Imperial College London

INTERIM REPORT

DEPARTMENT OF COMPUTING

IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

Voice Recognition Escape Room

Author:
Syretta Man Nga Yin

Supervisor:
Mark Wheelhouse

January 27, 2022

Submitted in partial fulfillment of the requirements for the MEng Computing of Imperial College London

Contents

Chapter 1

Introduction

1.1 Motivation

An escape room is a puzzle game where a group of players are locked in a room. Their goal is to escape the room by finding clues and solving puzzles. Escape room games are available both in physical and virtual form. However, during the COVID-19 pandemic, physical escape rooms have become less popular due to safety measures. While players can still turn to a virtual online escape room, the sense of involvement is weaker since the interaction with the game involves only pressing buttons or mouse clicking.

Voice recognition has been mostly used for virtual assistants such as Apple's Siri, Amazon's Alexa and Microsoft's Cortana, which will be activated when the "hot word" like "Hey Siri" is said. Other applications of voice recognition include, voice biometry, transcribing a conversation or learning a new language. With the use of virtual reality and motion sensing in games, the gaming industry is not only growing rapidly, but also becoming more and more immersive by the day. The rise of speech recognition technology might be a game-changer in this industry as it promotes an even more pleasurable gaming experience, using our voice.

In previous years, there are countless attempts of integrating voice recognition into video games. However, the game development process is long and challenging. It involves not only creating the plot, but also characters, animation and much more. Incorporating voice recognition into the game brings the challenge of development to the next level. Not only do they need a large library of voices to train the algorithm, but they also need to consider different approaches for different languages, dialects and accents if the game is to publish globally.

As a result, most games apply a rule-based natural language processing approach such as regular expression (REGEX) or Context-Free Grammar (CFG). These approaches make use of pattern matching and associate the input of the user to a specific command in the game. Since they result in high recall but low precision, they can only be applied in certain situations but cannot be generalised. Rule-based

natural language processing is also cumbersome to develop as the programme will have to consider all possible inputs or actions from the user. This would involve a lot of rules for a game. In this project, I would like to explore the use of machine learning and neural network to develop a Natural Language Processing interface that can be integrated into video games.

1.2 Objectives

1.2.1 NLP Interface

The first objective of the project is to compare different tools and algorithms available for different NLP tasks and pick the best ones for selected scenarios. In the end, we are hoping to obtain an NLP interface that could understand the intention of the users and gives out a accurate response. The project also aims to provide as much flexibility as possible so to minimise the need for hard coded rules for speech understanding.

1.2.2 Escape Room Game

The second main objective is to create a basic escape room game for the purpose of demonstrating the NLP interface. At the end of the project, the users should be able to progress through the levels with voice control only.

Chapter 2

Background

2.1 Components of Natural Language Processing

Natural Language Processing is a branch of Artificial Intelligence that enables the computer to understand the text as well as spoken words as much as humans do. [1]

2.1.1 Natural Language Understanding

Natural Language Understanding (NLU) is a subset of NLP which uses syntactic and semantic analysis of text and speech to determine the meaning of a sentence. [2]

2.1.2 Natural Language Generation

Natural Language Generation (NLG) is the process of producing a human language text response based on some data input. [2]

2.2 NLP Pipeline

2.2.1 Lexical Analysis

Lexical Analysis is also called tokenization, which is a process of converting a sequence of characters into a sequence of tokens. [3] Given a specific delimiter, it is easy to apply tokenization to an English sentence. For example, tokenizing the sentence, ”I want a cup of tea”, with space delimiter, will result in 6 tokens: ”I”, ”want”, ”a”, ”cup”, ”of”, ”tea”.

2.2.2 Syntactic Analysis

Syntax refers to the grammatical structure of a sentence, determining the role of work in a sentence or phrase. Syntactic Analysis, also known as parsing, ensures that a given piece of text is in a correct structure. At this stage, techniques like lemmatization, stemming, dependency parsing and sentence segmentation will be

applied. It also uses the Part of Speech(POS) generated from the previous step to analyse the sentence structure [4].The details of how the POS can be obtained is illustrated in Section 2.5.1 Part-of-Speech Tagging.

2.2.3 Semantic Analysis

Semantics mean the intended meanings of the words. So semantic analysis looks for meaning in the given sentence as well as combining words into phrases. For example, the phrase ”hot ice cream” will be disregarded. Lexical semantics is important for understanding the relationship between lexical items. [5] The followings are the relationships that we might encounter throughout the project:

Synonyms

When two or more words have a similar meaning, for example, ”happy” and ”delighted”.

Antonyms

When two or more words have opposite meanings like ”open” and ”close”.

Hyponyms

A word of more specific meaning than a general or superordinate term applicable to it such as ”spoon” is a hyponym of ”cutlery”.

Homonyms

When two or more words have the same spelling or pronunciation but different meanings and origins

Meronomy

A word that denotes a constituent part or a member of something like ”arm” is a meronomy of ”body”.

There are two approaches to the semantic analysis technique: text classification models and text extractors. Text classification models assign the text into predefined categories. For example, Sentiment Analysis is often used for categorising text, like user’s review, into positive, negative or neutral. On the other hand, text extractors take out specific information from the text. For example, Entity Extractor identify named entities from the text to predefined categories such as a person’s name, building’s name.

Word sense disambiguation in Section 2.5.3, Semantic Role Labelling in Section

2.5.5 and relationship extraction Section 2.5.4 in are example approaches that are semantic based.

2.2.4 Discourse Integration

Discourse is a sense of the context. [6] It looks at how the previous sentences affect the sentences afterwards. For example, ”Jeff is running. He is very healthy.” With discourse integration, we can know that the ”he” in the second sentence refers to ”Jeff”. Anaphora Resolution and Coreference Resolution in Section 2.5.6 will look into the problem.

2.2.5 Pragmatic Analysis

The pragmatic analysis deals with outside word knowledge, which means it is not limited to the documents. It finds out what the word meant. [7] For example, without pragmatic, a response to the sentence ”Can you pass the salt?” will be a simple ”yes” or ”no. However, with pragmatism, we should be able to recognise that is a request to pass the salt to the speaker. [8]

2.3 Existing Voice Recognition Games

The followings are the existing games that integrate voice recognition into their gameplay experience.

2.3.1 Albedo

Albedo is a first-person perspective survival game developed by APEStudio [9] Before players can escape from the abandoned manor, they have to explore the place and release the soul that got trapped in the manor. To do so, players are granted special powers that can be activated using their voice. There are in total 4 different magic spells that correspond to different powers. *Luxo* is the power of light, *VERA* is the power to disappear, *ADUL* is the power to kill the evil and *NIMA* is the power to reveal the path of evil.

There was not much NLP technique being used in the game as voice recognition is only used to recognise 4 spells named by the player. Drawing from my personal experience, the voice recognition algorithm in the game is neither sensitive nor accurate. It took a long time for the game to recognise which spell I am pronouncing, which makes it insensitive. Also each of the spells may have a different way of pronouncing, so even when I tried repeating ”ADUL” many times, it did not pick up what I am saying, demonstrating its incapability to handle different pronunciations.

Figure 2.1: A picture of the universe!

2.3.2 Starship Commander: Arcade

Human Interact released Starship Commander: Arcade in 2018. It is a cinematic VR experience where the players are allowed to speak to the non-player characters (NPC) in the story of a sci-fi setting. [10].

The game was released in conjunction with Microsoft [11]. Although the developers from Human Interact has considered developing their NLP technology, they decided to use the widely available tools for their games due to the unforeseeable challenges during the development process.

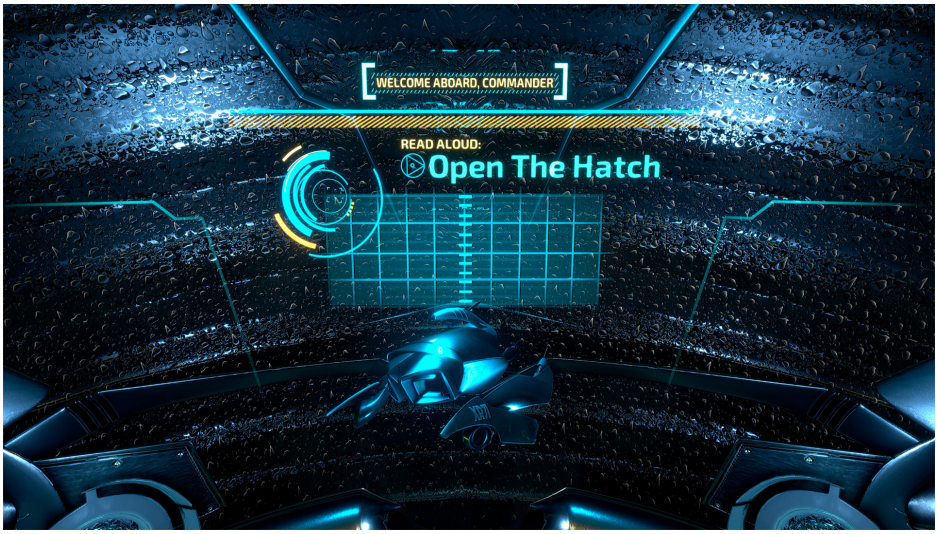
They used the Custom Recognition Intelligent Service (CRIS) for their speech recognition engine and Language Understanding Intelligent Service (LUIS) for the translation between spoken phrases and a number of discrete intention actions for the interactive narrative [12].

2.3.3 Lifeline

Lifeline is a video game released video game company, Konami for the PlayStation 2 in 2004. [13] It is the first video game that is solely controlled by voice.

The story of the game starts with the player joining a party in the space station, and the Space Station is attacked by monsters all of a sudden. The player character is trapped in the control room but has access to all Space Station mechanism and observe the place with the cameras. The player character then notices that Rio, a cocktail waitress from the hotel in the Space Station is held captive in a cell room and tries to contact the control room. The player character will then be able to contact Rio with the communication devices and give commands to Rio of getting out

Figure 2.2: Screenshot from the Gameplay of Starship Commander:Arcade



of the Space Station.

As the player cannot control Rio physically, they have to make sure of the voice control to tell Rio what to do. The player has to press the input mic button on the controller to notify the game that he is giving a command, then he can give commands such as ”hurry”, ”Stop”, ”dodge” etc. When Rio meets a monster, the player could instruct Rio to attack the monster with a combination of commands such as ”Shoot, Left Eye, Reload”, so that Rio could aim at a particular spot of the monster and shoot.

Throughout the game, the player will also have a short conversation with Rio and Rio will be guiding the player to the ending. During the conversation, we could see that the voice control in the game is not restricted to a list of commands. Rio can respond to whatever the player says.

It is great that Lifeline enables conversation in such a human-like way, but there are still some shortcomings in voice recognition in the game. During Rio’s combat, it is difficult to shout out commands in such a short time. Summarizing the reviews on Lifeline, the accuracy of the voice recognition seems to be quite low as Rio cannot comprehend the commands properly during the combat. Also, when the players are exploring the rooms, it seems to be a bit troublesome when specifying certain items. For example, if you told Rio to ”Look at the box”, Rio will likely reply with ”What?”, or if you told her ”The cigar box”, she will reply with something irrelevant like ”I think the chair is totally fine.” [14] Therefore, the another weakness of the voice recognition in Lifeline is its inability to generalise terms, so when you say it in different words, Rio will not able to respond properly.

Figure 2.3: Screenshot from Lifeline when the player is giving out commands to Rio

2.3.4 Facade

The facade is an artificial-intelligence-based interactive story created by Michael Meteas and Andrew Stern in 2005. [15] The story starts with the player character visiting the married couple, Trip and Grace, at their apartment, and becomes entangled in the high-conflict dissolution of their marriage. The player can interact with the characters by actually speaking to them.

Facade makes use of natural language processing technology to understand the players' input. According to [16], the NLP can be divided into two phases. Phase 1 maps surface text into discourse acts representing the pragmatic effects of an utterance while phase 2 maps discourse acts into one or more character responses. To understand the users' utterances, Facade makes use of surface text rules that map specific patterns directly to an intermediate meaning representation. For example, if the player said ”Hello Grace”, the word ”hello” will assert an iGreet fact and ”Grace” should assert an iCharacter(Grace) fact. The two facts combining will then produce a DAGreet(?x) discourse act fact, as shown in Figure 2.4. To capture more different ways of greeting, the rules for iGreet or DAGreet will have to be updated. Figure 2.5 shows some of the discourse acts representation in Facade as well as their corresponding pragmatic meaning.

There are currently approximately 800 template rules [16].

Figure 2.4: Simple greeting rules

```
"hello" → iGreet
"grace" → iCharacter(Grace)
iGreet AND iCharacter(?x) → DAGreet(?x)
```

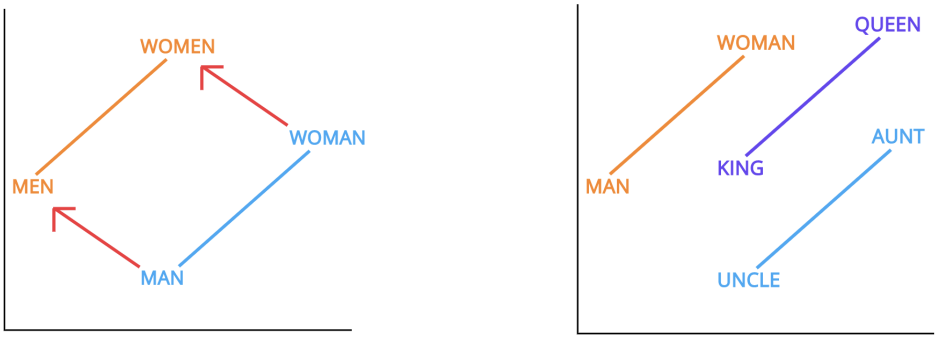
Figure 2.5: Examples of Facade discourse acts

Representation of Discourse Acts	*Pragmatic Meaning of Discourse Acts*
`(DAAgree ?char)`	Agree with a character. (e.g. "certainly", no doubt", "I would love to")
`(DADisagree ?char)`	Disagree with a character. (e.g. "No way", "Fat chance", "Get real", "Not by a long shot")
`(DAPositiveExcl ?char)`	A positive exclamation, potentially directed at a character. ("Yeah", "Wow", "Breath of fresh air")
`(DANegExcl ?char)`	A negative exclamation, potentially directed at a character. (e.g. "Damn", "That really sucks", "How awful", "I can't stomach that", "D'oh!")
`(DAExpress ?char ?type)`	Express an emotion, potentially directed at a character. The emotion types are *happy* ("I'm thrilled", ":-)"), *sad* ("That bums me out", ":-("), *laughter* ("ha ha"), and angry ("It really pisses me off", "grrrr").
`(DAMaybeUnsure ?char)`	Unsure or indecisive, potentially directed at a character. This discourse act is usually a response to a question. (e.g. "I don't know", "maybe", "I guess so", "You've lost me")
`(DAThank ?char)`	Thank a character (e.g. "Thanks a lot")
`(DAGreet ?char)`	Greet a character. (e.g. "Hello", "What's up")
`(DAAlly ?char)`	Ally with a character. (e.g. "I like you", "You are my friend", "I'm here for you")
`(DAOppose ?char)`	Oppose a character. (e.g. "Kiss off", "You're the worst, "I hate you", "Get out of my life")

2.4 Word Embedding

To process the text, we cannot directly input the text into the machine learning algorithm, so we have to find a way to represent the word. There is various way of doing it such as Bag-of-Words, using the features extracted from words, mapping the words to concepts etc. However, these methods have problems with ambiguation, generalisation as well as computation. On contrary, a vector space model can be used to represent the words in a continuous vector space. It also mapped the semantically similar words to nearby points.

We will be looking at four different unsupervised models for generating word vectors:



((a)) Relationship between words

((b)) Plurals of the word

Figure 2.6: Vector Space from Word2Vec

2.4.1 Word2Vec

Word2Vec is a predictive model that creates word embedding by using a 2-layer neural network. It takes in a large corpus of text and outputs a vector space with the words that share common context in the corpus located close to each another. [17] In this way, the vectors can capture relationships such as plurals, as shown in Figure 2.6(a) and 2.6(b).

Word2Vec is a combination of the Continuous Bag-of-Words model (CBOW) and Skip-gram model. CBOW predicts the target word given the context words, ie. the words surrounding the target word while the Skip-gram model instead predicts the context words given the target word. Both algorithms are trained similarly. After training, we can then obtain the embedding layer for that particular word by inputting the word and taking the hidden layer.

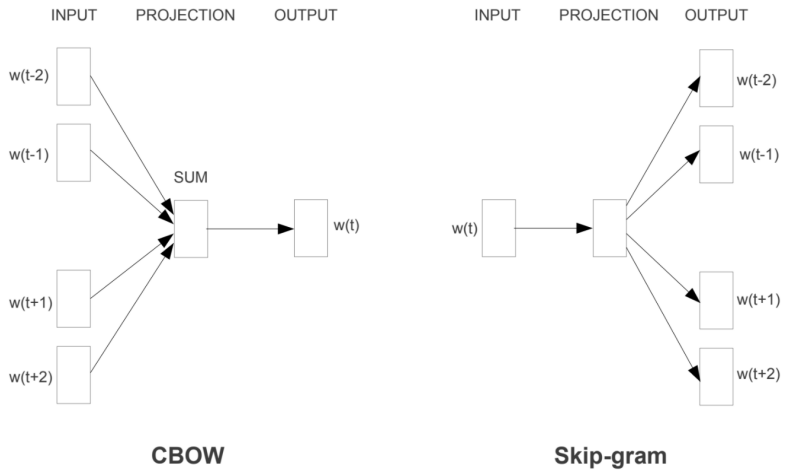
The limitation of Word2Vec is that summing across the entire corpus vocabulary will lead to a very expensive computation. Also, Word2Vec cannot represent words that are not in the vocabulary.

2.4.2 GloVe

Unlike Word2Vec, GloVe is a count-based vector-space model from Stanford University. It is based on the matrix factorization technique on the word-context matrix. For each word, we count the occurrence of that word in some context" in a large corpus, and this forms the word-context matrix that contains statistical information about the corpus. Then we factorize the matrix to obtain a lower-dimensional word-features matrix, where each row is a vector representation for the word. [18]

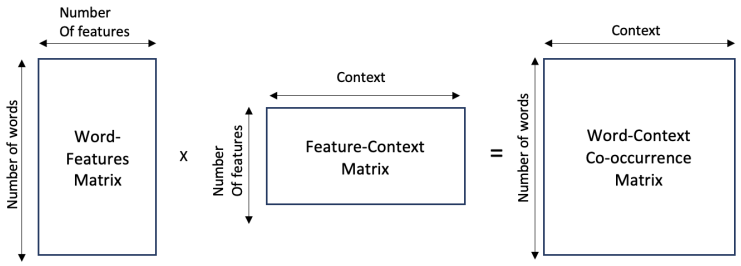
Overall, GloVe and Word2Vec both represent words in a vector form and have similar performance. However, GloVe may have outperformed Word2Vec as stated in the

Figure 2.7: Word2Vec - CBOW and Skip-gram Model



paper [18]. Also, GloVe is easier to parallelise during the training process, which will be an advantage over Word2Vec and having a large dataset.

Figure 2.8: GloVe - Matrix Factorization



2.4.3 BERT

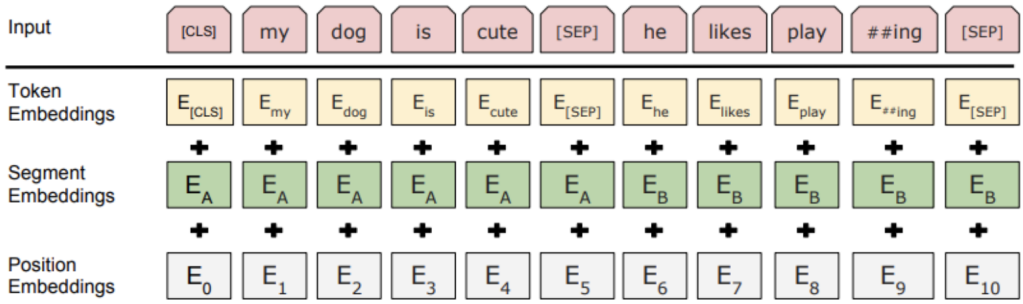
BERT (Bidirectional Encoder Representations from Transformers) is published by Google AI Language. It applies the bidirectional training of an attention model, Transformer, to language modelling. [19]. Instead of looking at the text from one direction, BERT reads the text from both left and right, which give the model a deeper sense of the context.

From Figure 2.9 [20], there are several embeddings layers in BERT's text processing stage. In the Segment Embeddings layer, BERT learns unique embedding for a different sentence to distinguish between them, which will be beneficial to tasks like Question-Answering. The Position Embeddings layer express the position of the words in a sentence which helps capture information of the words' order. Summing

all three embeddings will result in the representation of a particular word.

The text processing layers of BERT make it easier to train a BERT model for different NLP, without any major changes in the model. The biggest limitation of BERT is being compute-intensive at inference time. However, it should not be a major problem in this project since it only becomes costly when it is used in production at scale.

Figure 2.9: BERT's Text Processing



2.5 Key NLP Tasks

2.5.1 Part-of-Speech Tagging

Part-of-speech tagging is a crucial prepossessing step for other NLP task such as semantic analysis, syntactic analysis. The common English part-of-speech are noun, verb, adjective, adverb, preposition etc.

Although it seems to be simple to indicate if a word is a noun or a verb in a sentence, it becomes complex when there are several possible part-of-speech for that word. For example [21], in the sentence "The sailor dogs the hatch.", a correct grammatical tagging should show that "dogs" is a verb instead of a noun. There are several predefined tagsets that a tagger can use. The most popular ones are Penn Treebank tagset with 36 tags (Figure 2.10), the British National Corpus Basic (C5) tagset with 61 tags (Figure 2.11) , C7 tagset with 146 tags. They vary in level of detail. For example, Penn Treebank only differentiates 6 kinds of verbs while C5 has 24 different kinds of verbs because it takes into account the present, past and continuous forms of the verb as well.

The major challenge that a tagger need to overcome will be ambiguity. There are different approaches such as rule-based or probabilistic. [22] Rule-based tagger tags the words based on the handcrafting rule that specifies the condition while the probabilistic tagger estimates the probability of a specific tag for that given word under the observed context.

Figure 2.10: Table showing Penn Treebank Tagset

1	CC	Coordinating conjunction	19	PRP$	Possessive pronoun
2	CD	Cardinal number	20	RB	Adverb
3	DT	Determiner	21	RBR	Adverb, comparative
4	EX	Existential *there*	22	RBS	Adverb, superlative
5	FW	Foreign word	23	RP	Particle
6	IN	Preposition or subordinating conjunction	24	SYM	Symbol
7	JJ	Adjective	25	TO	*to*
8	JJR	Adjective, comparative	26	UH	Interjection
9	JJS	Adjective, superlative	27	VB	Verb, base form
10	LS	List item marker	28	VBD	Verb, past tense
11	MD	Modal	29	VBG	Verb, gerund or present participle
12	NN	Noun, singular or mass	30	VBN	Verb, past participle
13	NNS	Noun, plural	31	VBP	Verb, non-3rd person singular present
14	NNP	Proper noun, singular	32	VBZ	Verb, 3rd person singular present
15	NNPS	Proper noun, plural	33	WDT	Wh-determiner
16	PDT	Predeterminer	34	WP	Wh-pronoun
17	POS	Possessive ending	35	WP$	Possessive wh-pronoun
18	PRP	Personal pronoun	36	WRB	Wh-adverb

Figure 2.11: Table showing the C5 Tagset

1	AJ0	adjective	32	PUQ	punctuation - quotation mark (i.e. ` ' ")
2	AJC	comparative adjective	33	PUR	punctuation - right bracket (i.e.) or])
3	AJS	superlative adjective	34	TO0	infinitive marker TO
4	AT0	article	35	UNC	"unclassified" items which are not words of the English lexicon
5	AV0	adverb (unmarked)	36	VBB	the "base forms" of the verb "BE" (except the infinitive)
6	AVP	adverb particle	37	VBD	past form of the verb "BE")
7	AVQ	wh-adverb	38	VBG	-ing form of the verb "BE"
8	CJC	coordinating conjunction	39	VBI	infinitive of the verb "BE"
9	CJS	subordinating conjunction	40	VBN	past participle of the verb "BE"
10	CJT	the conjunction THAT	41	VBZ	-s form of the verb "BE", i.e. IS, 'S
11	CRD	cardinal numeral	42	VDB	base form of the verb "DO" (except the infinitive)
12	DPS	possessive determiner form	43	VDD	past form of the verb "DO"
13	DT0	general determiner	44	VDG	-ing form of the verb "DO"
›	DTQ	wh-determiner	45	VDI	infinitive of the verb "DO"
15	EX0	existential THERE	46	VDN	past participle of the verb "DO",
16	ITJ	interjection or other isolate	47	VDZ	-s form of the verb "DO"
17	NN0	noun (neutral for number)	48	VHB	base form of the verb "HAVE" (except the infinitive)
18	NN1	singular noun	49	VHD	past tense form of the verb "HAVE"
19	NN2	plural noun	50	VHG	-ing form of the verb "HAVE"
20	NP0	proper noun	51	VHI	infinitive of the verb "HAVE"
21	NULL	the null tag	52	VHN	past participle of the verb "HAVE"
22	ORD	ordinal	53	VHZ	-s form of the verb "HAVE"
23	PNI	indefinite pronoun	54	VM0	modal auxiliary verb
24	PNP	personal pronoun	55	VVB	base form of lexical verb (except the infinitive)
25	PNQ	wh-pronoun	56	VVD	past tense form of lexical verb
26	PNX	reflexive pronoun	57	VVG	-ing form of lexical verb
27	POS	the possessive (or genitive morpheme) 'S or '	58	VVI	infinitive of lexical verb
28	PRF	the preposition OF	59	VVN	past participle form of lex. verb
29	PRP	preposition (except for OF)	60	VVZ	-s form of lexical verb
30	PUL	punctuation - left bracket (i.e. (or [ )	61	XX0	the negative NOT or N'T
31	PUN	punctuation - general mark (i.e. . ! , : ; - ? ...)	62	ZZ0	alphabetical symbol

Figure 2.12: Example of Part-of-Speech Tagging

I_PRP want_VBP to open_VB the_DT box_NN and_CC see_VB what_WP is_VBZ inside_RB

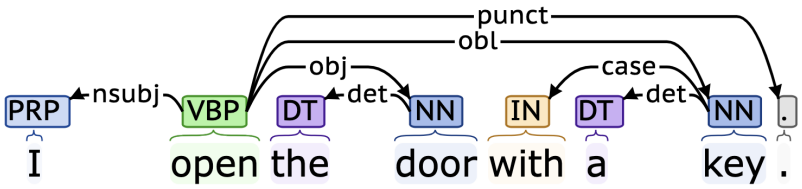
2.5.2 Dependency Parsing

Dependency Parsing is the process of examining the dependencies between words in a sentence to determine its grammatical structure. [23] With dependency parsing, we can understand what is the subject, the object of the verb as well as what is modifying the object.

Figure 2.13 shows an example of dependency parsing on the sentence ”I open the door with a key.”, ”open” is classified as the verb, ”I” is the subject and ”door” is the

object of the verb.

Figure 2.13: Example of Dependency Parsing



2.5.3 Word Sense Disambiguation

Word sense ambiguation is one of the biggest challenge when applying NLP techniques. For example [24], WordNet has 5 different sense for the word ”pen”:

1. a writing implement with a point from which ink flows
2. an enclosure for confining livestock
3. a portable enclosure in which babies may be left to play
4. a correctional instituion for those convicted of major crimes
5. female swan

We will discuss the most common approaches for word-sense disambiguation [24]:

Dictionary-based Method

Dictionary-based Disambiguation, also known as Knowlegde based Disambiguation. The most well-known method is the Lesk method. It is a seminal dictionary-based method. Among all sense, we choose the sense that share the most definition words with senses from neighbouring words. A frequent example for such algorithm is ”pine cone”, the dictionary definitions are following: [25]

Pine

1. kinds of evergreen tree with needle-shaped leaves
2. waste away through sorrow or illness

Cone

1. solid body which narrows to a point
2. something of this shape whether solid or hollow
3. fruit of certain evergreen trees

As we can see the first definition of Pine and the third definition of Cone contain the phrase ”evergreen tree”.

However, the limitation of the Lesk Algorithm is that the result depends heavily on the exact wording of definitions. Besides, the definitions from the dictionary are

usually brief, so it is difficult to compare the similarity.

Another option would be using a lexical knowledge base such as WordNet. It combines the characteristics of both dictionary and structured semantic networks, and groups synonymous words into synsets representing distinct lexical concepts. Therefore, it is easier to consider the word-sense relatedness and semantic similarity using WordNet.

Corpus-based Method

The corpus-based method is a supervised machine-learning algorithm that learns from sense-annotated corpora and it is created to overcome the limitation in the Knowledge-based Method. The context is represented as a set of features of the words, which also includes information about the surrounding words. [26] There are a lot of difficulties in the area of supervised WSD but among all attempts, support vector machine (SVM) and memory-based learning are the most successful approaches to WSD. However, a supervised method means it requires a lot of human effects to manually label the words. Also, it works based on an assumption that the context can disambiguate itself, which may not always be true.

2.5.4 Named Entity Extractions and Relationship Extractions

Named Entity Extractions is a subtask of Information Extraction that identify the entities in an unstructured text into pre-defined categories like person's, organisation, location etc. It may not be quite relevant to the project as Entities are less likely to be named. However, the Relationship Extractions that usually follow Named Entity Extractions will be closely related to our project. It extracts the relationship between two or more entities. For example, "The scissors are on the table." The "is on" is a relation between the scissors and the table. [27]

It can be done in various ways. It can be done in a rule-based which could be tailored specifically for our project. However, this would require a lot of manual work to create all the rules required. Other methods such as supervised RE or even unsupervised RE can solve the problem of hardcoding but would requires a huge library for training.

2.5.5 Semantic Roles Labelling

Semantic Role Labelling, also called shallow semantic parsing, is to understand how participants relate to events and able to answer questions like "who" did "what" to "whom" in "where", "when" and "how. [28] The Figure 2.14 shows a list of commonly used roles that get assigned to the predicate. As a result, we can understand the semantic role of the word in the sentence, such as Agent, Result, Source, Goal.

Figure 2.14: Commonly used theme roles

Thematic Role	Definition
AGENT	The volitional causer of an event
EXPERIENCER	The experiencer of an event
FORCE	The non-volitional causer of the event
THEME	The participant most directly affected by an event
RESULT	The end product of an event
CONTENT	The proposition or content of a propositional event
INSTRUMENT	An instrument used in an event
BENEFICIARY	The beneficiary of an event
SOURCE	The origin of the object of a transfer event
GOAL	The destination of an object of a transfer event

2.5.6 Coreference Resolution

Coreference resolution aims at resolving repeated references to an object in a document. [29] For example, ”Put the key into my inventory. Use it to open the lock.” With the help of coreference resolution, we can deduce that ”it” is referring to the ”key”.

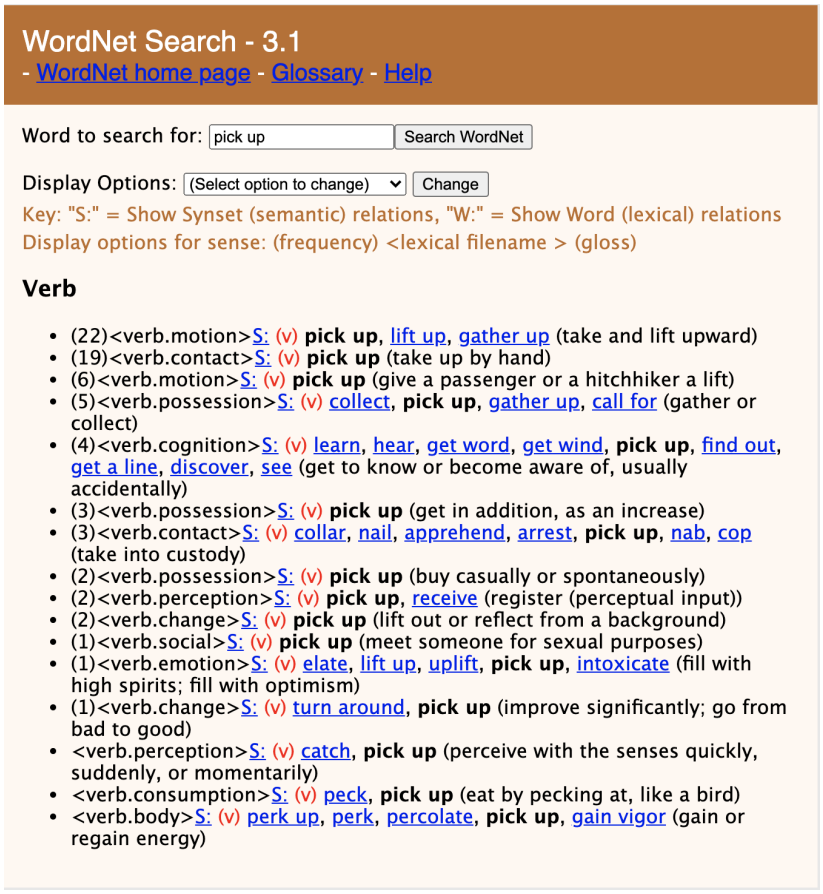
2.6 Tools

2.6.1 WordNET

WordNET is a large lexical database of English that is very useful to text analysis or natural language processing. [30] Each word is displayed with short definitions and examples as shown in Figure 2.15. Synonyms like ”close” and ”shut” are grouped into synsets, each expressing a distinct concept and are interlinked by conceptual-semantic and lexical relations. WordNET is not like a common thesaurus. It does not only find synonyms or antonyms of the word, it finds other relations such as hyperonymy, hyponymy etc. WordNET is a very useful tool in NLP tasks, such as word-sense disambiguation, information extraction, machine translation etc. It is also useful in determining the similarity between words.

As shown in Figure 2.16, the word-form ”funds” is recognized by WordNET as two lexical entry ”fund” and ”funds. The former belongs to three synsets: store, fund, fund, monetary fund and investment company, fund while the later belongs to one synset: monetary resource, funds. [31] The example also shows us the way WordNET connect synset as mentioned above. The synset money is related to another synset medium of exchange because ”money” is a hyponym of ”medium of exchange”.

Figure 2.15: Example Query Result in WordNET



2.6.2 NLTK

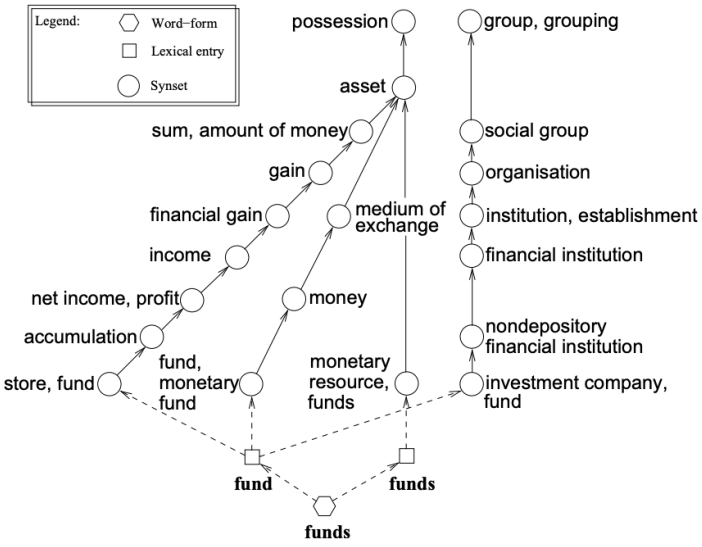
NLTK is one of the most popular libraries for building programs with language data in python. The interface is easy to use so it may be a good choice for a project that has a limited time. It also has rich corpora and lexical resources like WordNet, along with pre-models of text processing tasks like tokenization, stemming, tagging, parsing, etc., which will be very convenient when dealing with speech data. [32]

However, it does not analyze the semantic structure before splitting the text by sentences.[33] Also, another con of using NLTK is that they do not provide a neural network model.

2.6.3 spaCy

spaCy is another NLP library that is available in Python. It provides custom models for deep learning frameworks like TensorFlow and Pytorch. It also has pre-trained transformers like BERT, and pre-trained word vectors where in contrary, is not provided by NLTK. It also provides models for named entity recognition, POS tagging, dependency parsing, sentence segmentation, text classification, lemmatization, mor-

Figure 2.16: Mapping from word-forms and lexical entries to synsets and their hypernyms in WordNet



phological analysis etc. Therefore, spaCy might be a better choice when compared to NLTK as it contains all the tools that we might need in our project.

Chapter 3

Project Plan

23/1/2021	Finish writing up project plan, evaluation plan draft for introduction ready
25/1/2021	Draft for Background in interim report ready
27/1/2021	Interim report deadline
Feb	Design diagram
Feb-Mar	Sample text-based game, with speech-to-text implemented
Mar-Apr	Implementation of the NLP & Machine Learning techniques
Apr	Testing
Apr-May	Gather user experience
May	Software evaluation
May-Jun	Final report write-up
20/6/2022	Final report deadline
24/6/2022	Presentation slides deadline

Chapter 4

Evaluation Plan

The evaluation will be divided into two parts.

Evaluating NLP Techniques Implementation

1. Accuracy of speech-to-text - Word Error Rate (WER)
2. Performance of Word Embedding [34] - Word Similarity
3. Semantic Similarity using Pearson correlation coefficient and cosine similarity [35]
4. Accuracy of the action performed - check if the action performed match with the intention of the player

Evaluating Overall Escape Room Experience

1. User interface
2. Difficulty of the game - average time of finding the way out
3. Respond time of the game

Chapter 5

Ethical Issues

The main ethical issue of applying NLP will be the risk of socioeconomic stereotype. Bias could be introduced in a different stage of natural language processing, such as word embedding, coreference resolution etc.

It is suggested that stereotype related to racism and sexism is contained in the word embedding stage. [36] As shown in 5.1, ”Nurse” is likely to be associated with women while ”officer” is associated with men. Another example like ”Artist” and ”track driver” is categorised as Homosexual and heterosexual respectively. As a result, we can see how natural language processing could present an extent of prejudice in different areas. Besides, the collection of voice may also pose an ethical concern as the voice reveals information of the speaker such as gender, age, education, personality etc.

As a result, the data collection need to be handled with care so that the privacy of users can be well protected. To address these ethical issues, we could apply de-bias skills to reduce social discrimination appears in the text.

Figure 5.1: Words associated to social stereotype

Sexist prejudice			
Profession		Sentiment	
Woman	Man	Woman	Man
Nurse	Officer	Wedding	Reinforcement
Secretary	Hunter	Divorce	Attack
Teacher	Commander	Anulment	Combat
Saleswoman	Guard	Engagement	Power
Actress	Cameraman	Marry	Decrease
Population Prejudice			
Profession		Sentiment	
Foreigners	German	Foreigners	German
Aid official	Author	Refugee	Champion
Craftsman	Journalist	Unauthorized	Cooperation
Bank Assistant	Historian	Lawful	Union
Tour guide	Director	Tax	New
Foreman	Painter	Accumulate	Assignment
Sexual Orientation Prejudice			
Profession		Sentiment	
Homosexuality	Heterosexuality	Homosexuality	Heterosexuality
Artist	Singing teacher	Corruption	Unserious
Art dealer	Copywriter	Violence	Nice
Actress	Forest manager	Adultery	Fantastic
Cook	Track driver	Known	Smart
Shoemaker	Carpenter	Prohibited	Fair

Bibliography

[1] IBM Cloud Education. Natural language processing (nlp), 2020. URL https://www.ibm.com/cloud/learn/natural-language-processing. pages 4

[2] Eda Kavlakoglu. Nlp vs. nu vs. nlg: The differences between three natural language processing concepts, 2020. URL https://www.ibm.com/blogs/watson/2020/11/nlp-vs-nlu-vs-nlg-the-differences-between-three-natural-language-processing-co pages 4

[3] Wikipedia. Lexical analysis, 2021. URL https://en.wikipedia.org/wiki/Lexical_analysis. pages 4

[4] Deep Mehta. Stages of natural language processing, 2020. URL https://byteiota.com/stages-of-nlp/. pages 5

[5] Rachel Wolff. Semantic analysis, explained, 2020. URL https://monkeylearn.com/blog/semantic-analysis/. pages 5

[6] Daniel Johnson. Natural language processing tutorial: What is nlp? examples, 2022. URL https://www.guru99.com/nlp-tutorial.html#:~:text=to%20each%20other.-,Discourse%20Integration,upon%20the%20prior%20discourse%20context. pages 6

[7] Jalaj Thanaki. *Python Natural Language Processing*. Packt, 2017. ISBN 9781787121423. pages 6

[8] School of English. What is pragmatics? URL https://all-about-linguistics.group.shef.ac.uk/branches-of-linguistics/pragmatics/what-is-pragmatics/. pages 6

[9] APEStudio. URL https://apestudio.itch.io/albedo. pages 6

[10] Human Interact. Starship commander: Arcade, 2020. URL https://human-interact.com/starshipcommanderarcade/. pages 7

[11] Wikipedia. Starship commander: Arcade, 2021. URL https://en.wikipedia.org/wiki/Starship_Commander:_Arcade. pages 7

[12] Kent Bye. Bringing conversational gameplay interactive narrative to 'starship commander', 2017. URL https://www.roadtovr.com/conversational-gameplay-interactive-narrative-starship-commander/. pages 7

[13] Wikipedia. Lifeline (video game), 2021. URL https://en.wikipedia.org/wiki/Lifeline_(video_game). pages 7

[14] Miguel Lopez. 'lifeline' (ps2) review, 2004. URL https://web.archive.org/web/20041029012230/http://www.g4techtv.com/xplay/features/493/LifeLine_PS2_Review.html. pages 8

[15] Wikipedia. Facade (video game), 2021. URL https://en.wikipedia.org/wiki/Fa%C3%A7ade_(video_game). pages 9

[16] Andrew Stern Michael Mateas. Natural language understanding in facade: Surface-text processing. 2004. doi: https://link.springer.com/content/pdf/10.1007\%2Fb98252.pdf. pages 9

[17] Wikipedia. Word2vec, 2021. URL https://en.wikipedia.org/wiki/Word2vec#Approach. pages 11

[18] Richard Socher Jeffrey Pennington and Christopher D. Manning. Glove: Global vectors for word representation. 2014. doi: https://nlp.stanford.edu/pubs/glove.pdf. pages 11, 12

[19] Rani Horev. Bert explained: State of the art language model for nlp, 2018. URL https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270. pages 12

[20] Mohd Sanad Zaki Rizvi. Demystifying bert: A comprehensive guide to the groundbreaking nlp framework, 2019. URL https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/. pages 12

[21] Wikipedia. Part-of-speech tagging, 2021. URL https://en.wikipedia.org/wiki/Part-of-speech_tagging. pages 13

[22] Marco Maggini. Natural language processing, part 2: Part of speech tagging. URL https://www3.diism.unisi.it/~maggini/Teaching/TEL/slides%20EN/06%20-%20NLP%20-%20PoS%20Tagging.pdf. pages 13

[23] Prashant Sharma. Dependency parsing in natural language processing with examples, 2021. URL https://www.analyticsvidhya.com/blog/2021/12/dependency-parsing-in-natural-language-processing-with-examples/. pages 14

[24] Eneko Agirre Philip Edmonds. Word sense disambiguation, 2008. URL http://www.scholarpedia.org/article/Word_sense_disambiguation. pages 15

[25] Ruslan Mitkov. *The Oxford Handbook of Computational Linguistics*. Oxford University Press, 2003. ISBN 9780198238829. pages 15

[26] Wikipedia. Word sense disambiguation, 2021. URL http://www.scholarpedia.org/article/Word_sense_disambiguation#Methods. pages 16

[27] Andreas Herman. Different ways of doing relation extraction from text, 2019. URL https://medium.com/@andreasherman/different-ways-of-doing-relation-extraction-from-text-7362b4c3169e. pages 16

[28] Daniel Jurafsky & James H. Martin. Speech and language processing., 2019. URL https://web.stanford.edu/~jurafsky/slp3/old_oct19/20.pdf. pages 16

[29] Rhea Sukthanker, Soujanya Poria, Erik Cambria, and Ramkumar Thirunavukarasu. Anaphora and coreference resolution: A review. *Information Fusion*, 59:139–162, 2020. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2020.01.010. URL https://www.sciencedirect.com/science/article/pii/S1566253519303677. pages 17

[30] Princeton University. About wordnet., 2010. URL https://wordnet.princeton.edu/citing-wordnet. pages 17

[31] Dimitar Kazakov and Simon Dobnik. Inductive learning of lexical semantics with typed unification grammars. 01 2022. pages 17

[32] NLTK Project. Python nlp libraries: Features, use cases, pros and cons, 2021. URL https://www.nltk.org/. pages 18

[33] Tomasz Bak. Python nlp libraries: Features, use cases, pros and cons, 2019. URL https://medium.com/@tomaszbak/python-nlp-libraries-features-use-cases-pros-and-cons-da36a0cc6adb. pages 18

[34] Fenxiao Chen Yuncheng Wang Bin Wang, Angela Wang and c.-c. Jay Kuo. Evaluating word embedding models: methods and experimental results. 2019. doi: https://www.cambridge.org/core/services/aop-cambridge-core/content/view/EDF43F837150B94E71DBB36B28B85E79/S204877031900012Xa.pdf/div-class-title-evaluating-word-embedding-models-methods-and-experimental-results-div.pdf. pages 21

[35] April Shen & Nils Y. Hammerla Vitalii Zhelezniak, Aleksandar Savkov. Correlation coefficients and semantic textual similarity. doi: https://aclanthology.org/N19-1100.pdf. pages 21

[36] Juan Carlos Medina Serrano & Fabienne Marco Orestis Papakyriakopoulos, Simon Hegelich. Bias in word embeddings. doi: https://dl.acm.org/doi/pdf/10.1145/3351095.3372843. pages 22