension which are meant to be main dependencies in the question; this increases the chances of the algorithm failing the question. The questions classification also shows 3 depth of “*nmod*” dependencies which made the algorithm confused.

Good question comprehension has helped the algorithm in achieving good results as seen in table 6.2 and 6.4. The question understanding improvement was done by mixing keyword matching technique, POS matching technique, word relationships and dependencies structures together. This method is an improvement to the studied concepts described in chapter 2 where one or two of these techniques were only applied.

The aims of this research have been met by analysing questions deeper using dependencies as a main part of the question understanding which produces higher passage comprehension and in turn, produces better results.

7.3.2 Case 2 (Answer Retrieval)

Case 2 discusses the performance of the algorithm as demonstrated in section 6.2 when tested against other commercially used natural language processing applications mostly called personal assistants. To retrieve correct answers to questions, the algorithm made use of the question comprehension as evaluated in Case 1 above. Three evaluation matrixes were used for this case study; Accuracy, Error and Recall (in percentages -%) with the aim to justify that the algorithm’s model will perform better than the current existing NLP algorithm as discussed in Chapter 4.

In Table 6.4, the performance of the algorithm was better than all the other systems followed by Google's personal assistant. From Table 5.3, 46% of questions failed by algorithm were failed by GPA as well. The algorithm correctly answered 65% of the question's GPA failed whereas GPA was only able to answer 53% of the algorithm's failed questions correctly. The results obtained from the other test applications were relatively low as shown in Table 6.4 compared to an average of 50%. The best result was produced by the algorithm which can be used as evidence to proof the knowledge representation process (which accounts for mapping word dependencies amongst other variables as explained in chapter 4) has helped the algorithm to successfully retrieve more correct answers from the experimental questions than any of the other test cases.

From these use cases and comparison analysis, the initial aims outlined in chapter one of this thesis has fully been met and knowledge contributed has been identified.

7.4 Algorithms Complexity Analysis

In theoretical computing, two of the main areas for algorithm evaluation are **The Big O Notation** and **Algorithm Complexity Analysis.** They have their uses and usually turn out to be practically relevant for applications. Some of these methods mentioned above **Asymptotic Behaviour, Worse Case Analysis and the Big O** will be used in evaluating this algorithm. Complexity analysis is the formal process used in this research to evaluate and measure how fast the algorithm works independent of an operating system, programming language or a computers resource. Using complexity analysis seems to be one of the best ways to independently measure the performance of the algorithm. Other profiling software exists, which can measure the running time and help a developer in code optimisation by identifying bottle necks. While these tools prove useful, they are not the same as complexity analysis because complexity analysis is used to compare algorithm at an ideal level ignoring basic details like instruction sets of a CPU, the hardware powering the algorithm or the implemented programming language (Hogan, 2011). Using this technique to evaluate the algorithm will equally show how the algorithm behaves as input grows larger or its behaviour when fed with different sets of inputs.

For this analysis, the major areas to be evaluated in the algorithm will be

1. Knowledge representation
2. Answer retrieval

Counting instructions is one of the processes used while doing complexity analysis. This approach will not be used in this evaluation. This is because it will be very inaccurate to count all the instructions a large algorithm will have. So this method is prone to a lot of mistakes. An example of counting instruction method is shown below. All the fundamental instructions of the pseudo code shown below are counted; this will only be done once.

```
var k = P[0]
for (i=0; i<n; i++;) {
//some condition
}
```

If number of words (n) equals 10, the formula for this instruction will be f(n) = 4+2n; where 4 is the number individual instructions the algorithm processes.

- *var k*
- *P[0]*
- *I = 0*
- *i < n*

and 2n

- *i++*
- *i < n*

is dependent on how many times the loop is run. Because this process of counting instruction still depends on the compiler of each individual programming language used, and complexity analysis is about calculating independent analysis, therefore, the result will do away with dependencies. The $f(n)$ will now become $f(n) = n$

The **Asymptotic Behaviour** obtained after dropping all constant factors will only account for the function n as it tends to infinity. If the program does not have a loop, it automatically becomes $f(n) = 1$ and any program that has a single loop will be $f(n) = n$ (**linear complexity**). When 2 nested loops are found, in an algorithm, the $f(n) = n^2$ (**rule of thumbs**). If $f(n) = n$, therefore it can be said that $f(n) = \theta(n)$ (**time complexity or complexity of the algorithm**). An algorithm of $\theta(n)$ is of n complexity. A program with larger θ run slower than programs with smaller ones.

It is often difficult to determine the exact behaviour of a very complex algorithm but it is safe to assume that while using this method, the behaviour will never exceed certain bounds. For nested loops that do not have the same limit, an assumption to alter the algorithm to its worst "n" state will be done. By doing this, it is known that the algorithm cannot perform worse than its altered state **Upper Bond -** $O(n^2)$ pronounced $Big\ Oh\ of\ (n^2)$. Because the algorithm cannot perform worse than the alter state, this means the performance of the algorithm is either the same as its altered state or better.

Therefore, $O(n^2)$ may not be equal to $\theta(n^2)$ but $\theta(n)$ is also $O(\mathrm{n})$ and $O(n^2)$; taking note that $O(n^2)$ is the modified maximum error upper bond and the algorithm cannot behave worse. In addition to finding the worst state, the best performance will also be found represented by the notation, $\Omega(\mathrm{n})$. This is not always easy to achieve but it helps to show within the bonds the algorithm operates.

Assumptions made for the calculation:

1. Average word count for text of a question will not be longer than 15 words.
2. Average word count for text of a sentence will not be longer than 15 words.

3. Best average word count for a normal sentence will be 8words.
4. Maximum number of corresponding answers to a question will be 10 rows

Phase one (knowledge representation), a scenario of 1,000, 5,000, 10,000, 20,000, 30,000, 40,000, 50,000 words will be calculated to determine software complexity. Phase two (answer retrieval algorithm) will be tested with a question of 8 and 15 words of length to calculate the difference.

7.4.1 Knowledge Representation algorithm

Table 7.2 and 7.3 shows the results for the algorithm when it processes a passage of "n" number of words.

Table 7.2 - Complexity calculation for knowledge representation

Loops found	**Complexity** $\boldsymbol{\theta(n)}$	**Lower Bond** $\boldsymbol{\Omega(n)}$ **(Best Case)**	**Upper Bond** $\boldsymbol{O(n)}$ **(Worst Case)**
$\boldsymbol{f(n) = n + (n^5)}$	$f(n) = \theta(n^5)$ *from Asymptotic Behaviour*	$f(n) = \frac{\Omega(n^3)}{8} + \Omega(n^2)$	$f(n) = O(n^5)$
		Assumptions	
where 3 of the limits $(\boldsymbol{n^5})$ are dependent on sentences while 2 are dependent on words		The sentences are maximum of 8 words.	where all limits are dependent of word count.

Table 7.3 - Complexity calculation for knowledge representation

Words	**Complexity** $\boldsymbol{\theta(n)}$	**Lower Bond** $\boldsymbol{\Omega(n)}$ **(Best Case)**	**Upper Bond** $\boldsymbol{O(n)}$ **(Worst Case)**
1,000	$1000^5 = 1 x 10^{15}$	$\frac{(1000^3)}{8} + 1000^2 = 126000000$	$1000^5 = 1 x 10^{15}$
5,000	$5000^5 = 3.125 x 10^{18}$	$\frac{(5000^3)}{8} + 5000^2 = 1.565 x 10^{10}$	$5000^5 = 3.125 x 10^{18}$
10,000	$10000^5 = 1 x 10^{20}$	$\frac{(10000^3)}{8} + 10000^2 = 1.251 x 10^{11}$	$10000^5 = 1 x 10^{20}$
20,000	$20000^5 = 3.2 x 10^{21}$	$\frac{(20000^3)}{8} + 20000^2 = 1.0004 x 10^{12}$	$20000^5 = 3.2 x 10^{21}$

30,000	$30000^5 = 2.43\ x\ 10^{22}$	$\frac{(30000^3)}{8} + 30000^2 = 3.3759\ x\ 10^{12}$	$30000^5 = 2.43\ x\ 10^{22}$
40,000	$40000^5 = 1.024\ x\ 10^{23}$	$\frac{(40000^3)}{8} + 30000^2 = 8.0016\ x\ 10^{12}$	$40000^5 = 1.024\ x\ 10^{23}$
50,000	$50000^5 = 3.125\ x\ 10^{23}$	$\frac{(50000^3)}{8} + 50000^2 = 1.56275\ x\ 10^{13}$	$50000^5 = 3.125\ x\ 10^{23}$
100,000	$100000^5 = 1\ x\ 10^{25}$	$\frac{(100000^3)}{8} + 100000^2 = 1.2501\ x\ 10^{14}$	$100000^5 = 1\ x\ 10^{25}$

The graphical images in the Figure 7.1 and 7.2 show this mathematical reading in graphical line form. In Figure 7.3, the algorithm is expected to operate anywhere within the best state and worse state. These images show the number of instructions the algorithm will run in best and worst state as the n tends to infinity.

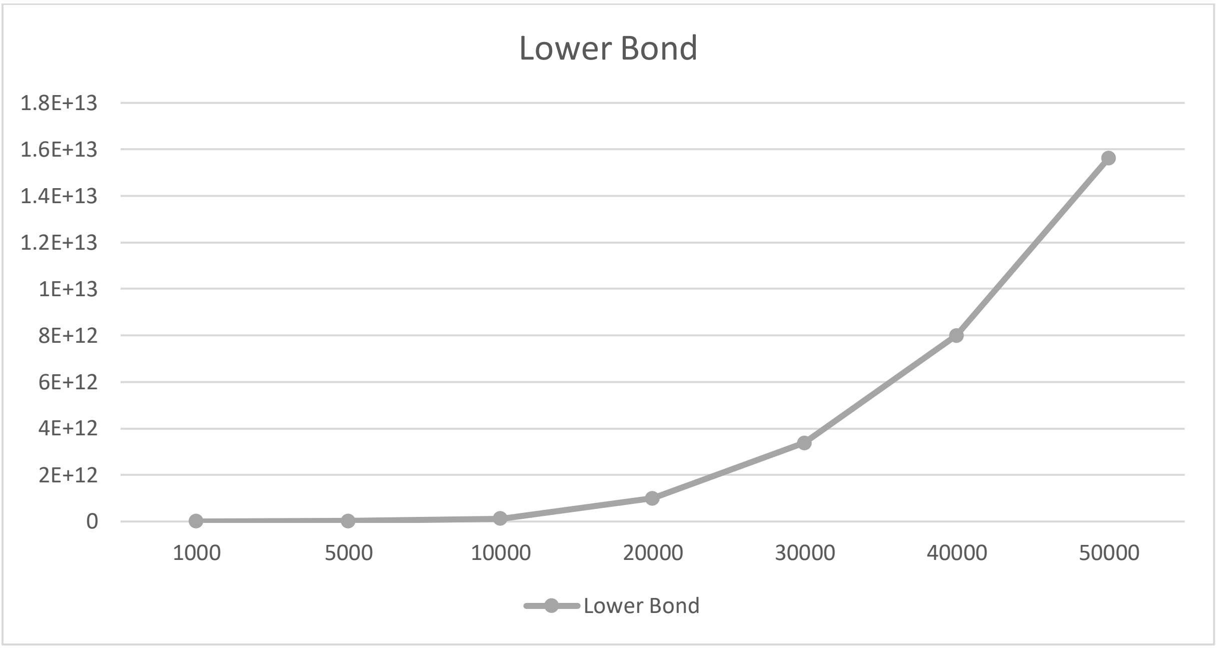


Figure 7.1 - Graphical result of knowledge representation complexity (Lower Bond)

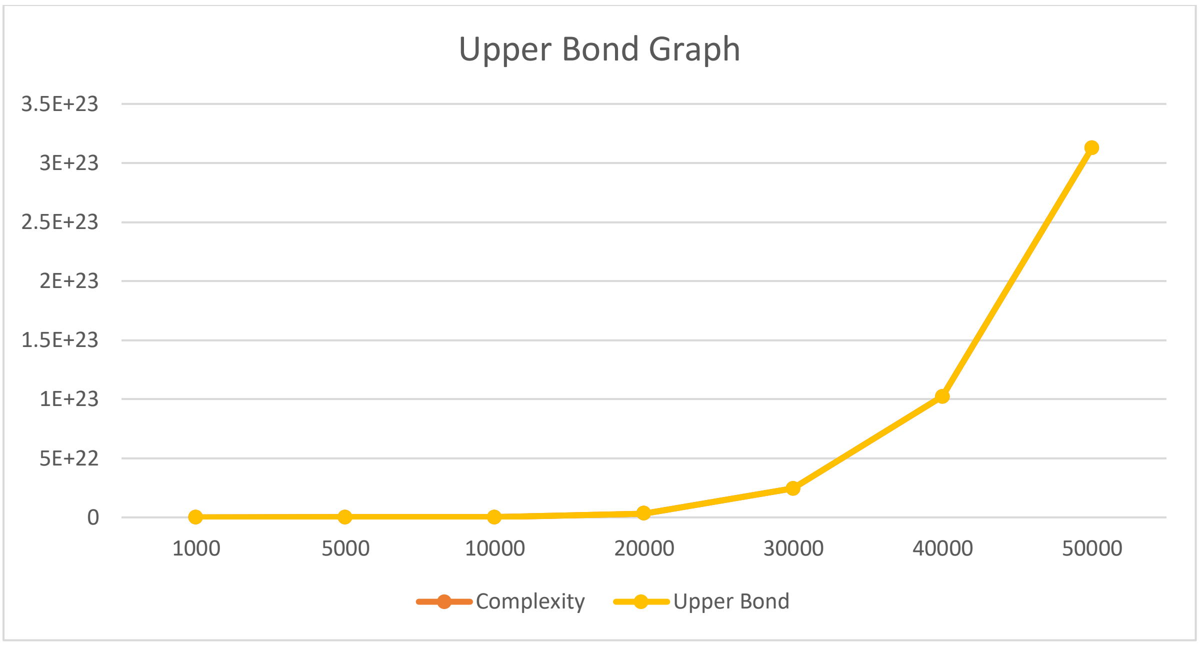


Figure 7.2 - Graphical result of knowledge representation complexity (Upper Bond)

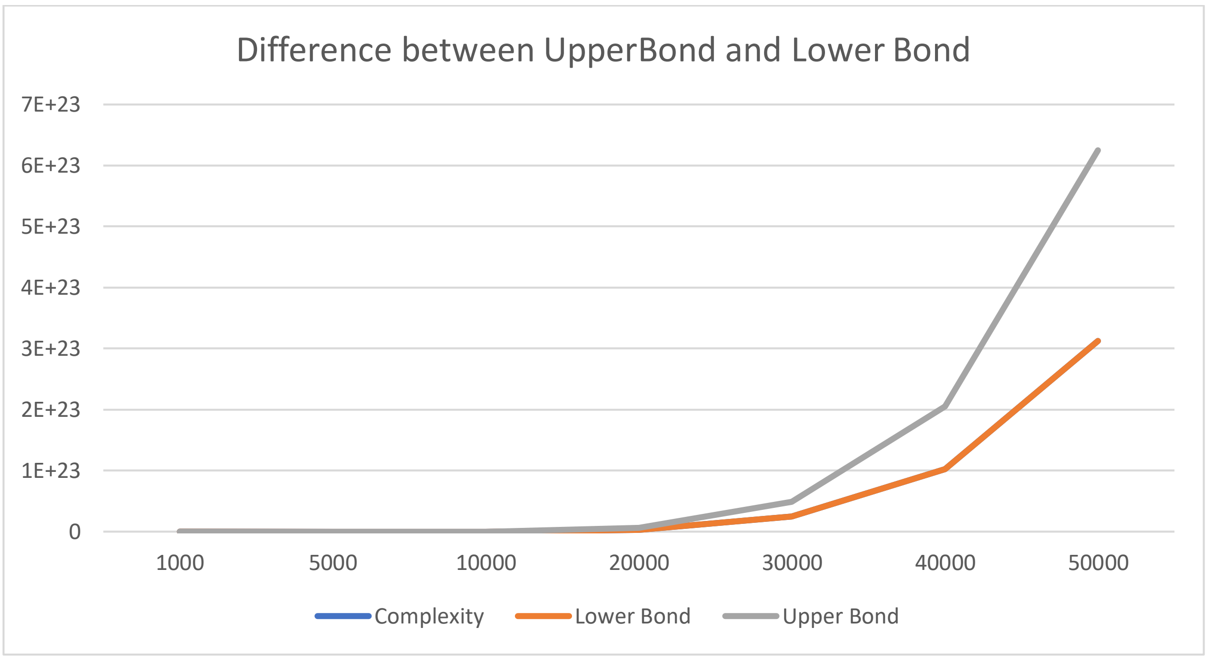


Figure 7.3 - Showing the difference between the lower bond and upper bond

The growth that exists in this is depending on: the performance of a CPU and programming language. These instructions can be converted to time values. The growth shown in the graph as n tends to infinity is an **exponential growth**.

7.4.2 Answer retrieval algorithm

Table 7.4 – explanation of the answer retrieval algorithm calculation

Loops found	**Complexity $\theta(n)$**	**Lower Bond $\Omega(n)$ (Best Case)**	**Upper Bond $O(n)$ (Worst Case)**
$\boldsymbol{f(n) = 5n + (n^5) + (n^3)}$	$f(n) = \theta(n^5 + n^3)$ from Asymptotic Behaviour	$f(n) = \Omega(n^5)$	$f(n) = O(n^5 + n^3)$
Assumptions			
where 3 of the limits $(\boldsymbol{n^5})$ are dependent on sentences while 2 are dependent on words). $(\boldsymbol{n^3})$ is the limit of nested loops for machine learning process to generate an answer to a question.		$\theta(n^3)$ is a condition which if not met will = $\theta(\mathrm{n})$ which is n. The assumption made is that words in the questions are a maximum of 8.	where all limits are dependent of word count and the assumption made is that words in the questions are a maximum of 15. For the upper bond, it is assumed that $\theta(n^3)$ will always meet the condition

Table 7.5 - calculating the difference between upper and lower bonds for answer retrieval algorithm

Words	**Complexity $\boldsymbol{\theta(n)}$**	**Bond $\boldsymbol{\Omega(n)/O(n)}$**
5 (Lower Bond)	$8^5 + 8^3 = 33280$	$8^5 + 8^3 = 33280$
8 (Upper Bond)	$15^5 + 15^3 = 762750$	$15^5 + 15^3 = 762750$

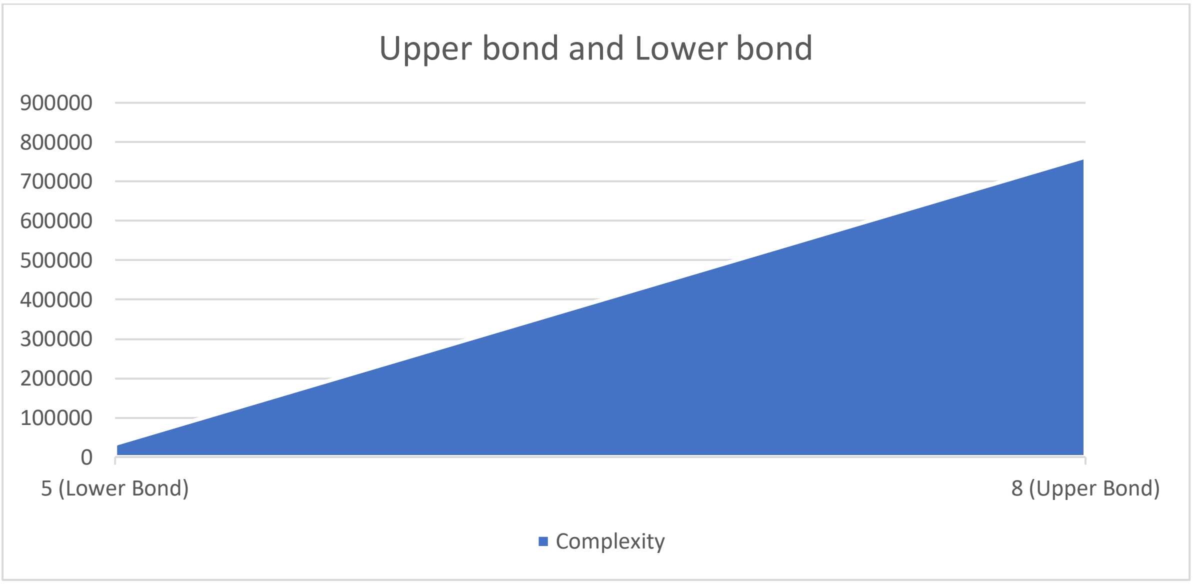


Figure 7.4 - Showing the difference between the upper and lower bond for answer retrieval algorithm

Figure 7.4 shows the difference in calculation of complexity that will arise when the algorithm is operating at its best state against its worst stage. Mathematically, this is a **linear growth** difference that exists between the system at its best and worst.

7.5 Conclusion

From the evidences shown in the evaluation, several observations can be made in this thesis. The results are positive as indicated by the matrix in favour of the algorithm when compared to matrix of other systems. When very complex questions were asked, the negative score of the algorithm increase as opposed to when the questions were less complex. It would be evident at this point to justify that combining word dependencies in a sentence with key words and part of speech matching will produce more accurate results which was demonstrated in the experiment conducted in Chapter 5.

All research questions as identified in chapter one have been answered. The list below shows the questions and how they have individually been addressed.

1. Is it possible for an algorithm to understand a sentence semantic structure?

   Yes, from the background review done in the second chapter, section 2.6 has addressed this question.

2. Is it possible for an algorithm to identify a sentence subject and other sentence attributes?

   Yes, it is possible. Section 2.5 discusses in detail how sentence analysis solves this problem.

3. What are the existing NLP methods available used in question and answer systems?

   Section 2.1-2.6 reviewed and discussed existing NLP methods currently used in existing autonomous and question answering systems.

4. What are the limitations of these methods?

   By carrying out preliminary tests on different systems in section 3.1.2, certain limitations of these systems were identified and discussed.

5. Is it possible to improve these methods and reduce the knowledge gap?

   Yes. Chapter 4 discusses the process adopted in this thesis to answer this question. Chapter 5 and 6 uses experimentation and analysis to show the performance of the new method.

Chapter 8. Conclusion

This research has used different combinations of research methodologies such as experimental, theoretical and systems design methods which helped in eliminating possible limitations of individual methods. The principle contribution of this thesis is the definition of an alternative process on how a machine can understand natural language questions easier:

- *Creating a knowledge base designed to make answer retrieval easier*. Unlike traditional knowledge bases designed for multiple scenarios, the knowledge base discussed in section 4.4.1 was created specifically to make answer retrieval easier using dependency models. This design was done to save all the properties of a sentence or passage normally obtained from language classifiers. The design equally considered ways of improving the querying relationships between entities.
- *Using several techniques for natural language processing*. Passage classification was primarily done using language classifiers. Word dependencies were then used to map sentence words to form meaningful relationships which helped the querying process for answer retrieval.
- *Adding dependency technique to get correct answers to questions*. The algorithm was engineered to use unique dependency patterns to find correct answers. This is used in addition to old methods like keywords matching and part of speech matching.

Hence, this chapter concludes the research undertaken in improving answer predictions in question and answer domain. The aim of this thesis has been met by creating a method that records a higher accuracy in question answering.

8.1 Future Work

A thesis can be seen as an apprenticeship where skills are initially developed and on completion of the apprenticeship, the gained skill set can then be expanded and advanced further until the outcome becomes very good. This thesis should be seen as the beginning of a refined process which still needs more refinement. Research in NLP is continually and actively in progress. Since the inception, several systems and methods of processing NLP have come into existence. To date, most of the methods used or applications released have had drawbacks to properly understand complex questions. There is a lot of potential for future work in on this thesis, both theoretically and experimentally. The summary of future work for updates on this research is as follows:

- During analysis of results carried out in Chapter 6, the algorithm was observed to have more fails than passes with very complex questions. These types of questions were questions that usually have more than 3 level depth of dependencies. For example, “Who was the person who dropped their red bag on my reading table yesterday”. Following the algorithms answering process, this question has about 5 depths of dependencies.

    1. Red
    2. Bag
    3. Reading
    4. Table
    5. Yesterday

  For the algorithm to produce the correct answer it needs to appropriately match all these dependencies to correctly get the right answer. At this point, from experimentation, more fails were recorded than passes. On some occasion, the algorithm recorded passes on extremely complex questions, but this was not consistent as the passes recorded on questions with 1-3 levels of dependency. Improvements can still be made on the methods used on the algorithm to account for deeper levels of questions understanding. However, compared to other experimented systems, the algorithm scored more points in 2 and 3 level dependencies, than any other system.

- Another area that needs further work is the question types. This research focused on question types about *"who or whom"* has a case but there are still more question types like What, Where, When, Why and How that needs to be investigated to find out how this current method can fit into those question types.
- Because the algorithm depends on classifiers for its initial classification, if something is classified incorrectly, then the whole algorithms process afterwards could be jeopardised. Figure 3.3 sheds light on this; when the classifier was fed with random bogus meaningless words, these meaningless words were still classified. In such scenario, the implement in this research can be further investigated to handle issues like this.

To this end, there is still a lot to do in future relating to this research and possibly making it into a household application at some point.

References

A* search algorithm for question answering. (2013) (1st ed.). Japan. Retrieved from http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings3/NTCIR3-QAC-MoriT.pdf

AIML Tutorial. (2017). Retrieved 17 April 2017 from https://www.tutorialspoint.com/aiml/

Alchemy Language demo (2017). *Alchemy-language-demo.mybluemix.net*. Retrieved 6 April 2017, from https://alchemy-language-demo.mybluemix.net/

Alice the Chatterbot. (2017). Retrieved 24 March 2017 from https://www.pandorabots.com/pandora/talk?botid=a847934aae3456cb

Allen, J. (1987). *Natural language understanding*. Menlo Park, Calif.: Benjamin/Cummings Pub. Co.

Avison, D., & Fitzgerald, G. (2003). *Information systems development (pp. 1-45)*. New York: McGraw-Hill.

Ballard, K. (2001). *The frameworks of English*. Basingstoke: Palgrave.

Bambuto, R. (2019). 50 General Knowledge Questions Only A Know-It-All Can Answer. Retrieved 08 January 2019 from https://www.thequiz.com/50-general-knowledge-questions-only-a-know-it-all-can-answer/

Bubien, M. (2017). *Story Bytes - Very Short Stories - "A Prayer so Powerful"*. *Storybytes.com*. Retrieved 27 July 2017, from http://storybytes.com/view-stories/2004/prayer-so-powerful.html

Bureman, L. (2012). *Subjects and Predicates: Breaking Down Sentences - The Write Practice*. *The Write Practice*. Retrieved 25 February 2016, from http://thewritepractice.com/breaking-down-sentences/

Caplinskas, A., & Vasilecas, O. (2014). *Information systems research methodologies and models*. *Ecet.ecs.uni-ruse.bg*. Retrieved 14 August 2016, from http://ecet.ecs.uni-ruse.bg/cst04/Docs/sIV/44.pdf

Center, T. Open Information Extraction. Retrieved from http://openie.allenai.org/

De Marneffe, M., & Manning, C. (2008). *Stanford typed dependencies manual* (1st ed.). Retrieved from http://nlp.stanford.edu/software/dependencies_manual.pdf

Different Types Of Sentences & Their Examples. (2016). *Fos.iloveindia.com*. Retrieved 30 March 2016, from http://fos.iloveindia.com/types-of-sentences.html

Duma, D., & Klein, E. (2017). *Cite a Website - Cite This For Me*. *Aclweb.org*. Retrieved 8 November 2016, from http://www.aclweb.org/anthology/W13-0108

Death on the Nile. (2018). *En.wikipedia.org*. Retrieved 6 March 2018, from https://en.wikipedia.org/wiki/Death_on_the_Nile

Engadget,. (2011). *IBM's Watson Supercomputer Destroys Humans in Jeopardy | Engadget*. Retrieved from https://www.youtube.com/watch?v=WFR3lOm_xhE

Farghaly, A., & Shaalan, K. (2009). Arabic Natural Language Processing: Challenges and Solutions. *ACM Transactions on Asian Language Information Processing (TALIP), 8(4), 1–22*. http://doi.org/10.1145/1644879.1644881

General Knowledge Quiz with Answers (160 Questions) - Fun Quizzes. (2018). Retrieved from https://www.funquizzes.uk/general-knowledge-quiz-answers/

General Knowledge Quizzes - ReadyMadePubQuiz.com. (2018). Retrieved from https://readymadepubquiz.com/tag/general-knowledge/

Goldberg, A. (2003). Constructions: a new theoretical approach to language. *Trends in Cognitive Sciences*, 7(5), pp.219-224.

Grammar, E. *Question Words in English - Who When What Why Which Where How*. *Grammar.cl*. Retrieved 6 April 2016, from http://www.grammar.cl/Notes/Question_Words.htm

Gray, T. (2017). *The Secret behind the Computational Engine in Wolfram|Alpha—Wolfram|Alpha Blog*. *Blog.wolframalpha.com*. Retrieved 26 October 2017, from http://blog.wolframalpha.com/2009/05/01/the-secret-behind-the-computational-engine-in-wolframalpha/

Hasan, H. (2004). *Information Systems Development as a Research Method*. *Journal.acs.org.au*. Retrieved 20 November 2015, from http://journal.acs.org.au/index.php/ajis/article/view/142/122

Hinaut, X., Petit, M., Pointeau, G. and Dominey, P. (2014). Exploring the acquisition and production of grammatical constructions through human-robot interaction with echo state networks. *Front. Neurorobot.*, 8. – Page 12 of (Hinaut et al., 2014) about grammatical structure

Hogan, S. (2011). *A gentle introduction to computational complexity theory, and a little bit more*[Ebook]. Retrieved from https://www.math.uchicago.edu/~may/VIGRE/VIGRE2011/REUPapers/Hogan.pdf

Hu, S., Zou, L., Yu, J., Wang, H., & Zhao, D. (2017). Answering Natural Language Questions by Subgraph Matching over Knowledge Graphs. *IEEE Transactions on Knowledge And Data Engineering*, 1-1. http://dx.doi.org/10.1109/tkde.2017.2766634

IBM Watson. (2017). *IBM Watson*. Retrieved 6 April 2017, from https://www.ibm.com/watson/

IBM, (2017). *Watson Conversation Service Overview*. Retrieved from https://www.youtube.com/watch?v=1rTl1WEbg5U

Kilani, M., & Kobziev, V. (2016). *An Overview of Research Methodology in Information System (IS)*. *Scientific Research Open Access*. Retrieved 20 November 2017, from http://file.scirp.org/Html/71775_71775.htm

Microsoft Store. (2016). *Eliza – Windows Apps on Microsoft Store*. Retrieved 9 February 2016, from https://www.microsoft.com/en-us/store/apps/eliza/9wzdncrdf9pn

Misra, D., Sung, J., Lee, K., & Saxena, A. (2015). Tell me Dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, *35*(1-3), 281-300. http://dx.doi.org/10.1177/0278364915602060

MORI, T., OHTA, T., FUJIHATA, K., & KUMON, R. (2001). A* search algorithm for question answering. *Proceedings Of The Third NTCIR Workshop*, 1-8. Retrieved from http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings3/NTCIR3-QAC-MoriT.pdf

MySQL Top Reasons to Use MySQL. (2016). *Mysql.com*. Retrieved 27 April 2016, from https://www.mysql.com/why-mysql/topreasons.html

Negro, A. (2017). *Mining and Searching Text with Graph Databases*. *GraphAware*. Retrieved 7 August 2016, from https://graphaware.com/neo4j/2016/07/07/mining-and-searching-text-with-graph-databases.html

Neo4j. (2017). Retrieved from https://www.youtube.com/watch?v=U8ZGVx1NmQg

Nicula, B., Ruseti, S., & Rebedea, T. (2015). *Enhancing Property and Type Detection for a QA System over Linked Data*. *Research Gate*. Retrieved 9 November 2017, from https://www.researchgate.net/publication/289674143_QAnswer_-_Enhanced_Entity_Matching_for_Question_Answering_over_Linked_Data

Nunamaker Jr, J. F., & Chen, M. (1990, January). Systems development in information systems research. In System Sciences, 1990., *Proceedings of the Twenty-Third Annual Hawaii International Conference on (Vol. 3, pp. 631-640)*. IEEE.

Princeton University, P. (2019). WordNet | A Lexical Database for English. Retrieved 12 April 2016, from https://wordnet.princeton.edu/

RDF - Semantic Web Standards. (2017). *W3.org*. Retrieved 8 November 2017, from https://www.w3.org/RDF/

Russell, S. J., & Norvig, P. (2010). *Artificial intelligence: A modern approach* (3rd, International ed.). Boston, [Mass.]; London;: Pearson.

Selecting a development approach. (2016) (1st ed., pp. 1-10). USA. Retrieved from https://www.cms.gov/research-statistics-data-and-systems/cms-information-technology/xlc/downloads/selectingdevelopmentapproach.pdf

SEMPRE: Semantic Parsing with Execution. (2017). *Nlp.stanford.edu*. Retrieved 8 November 2017, from https://nlp.stanford.edu/software/sempre/

Soon, W., Ng, H., & Lim, D. (2001). A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, *27*(4), 521-544. http://dx.doi.org/10.1162/089120101753342653

Stanford CoreNLP. (2015). Retrieved 09 March 2017 from http://nlp.stanford.edu:8080/corenlp/process

Stanford CoreNLP – *Natural language software | Stanford CoreNLP*. (2017). *Stanfordnlp.github.io*. Retrieved 20 October 2017, from https://stanfordnlp.github.io/CoreNLP/

Christopher D., M., Mihai, S., John, B., Steven J., B., David, M., & Jenny, F. (2014). *The Stanford CoreNLP Natural Language Processing Toolkit* [Ebook] (pp. 55-60). Retrieved from http://www.aclweb.org/anthology/P/P14/P14-5010

Tellex, S., Dickerson, S., Walter, M., Banerjee, A., Teller, S., Roy, N., & Kollar, T. (2011). Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation. *National Conference On Artificial Intelligence*, *MA 02139*(AAAI 201), 1-7. Retrieved from https://cs.brown.edu/~stefie10/publications/tellex11.pdf

The Systems Development or Engineering Approach to Research in Information Systems*: An Action Research Perspective*. (2016) (10th ed., pp. 122-134). Australia. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.3.5361&rep=rep1&type=pdf

Thesaurus.com - The world's favorite online thesaurus!. Retrieved 31 March 2016, from https://www.thesaurus.com/

Thompson, M., & Winograd, T. (1974). Understanding Natural Language. *Leonardo*, *7*(2), 174. http://dx.doi.org/10.2307/1572810

UIMA, A. Apache UIMA - Apache UIMA. Retrieved from http://uima.apache.org/

Unger, C., Forascu, C., Lopez, V., Ngonga Ngomo, A., Cabrio, E., Cimiano, P., & Walter, S. (2017). *Question Answering Over Linked Data (QALD-5)*. Retrieved from http://ceur-ws.org/Vol-1391/173-CR.pdf

Wallace, R. (2019). Chatbot A.L.I.C.E., A.L.I.C.E. A.I Foundation | Virtual Assistant A.L.I.C.E. | Virtual agent A.L.I.C.E. | Chat bot A.L.I.C.E. | Conversational agent A.L.I.C.E. | (4510). Retrieved from https://www.chatbots.org/chatbot/a.l.i.c.e/

WANG, Y. (2008). Ontology matching approach based on RDF graph. *Journal Of Computer Applications*, *28*(2), 460-462. http://dx.doi.org/10.3724/sp.j.1087.2008.00460

Wolfram|Alpha: Making the world's knowledge computable. *Wolframalpha.com*. Retrieved 21 March 2018, from https://www.wolframalpha.com/

YourDictionary, (2016). *Free Diagramming Sentences Worksheet*. Retrieved 1 March 2016, from http://grammar.yourdictionary.com/sentences/free-diagramming-sentences-worksheet.html

Zindros, D. (2018). A Gentle Introduction to Algorithm Complexity Analysis. Retrieved from https://discrete.gr/complexity/

Appendix

A

Below is outlined a list of helping verbs

- am, are, is, was, were, be, being, been
- have, has, had
- shall, will
- do, does, did
- may, must, might
- can, could, would, should

B

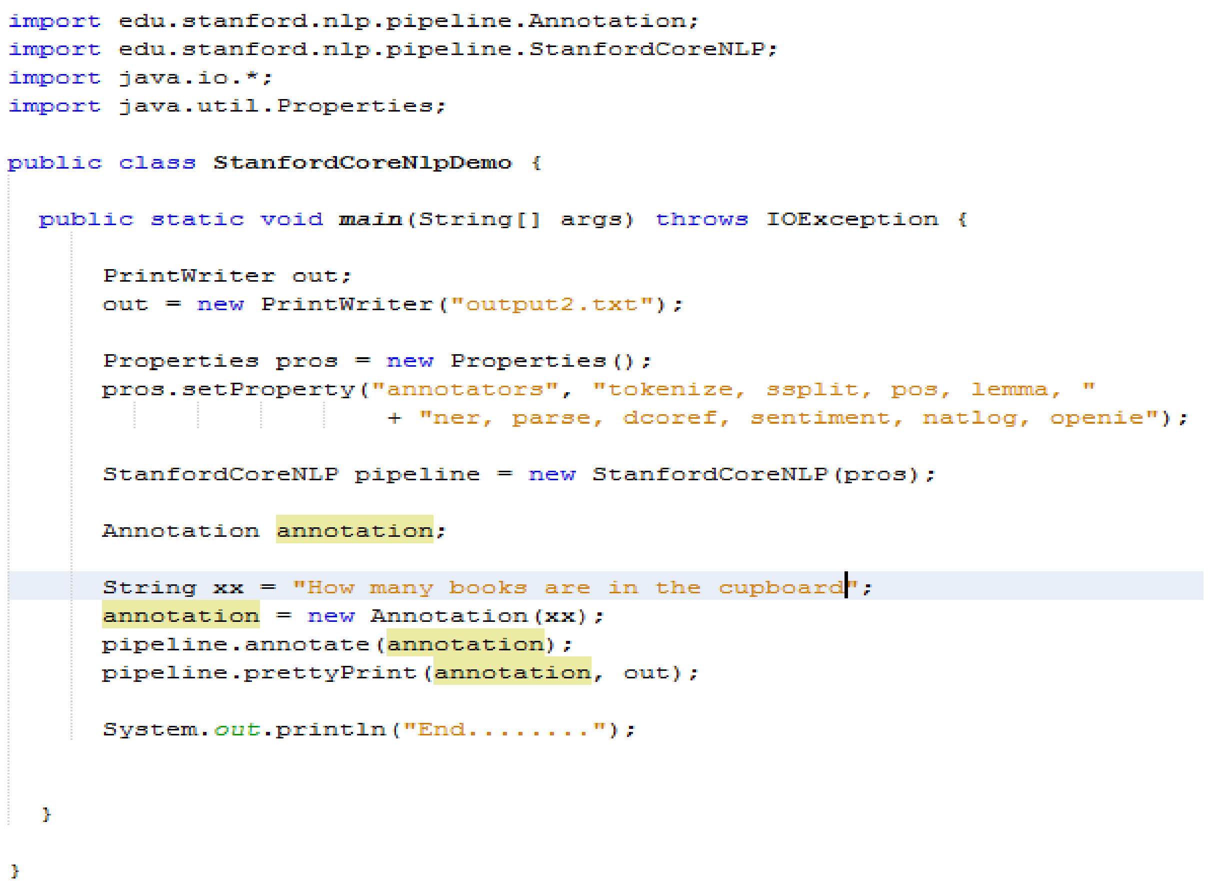
What people would usually assume contributed to word meaning

If one asked other people what their thoughts were on the constitution of language, the most common reply would be words. (Ballard, 2001) For most of the researches done over the years, close study of language has been perceived to be words as the building block of language. Language offers a lot more than simply the study of its lexicon. One can choose to study the smaller units that makes up words or the units of sentence formation. However, words can be a good starting point.

There are nine major word classes, which are nouns, verbs, adjectives, pronoun, preposition, adverbs, auxiliary, conjunction, determiners and two minor ones, which are interjection and particle. There is only a hand full of particles e.g. “to be”, “not to be” etc.

C

Figure C1 below shows a screenshot of the code ran to test The Stanford CoreNLP library using NetBeans IDE.



```
import edu.stanford.nlp.pipeline.Annotation;
import edu.stanford.nlp.pipeline.StanfordCoreNLP;
import java.io.*;
import java.util.Properties;

public class StanfordCoreNlpDemo {

  public static void main(String[] args) throws IOException {

      PrintWriter out;
      out = new PrintWriter("output2.txt");

      Properties pros = new Properties();
      pros.setProperty("annotators", "tokenize, ssplit, pos, lemma, "
                                  + "ner, parse, dcoref, sentiment, natlog, openie");

      StanfordCoreNLP pipeline = new StanfordCoreNLP(pros);

      Annotation annotation;

      String xx = "How many books are in the cupboard";
      annotation = new Annotation(xx);
      pipeline.annotate(annotation);
      pipeline.prettyPrint(annotation, out);

      System.out.println("End........");


  }

}
```

Figure C1 - Sample CoreNLP code ran on NetBeans IDE

The code explanation is:

1. Creates a New instance of the PrintWriter Class
2. Sets the Stanford properties; for example, the property “POS” which is set simply tells the program that it should output the Parts of Speech for each word in the sentence when it is run.
3. The properties are passed as a variable to the Stanford pipeline
4. Annotations hold the sentence(s)

5. One or more annotations can then be passed to the pipeline in this case, it is just one.
6. The "Pretty Print" command prints out the properties of the annotation in human readable format through the preferred output stream transfer which has been added as a parameter (out) to the pipeline.

D

Below is a list showing 3 popular research methods applied mostly in the field of management research. Due to the popularity of these methodologies, their application in this research was considered.

Qualitative Approach

(Kilani & Kobziev, 2016) explains the qualitative research approach as an approach that often depends on the explanatory or critical paradigm in social sciences. Qualitative research helps researchers analyse cultural and social occurrences; this approach represents collection of data which is field dependent on several life situations for example human behaviours, experiences, values etc. Data in qualitative research can be produced in form of sentences, words, narrations, phrases e.g. conversations, explanations, interviews etc. rather than only numbers. This makes the collected data have qualitative richness and strong complexity potentials by focusing on problems that exist in their direct cultural and social environments.

Quantitative Approach

This approach involves interpretation of numeric data such as numbers, interval, percentages, ratio etc. and representing the readings with items like diagrams and graphs to present perfect and meaningful results. The approach is described as taking extreme practicality because it depends on explanation and control of the occurrence. Researchers that apply this form of research are usually interested in measuring "how often?", "how many?", or "to what extent?"

Mixed Method

This simply involves the use of more than one form of data collection. It is usually a mix of qualitative and quantitative method, data, methodologies to study a set of events, either related or unrelated events.

E

The passage below was used as one of the test data set used to initially trained the algorithm. This was a random passage obtained from (Gray, 2017).

Margaret Thatcher was the Prime Minister of the United Kingdom from 1979 to 1990 and Leader of the Conservative Party from 1975 to 1990. She was nicknamed Iron Lady. Initially it was termed as an insult but by the time she left office, it became a term of respect and affection for the Russians. Theresa May is the current prime minister of United Kingdom. The previous prime minister of United Kingdom before her was David Cameron. Jeffrey P. Bezos is the president, chief executive officer and chairman of the board. Jeffrey A. Wilke is the chief executive officer, worldwide consumer. Andrew Jassy is the chief executive officer, amazon web service. Benjamin Franklin had one of the greatest scientific minds of his time. He was interested in many areas of science, made many discoveries, and invented many things, including bifocal glasses. In the mid-1700s, he became interested in electricity. He was the man who discovered electricity. A German, Konrad Zuse, created the first programmable computer, the Z1, in 1936.

F

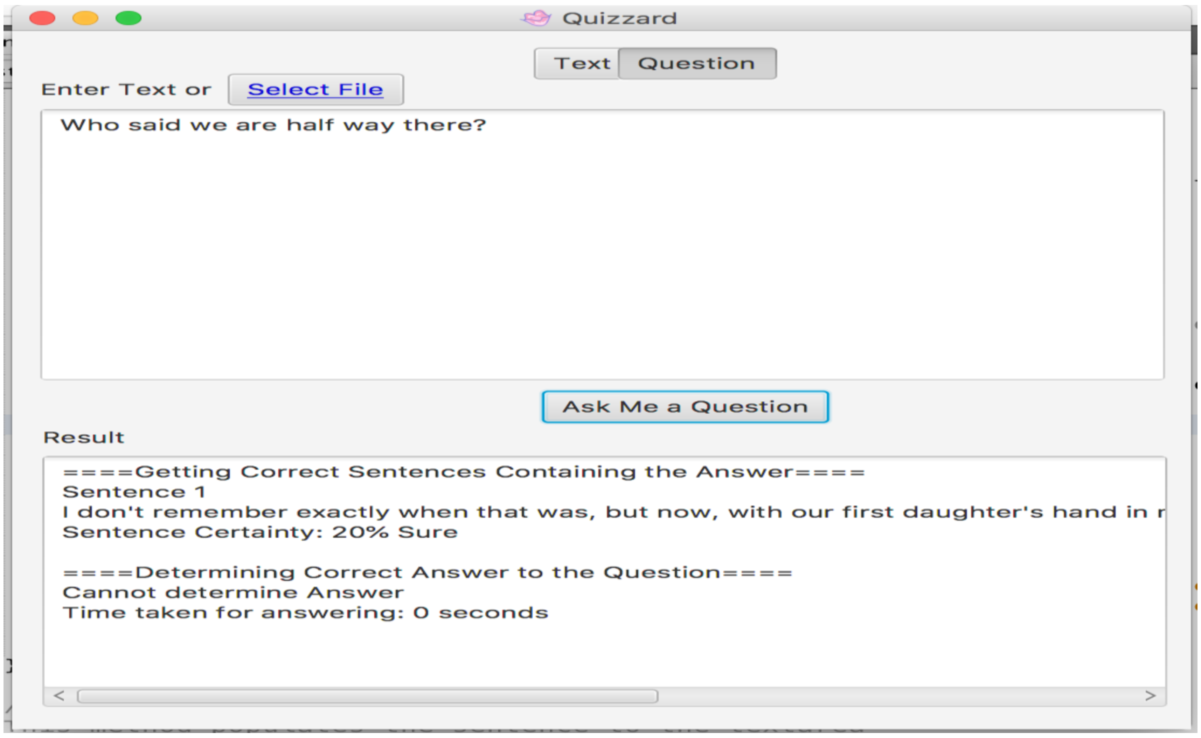
Table F1 below shows a few preliminary questions tested on public domain systems in Chapter 3. This test was conducted to understand the current capabilities and limitations of different NLP systems.

Table F1 - Simple questions asked to public domain systems for preliminary test

SN	Questions
1	What is the time?
2	What is the current exchange rate of USD to GBP?
3	What was the highest exchange rate for USD to GBP yesterday?
4	What is the best seller movie in 2018?
5	Who was the last president of Gambia?
6	What was the price of bitcoin on the 1st January, 2019

G

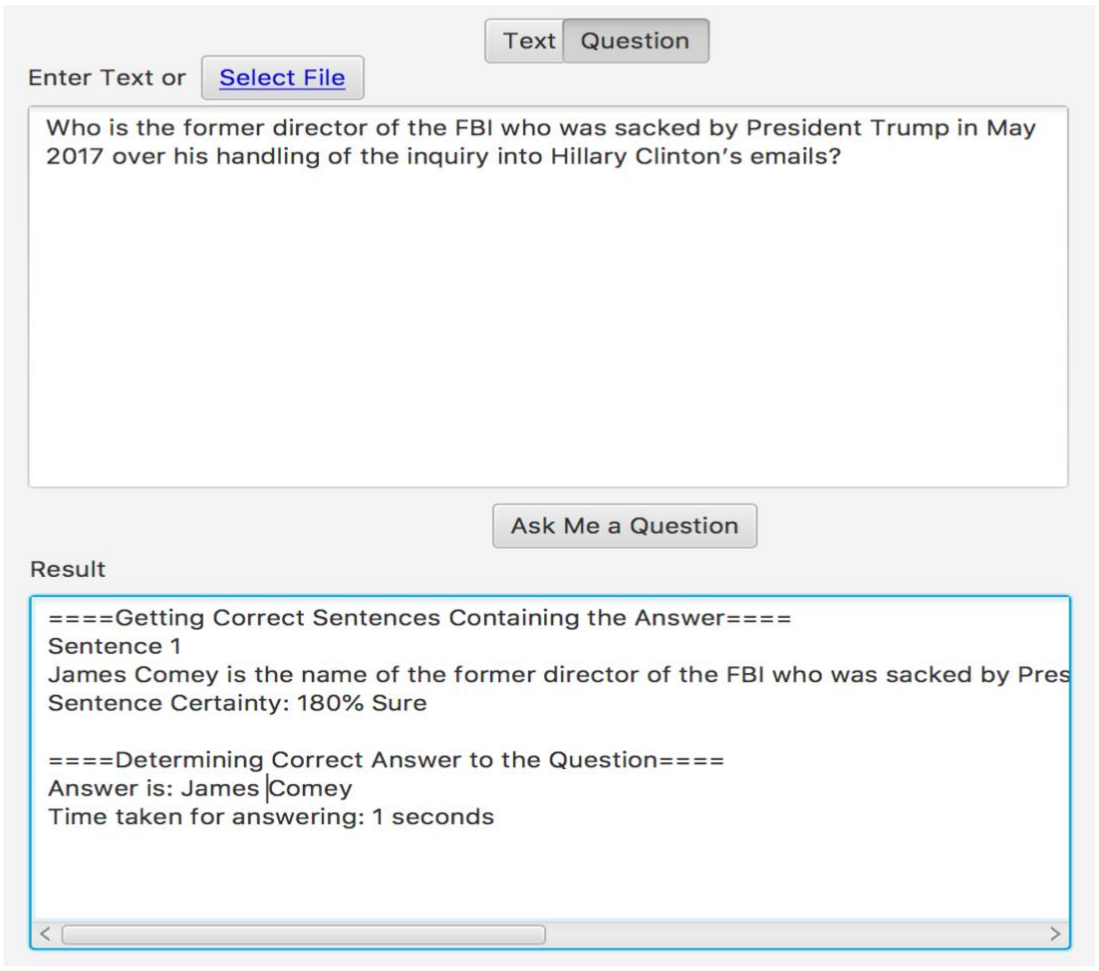
Evidence of questions obtained during the experimental phase of these thesis. Figure G1 and G2 show samples of the prototype when questioned during experimentation.



Who said we are half way there?

```
====Getting Correct Sentences Containing the Answer====
Sentence 1
I don't remember exactly when that was, but now, with our first daughter's hand in r
Sentence Certainty: 20% Sure

====Determining Correct Answer to the Question====
Cannot determine Answer
Time taken for answering: 0 seconds
```

Figure G1 - Sample screenshot of conducted self-test with wrong answer to a question



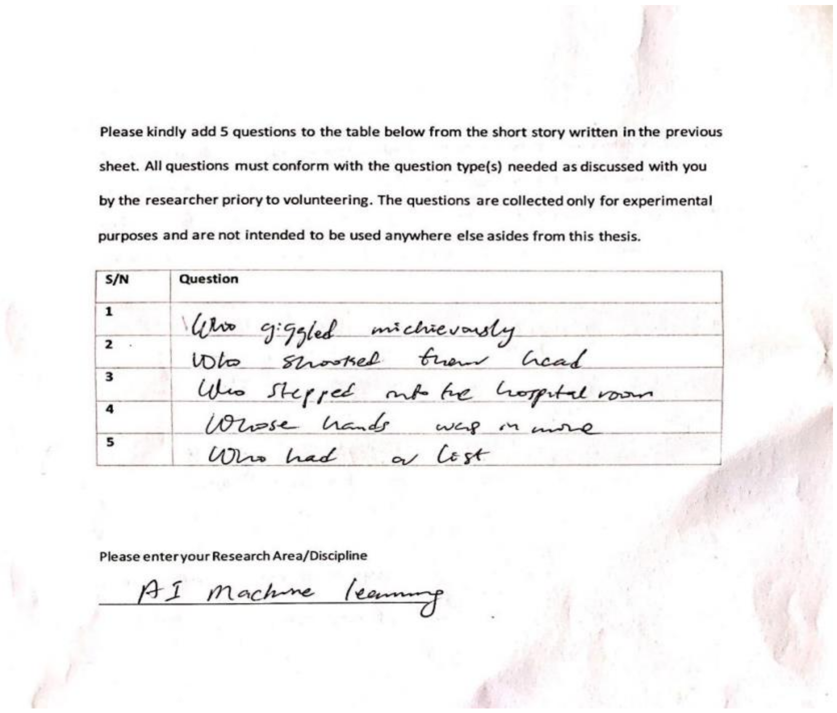
Who is the former director of the FBI who was sacked by President Trump in May 2017 over his handling of the inquiry into Hillary Clinton's emails?

```
====Getting Correct Sentences Containing the Answer====
Sentence 1
James Comey is the name of the former director of the FBI who was sacked by Pres
Sentence Certainty: 180% Sure

====Determining Correct Answer to the Question====
Answer is: James Comey
Time taken for answering: 1 seconds
```

Figure G2 – Another Sample screenshot of conducted self-test with the algorithm showing the correct answer to a question

H

Below are two sample images of questions collected from members of the PARK team during the experimentation stage.



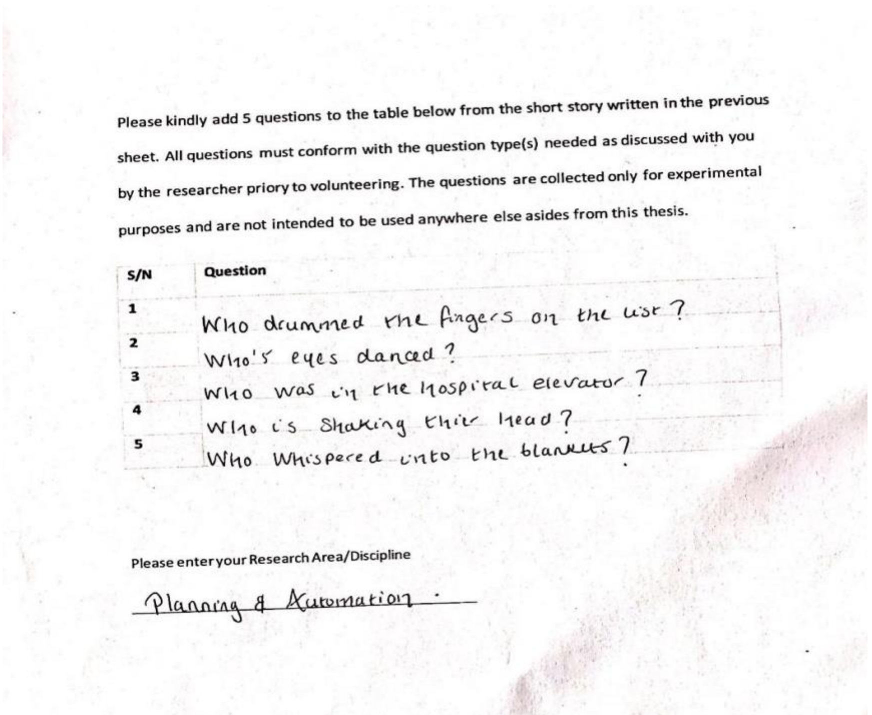
Please kindly add 5 questions to the table below from the short story written in the previous sheet. All questions must conform with the question type(s) needed as discussed with you by the researcher priory to volunteering. The questions are collected only for experimental purposes and are not intended to be used anywhere else asides from this thesis.

S/N	Question
1	Who giggled michievously
2	Who shooked their head
3	Who stepped into the hospital room
4	Whose hands was in mine
5	Who had a list

Please enter your Research Area/Discipline

AI Machine learning

Figure H1 - Sample one of questions collated for experimenting



Please kindly add 5 questions to the table below from the short story written in the previous sheet. All questions must conform with the question type(s) needed as discussed with you by the researcher priory to volunteering. The questions are collected only for experimental purposes and are not intended to be used anywhere else asides from this thesis.

S/N	Question
1	Who drummed the fingers on the list?
2	Who's eyes danced?
3	Who was in the hospital elevator?
4	Who is Shaking their head?
5	Who Whispered into the blankets?

Please enter your Research Area/Discipline

Planning & Automation.

Figure H2 - Sample two of questions collated for experimenting