Designing Service-Oriented Chatbot Systems Using a Construction Grammar-Driven Natural Language Generation System

Marie-Claire Jenkins

A thesis submitted for the Degree of
Doctor of Philosophy

University of East Anglia
School of Computing Sciences

30th April, 2011

©This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with the author and that no quotation from the thesis, nor any information derived therefrom, may be published without the author's prior written consent.

Abstract

Service oriented chatbot systems are used to inform users in a conversational manner about a particular service or product on a website. Our research shows that current systems are time consuming to build and not very accurate or satisfying to users. We find that natural language understanding and natural language generation methods are central to creating an efficient and useful system. In this thesis we investigate current and past methods in this research area and place particular emphasis on Construction Grammar and its computational implementation. Our research shows that users have strong emotive reactions to how these systems behave, so we also investigate the human computer interaction component. We present three systems (KIA, John and KIA2), and carry out extensive user tests on all of them, as well as comparative tests. KIA is built using exisitng methods, John is built with the user in mind and KIA2 is built using the construction grammar method. We found that the construction grammar approach performs well in service oriented chatbots systems, and that users preferred it over other systems.

Acknowledgements

It took a big team of special people to allow this Ph.D thesis to be produced. Although I am the author, there are many other names that should be mentioned here. Without the following people, none of this would have been possible.

I would like to thank my advisor, Professor Stephen Cox for always having faith in me, and for being reliable and accommodating throughout the duration of my research. I would also like to thank Dr Dan Smith, without whom this Ph.D thesis would never have even begun. I could never have done this without Richard Dean, my partner, editor, data sorter, coder, dinner maker and counsellor. Thank you. Special thanks to Dr Daniel Mapleson for unwavering support. My greatest gratitude goes to Ross Tailby for all of his help and for turning things around when they seemed too complicated, and to Sophie Jenkins, my sister, for all of her unrelenting help with the statistics. Many thanks to Ivan Benko for late night help and for being a great cheerleader and to Matthias for having it printed and handing it in on my behalf. Special thanks goes to Dr Rosie Chance for unwavering support and the occasional good kick up the backside. I'd like to thank my family, especially my mother, for their unwavering support and for talking me back into it, every time I talked myself out of it. Thank you also to Dr Daniel Tunkelang, for inspiring me and exciting me about computer science through his wonderful blog. I'd like to thank Adele Goldberg for her help on Construction Grammar, and for her encouragement.

Finally thank you to all those of you who took an interest in my work, spared a few words of encouragement, or listened to me talk about it, even if you had no idea what I was going on about.

You have my deepest gratitude.

“No computer has ever been designed that is ever aware of what it’s doing; but most of the time, we aren’t either.”

Marvin Minsky

Contents

Abstract **ii**

Acknowledgements **iii**

List of Tables **xi**

List of Figures **xii**

Abbreviations **xiv**

1 Introduction **1**

1.1 Research objectives 2

1.2 Summary of research objectives 3

1.3 Structure of this thesis 4

2 Chatbot architectures **5**

2.1 Introduction 5

2.2 Early chatbots 6

2.3 Chatbot specific programming languages 6

2.3.1 AIML 6

2.3.2 DMML 8

2.4 Learning algorithms and tagged corpora 9

2.5 Semantic web technology 9

2.6 The natural language processing techniques commonly used for query disambiguation and response generation 11

2.7 Stemming 12

2.7.1 Tokenizing 13

2.7.2 Ngrams 14

2.7.2.1 N-grams in text processing 15

2.7.2.2 Term frequency / inverse document frequency 16

2.7.3 Parsing 17

2.7.3.1 Parsing in information systems 17

2.7.4 Context-free grammars (CFG) 18

2.7.4.1 Parsing context-free grammars: 19

2.7.4.2 Latent Semantic indexing 21

2.8 The part-of-speech tagger 24

2.8.1 Tag sets 24

2.8.2 Rule-based tagging 25

2.8.3 Stochastic taggers 27

2.8.4 Transformation-based Tagging 28

2.9 Semantics 29

2.9.1 Word disambiguation and establishing semantic dependency 29

2.10 Using summarization techniques for response production 31

2.10.1 Summary 35

2.11 Query reformulation 35

2.12 Knowledgebases 36

2.12.1 Acquiring new data 38

2.12.2 Response generation 39

2.13 Engaging the user 41

2.13.1 The Turing test 41

2.13.1.1 Objections to the Turing test 41

2.13.1.2 The Chinese room argument 44

2.13.2 Creating chatbots with human behaviour 45

2.13.3 Grice's maxims 47

2.13.4 Small talk 48

2.14 Summary 49

3 Our early experiments and observations 50

3.1 John 50

3.1.1 John system characteristics 51

3.1.1.1 Mauldin's model 51

3.1.2 The knowledgebase 52

3.1.3 The John system architecture 55

3.1.3.1 The UEA-Lite stemmer 55

3.1.3.2 Using a vector model rather than the ALICE Graphmaster 59

3.1.4 Performance of the John system . . . 61
3.2 The KIA1 system . . . 63
3.2.1 KIA system characteristics . . . 63
3.2.2 KIA: system architecture . . . 63
3.2.3 Description of the human interaction experiment and results . . . 64
3.2.4 Results of the experiments . . . 66
3.2.4.1 Findings . . . 70
3.3 Summary . . . 71

4 Natural language understanding and generation 73

4.1 Natural language understanding and generation overview . . . 73
4.2 Natural language understanding . . . 74
4.2.1 Semantics . . . 74
4.2.1.1 Hartholt et al. on using an Ontology in a Natural Language Oriented Virtual Human Architecture . . . 74
4.2.1.2 Dahlgren's research on cognitive NLU . . . 76
4.2.1.3 Denis's dialogue understanding system . . . 76
4.2.1.4 The Delphi system . . . 77
4.2.1.5 Traum's work with Virtual humans . . . 78
4.2.1.6 The AT&T System . . . 79
4.2.1.7 Aleven's geometry tutor . . . 79
4.2.2 Syntax analysis . . . 80
4.2.2.1 Grammar formalisms: background . . . 80
4.2.2.2 NLU systems using syntactic theory . . . 82
4.2.3 Trainable systems . . . 83
4.2.3.1 Allen's TRIPS: The Rochester Interactive Planning System . . . 83
4.2.3.2 Allen's PLOW: A Collaborative Task Learning Agent . . . 84
4.2.3.3 A Knowledge-base NLU tutor . . . 85
4.2.3.4 Using Hidden Understanding models . . . 86
4.2.3.5 Using alignment templates and maximum entropy models (ME) . . . 87
4.3 Natural language Generation . . . 87
4.3.1 Hand crafted application-specific grammar approach . . . 88
4.3.2 Template-based approach . . . 89
4.3.2.1 The FLIGHTS system . . . 90
4.3.2.2 The WEATHERREPORTER system . . . 90
4.3.3 Rule-based approach . . . 91

4.3.4 Trainable generation approach 92
4.3.4.1 Automatically learning content selection rules from a corpus and its related database 92
4.3.4.2 Grammar-based generation 93
4.3.4.3 Using a Markov Decision Process (MDP) 93
4.3.4.4 Using a wide coverage realizer 94
4.3.5 Corpus-based approach 94
4.3.6 Summary 95

5 Construction Grammar **96**
5.1 Introduction 97
5.2 Cognitive linguistics 97
5.2.1 Frame semantics 98
5.3 Construction Grammar 103
5.3.1 Fluid construction grammar 109
5.3.2 Differences between construction grammar and other grammars 113
5.3.2.1 Formal grammars 113
5.3.2.2 Context-free grammars 113
5.3.2.3 Transformational grammar 114
5.4 Conclusion: The Construction grammar method in our SOCS 114

6 System Design **118**
6.1 KIA2 system overview 118
6.2 The architecture 120
6.2.1 Getting started 120
6.3 The user interface 122
6.4 Canned response phase 122
6.5 The construction grammar method 123
6.5.1 The syntactic pole 124
6.5.1.1 Parsing 124
6.5.1.2 Identifying constructions 124
6.5.2 The semantic pole 125
6.5.2.1 Word sense disambiguation 125
6.5.2.2 Named entity extraction 126
6.5.2.3 Semantic information 127
6.6 Generating a response 128
6.6.1 The taxonomy of constructions 128
6.6.2 The response engine 131

6.7 Continual performance improvement and training . . . 132
6.8 An example KIA2 system run . . . 133
6.9 Summary . . . 135

7 Evaluation **136**
7.1 Introduction . . . 136
7.2 Coniam's system evaluation . . . 137
7.2.1 Method: . . . 137
7.2.2 Considerations: . . . 138
7.2.3 Results: . . . 138
7.3 Human-computer interaction evaluation . . . 143
7.3.1 Method: . . . 144
7.3.1.1 Pre-test questionnaire . . . 144
7.3.1.2 The testing environment . . . 144
7.3.1.3 The chatbot setup . . . 145
7.3.1.4 The task-based user test . . . 145
7.3.1.5 The user evaluation procedure . . . 147
7.3.2 Considerations: . . . 148
7.3.2.1 User fatigue . . . 148
7.3.2.2 Bias: . . . 148
7.3.2.3 Demographics: . . . 148
7.3.2.4 Controlled environment testing: . . . 152
7.3.2.5 Chatbot Interfaces: . . . 153
7.3.3 Results . . . 153
7.3.3.1 User frustration threshold: . . . 154
7.3.3.2 Previous use of chatbots: . . . 155
7.3.3.3 Demographic effect: . . . 155
7.3.3.4 User feedback: . . . 155
7.3.3.5 Emotion: . . . 155
7.3.3.6 Unexpected events: . . . 155
7.3.4 Comments and conclusions: . . . 156
7.4 Summary . . . 156

8 Conclusions and future work **157**
8.1 Review of thesis objectives . . . 157
8.2 Summary of the research in this thesis . . . 157
8.3 Further work . . . 159

8.3.1 Interface . . . 159
8.3.2 OWL ontologies for knowledgebase engineering . . . 160
8.3.3 Integrating voice technology . . . 160
8.3.4 Topic tracking . . . 160
8.3.5 Sentiment analysis . . . 161
8.3.6 Open domain . . . 161
8.3.7 Integration with mobile devices . . . 161

Bibliography **162**

A KIA 1 Chatbot Logs Sample **175**

B KIA 2 John System conversation Sample **179**

C KIA1 User tasks and instructions **182**

D KIA1 User feedback form **187**

E Oral brief transcript samples **191**

F Results from the KIA user experiment **195**

G KIA2 Pre-test form **198**

H KIA2 pre-test responses **200**

I KIA2 Tasks **206**

J KIA conversation sample **208**

K KIA2 conversation sample **212**

L HCI User Experiment Raw Data **216**

List of Tables

3.1 HCI experiment results . . . 67

6.1 Table: Constructions . . . 129
6.2 Table: Construction Relationships . . . 129

7.1 Coniam Avivabot results . . . 140
7.2 Coniam KIA results . . . 141
7.3 Coniam KIA2 results . . . 142

List of Figures

2.1 Tokenizer example by Ying He and Mehmet Kayaalp 13
2.2 Google N-Gram viewer example 14
2.3 Example Parse tree 19
2.4 Top-Down Parsing by Andrew McCallum and Chris Manning 20
2.5 Bottom-up parsing by Andrew McCallum and Chris Manning 21
2.6 Chart Parser example by Bill Wilson 21
2.7 CKY chart example by Sariel Har-Peled and Madhusudan Parthasarathy 22
2.8 LSI equation 1 by Landauer 22
2.9 LSI equation 2 by Landauer 23
2.10 Penn Treebank tagset 25
2.11 ENGTWOL lexicon entry example 26
2.12 CLAWS Tagger example 27
2.13 TNT Example Output 28
2.14 A generalized framework for Word Sense Disambiguation by Pedersen et al 30
2.15 Web Mining categories by Kosala et al. 37
2.16 A four-layer architecture with the emotion engine as the core. Dark grey circles represent information assimilation. Light grey circles represent mappings onto internal and external behaviours. White triangles are filters. Dashed lines represent feedback. 46

3.1 John System Architecture 55
3.2 Alice Graphmaster 60
3.3 The overall Alice system 61
3.4 Kia system Diagram 65
3.5 The KIA interface 65

5.1 The “origin” frame in FrameNet 98
5.2 Legend for FrameNet diagrams 99

5.3 Robbery Frame 100
5.4 The “Heather sings” construction 105
5.5 Example taxonomy 106
5.6 Lexicon 107
5.7 Revised componential model 108
5.8 Symbolic links in CxG 108
5.9 Form & Meaning in the componential model 109
5.10 Form & Meaning in CxG 109
5.11 Example lexical entry for put and illustration of the J-operator. 112
5.12 Left: decomposition of the constraint program for the ball in the semantic structure. Right: related syntactic structure. In reality both structures contain a lot more information 115
5.13 Context-Free Grammar 115
5.14 Context-Free Grammar example 116
5.15 Application of Grammar re-write rules 116
5.16 CFG Parse tree 117

6.1 KIA2 system diagram 119
6.2 KIA2 system architecture 121
6.3 KIA2 system user interface 122

7.1 Coniam T-Test Results 143
7.2 User professions 149
7.3 Hours spent on search engines 150
7.4 Hours spent on the web 150
7.5 User computer habits 151
7.6 Users who used chatbots previously 151
7.7 How users had previously purchased insurance 152
7.8 HCI T-Test Results 154

List of Abbreviations

AIML	Artificial Intelligence Markup Language
AVM	Attribute Value Matrix
CCG	Combinatorial Categorical Grammar
CxG	Construction Grammar
CLAWS	Constituent Likelihood Automatic Word-tagging System
CFG	Context free Grammar
DMML	Dialogue Manager Markup Language
FCG	Fluid Construction Grammar
GPML	Genia Project Markup Language
HC	Human Chatbot
HMM	Hidden Markov Model
ICM	Idealized Cognitive Model
KIA	Knowledge Interaction Agent
LCA	Local Context Analysis

LSI	Latent Semantic Indexing
LTAG	Left Tree Adjoining Grammar
MDP	Markov Decision Process
ME	Maximum Entropy
MMIL	Multimodal Interface Language
MT	Machine Translation
NLG	Natural Language Generation
NLU	Natural Language Understanding
OWL	Web Ontology Language
POS	Part-of-speech
RDF	Resource Description Framework
RST	Rhetorical Structure Theory
SFL	Systemic Functional Linguistics
SOCS	Service Oriented Chatbot System
SynSem Frame	Syntactic-Semantic Frame
SVD	Standard Vector Deviation
TF	Term Frequency
TNT	Trigrams'n'Tags

Chapter 1

Introduction

A chatbot is a computer program designed to simulate the conversation that a human would have with another human in a chatroom, or in an instant messenger type of application, or for that matter, any real-time communication service between computers. The word ”chatbot” comes from the merging of the words ”chat” and ”robot”, which together describe what this computer program does quite well.

In his seminal paper ”computers, machinery and intelligence” in 1950, Alan Turing proposed that if such a chatbot were able to fool a human into thinking that s/he was talking to another human, then it could be considered as intelligent. This idea triggered much discussion on the matter of machine intelligence for many decades to come. There were strong objections to this test, which become known as ”The Turing test”, as well as a relatively large number of people building increasingly more sophisticated chatbots in order to try and pass it. To date, no chatbot has been able to pass the Turing test. We discuss the Turing test in much more detail in chapter 1 (section 2.12.1).

Chatbots have been used for a great number of tasks other than attempts at passing the Turing test, such as customer service on websites for example. This type of chatbot is referred to as a ”Service Oriented Conversational System” or ”SOCS”. SOCS have the particular job of helping people to accomplish a task on the website. They deal with customer queries and respond accurately and informatively to them, giving a natural language response rather than simply a list of possibly relevant web pages to visit, like a search engine does.

Creating such systems presents interesting challenges which are highly complex in nature. Natural language understanding and generation technologies are core to this type of system, and are currently still unresolved challenges in computing. There is also a strong human computer interaction component that requires development. Humans must be able to engage with and use the SOCS with ease. Current systems are not able to satisfyingly carry out their tasks due to the difficulties involved in creating such a system. Additionally there are few metrics in place allowing us to measure the impact of the performance of the system on the human and also to measure accuracy from a linguistic perspective.

The fact that no chatbot has ever passed the Turing test, suggests that they do not currently display enough human characteristics. Montero and Kenji Araki analysed human and chatbot conversations and concluded that the computer lacked criticality, making the conversation ”interaction vague and loosy” [87]. Many chatbots are based on pattern matching techniques such as AIML for example, which means that they do not have any reasoning capabilities. AIML (Artificial Mark-up Language) [195] is a templating language based on XML. Some chatbots such as Kyle [125] or Verbot [131] claim to use evolutionary algorithms to learn from user interaction rather than relying on a set number of responses. The problems of dialogue management and natural language understanding and generation means that the performance of these systems is fairly poor.

This thesis examines the issues with previous and current systems, evaluates testing methods, and proposes a new computational direction to improve their performance. We also observe human computer interaction factors that affect system perception for users. Our aim is not to pass the Turing test, but to allow for the creation of better SOCS, to improve their service to humans.

1.1 Research objectives

The research in this thesis has been motivated by the gaps in performance of existing systems, and the opportunity to investigate alternative methods. We have studied the problems from a usability, linguistic and technical point of view, in order to find a plausible solution to the problem of efficient SOCS creation. We have tested current chatbot technology in the lab and on the international stage of the Loebner prize (the yearly Turing test competition). We observed human interaction with chatbots of different types (conversational, service oriented, educational) and identified where the issues stemmed from.

The problems we observed and address in this thesis pertain to the areas of natural language understanding, natural language generation and user testing. The human computer interaction work carried out allowed us to better understand what the user expected from the system and what stopped him/her using such systems. With this information we were able to work on the linguistic aspect in further detail.

The basic SOCS operating process is as follows:

1. The user enters an utterance into the system
2. The utterance is processed by the machine
3. The machine retrieves relevant knowledge to include in a response
4. The machine processes the knowledge and creates a natural language response

Natural language understanding has only been addressed at a basic level in current SOCS. This stage of the process involves the machine understanding what the user is asking and this stage determines the knowledge retrieval stage. In order for this to be achieved, many systems use pattern matching techniques such as AIML for example. This will trigger either a response (bypassing the knowledge retrieval stage) or will trigger a course of action within the system knowledgebase. The natural language generation phase is similarly addressed by current systems by using keyword triggers and cannned responses. These responses are already prepared and available in correct English language in the database. We studied the current methods and built systems using these to test them with and learn more from them. Our conclusions show that there are many limitations to these methods and these are discussed in this thesis.

We have sought out an alternative method to the natural language understanding and generation phases. Construction Grammar (CxG), which is a cognitive linguistic method, was used to address the problems of these phases. It allows us to make use of syntactic and semantic information simultaneously. Construction Grammar (CxG) is described by Fried et al. as ”constraint based, generative, non-derivational, monostratal grammatical model, committed to incorporating the cognitive and interactional foundations of language”. [69]. There is no hierarchy of expressions, all are considered equally as important in understanding grammatical patterns. It also inlcudes all aspects of language (syntax, semantics, pragmatics, discourse, morphology, phonology, prosody). CxG consists of taxonomies of families of constructions, and uses entire constructions as the primary unit of grammar rather than working with words as the atomic unit. Pairings of form and meaning (semantic and syntactic information) are central to CxG and allow for machines to have more specific information when disambiguating and reconstructing a natural language utterance.

This thesis describes the process used to investigate this technique and the method used to compute it. The results of the tests show that, in a closed environment at least, the majority of these limitations are eradicated by our innovations.

Knowledge engineering was also an area of concern for SOCS because there needs to be some kind of intuitive way of storing information belonging to the domain and also to store the constructions. We investigated using of an OWL ontology to allow the system to verify constraints and alternatives in the domain and in the language as well. OWL is a semantic markup language used to share ontologies on the web [135]. It is a vocabulary extension of RDF (Resource Description Framework) and is derived from the DAML and OIL Web Ontology Language. There are two subsets of the OWL language, OWL Full, OWL Lite (containing just a functional subset of the OWL Full) and OWL DL (where DL stands for ”Description Logic”). OWL DL supports the existing Description Logic business segment and has good computational properties for reasoning systems. These are the reasons why we researched the use of OWL DL. In OWL ontotolgies, information is organised in a hierarchical manner so that it can be parsed and understood by software. In such a system, one ontology could obtain all of the domain knowledge while the other could contain all of the constructions commonly found in this domain. The advantage of using OWL is that many different types of knowledge from different domains could easily be absorbed by the system, whereas databases and spreadsheets do not offer the same flexibility. Although we did not construct an OWL ontology for the KIA2 system, our initial research shows that this kind of technology would be potentially very useful.

1.2 Summary of research objectives

The main goals of the work described in this thesis were to use CxG for user utterance processing and disambiguation as well as for machine response processing and delivery. The end result is a closed-domain SOCS able to carry out the task of advising on a user on car insurance using natural language. A by-product of this work has been a discussion of the metrics and testing methods associated with SOCS.

1.3 Structure of this thesis

The motivation for this thesis has been based on the relatively new adoption of SOCS by large corporate websites and businesses.

In chapter 1, we introduce the reader to the concept of the chatbot and the different types available as well as their differences. This chapter also gives some statistics and figures which serve to explain why SOCS are so useful to businesses today. We also explain how such systems currently function and what their drawbacks are.

In chapter 2, we describe our early experiments building SOCS, and give details on how and why we experimented with these particular techniques. Our two early systems are "John", which was built to compete in the Loebner prize, and KIA that was built to evaluate users and existing SOCS methods and technology.

In chapter 3, we discuss the current and previous natural language understanding and generation techniques in conversational systems and other types of language system. This body of research contributed to the decision to make use of CxG.

In chapter 4, we present and discuss CxG. This type of grammar has evolved from frame semantics, and other types of grammar. It is important to present some of the history of CxG in order to better explain why we have chosen it in preference to other types of grammar.

In Chapter 5, we present our final system, KIA2, complete with system diagrams and further information concerning the computational implementation of CxG. We also discuss the practicalities of such a method.

In chapter 6, we discuss our results and do so by comparing our system to an existing SOCS in the exact same domain, by HCI testing, and also through individual response assessment. We also take this opportunity to discuss different metrics for SOCS evaluation.

Finally we conclude by summarising the research and findings available in this thesis. We finish by discussing further work which is to be undertaken outside of the scope of this thesis. It features developing an open-domain system, using the CxG implementation in a sentiment analysis system, using OWL ontologies for knowedgebase engineering, automating construction identification and storage, and improved interface design.

Chapter 2

Chatbot architectures

2.1 Introduction

In this chapter we investigate previous work undertaken in building SOCS and other types of conversational system. We look at chatbot specific programming languages, the different types of templating solutions that have been tested, the semantic web approach and we also the natural language processing approach. We look at solutions involving semantic networks and summarization techniques for response disambiguation and formulation. We will identify and discuss the contributions of previous chatbot research in terms of their shortcomings but also in terms of the promising methods they offer for future systems. This work has served as the basis of our research and was the starting point in our investigations.

Conversational systems, both service oriented and social, are built in a very similar way at this time. They have some kind of database or knowledgebase, which serves as a repository giving the system a number of keywords, sentences or patterns to extract. They also, more often than not, have a number of canned responses, which are already written out and are returned as they appear in the repository. One such example is Verbot [131], where the system includes a collection of rules within a database. These help determine what an individual query is about and uses one or more inputs to produce one or more outputs. At this point regular expressions are used to enable matching. Once the correct response is matched, it is output to the user. The system can be manually trained, as Verbot state on their website, to: "Easily create your own virtual personality for free by filling out your bot's answers". Alternatively the user can upload a knowledgebase. The problem with this approach is that it relies heavily on manual input into the database which is not a long term scalable solution.

2.2 Early chatbots

Computer scientist Joseph Weisenbaum [197] released a chatbot called ELIZA in 1966. ELIZA simulated a psychotherapist, and would recognise keywords, which would trigger pattern-matching rules, prompting the user to say more about a particular topic. Responses were generated by reassembly rules, associated with selected decomposition rules. ELIZA the virtual psychotherapist, kept users captivated as they talked about their problems.

In 1972, Keneth Colby released PARRY, a chatbot modelled on schizophrenic paranoid patients. PARRY had 3 main affective states, namely fear, anger and mistrust. PARRY was a more sophisticate chatbot than ELIZA, because it had a conversational strategy: it simulated paranoid schizophrenic behaviour achieved through tracking its own internal emotional state on different dimensions. Colby [38] described the functioning of his system as follows:

”In the current version, the model consists of two modules, one for recognition of natural language expressions and one for response. Once the recogniser decides what is being said, the response module, using a number of tests and rules, decides how to respond. The output action of the model is now a function of the input, beliefs, affects and intentions. Thus a ”Hello” no longer received an automatic ”Hi” but may receive a variety of responses depending on a large number of conditions, including a ”model” of the interviewer which PARRY builds up during the interview. This representation of the interviewer involves making references about his competence, his helpfulness, etc...”

ELIZA and PARRY both paved the way for chatbots to come later, and allowed researchers to identify areas of research that would be useful to improving the performance and perceived performance of chatbots.

2.3 Chatbot specific programming languages

There are a number of programming languages that have been designed especially for chatbot development. Here we present two such languages, AIML and DMML that are the most widespread at this time.

2.3.1 AIML

AIML (Artificial Mark-up Language) [195] is an XML-compliant language designed for chatbots. It enables the storing of questions, answers and knowledge in a way that can be easily accessed by the system. This method was developed by the A.L.I.C.E foundation, and is constantly being modified and improved. The objects are identified by categories and topics. At this time there are over 41,000 categories. These contain data which is used to trigger responses by the chatbot when the correct cue is identified. The categories contain both possible questions and answers.

Here is an example of using AIML to answer the question ”What is water?”:

```
<category>
    <pattern>WHAT IS water</pattern>
    <template>
Water is a clear, colorless, odourless, and tasteless liquid, H 2 O,
essential for most plant and animal life and the most widely used of all.
```

```
    </template>
</category>
```

The words in all capitals are the AIML cases. The same question entered with a different pattern is configured by using the <srai> tags.

”Tell me what water is?” will produce the same answer as ”What is water?”

The ¡srai¿ tag always appears in the response template, but the chatbot treats X as if it is an input to the chatbot.

Wallace gives an example of the ¡sri¿ tag with the following example and explanation:

Client: You may say that again Alice. Robot: Once more? ”that.”

”The robot has no specific response to the pattern ”You may say that again Alice.” Instead, the robot builds its response to the client input in four steps. This simple sentence activated a sequence of four categories linked by ¡srai¿ tags. The robot constructed the reply ”Once more? ’that’” recursively as each sub-sentence triggered the next matching pattern”. [196]

```
<category>
    <pattern>TELL ME WHAT WATER IS</pattern>
    <template><srai>WHAT IS WATER</srai></template>
</category>
```

It is possible for a random response to be triggered by using a list (of list items <li>) wrapped by <random> tags.

```
<category>
    <pattern>WHAT IS WATER</pattern>
    <template>
        <random>
            <li>First answer</li>
            <li>Second answer</li>
            <li>Third answer</li>
        </random>
    </template>
</category>
```

A wildcard * can also be included in the answer <star>

```
<category>
    <pattern>TELL ME WHAT * IS</pattern>
    <template>I don’t know what <star/> is.</template>
</category>
```

There are approximately 200 such tags available to developers to create responses. The system tag enables a program to be executed as an operating system shell command, which then forms the reply. The JavaScript tag enables arbitrary scripting inside the templates. Tags such as The and That are both able to match the last utterance. AIML supports recursion, which means that the same rules can be applied again and again to increasingly smaller problems. The chatbot is able, through this, to break the problem down until it is as small as it can get. A question can simply be reduced to the type of w-question identified followed by the noun. Similarly several sentences can be correlated into one, synonyms are found in the script, keywords can be detected and matched to categories, and conditionals are dealt with through a ”condition tag”.

In this example, the chatbot is asked what it is, a common question to chatbots. The ”think” and ¡set¿ tags correspond to AIML categories and patterns.

```
<category>
    <pattern>WHAT ARE YOU</pattern>
    <template>
        <think><set name="topic">Me</set></think>
        I am a chatbot and I love chatting to people.
    </template>
</category>
```

Quarteroni and Manandhar [158] created a system that extracts information from the top 20 results relating to the user query. They used AIML as a base component of their system. The user provides natural language input, which triggers a pattern matching component to find the corresponding template response. This method does not deal well with anaphora (i.e. subject resolution after the speaker has avoided repeating the word) but it is still a work in progress. Other chatbots have issues with scale because storing information in templates or in a database means that when the domain is open, there is a lot of manual labour to be done in order for the systems to work. The larger the domain, the worse their performance due to the heavy reliance on the database. This is because the database would need to contain every single possible word or patterns that the chatbot might encounter. This would mean a lot of up front work over a long time. Another issue is that natural language generation is knowledge-intensive, flexible and highly context-sensitive [14]. The type of system described by Quateroni and Manadhar cannot offer enough flexibility because of its reliance on templates and hard coded patterns.

2.3.2 DMML

The DMML system [208] was devised by IBM’s research in conversational dialogue systems. DMML (Dialogue Moves Mark-up Language)is a form of middleware, inspired by speech acts and XML. IBM state that one of the limitations with natural language dialogue systems is that ”repositories of knowledge are not designed for NL interaction”. This middle layer contains a dialogue manager (used to maintain a history of the dialogue, retrieve content) , an action manager (to change the strategy according to events), a language-understanding module, and a layer for conveying messages between them. DMML has been developed to facilitate the intent of dialogue systems in the context of dialogue. Dialogue cues are stored in tags, which interact with the machine understanding module. The system is able to identify positives and negatives and so act appropriately. This illustrates recognition for the use of pragmatics in such systems. We begin to see the incorporation of how the user perceives and acts with the system. Chatbot systems are made for use by people, so taking into account pragmatics is desirable. Reviewing the DMML method described by IBM, we notice that in some senses the technique is similar to AIML, but focuses on intent rather than the execution and delivery of dialogue.

2.4 Learning algorithms and tagged corpora

Information extraction on a corpus uses a collection of documents to work from. There is usually a learning algorithm which processes the data and splits it into topic areas. This to allow for easier pattern matching.It is also able to produce some kind of semantic similarity between all of the terms in the document and identify any passages or texts that are similar to each other. This means that when a query is submitted, it is able to match the terms within it to the structured format of the corpus. Information extraction systems can use various methods such as keyword search, or analysis of noun and verb groups [18]. An untagged or tagged corpus can be used, and these methods can use supervised or unsupervised learning techniques.

A tagged corpus involves some degree of manual involvement or a program which can tag words such as a pos-tagger, or a parser. A tagged corpus can be used to build ontologies, such as the GENIA corpus [102] which was marked-up using the Genia project markup language (GPML). They annotate the corpus according to linguistic information and document structure. Names were marked up with their respective semantic classes and tagged according to their relationships. A hybrid semantic tagging system was proposed by Feldman et al [63]using the "trainable extraction grammar". This method uses both manual tagging and tagging rules. They explain that a human writes the SCFG rules and then these are trained on the data. The authors found that this technique enables developers to write rules in a more efficient manner. Tanabe et al. [183] tagged the MEDLINE corpus using GENETAG [182] (a gene/protein tagger), in order to perform named entity recognition on scientific data. They identified gene and protein names using this tagging method. An untagged corpus involves analyzing data on the fly without depending on tagging.

2.5 Semantic web technology

More recent chatbots have been making use of the semantic web. Tim Berners-Lee, inventor of the semantic web, presents it as follows [19]:

"The Web was designed as an information space, with the goal that it should be useful not only for human-human communication, but also that machines would be able to participate and help. One of the major obstacles to this has been the fact that most information on the Web is designed for human consumption, and even if it was derived from a database with well defined meanings (in at least some terms) for its columns, that the structure of the data is not evident to a robot browsing the web. Leaving aside the artificial intelligence problem of training machines to behave like people, the Semantic Web approach instead develops languages for expressing information in a machine processable form".

Some of the most well known languages used in the semantic web are RDF (Resource Description Framework), used to describe information and resources on the web; OWL (Web Ontology Language), a semantic markup language for publishing and sharing ontologies on the World Wide Web; SPARQL (SPARQL Protocol and RDF Query Language), a protocol and query language for semantic web data sources; XML (Extensible Markup Language), providing a syntax for the content of semantic web documents.

This is interesting from a chatbot perspective, because it could give them the ability to expand their knowledge of our world considerably, by tapping into knowledge stores on the web. This is because instead of using traditional databases that required a lot of maintenance, it would be possible to use the semantic web, which is created to be machine readable. Here, we present research done using the semantic web, because it a growing area of research that deserves a place in this thesis. We made some early attempts at using OWL ontologies ourselves, but as the scope of our work was already broad, this remains to be built and tested in our system.

The Senior Companion [153], for example, uses a dialogue manager to run the conversation. A series of questions are asked of the user and the responses are stored as knowledge. The world knowledge in this case is the world of the user essentially. Named entity recognition allows for the main subject of the conversation to be identified. The system uses RDF triples to store all of the knowledge, and certain inference rules are also included. The Resource Description Framework (RDF) is a framework for representing information in the Web [104]. RDF is a simple data model (independent of any specific serialization syntax), it has a formal syntax (which provides the basis for reasoning about RDF expressions), extensible URI-based vocabulary (used for naming things in RDF), an XML based syntax and XML schema data types (allowing for data exchange between RDF and XML). It is an open-world framework that allows anyone to make statements about any resource. RDF triples consist of a subject (identifying what the triple is about about), a predicate (defining the data in the object we are giving value to), and an object (the actual value). The Senior Companion system, for example, stores information about the family tree of a user in RDF triples. A reasoner is then used by the dialogue manager to infer new information.

The WKUP system (Virtual Consultant of Public Services) [200] assists users when searching and selecting services. The system allows for natural language communication in at least the first stage of interaction. The Conversational part of the system is built using an ontology built using the Simple Knowledge Organization System (SKOS) model [140]. SKOS provides a model useful for expressing the basic structure and content of concept schemes like dictionaries, classification schemes, taxonomies, folksonomies, and other types of controlled vocabulary. It allows us to link and share knowledge organisation systems via the semantic web. It is also useful for describing models that have some hierarchy and structure but that are not sufficiently discrete and formal to map directly into OWL. The chatbot makes polite chat about things such as the weather for example. Information retrieval is done using the Lucene engine [85], and this is also used to cluster the results. Because the WKUP conversational system deals with FAQ's, this is a good framework. The authors found that they could define ready-made answers to questions in the

FAQ list quite easily and effectively.

Tarau and Figa [184] built a story telling chatbot using Prolog refactorings of the WordNet lexical knowledgebase, FrameNet and the Open Mind common sense knowledge repository, together with Internet data stored in XML/RDF to provide high-quality knowledge sources. Ontology specific knowledge is extracted from the RDF store and content from FrameNet, WordNet and OpenMind is matched up with that. The inference engine uses a dynamic knowledge base which accumulates facts related to the context of the interaction. The XML/RDF meta-data allows users to search the story database by title, abstract, story type, performer name and so on. The authors found that because repositories like FrameNet [170] and OpenMind [178] contain so many possible conversation patterns, it was possible to rely on this rather than working towards automatic language generation.

Clearly while still in its infancy, this area of research has great value for SOCS and should be explored further in the future.

2.6 The natural language processing techniques commonly used for query disambiguation and response generation

The Natural Language Processing (NLP) is used for query disambiguation, which means that a query must be examined before it can be delivered in a format which can be used by the information matching part of the system. Many autonomous agent systems (a system that reacts to a changing environment of its own accord) and information retrieval systems (such as search engines) use pre-processing techniques in order to identify the meaning of a sentence so as to retrieve information relevant to the query. Being able to return the correct information to the user will rely on pattern recognition techniques which are created from various preprocessing techniques. During pre-processing, text is processed by various software programs to output the data in a format that further software programs can use effectively to produce yet another output of some kind. Pattern recognition allows us to identify patterns in data or in text, so that we can trigger a behaviour or an action (i.e. find relevant information). Pattern matching in contrast requires finding the exact match to a given pattern. We explored these techniques, and experimented with them in order to determine how successful they were when used in an autonomous agent system.

In this section, we will present the NLP components used in query disambiguation (i.e. when a user types in an ambiguous query, the system must be able to disambiguate it to allow for matches to be found) and describe the preferred variant or method of implementing each of them. They are parsing, tokenizing, context-free grammars, n-grams, term frequency/inverse document frequency, context-free grammars, latent semantic indexing, semantic dependency and semantic networks. These methods are for the most part used as the foundation of our final system KIA2.

2.7 Stemming

Stemming is used to reduce words to their root form, making it easier for the system to match common roots in the query and document corpus. A knowledge retrieval system needs a set of keywords to be extracted from the user utterances. This in order to find possible answers containing the same sets of words which serve to indicate a correct match. In order for the matching to be effectively conducted the correct keywords must be parsed in formal English spelling.

There are many morphological variants of words used in documents, that stemming algorithms are the logical approach to dealing with information retrieval. The root form of the word can be found and the system is then able to match words and their meaning based on this approach. Stemming algorithms have the advantage of reducing the corpus size thus making information retrieval a faster process. There are a number of stemmers available, notably the Lovins stemmer [119], Paice/Husk stemmer [148] and the Porter stemmer [156].

Porter [156] defines a stemmer as being a “process for removing the commoner morphological and inflexional endings from words in English.” The purpose of a stemmer is to enable the system to connect different topics, as morphological variants of a word are mapped to the same word. It is important to note that although the query is stemmed in order to identify closely related terms in the document sets, the query itself remains in its original format so that the grammatical structure stays intact. This facilitates the utterance generation in the later stages. Key features of the different stemmers available include The Porter Stemmer by Martin porter [156], which removes the morphological and inflexional endings from words. The Paice/Husk stemmer [148] is a more aggressive stemmer which works using an iterative process, removing endings in an indefinite number of steps. The Lovins stemmer[119] is a single pass, context-sensitive, longest-match stemmer. It’s rule list is limited and can only remove one suffix each time due to its being a single pass stemmer.

Overly aggressive stemmers tend to over-stem the given words, thus leading to a large number of different classes. The words that they produce are heavily conflated, and this leads to many different choices for the system. This could in turn lead to confusion and error, because each word has a very different meaning. A more conservative stemmer produces fewer classes, so it more probable that the stemmed words will still share the same meaning. The Porter stemmer has a number of drawbacks, namely:

1. It’s not easy to understand and difficult to modify
2. It makes errors by being too aggressive in conflation (e.g.policy/police)
3. It misses other words that should be conflated (e.g.matrices/matrix)
4. It produces stems that are not words (e.g.”’Rubies”’/”’Rubi”’)

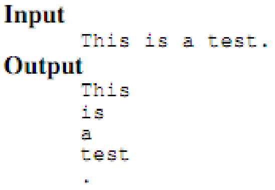
A conflation (or equivalence class) is a group of words that have a common root after a specific stemming algorithm is applied to them. Equivalence classes do not overlap: each word belongs to exactly one class. The basic hypothesis is that the word forms that should be conflated for a given corpus will co-occur in documents from that corpus. When a a general thesaurus is used for automatic query expansion, it does not improve the effectiveness of the system. The "connected component and optimal partition algorithm" tested by Xu and Croft [203] improves the Porter stemmer. They describe their results:

> On WSJ, the 39,949 Porter equivalence classes are broken up into 64,821 classes by the Connected Component algorithm, with average class size 1.17, which are further broken up by the Optimal Partition algorithm into 73,015 classes of average size 1.04. The expansion factor is reduced from 4.5 to 2.2 to 2.06. On WEST, the 27,117 Porter equivalence classes are broken up into 40,215 classes by the Connected Component algorithm, with average class size 1.24, which are further broken up by the Optimal Partition algorithm into 46,632 classes of average size 1.07. The expansion factor is reduced from 5.5 to 2.9 to 2.6.

They concluded that less expansion not only means faster retrieval, but also results in better retrieval effectiveness. The Connected Component and Optimal Partition algorithms produced small improvements in retrieval effectiveness on both WEST and WSJ. This experiment shows that aggressive stemming alone produces far more errors than conservative stemming.

2.7.1 Tokenizing

The tokenizer is usually the first stage of pre-processing. It is the process of breaking down text into smaller components (phrases, words or symbols for example). These smaller components are called tokens. Figure 2.1 gives an example of how a tokenizer operates by showing the output.



Input

```
This is a test.
```

Output

```
This
is
a
test
.
```

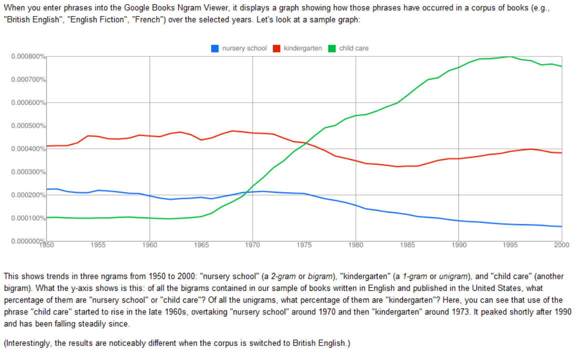
Figure 2.1: Tokenizer example by Ying He and Mehmet Kayaalp

Ying He and Mehmet Kayaalp carried out a study on tokenization of MEDLINE [207]. They tested 13 freely available tokenizers on various scenarios taken from actual MEDLINE abstracts. Their tests showed that the performance varied significantly from tokenizer to tokenizer. One of the reasons for this is that tokenizers make different choices on what constitutes a token. Choosing the right tokenizer is important as it will affect the subsequent stages in the greater system, such as for example parsing and indexing.

2.7.2 Ngrams

An ngram is a co-occurring set of terms. A term that is part of a sequence n is an ngram. A term can be a phoneme, a letter, a word, or even a sequence of words. The ngram is defined by the application. Pairs of ngrams are called bigrams, triples are trigrams and structure of 4 or more and simply called ngrams. An ngram model is a probabilistic model for predicting the next item in the sequence. Ngram models are useful for modelling sequences in natural languages.

The "Google Ngram viewer", in figure 2.2, allows for ngrams to be mapped over time in a large collection of books:



When you enter phrases into the Google Books Ngram Viewer, it displays a graph showing how those phrases have occurred in a corpus of books (e.g., "British English", "English Fiction", "French") over the selected years. Let's look at a sample graph:

This shows trends in three ngrams from 1950 to 2000: "nursery school" (a *2-gram* or *bigram*), "kindergarten" (a *1-gram* or *unigram*), and "child care" (another bigram). What the y-axis shows is this: of all the bigrams contained in our sample of books written in English and published in the United States, what percentage of them are "nursery school" or "child care"? Of all the unigrams, what percentage of them are "kindergarten"? Here, you can see that use of the phrase "child care" started to rise in the late 1960s, overtaking "nursery school" around 1970 and then "kindergarten" around 1973. It peaked shortly after 1990 and has been falling steadily since.

(Interestingly, the results are noticeably different when the corpus is switched to British English.)

Figure 2.2: Google N-Gram viewer example

The probabilities of ngrams are established by running an evaluation on a training corpus. In mathematical terms we predict the term Xi based on:

$$\mathrm{P}(Xi|Xi-1,...,Xi-n) \tag{2.1}$$

Every word in any language occurs with a different frequency. Zipf's law states that if f is the frequency of the word and r is the rank of the word in the list ordered by the frequency:

$$\mathrm{f} = \mathrm{k}\frac{\mathrm{r}}{} \tag{2.2}$$

In reality however, the probability is not a discrete distribution, and it is necessary to smooth the probability distributions using "add one" or more complex models. New terms and new n-grams are assigned non-zero probabilities. This is because, if we use Maximum-Likelihood distribution and assign to each term the exact probability of occurring based on the n-gram count in the training corpus, and assign zero to any term not occurring in it, the results are not good. Just because an ngram did not occur in the training corpus does not mean that it can never occur. "Smoothing" adds some probability to the unseen ngrams in the training corpus and removes some probability from the seen ngrams. "Discounting" refers to the act of removing probability mass away from terms. Back-off and deleted interpolation are methods used to redistribute the probability mass taken away by discounting. Witten-Bell discounting [199] estimates the probability of unseen ngrams and views the training corpus as events. Each event is an ngram. The number of times such an event occurs in the corpus is measured and assigned a probability. This is then distributed among the yet unseen types. This way we can estimate the total probability mass assign to all as yet unseen bigrams:

$$\mathrm{p}i* = T/[Z*(N+T)] \tag{2.3}$$

Where T is the number of observed types and N is the number of tokens and Z is number of zero-count bigrams. Good-Turing discounting estimates yet again the probability mass for unseen ngrams by estimating the total probability of unseen n-grams to be: the total number of 1-grams seen = N1/N. It gives some of the probability mass assigned to higher probability events to lower probability events.

2.7.2.1 N-grams in text processing

Agyemang et al. /citeAgyemang used ngram techniques in order to locate web content outliers. They particularly focused on data captured in ¡Meta¿ and ¡Title¿ tags in the HTML code, to reduce the huge number of n-grams vectors. Agyemang et al. state that the benefits of using ngrams are that (1) many pages on the web are published with spelling and grammatical errors, making full word matching less efficient in comparing strings, because ngrams support partial string matching, and (2) the ngram technique proved faster, more efficient as they all have the same length whereas whole words do not. Their experiments showed that using 5-grams was the most efficient ngrams length because shorter ngrams captured similarities between words due to other factors than semantic similarity (”commodity” and ”community” have five out of eight 2-grams in common suggesting that they are related but they are not). Longer ngrams failed to capture similarities between similar but different words. Their experiments showed that the ngram method was useful for data mining, and capable of detecting web content outliers in web data. They found that using ngrams was computationally efficient (speed and memory), efficient in determining similarity between different but related words in text processing, and the fact that ngrams support partial matching of strings with errors allowed for a better performance than full-word matching.

Kit and Wilks [103] used an implementation of a virtual corpus to derive ngram statistics for ngrams of any length from large-scale corpora based on the suffix array data structure. It is called a ”Virtual Corpus” (VC) because of its heavy reliance on a virtually existing sorted corpus for ngram counting. Corpus tokens are converted into integer codes before constructing the VC, thus enabling it to handle large-scale corpora with a vocabulary of various sizes. The method is able to derive statistics for ngrams of any length, based on the suffix array data structure.

Banko et al. [11] used ngrams in order to measure the similarity between human summaries and the original documents. They identified all ngrams common to the summary and text and placed no restriction on the size of the ngrams, but placed more importance on the longer ngrams. They discovered that 61 sentences used in summaries were identical to ones in the 1667 original texts. Their work supports the contention that ngram-based evaluation measures correlate strongly with human judgements when unigrams and bigrams are considered.

2.7.2.2 Term frequency / inverse document frequency

N-gram frequency distributions are useful for identifying similarities in documents. Inverse document frequency (IDF) can be used to help to choose most significant N-grams for the subject identification in these documents. Each ngram IDF is calculated as the ratio of documents in the collection to the number of documents in which the given N-gram occurs. TF-IDF corresponds to the importance of an N-gram for the document collection. The tf-idf weighting scheme assigns to term t a weight in document d given by

$$TF - IDFt, d = TFt, dxIDFt \tag{2.4}$$

1. The weight assigned to t in d will be highest when it occurs many times in a small number of documents.
2. The weight assigned to t in d will be lower when it occurs few times or in many documents.
3. The weight assigned to t in d will be lowest when it occurs in alll or almost all the documents.

Each document can be mapped as a vector with one component corresponding to each term in the dictionary, together with a weight for each component.

Bants et al. [22] used tf-idf to determine new event detection in news stories. They extended the tf-idf model by generating source specific models, normalizing the similarity score based on source-pair averages, normalizing the similarity score based on source-pair specific averages, term re-weighting based on inverse event frequencies and the segmentation of the documents. Very low term frequencies were viewed as unimportant. They only found a an improvement of 18 percent in the results yielded by each extension.

Ponte and Croft [155] however, found that their technique using probabilistic language modelling, was considerably superior in performance to the tf-idf technique. Ponte and Croft used three sets drawn at random from the "What's News" articles from the Wall Street Journal 1989. They considered that the smallest unit was a sentence, with a segment consisting of one or more sentences. They began by using Local Context Analysis (LCA) (as described in Xu and Croft [203]), to find words and phrases related to each sentence. In this method, standard tf and idf scores are not used because term frequency and document length are not taken into consideration.

Xu and Croft describe LCA in the following way:

1. Retrieve the top N passages
2. Extract words and phrases from these passages using a variant of EMIM
3. Rank the extracted concepts according to their co-occurrence with the query terms and their collection statistics
4. Return the top M concepts

In this particular study, N was 2000 and M was 100. The words and phrases returned by this method were then used instead of the original sentence and pairwise similarity of all of the sentences in their dataset was then calculated. These pairwise similarity measures were then sued to score individual segments of various sizes. The scores were used to rank each possible block of text of size n beginning at each position in the text. A final score was given to each block based on its position and length. This method of segmenting text by topic worked well despite small segments with few common words. LCA proved useful in augmenting short passages with related words for the purposes of text segmentation.

2.7.3 Parsing

Parsers use a grammar to perform syntactic analysis on a string of words. Different parsers vary in complexity. The choice of parser depends on the complexity of the task at hand. Shallow parsers are low complexity parsers, their input is split into chunks and each word sequence is labelled according to its syntactic nature.

Dependency parsing uses a dependency grammar, which assumes that syntactic structure consists of lexical elements linked by binary asymmetrical relations called dependencies [144]. When two words are connected by a dependency relationship, we refer to one of them as the Head and the other as the Dependent. Dependency trees have one node per word, because the parser only connects existing nodes and does not create any new ones. Dependency parsing also functions at the individual word level. Examples of using dependency parsing cane be seen in the Link grammar system proposed by Grinberg, Lafferty and Sleator [55] where each word is associated with a number of possible links. As Briscoe et al. [26] point out, when additional pruning of possible linkages is performed, the technique is very effective for large grammars.

The functional dependency grammar system [150] labels each word with all its possible functions and then applies rules which produce links between the types available in the context. Improbable links are removed.

2.7.3.1 Parsing in information systems

Novichkova et al. [145] proposed a system called MedScan. This is an information extraction system with the role of helping researchers understand protein functions. The information on protein functions is available across a large corpus, which makes the identification of the relevant passages of text difficult. Novichkova et al. constructed an NLP system to deal with this problem and achieved a ”recall rate” (the number of relevant instances retrieved) of 50-80 percent using shallow parsing techniques. The parser processes sentences from the Medline corpus abstracts and organizes them into semantic trees. A semantic tree is an ordered tree which has nodes representing relations and concepts. Sentences are parsed into semantic trees that represent their semantic relationships between the sentences, within the concepts that occur in the original text. The NLP system is based on a context-free grammar and a lexicon which was developed especially for MedLine. The syntactic parser creates alternative syntactic structures of the sentences. Because of the ambiguity of this technique, they also used a semantic processor to yield semantic trees. Briscoe and Carroll (1993) developed a probabilistic parser based on LR parsing methods (left to right, bottom-up parse). It used a unification based grammar which used part-of-speech and syntactic features. The tagger used was the Acquilex HMM tagger developed by Elworthy [62].

Charniak [61] developed a parser based on a Markov grammar. It is a bottom-up best-first chart parser. It derives its probabilities from a manually tagged corpus. It assigns probabilities of parses for sentences. The parser is used in a system by identifying the parse for the sentence with the highest probability. Collins [137] developed two CKY type chart parsers which used a generative form of lexicalized context-free grammar. CKY parsers parse according to the CockeYoungerKasami (CYK) algorithm. It only operates on context-free grammars expressed in Chomsky normal form (CNF), because it needs to test for possibilities to split the sequence it is running in half. It's a bottom-up parser that makes use of dynamic programming.

Preis [98] found, during her evaluation of parsers for anaphoric resolution for the Lappin algorithm, that the Collins parser [137] outperformed the Briscoe and Carroll [26] parser with a statistical significance of 85 percent. As a side note, the Lappin algorithm is designed to resolve pronominal anaphora. The Collins parsers are described by Collins as a " Head-Driven Statistical Models for Natural Language Parsing" [39]. The Collins parser operates in a bottom-up manner, decomposing the generation of the parse tree into increasingly smaller steps. It uses interdependence assumptions, which make parameter estimation easier. Preis found that the Charniak parser performed better then the Collins parser. She found that the Briscoe and Carroll parser did not much improve the Lappin algorithm. She found that it was possible that the Charniak tokenizer was detrimental to its performance. Errors in both the Charniak parser and the Collins parser were due to erroneous tagging. All 3 parsers (Collins, Briscoe-Carroll and Charniak) were found to be under performing compared on a corpus they had not been trained on. She concludes that the metrics used to evaluate parsers are not indicative of their performance in different situations.

2.7.4 Context-free grammars (CFG)

A CFG (also known as phrase structure grammar), is a grammar that generates a formal grammar where clauses can be nested inside clauses. Each production consists of a symbol which is to be replaced, and the symbol that replaces it. Each state can be replaced by a set of symbols. In this formal grammar, every production rule is such that the left-hand side is exactly one-terminal symbol and the right hand side is zero or more terminal symbols and/or non-terminal symbols. The grammatical structures of these clauses cannot overlap, but they can be as deep as required. This type of grammar relies solely on form and they are expressed using production rules. A parse tree is created when these rules are applied. In this parse tree, the symbols are nodes and the leaves are the terminal symbols. Each node expands into the next level of the tree when a production rule is applied. We describe it as "context-free" because all of the rules have nonterminal states on the left hand side, meaning that a nonterminal state on the right hand side can be replaced with it. Context is therefore unimportant.

A context-free grammar provides a simple and precise mechanism for describing how phrases in natural language are built from smaller blocks, capturing the structure of sentences in a natural way. It does however omit to take into consideration agreement and reference. It also does not take pragmatics into consideration.

The following is an example of CFG, as explained by Andrew McCallum and Chris Manning [133]

G = T,N, S,R T is set of terminals (lexicon) N is set of non-terminals For NLP, we usually distinguish out a set P ? N of preterminals which always rewrite as terminals. S is start symbol (one of the nonterminals) R is rules/productions of the form X ? ?, where X is a nonterminal and ? is a sequence of terminals and nonterminals (may be empty). A grammar G generates a language L

Therefore:

G = hT,N, S,Ri T = that, this, a, the, man, book, ?ight, meal, include, read, does N = S, NP, NOM, VP, Det, Noun, Verb, Aux S = S R = S ? NP VP Det ? that — this — a — the S ? Aux NP VP Noun ? book — ?ight — meal — man S ? VP Verb ? book — include — read NP ? Det NOM Aux ? does NOM ? Noun NOM ? Noun NOM VP ? Verb VP ? Verb NP

S ? NP VP ? Det NOM VP ? The NOM VP ? The Noun VP ? The man VP ? The man Verb NP ? The man read NP ? The man read Det NOM ? The man read this NOM ? The man read this Noun ? The man read this book

Figure 2.3 shows an example parse tree.

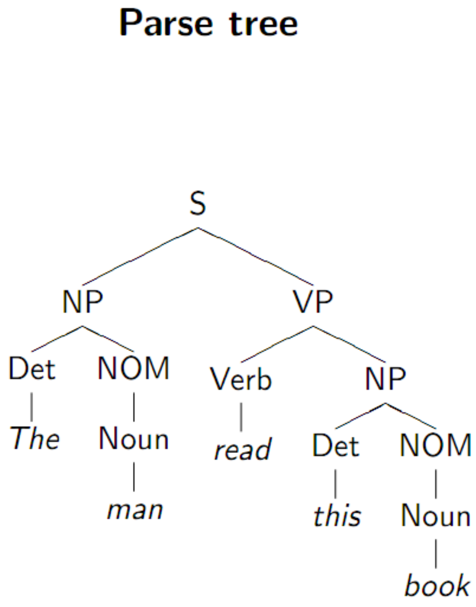


Figure 2.3: Example Parse tree

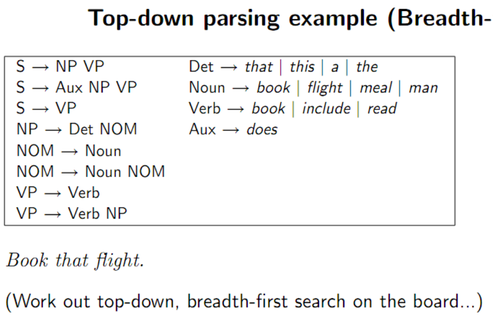
2.7.4.1 Parsing context-free grammars:

Parsing is described by Shieber,Schabes and Pereira as "...a deductive process that seeks to prove claims about the grammatical status of a string from assumptions describing the grammatical properties of the string's elements and the linear order between them" [177]. Parsing allows us to find the structure by running the grammar backwards, and by running it forwards we can understand the parsing algorithm and view it as a search problem. We can look for all of the structures matching a particular pattern, for example. In this case we use it as a search tree rather than a parse tree.

A recogniser has the task of testing whether a string matches the grammar, and returns a YES if it does and a

NO if not. The parser also returns parse trees for the string as well as doing the work of the recogniser. The parser is deemed Sound if every parse it returns is valid. It must terminate and not get caught in an infinite loop. A parser that for every given grammar and sentence produces a valid parse for that sentence and terminates, is deemed Complete. The parse can happen top-down or bottom-up and both of these approaches are independent of depth or breadth first, as both of these may be used.

Top-down parsing is data driven. The parser begins with a list of elements to build, and rewrites the goals in the goals list, so that they are in line with the grammar rules in the left-hand side. It then expands the rules with the right-hand side, and tries to match the sentence to be derived. If it is found that a goal can be derived in many different ways, then there is a choice of rule to apply and it becomes a search problem. The problem with this approach is that left recursive rules can lead to infinite loops. If there are many different loops for the same left-hand side, then it may have trouble matching the correct rule. Top-down parsers also struggle with rewriting parts-of-speech and expand top-down for instances where there is no bottom-up evidence for them. An example of top-down parsing is shown in figure 2.4 below.



Top-down parsing example (Breadth-first)

S → NP VP
S → Aux NP VP
S → VP
NP → Det NOM
NOM → Noun
NOM → Noun NOM
VP → Verb
VP → Verb NP

Det → *that* | *this* | *a* | *the*
Noun → *book* | *flight* | *meal* | *man*
Verb → *book* | *include* | *read*
Aux → *does*

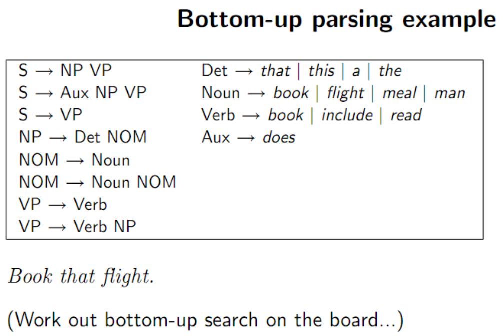
Book that flight.

(Work out top-down, breadth-first search on the board...)

Figure 2.4: Top-Down Parsing by Andrew McCallum and Chris Manning

Bottom-down parsing begins with the string to be parsed in the goal list. If the sequence in the right-hand side of the rule matches a sequence in the left-hand side of the rule, then it is replaced. The parse is finished when the goal list contains only the start symbol. If there are many rules in the right-hand side that match the left-hand side, then this becomes a search problem. Bottom-down parsing struggles with empty categories, and where there is lexical ambiguity. It also works very much at a local level which means that the work is repeated every time the structure is discovered globally. An example of bottom-up parsing is shown in figure 2.5

A solution to the problems which occur in both the bottom-up and top-down approach is to combine both approaches. In order to do this there needs to be some form of common representation between both methods. Chart parsing, invented by Martin Kay, can then be used in order to combine both techniques. The hypothesised results are stored in a chart structure (spanning the sentence) and re-used when necessary, thus eliminating the need for backtracking. It finds the edges which are consistent with its own knowledge and then adds them to the chart until



Bottom-up parsing example

S ⟶ NP VP
S ⟶ Aux NP VP
S ⟶ VP
NP ⟶ Det NOM
NOM ⟶ Noun
NOM ⟶ Noun NOM
VP ⟶ Verb
VP ⟶ Verb NP

Det ⟶ *that* | *this* | *a* | *the*
Noun ⟶ *book* | *flight* | *meal* | *man*
Verb ⟶ *book* | *include* | *read*
Aux ⟶ *does*

Book that flight.

(Work out bottom-up search on the board...)

Figure 2.5: Bottom-up parsing by Andrew McCallum and Chris Manning

parse edges are apparent. A set of grammatical rules and a dictionary listing all grammatical senses are used. An example of a chart parser is shown in figure 2.6

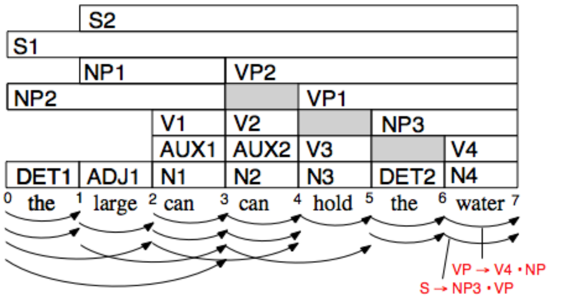
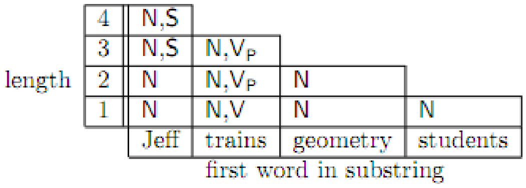


Figure 2.6: Chart Parser example by Bill Wilson

The CKY (Cocke-Kasami-Younger) parser's [31] table-data structure is in the form of a triangular chart. It can incorporate constituents of infinite length. CKY and bottom-up passive chart require that the grammar be expressed in Chomsky normal form (CNF), which means that the parse tree must always be binary. The CKY algorithm can be extended to handle grammars that are not in CNF (such as Earleys algorithm) and to linear-time parsing for special types of CFGs. CYK creates a table summarising the possible parses for each substring. From the table, it is possible to tell whether an input has a parse and extract one representative parse tree. It can prove whether a string belongs to the formal language described by the CFG. An example of the CKY chart is shown in figure 2.7

2.7.4.2 Latent Semantic indexing

Latent semantic indexing by singular value decomposition is a probabilistic method which is based on the spectral analysis of the term-document matrix. it is a type of vector retrieval method of text indexing and retrieval. A matrix



	4	N,S			
	3	N,S	N,V_P		
length	2	N	N,V_P	N	
	1	N	N,V	N	N
		Jeff	trains	geometry	students

first word in substring

Figure 2.7: CKY chart example by Sariel Har-Peled and Madhusudan Parthasarathy

is constructed in which each row corresponds to a unique word and each column represents a document. Each cell contains the frequency of appearance of the word in that particular column. The cell values are then weighted in order to identify their relative importance in the text. The matrix then has singular value decomposition applied to it, and is represented in a lower dimensional space. Dimensionality refers to the number of parameters by which a word or paragraph is described. The purpose of this method is to identify underlying semantic concepts in the corpus. It uses the implicit higher-order associations of words with text objects. This enables identification of major semantic categories within a corpus. The corpus can be organized into a semantic structure, which means that although the information contained in a document does not contain the exact words in the query, it can still be identified as relevant. It is assumed by LSI that this yields a better representation of how humans identify contexts. Deerwester et al. [52] found that by using this method they were able to overcome the problem of polysemy through using a semantic representation of the data and that the method was also superior to the simple keyword matching method or the use of controlled vocabularies and human intermediaries to act as translators.

Singular value decomposition or SVD, is used within the framework of the LSI algorithm in order to pinpoint the associative relationships. This is derived from the mathematical processing of the text, which eliminates any need for dictionaries and other such external resources. SVD is a mathematical decomposition technique, closely resembling eigenvector decomposition and factor analysis [44]. It represents the input matrix as a product of 3 matrices, which present “the row entities as vectors derived from orthogonal factor values, the original column entities in the same way, and the diagonal matrix containing scaling values such that when the three components are matrix multiplied, the original matrix is reconstructed” [187]. This is a least-squares best-fit matrix.

The mathematical description as expressed by Landauer [109]:

Consider a rectangular x matrix of terms and passages, X. Any rectangular matrix can be decomposed into the product of three other matrices using the singular value decomposition. Figure 2.8 by Landauer shows the equation.

$$X = T * S * P^T$$

Figure 2.8: LSI equation 1 by Landauer

is the SVD of a matrix where X is a x matrix with orthonormal columns, T is a t x r matrix with orthonormal columns, and P is an p x r diagonal matrix with the entries sorted in decreasing order. The entries of the S matrix are the singular values (eigenvalue), and the T and P matrices are the left and right singular vectors, corresponding to term and passage vectors. This is simply a re-representation of the X matrix using orthogonal indexing dimensions. LSA uses a truncated SVD, keeping only the k largest singular values and their associated vectors. The equation in figure 2.9 by Landauer shows the equation.

$$X = T_k * S_k * P_k^T$$

Figure 2.9: LSI equation 2 by Landauer

The equation in figure 2.8 is the reduced-dimension SVD, as used in LSA. This is the best (in a least squares sense) approximation to X with k parameters, and is what LSA uses for its semantic space. The rows in Tk are the term vectors in LSA space and the rows in Pk are the passage vectors.

The LSI technique has been used in a number of different applications:

Dumais et al. [52] applied it to cross-language information retrieval. They described the approach as “a method for fully automated cross-language information retrieval in which no query translation is required”. A multilingual semantic space is produced using the technique. The training set comprises of a set of dual language documents, which they drew from the Hansard collection. Words shown to be related using the LSI technique in both languages are represented, because the training set is bilingual. This method therefore avoids the document translation stage. Words which appear frequently similar have a similar representation in the LSI space, and those often seen to be associated are also given high representations. They explain that the new document is located at “the weighted vector sum of its constituent terms”. This ensures that each document has a numerical representation. This method means that the queries can be posed in either language. The results were promising and the method was also used on a number of different language pairs.

Gong and Liu [75] applied LSI to text summarization. They observed that because SVD obtains the semantic structure from the document matrix, it also represents “a breakdown of the original document into r linearly-independent base vectors of concepts”. SVD allows the document to be represented by a number of salient topics and the greater the value of its singular value, the more important the concept is. They used two months of CNN world view news programs as a corpus, and manually summarized it in order to obtain a benchmark measurement. Precision and recall was their measure of success for their technique. The standard information retrieval method, the relevance measure, took the sentence with the highest relevance score as the most relevant sentence. This was measured against the LSI summarization method. They found that the two methods performed equally well, proving that the LSI singular value method is in fact capable of capturing the general topic of a document.

The limitations of LSI lie in its inability to deal with syntactic relationships, morphology, polysemy or logic due to the fact that it does not take into account the word order. It captures the differences in the relationships between word choice and passage meanings. Hoffman [89] proposed a method called probabilistic latent semantic indexing. This method is described as a method based on a statistical latent class model for factor analysis of count data which is based on the likelihood principle. It is obtained from a training corpus by “a generalization of the Expectation Maximization algorithm” which has the task of dealing with domain-specific synonymy as well as the polysemous aspect. The results of the experiment showed that the method was much more effective in dealing with such words. He also found that the combination of models with different dimensionalities was even more efficient.

2.8 The part-of-speech tagger

A part-of-speech (POS) tagger is a program which is used to attach tags to words indicating their syntactic category in the sentence. This is used to identify content rich words such as nouns, verbs and noun phrases. It allows for the text to be processed into a machine readable format. These can be used by a search engine for further processing directly, for example, or pushed into a stemmer for further processing. POS tagging can be regarded as a word-level parser and is important for decomposing the user query. Words can have different POS tags depending on their contexts. POS can be a challenging task, language is very ambiguous, for example the word ”‘book”’ can be a verb or a noun.

In the next section, we describe different types of part-of-speech taggers. The following example of a tagged text shows how a POS-tagger might assign tags to each word in the sentence “The cat was sitting in the sunshine”:

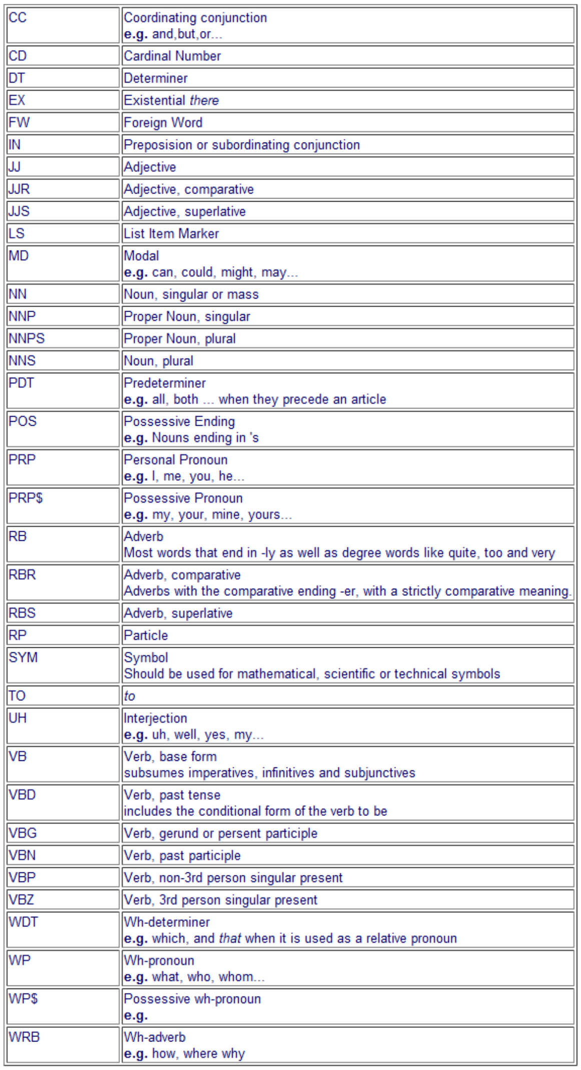
```
([ The_DT cat_NN ])
<: was_VBD sitting_VBG :>
in_IN ([ the_DT sunshine_NN ])._.
```

Supervised learning in taggers relies on pre-tagged corpora and dictionaries to create the rules to be used. Tag sequences, word and affix frequencies are identified to eliminate ambiguity. The limitation of this approach is that tagged corpora are not readily available and tagging a corpus takes a long time. These types of taggers also work best on the corpus they were trained on rather than another. Unsupervised learning involves the use of sophisticated methods in order to identify tag sets. These involve techniques such as Brill’s transformation-based tagger uses for example. The dictionary is created by the identification of the tag set. As with supervised learning, word frequencies, affix frequencies and word sequence probability are used to eliminate ambiguity. Based on these they can induce the rules needed by stochastic taggers or provide the context rules used by rule-cased systems.

2.8.1 Tag sets

The tag set chosen is a fundamental component. There are a number of different tag sets which can be chosen from depending on the task at hand. The tag sets differ according to their degree of granularity. Larger tag sets typically lead to less accuracy, and smaller ones are typically more accurate but also less useful as they tag fewer words. However, to cope with fine grained tags it is possible to reduce the tag set. Some widely used tag sets include Penn

Treebank (48 tags), Brown corpus (87), London-Lund Corpus (197 tags), Lancaster UCREL (165 tags). Some sets are designed for specific languages, such as for example Asahara [130] et al. who used a tag set of 500 different tags for the tagging of a Japanese corpus. The Penn Treebank tag set is listed here as an example what a tag set might comprise. Please see figure 2.10 for Penn Treebank tag set.



CC	Coordinating conjunction **e.g.** and,but,or...
CD	Cardinal Number
DT	Determiner
EX	Existential *there*
FW	Foreign Word
IN	Preposision or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List Item Marker
MD	Modal **e.g.** can, could, might, may...
NN	Noun, singular or mass
NNP	Proper Noun, singular
NNPS	Proper Noun, plural
NNS	Noun, plural
PDT	Predeterminer **e.g.** all, both ... when they precede an article
POS	Possessive Ending **e.g.** Nouns ending in 's
PRP	Personal Pronoun **e.g.** I, me, you, he...
PRP$	Possessive Pronoun **e.g.** my, your, mine, yours...
RB	Adverb Most words that end in -ly as well as degree words like quite, too and very
RBR	Adverb, comparative Adverbs with the comparative ending -er, with a strictly comparative meaning.
RBS	Adverb, superlative
RP	Particle
SYM	Symbol Should be used for mathematical, scientific or technical symbols
TO	*to*
UH	Interjection **e.g.** uh, well, yes, my...
VB	Verb, base form subsumes imperatives, infinitives and subjunctives
VBD	Verb, past tense includes the conditional form of the verb to be
VBG	Verb, gerund or persent participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner **e.g.** which, and *that* when it is used as a relative pronoun
WP	Wh-pronoun **e.g.** what, who, whom...
WP$	Possessive wh-pronoun **e.g.**
WRB	Wh-adverb **e.g.** how, where why

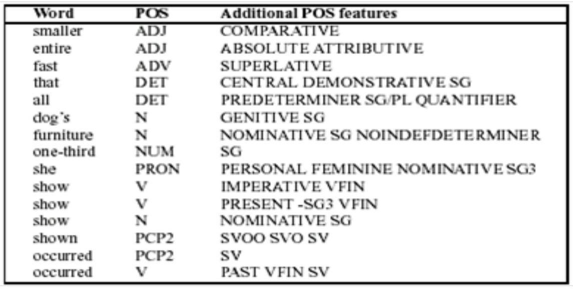
Figure 2.10: Penn Treebank tagset

2.8.2 Rule-based tagging

The concept of Rule-based tagging is to assign tags to all words and then remove tags based on a set of rules:

1. Start with a dictionary
2. Assign tags to all words
3. If word+1 is an adj, adv, or quantifier and the following is a sentence boundary and word-1 is not a verb like consider then eliminate non-adv else eliminate adv.

The ENGTWOL morphological parser (ENGlish TWO Level analysis) [193] is a rule-based tagger. It uses a lexicon of over 56,000 entries for English word stems and a morphosyntactic description based on Quirk et al. [160], an example of which is shown here, see figure 2.11 for an ENGTWOL lexicon entry example.



Word	POS	Additional POS features
smaller	ADJ	COMPARATIVE
entire	ADJ	ABSOLUTE ATTRIBUTIVE
fast	ADV	SUPERLATIVE
that	DET	CENTRAL DEMONSTRATIVE SG
all	DET	PREDETERMINER SG/PL QUANTIFIER
dog's	N	GENITIVE SG
furniture	N	NOMINATIVE SG NOINDEFDETERMINER
one-third	NUM	SG
she	PRON	PERSONAL FEMININE NOMINATIVE SG3
show	V	IMPERATIVE VFIN
show	V	PRESENT -SG3 VFIN
show	N	NOMINATIVE SG
shown	PCP2	SVOO SVO SV
occurred	PCP2	SV
occurred	V	PAST VFIN SV

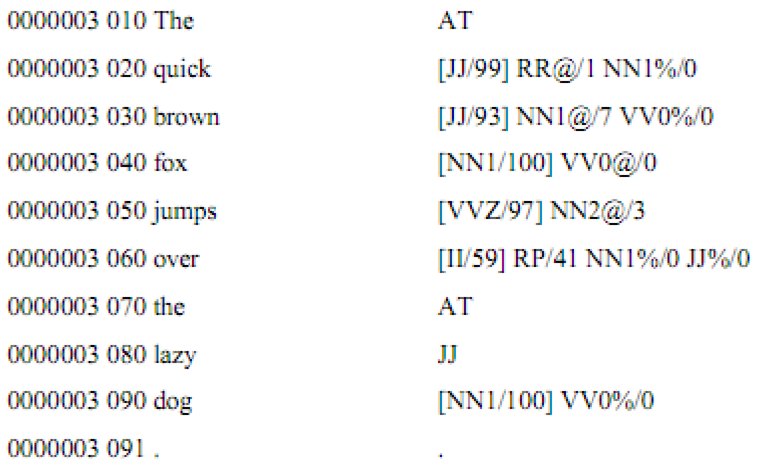
Figure 2.11: ENGTWOL lexicon entry example

The morphological disambiguator module applies a grammar that has 1,100 ’grammar-based’ constraints, and a set of 200 ’heuristic-based’ constraints. When all of the constraints have successfully been applied, it performs very well and is capable of a 96 percent accuracy rate. An example of output from ENGTWOL looks like this:

```
"the" <Def> DET CENTRAL ART SG/PL @DN>

"<cat>" "cat" N NOM SG

"<was>" "be" <SV> <SVC/N> <SVC/A> V PAST SG1,3 VFIN

"<sitting>" "sit" <SV> <SVO> <P/on> PCP1
```

The following is an example of an actual constraint, given the input “that”. This rule eliminates the adverbial sense of that as in ”I find that odd”’:

```
If

(+1 A/ADV/QUANT)

(+2 SENT-LIM)

(NOT -1 SVOC/A)

Then eliminate non-ADV tags

Else eliminate ADV tag
```



```
0000003 010 The          AT
0000003 020 quick        [JJ/99] RR@/1 NN1%/0
0000003 030 brown        [JJ/93] NN1@/7 VV0%/0
0000003 040 fox          [NN1/100] VV0@/0
0000003 050 jumps        [VVZ/97] NN2@/3
0000003 060 over         [II/59] RP/41 NN1%/0 JJ%/0
0000003 070 the          AT
0000003 080 lazy         JJ
0000003 090 dog          [NN1/100] VV0%/0
0000003 091 .            .
```

Figure 2.12: CLAWS Tagger example

2.8.3 Stochastic taggers

Stochastic taggers use probabilistic techniques. They will assign a tag to a word, given the probability that a particular one has a higher statistical likelihood of being the correct one. Basically their goal is to pick the most likely tag for a word. Stochastic taggers therefore require a training corpus. One issue with this is that words that are not in the training corpus are not assigned a probability. Smoothing methods are employed to deal with the eventuality of new words. Hidden Markov Models are an example of a stochastic tagger. They are described as “hidden” because the grammatical categories are unobservable. They estimate how often a word is associated with a particular tag in a particular sequence, and this results in a 90 percent accuracy rate. Stochastic taggers involve counting cases (such as from the Brown Corpus), and making a table of the probabilities of certain sequences.

The CLAWS tagger uses a lexicon of words which have attached all of the possible parts of speech which they might have, and a list of multi-word syntactic idioms. It achieves a accuracy in the 93-95 percent range. It is however quite expensive, since it enumerates all possibilities. An example sentence “The quick brown fox jumps over the lazy dog” is shown in figure 2.12.

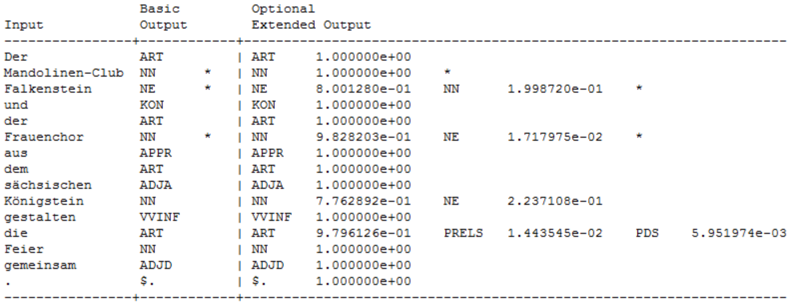
The following tagged sentence is the result of the processing:

```
The_AT quick_JJ brown_JJ fox_NN1 jumps_VVZ over_II the_AT lazy_JJ dog_NN1 ._
```

Many other taggers are available and each is chosen for use according to the results achieved on certain corpora by the developers. Briscoe et al. presented stochastic tagger which parses sequences of POS tags using a unification-based grammar coupled with a probabilistic LR parser. It was based on identifying the correct word category based on a statistical estimation. It is described as an n-gram tagger, and a ”probabilistic model implementing Markov chains” [26]. They achieved good parse selection accuracy, but at the expense of poor coverage of free text.

TNT (Trigrams’n’Tags) tagger [22] is an implementation of the Viterbi algorithm for second order Markov models.

It is not language dependent because it functions using a tagged corpus (which can be in any language). Smoothing is done using linear interpolation, where the respective weights are determined by deleted interpolation. An example of the output from the TNT tagger is shown in figure 2.13.



```
                 Basic         Optional
Input            Output        Extended Output
----------------+------------+--------------------------------------------------------------------
Der              ART         | ART     1.000000e+00
Mandolinen-Club  NN      *   | NN      1.000000e+00    *
Falkenstein      NE      *   | NE      8.001280e-01    NN      1.998720e-01    *
und              KON         | KON     1.000000e+00
der              ART         | ART     1.000000e+00
Frauenchor       NN      *   | NN      9.828203e-01    NE      1.717975e-02    *
aus              APPR        | APPR    1.000000e+00
dem              ART         | ART     1.000000e+00
sächsischen      ADJA        | ADJA    1.000000e+00
Königstein       NN          | NN      7.762892e-01    NE      2.237108e-01
gestalten        VVINF       | VVINF   1.000000e+00
die              ART         | ART     9.796126e-01    PRELS   1.443545e-02    PDS    5.951974e-03
Feier            NN          | NN      1.000000e+00
gemeinsam        ADJD        | ADJD    1.000000e+00
.                $.          | $.      1.000000e+00
----------------+------------+--------------------------------------------------------------------
```

Figure 2.13: TNT Example Output

The tag probabilities are set according to the word suffix, as they found that suffixes are good indicators of word classes (e.g.all words ending in “able” are listed as adjectives (JJ) in most cases). Smoothing is achieved with linear interpolation of unigrams, bigrams and trigrams. Setting all previously unseen words to zero is not efficient as it causes the probability of an entire sequence to be also set to zero, if it needs to be used for a new text sequence. Additionally, it is hard to rank sequences that contain zero probability. The average accuracy rate for assigning the correct tag using TNT was between 96 and 97 percent, depending on the tag set used and the language.

2.8.4 Transformation-based Tagging

Brill’s tagger [25] is described as an “error-driven transformation-based tagger”. It is “error driven”’ in the sense that it uses supervised learning. It is “transformation-based” in the sense that a tag is assigned to each word and changed using a set of predefined rules. Transformation-based learning (TBL) is a supervised learning technique and it requires a pre-tagged corpus. Brill’s tagger makes use of lexical and contextual rules in order to identify the correct tag. A number of extensions were added to the tagger which included lexicalizing the tagger, tagging unknown words and assigning the k-best tags to a word. The input is processed through an initial-state annotator, and it is then compared to the “truth” which is an annotation specified manually in the corpus. To lexicalize the tagger, Brill added contextual transformations which reference words as well as part-of-speech tags. The lexicalized transformation corrects mistagging and verb contractions such as “don’t”, meaning that the verb is recognized as such and not mistagged. Not using lexicalization led to an increased error rate (11 percent more than without). The accuracy of the tagger was measured at 96.9 percent. The tagger was also improved to be able to tag more accurately unknown words by guessing their most likely tag, which was estimated from a learning corpus. On the Penn Treebank tagged Wall Street corpus, the accuracy rate on unknown words was at 85 percent and the overall accuracy at 96.5

percent. The Brill tagger has 3 main steps to achieve tagging:

1. Label every word with the most likely tag
2. Check every possible transformation and choose the one that leads to improved tag accuracy
3. Tag all the data once again based on the rules

The sentence “The cat was sitting in the sunshine” is processed by the Brill tagger in the following way:

```
The/DT cat/NN was/VBD sitting/VBG in/IN the/DT sunshine/NN ./.
```

2.9 Semantics

Semantics have long been used in chatbots in order to establish topic during a conversation, and to find links between those topics. They allow the SOCS to identify which topic area is being discussed by the user, and what subject areas might come up during the conversation. Here we specifically look at chatbots using semantic dependency and semantic networks.

2.9.1 Word disambiguation and establishing semantic dependency

In order to establish more accurately which texts, paragraphs, sentences are appropriate for an answer, a SOCS evaluates semantic mappings between all of the topic sets in the training corpus. One word could have many other synonyms and ways of referring back to it. If we are able to establish links between topics through semantic dependency, we can increase the probability of finding an answer. If we used a canned answer approach, there is one response for every specific question. In this model, using semantic dependency between topics, if there is not an exact match, we can still find a closely related response. There are different ways to determine semantic dependency such as using ngrams, or LSI for example. In this section, we look at different approaches to semantic dependency.

Quillian [159] in the 1960s proposed a method which used the content of a machine readable dictionary to identify word meanings. The content of an electronic dictionary was represented by semantic networks associated with a node, which is turn connected to words defining concepts in dictionaries, and these are connected to others in the same way, creating a web of words. The method makes use of gloss overlaps (dictionary definitions which share the same words) to determine word meanings. In the 1960s there were very few electronic dictionaries so this method was not used a great deal but was built upon to improve the identification of semantic dependency.

Lesk [114] proposed a method which identifies the most likely meaning for a word in its context based on the contextual overlaps between dictionary definitions of ambiguous words or between the overlaps of dictionary definitions for the target word.

Rada et al. [161] extended Lesk’s algorithm by measuring semantic distance based on local (connected by syntactic features in the sentence) and global (derived by lexical chains throughout the whole text) context.

Pedersen et al. compute the intended sense of a target term by using WordNet-based measures of semantic relatedness. The process of disambiguating the words is done through selecting the sense of a word which is most related to the context words.

Pedersen and Banerjee [151] also extended the Lesk algorithm, as did many others. Their method involved measuring the similarity of words in the WordNet relations as well as those in the surrounding words, so glosses of word senses are also considered. The authors proposed giving a numeric value so that consecutive glosses are given more weight than others. Pedersen and Banerjee extended Lesks's method by expanding the glosses of words according to the structure of WordNet [141]. Measuring gloss overlaps is one way of measuring semantic relatedness, but the surrounding words give additional contextual information that can be used to improve word sense disambiguation. See figure 2.14 for a generalised framework for word sense disambiguation.

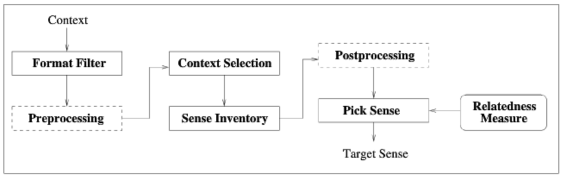


Figure 2.14: A generalized framework for Word Sense Disambiguation by Pedersen et al

Microsoft Research used semantic dependency graphs [205], for their machine translation research in order to address the issue of semantic connectivity. Both the source language text and target language text were aligned using a parsing algorithm. The dependency graphs are then aligned and a set of dependency mappings. The nodes in the graph represent the lemmas (word stems) of all the concept words in the sentence. A pair of nodes can be connected by an edge which is labelled by a semantic or deep-syntactic dependency relationship. Additional syntactic or semantic relationships are represented by binary features which indicate that there are further semantic relationships. The dependency graphs are aligned using a learned translation dictionary. Any input string from then on is converted to a dependency graph which is matched against the database of learned dependency graph fragments. It can then be matched to any mappings in the source data. This is used to map to the target language dependency graph. A text generation module then generates a baseline sentence in the target language. They found that their method was efficient and fast.

Silvio Ceccato [30] proposed a method called "correlational nets", which is based on 56 different relations, examples being case relations, type-subtype, type-instance, part-whole, miscellaneous logical, numerical, causal, spatial, and temporal relations. He built dependency trees, which were checked for correlation against pre-defined nets. These correlations were used to guide the parser thus resolving syntactic ambiguities.

2.10 Using summarization techniques for response production

Automatic summarization can be used to extract the most meaningful extracts from a relevant piece of text so as to compress the information necessary for the formulation of an answer. Once the information is available there is the possibility of using it as it is and giving it as an answer, or a reformulation of the information can be constructed using the grammatical cue from the user query. For example, if the user asked ”where”, the answer would be a location, so a single noun may be sufficient. Typically, automatic summarization systems incorporate a redundancy removal stage, a topic detector, and a text reformulation stage. Other elements may include further NLP techniques or data mining techniques.

Halliday and Hassan [82] established that certain language elements such as ellipsis (omitting words from a sentence, that are deemed to be implied by the context), lexical cohesion and lexical chains ”hang together as a whole”, which they call ”semantic unity”. This means that in text there are certain elements which can be used to identify topic. There have been several different attempts to produce automatic summaries some of which are described below.

SOCS require a knoweledgebase in order to answer user queries in dialogue. A knowledgebase is a database used in knoweldge management, a centralised repository for information. In order to be able to give an answer, SOCS need to process a significant amount of often unstructured documents to gather the required knowledge needed to give an answer. For this reason, it is interesting and useful to look into methods of automated text summarization, which we present in this section.

Automated text summarization involves topic identification, interpretation and generation. The text is reduced to a small number of concepts. Keywords are all important, as they indicate the topics included in the summary. The Summarist system by Hovy et al. [92] used information retrieval and statistical techniques in order to produce automatic summaries. They used both techniques to overcome problems that the other methods produced. The authors found that they could avoid misconceptions and generalizations using NLP. Typically NLP operates at a word level, and the problems produced by acquiring symbolic world knowledge, because it is difficult to acquire a large enough repository. Hovy et al. identified the most important topics using word frequency and cues, discourse structure and stereotypical text structure. They then processed this by condensing further the results through rephrasing sentences and removing redundancies. The authors then reformulated the result into a coherent structure. The results of their work were very promising and the methods they used are still being investigated (e.g.using decision trees).

Luhn and Edmunson [120] used a much simpler technique than the Summarist system. They extracted information from specific locations such as the abstract, the introduction and the conclusion. They placed more importance on the first and last sentences of the text and picked up on lexical cues which identified other important sentences within these sections. The problem with this method is that it relies on texts being very structured. This means that the method cannot reliably be used with web data and can be particularly difficult to apply to other text structures. The most important discovery that Luhn made was that important sentences in texts have frequently used words in them such as "the", "it", "a" for example.

Kupiec et al. [107] used sentence length, the occurrence of cue words, the position of the sentence in the text and the paragraph, term frequency, and identified words beginning with capital letters excluding those at the beginning of the sentence. The authors developed a trainable statistical summarization program, which requires a training set. The authors used abstracts of technical articles on online information services from Engineering Information Co. They developed a classification function that estimates the probability that a given sentence is included in an extract. They created new extracts by ranking sentences according to this probability, and then selected a user-specified number of top scoring ones. The evaluation criteria was precision (or the ability of the program to correctly classify sentences). They used an automatic alignment program to obtain correspondences which were then manually checked and corrected. The performance of the program was also checked using the manually corrected correspondences, however training was done using only the correspondences created by the alignment program.

The performance was 216 sentences (43 percent) correct. This suggests that for corpora such ours, summarizers can be trained automatically from document/summary pairs without manual intervention. For summaries that were 25 percent of the size of the average test document, the program selects 84 percent of the sentences chosen by human evaluators. Smaller summary sizes showed an improvement of 74 percent when presenting the beginning of a document. This method is useful for identifying in a relatively simple manner which parts of the text should appear in the summary. This technique requires a training set, and also that the sentences be presented in a certain order,for the program to be able to assign importance to them (i.e. title, abstract, introduction). In order to ensure that a structured corpus was available to their program, they had to carry out preprocessing in order to ensure that it would work. This reliance on structured text is a problem in systems that need to use the web as a corpora, as web data is very unstructured.

Teufel et al. [186] used a method that involved identifying a subset of sentences which contain the most information about the overall rhetoric structure of the text. They used structured texts, having a corpus of 202 technical papers with summaries, the CMP-LG corpus. The system would then identify sentences in the text which also occur in the summary. It then creates a model of "abstract-worthiness" of a sentence as a combination of a number of properties of that sentence. Human judges select the N most meaningful sentences to be used as training and evaluation material because this was shown to improve the performance of the automatic abstract generation program. They semi-automatically mark up the structural information of the documents (title, paragraph, headings, summary and so on). They used 5 different heuristics: Cue phrase, location method, title method, sentence length and thematic word method. The Cue phrase method seeks to filter out meta-discourse from subject matter. They used a list of 1670 negative and positive cues and indicator phrases or formulaic expressions. These Cue phrases were fixed strings for the sake of simplicity, and the list was manually created. The cue phrases were then classified in 5 groups identifying whether they were likely or less likely to be included in the summary. The location method identified if the sentence was likely to be important based on where it appeared in the document. For example, sentences at the start of a document are considered more likely to contain these sentences. Sentence length was used to filter out captions, titles and headings (threshold: 15 tokens including punctuation). The thematic word method involved trying to identify keyword that are characteristic of a document. These are identified by being non-stop words that occur frequently in the document but rarely in the document collection. They used TF-IDF to identify these thematic words. The title method involved giving words that were found in the title of a document more weight than other words during TF-IDF calculation. They found considerably less overlap between sentences in the summary and sentences in the text than Kupiec et al. did, because the documents in their training set were be many different authors. Using a mixture of the heuristics described here, they achieved a 68.4 percent recall and precision on the text material, compared to a 28 percent baseline and a best individual method of 55 percent. Additionally, increased training did not improve the performance of the program.

Mani et al. [126] used the C4.5 algorithm to identify similarities between each sentence and the query, and selected the n most similar ones. The authors took into consideration the unstructured nature of documents on the web, and tried to provide a corpus-based machine learning approach to summarization that would work on partially structured source text. They used location features, thematic features and cohesion features (relations between words or referring expressions). Their program used the raw frequency term in a document, its frequency in the corpus, number of terms in the document, and the sum of all term counts in the corpus. Cohesion is calculated by using synonymy and co-occurrence based on bigram statistics. They use WordNet to determine whether nouns have a synset in common. At a 20 percent compression rate, analysis of C4.5-Rules learning curves showed that there was a learning improvement in the generic case of Predictive Accuracy (.65-.69). They concluded that rules learned for user-focused interests tended to use a subset of the location features found in rules for generic interests, along with user specific keyword features. Additionally, the rules were easy to understand, and thus suitable for human-use. The approach has the advantage of being widely applicable, because it does not require manual tagging or sentence-level text alignment.

Marcu [128] used the Rhetorical Structure theory (RST), which identifies relations between 2 non-overlapping segments of text which he called the nucleus and the satellite. He derived an algorithm that constructed all the valid rhetorical trees that can be associated with a given discourse. The RS-tree is a binary tree whose leaves denote elementary textual units. Each of the nodes had a nucleus or satellite associated with it. It also had a type (the rhetorical relation that holds between the text spans that that node spans over), and a salience or promotion set (the set of units that constitute the most important part of the text that is spanned by that node). For each leaf node, the type is leaf and the promotion set is the textual unit to which it corresponds. The nucleus contained the essential arguments of the text and the satellites less important information. The problem with this method is that it was never tested on a large corpus.

Yaari [204] used lexically related paragraphs by segmenting the text into basic segments and applying a proximity test (cosine similarity) to find the most similar consecutive segment, which he then merged into one. The problem with this method is that it is only applied to neighbouring segments of text.

Neto and Santos [143] combined the output of an agglomerative clustering algorithm with the detection of sentences which are either relevant or not. They extracted proper nouns, anaphora, cue words, and then applied the vector similarity method to establish connectivity. The system was trained and tested using Ziff-Davis texts from the TIPSTER collection. They used a Naive-Bayes approach, which computed the probability that each sentence belonged to the summary, the N sentences with the largest value of this probability were selected for the summary. For the C4.5 algorithm, they used the -p option which generated soft-threshold decision trees providing an estimate of the class probabilities. The system selects the N sentences with largest value of this probability to be included in the summary. They reported that using the Naive-Bayes algorithm gave them an accuracy of 38.49 percent and using C4.5 they achieved an accuracy rate of 33.8 percent, - for 10 percent summaries using all features.

Hearst et al. [86] introduced the TextTiling algorithm which consisted of detecting topics linearly in a text, using the TF-IDF technique. The core system comprised tokenization, Similarity Determination, and Boundary Identification. They also made use of the WordNet thesaurus, a part-of-speech-tagger, and a shallow parser to identify nominal groups. They then made use of a segmentation algorithm. Their method used domain-independent lexical frequency and distribution information in order to determine the overlaps between topics. The ”tiles” were found to correspond to human identification of the subtopic boundaries of science magazine articles. One issue with this approach, however, is that was not efficient for dealing with multi-document summarization, as it only deals with the sentences in the order that they appear in the text.

Barzilay et al. [166] used lexical chains, which provide a lexical representation of the cohesive structure of the text. They found that related words occur in the same discourse unit of text, and that lexical chains would be useful to identify this phenomenon. When a new word is found in the text, a related chain is found according to some relevance criterion. They used the Hirst-StOnge method which involved selecting candidate words, finding an appropriate chain for them relying on the relatedness criteria among members of the chain, and inserting it into the chain as appropriate. They tried three different approaches for extracting the most relevant sentence. The first assumed that the sentence in the text with the strongest lexical chain was the most important. In the second, they collected the sentence containing the first appearance of representative chain member in the text for each sentence. In the third, they assumed that the same topic could be discussed at various points, and that the chain was distributed. They found the text unit where a chain was highly concentrated and extracted the sentence with the first chain appearance in this central unit. The third approach favoured long sentences, and did not take into account the fact that the extracted sentences could be anaphoric to other sentences and that the length and level of detail was not specified. They found that overall the first approach was the most efficient.

Hirst and StOnge [88] also used lexical chains but used WordNet to determine the relatedness of words that appeared in each synset by taking into account synonymy, polsemy, and hyponymy. Synsets refer to the logical groupings of information in WordNet. Each Synset has a list of synonymous words or collocations, and pointers that describe the relations between synets /citeMiller90wordnet. Hirst and StOnge selected candidate words, found the correct chain for each word relying on relatedness, and then inserted the word into the chain. The limitations of this approach is that it didn’t take into account the whole picture but rather remained at a word level which is not sufficient for a good summary.

2.10.1 Summary

The techniques used above indicate that there are still problems in the identification of important topics in a text, and problems with reformulation also. Extracting the right information from a corpus is very important for a chatbot system, as this is the substance of the answer it will deliver. It is also necessary for the summarization system to deal with multi-document summarization as often a general question may be asked, making it important to not center on a particular text but rather determine the similarities between all of the relevant texts and deliver a complete but short answer. Due to the fact that chatbots must deliver answers in a spoken language, it is important that the reformulation follow natural speech rather than the written word. The reformulation methods described above are not representative of human speech and have not been devised in such a way. It is therefore necessary to reformulate the summary using patterns learned through training, or using cues which may trigger certain behaviour in the utterance. The query is a natural language query which therefore does not simply contain keywords. Cues within the query such as “why” (which would lead to ”because”) and ”what” (which would lead to a definition or another more complex answer) appear and should be addressed. The amount of information and the sentence cut-off point are likely to be smaller because an utterance is typically short and concise. The summarization methods above can be used at the core but they need to be altered so as to cater for the particularities of human-chatbot interaction.

2.11 Query reformulation

Query reformulation is necessary when the system is unable to gain enough information from the user’s query in order to correctly answer the question, or even to answer the question at all. When the user input cannot be matched to any given information, the system will prompt the user for further information. This can be carried out in various ways. The system may reformulate the question using the same keywords as those provided by the user, much like the Ultra Hal assistant. The keywords are placed in the reformulation template, which is in turn given to the user. Another method is to give a standard prompt to the user. It is assumed in both cases that the user will volunteer more information containing additional keywords. However, when this is not the case, the user and the system are stuck in an endless loop of query reformulation, and query reformulation request.

Spink [147] studied the user sessions from the Excite data set. Her research showed that only one in five users chose to reformulate their query. They were given the choice of asking for similar results, which they did not elect to use, even though they conducted several related searches. The most common reformulation found was a change of term. This would be useful in a chatbot environment. However, if this term also is unknown, the problem is not resolved. This study would suggest that users are generally unwilling to spend time reformulating questions. It also suggests a low patience threshold for chatbot query reformulations. Jansen et al. [94] used transaction log analysis to examine the behaviour of AltaVista Web users, when they searched for information, that occurred from 1998 to 2002. They found that the percentage of users who modified queries increased from 20% of all users in 1998 to 52% in 2002.

Boldi et al. [20] explain that ”Understanding query reformulation patterns is a key step towards next generation web search engines: it can help improving users web-search experience by predicting their intent, and thus helping them to locate information more effectively". They built a model that could accurately classify user query reformulations into four different classes: generalization, specialization, error correction or parallel move. They achieved an accuracy rate of 92%. They applied their classifier to a large query log and were able to examine query reformulation patterns. They noticed that query generalisations and specialisations appeared frequently in alternating order, and that error corrections occurred more frequently at the beginning of a search session rather or after an error correction.

Paradiso [149] used a "fallback" method which meant that the system backtracked up into the topic category until the context was recognized. This was done by measuring how far the new context is to the previous one. The method can work reasonably well, however if the keywords are not recognized in the discourse structure to begin with, the system is unable to proceed with the conversation.

2.12 Knowledgebases

Information extraction (IE) differs from information retrieval (IR). Information retrieval produces document sets which are of relevance to a user, whereas information extraction produces facts from the data. In a question answering system, facts are needed in order to be able to formulate a response. Cowie et al. [42] define IE in the following way:

"Information Extraction (IE) is the name given to any process which selectively structures and combines data which is found, explicitly stated or implied, in one or more texts. The final output of the extraction process varies; in every case, however, it can be transformed so as to populate some type of database. Information analysts working long term on specific tasks already carry out information extraction manually with the express goal of database creation."

Baeza-Yates [9] defines IR ystems as:

"Information retrieval deals with the representation, storage, organization of, and access to information items such as documents, Web pages, online catalogs, structured and semi-structured records, multimedia objects. The representation and organization of the information items should be such as to provide the users with easy access to information of their interest."

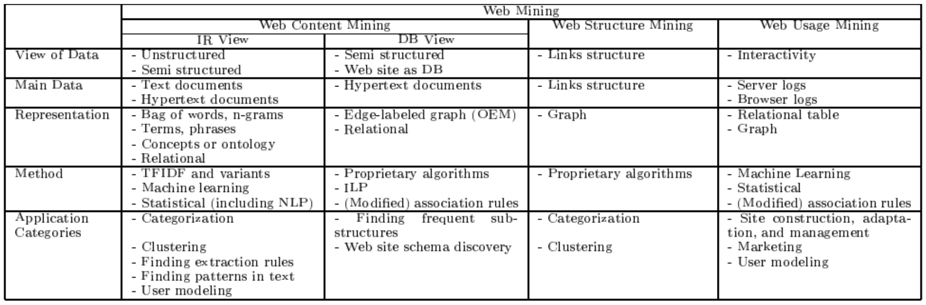
Etzioni [28] hypothesized that web data contained enough structure to enable efficient information extraction. Kosala and Blockeel [165] surveyed web mining and investigated the use of the web as a large collection of knowledge. The authors state that in the discipline of web mining, there is a crossover between database, information retrieval and AI (especially in the areas of machine learning). They list the following information problems when interacting with the web:

- Finding relevant information
- Creating new knowledge out of available information
- Personalization of the information
- Learning about consumers and individual users

The authors state that web mining techniques could be used to solve all of these problems directly or indirectly, but they also state that web mining techniques are not the only tools available to solve these problems. They decompose web mining tasks into 3 subtasks:

- Resource finding
- Information selection and pre-processing
- Generalisation (automatically discovering user patterns)
- Analysis (validating or interpreting mined patterns)

The biggest challenge with web mining is that the majority of web content data consists of unstructured data such as free texts, semi-structured data (HTML documents) and more structured data (such as that found in databases generated by HTML pages). They define web mining as a kind of preprocessing task in the information extraction procedure. They note that increasingly IE systems are constructed for using machine learning methods or other algorithms, as building these systems manually is no longer necessary. See figure 2.15 for Kosala et al. [165] web mining categories.



	Web Mining			
	Web Content Mining		Web Structure Mining	Web Usage Mining
	IR View	DB View		
View of Data	- Unstructured - Semi structured	- Semi structured - Web site as DB	- Links structure	- Interactivity
Main Data	- Text documents - Hypertext documents	- Hypertext documents	- Links structure	- Server logs - Browser logs
Representation	- Bag of words, n-grams - Terms, phrases - Concepts or ontology - Relational	- Edge-labeled graph (OEM) - Relational	- Graph	- Relational table - Graph
Method	- TFIDF and variants - Machine learning - Statistical (including NLP)	- Proprietary algorithms - ILP - (Modified) association rules	- Proprietary algorithms	- Machine Learning - Statistical - (Modified) association rules
Application Categories	- Categorization - Clustering - Finding extraction rules - Finding patterns in text - User modeling	- Finding frequent substructures - Web site schema discovery	- Categorization - Clustering	- Site construction, adaptation, and management - Marketing - User modeling

Figure 2.15: Web Mining categories by Kosala et al.

Lin et al. [115] studied the use of web data in question-answering systems. They used a ”federated” approach, using techniques for handling semi-structured data on the web as if it were a large database of resources, able to answer common questions. The distributed approach involved large scale text-processing techniques to extract answers from unstructured web data. Lin et al. found that these approaches worked well together. The federated approach showed that by applying the Penn Treebank tag set to questions, the accuracy of the parsing stage could be boosted to 95 percent or more (Hermjackob, 2001). The amount of knowledge engineering necessary was calculated by counting the number of schemas required. The distributed approach used passage retrieval to reduce the corpus. They used an information retrieval system which was efficient in capitalizing on data redundancy, by becoming a surrogate for natural language. It also allowed documents of poor quality to be efficiently processed. The approach worked well. However, further work involved more natural language work to deal with the ambiguity in language and machine learning to automate the pattern matching process.

Hong et al. [90] used Google as a back-end to their system. Their system focused on speech delivery through a chatbot. However, the problem of delivering the correct information was still an issue. In order to narrow down results they created a WebCrawler to crawl local web pages and extract the exact words necessary.

Radev et al. [162] used search engines in order to retrieve the correct answers to questions. They built an algorithm called QASM (Question Answering using Statistical Models) with the ability to learn "the best query paraphrase of a natural language question". QASM converts a natural language question into a search engine query. It is based on an expectation maximization technique that learns which paraphrase has the highest probability score on the standardized benchmark. It uses all of the available data in order to estimate the values of missing parameters. The authors assume that the paraphrase is the closest approximation of the original query. QASM uses precision and recall in order to produce a more adequate query. They found that the algorithm provided good results and is likely to produce a list of hits which contain the answer to the question rather than simply the keyword.

Yang et al.[206] explored the use of the web and of WordNet in order to extract terms highly correlated with the query. They introduced their system "QUALIFIER" (QUestion Answering by LexIcal FabrIc and External Resources)which uses external resources in order to provide an answer. It used the method of extracting known elements in the query and the expected answer. This was achieved by extracting several sets of known elements from the query and then using a rule-based classifier in order to find the answer. It employs a two-tiered question taxonomy which corresponds to named entities. It merges the knowledge found on the web and in WordNet in order to extract contextual information from the query. The web is used to retrieve relevant information to the query. They found that the web was very useful for finding world knowledge but lacked information on lexical relationships among terms such as synonyms. WordNet was then used to address this problem. It was used to get the glosses and the synsets of useful terms related to the query. For each run, the authors computed P,the precision, and CWS, the confidence-weighted score. Their results show that use of lexical knowledge from WordNet without constraint does not seem to be effective. This is also true of using the WordNet resource without constraint in conjunction with Web. Using the web in QA improves performance, because Web-based query formulation improved the baseline performance by 25.1 percent in Precision and 31.5 percent in CWS. The best performance (P: 0.588, CWS: 0.610) is achieved when combining the Web and WordNet with constraint.

2.12.1 Acquiring new data

A chatbot system relies on its knowledgebase in order to be able to understand utterances and give meaningful ones in response. A chatbot with no knowledgebase is like a human with no memory or experience of the world. In a closed domain, such as the one considered in this thesis, the knowledgebase contains information about car insurance. This is the domain knowledge. IN our case, it is gathered from the company website (AVIVA), and from the customer service logs that we have access to. Having access to the appropriate data is crucial as without that, the machine cannot learn or store the information that will allow a customer to interact successfully with it. The domain, is relatively small so we can can create an ontology manually to serve as the basis of the system. The main issue, however, resides in open domains, because they need to store so much data that it is impossible to do this manually. Within our closed domain, it is relatively easy have knowledge that is of high quality and reliability. This is because it often comes directly from structured data provided by the service or product owner. In an open domain, it is disproportionately difficult to cover every topic and to predict the topic of conversation. Additionally the user intent is clear in SOCS but not so in open conversation systems for example.

It is possible to use automated or semi-automated approaches to knowledge acquisition on a large scale. Jabberwacky [29] for example acquires information by storing all of the conversations it has, and then adding new information to its knowledgebase. Using template and pattern matching on the queries means that information can be added automatically to the existing body of knowledge. The problem with the method is that a substantial amount of information must be stored manually to start with. Additionally, the user input must be provided in a machine-friendly manner, meaning that it has to be structured and delivered in a way that the machine understands best. This does not allow for a free and natural conversation, if the method is to work well.

The Internet offers a wide body of knowledge to exploit but the main drawbacks are data quality and reliability. Rzepka et al. [87] use snippets from Google search results in their system to stay within the context of the conversation. They extracted nouns and noun phrases from the utterances and then extracted word association lists from the Web. This allowed them to generate a response using the extracted associations and by adding modality to the sentence ("Oh, smoking is unhealthy, isn't it"). Only the strongest association from the nouns, verbs and adjectives association groups is chosen and used for a response. Over 80 percent of the extracted word associations were correct. The limitation though was that the system didn't take into consideration semantics, speech acts and context or any real particularities in language which would make word association more ambiguous and would probably lower system precision and recall. The authors do not cover the quality of the snippets extracted in their paper unfortunately.

Richardson et al. [168] looked at building knowledgebases by mass collaboration over the Internet. They used first-order probabilistic reasoning techniques to combine potentially inconsistent knowledge sources of varying quality, and machine-learning techniques to estimate the quality of knowledge. In this model, quality, relevance and consistency of the data is again an issue. It also relies heavily on users being motivated to update the knowledge base with good input.

There is, however, a good chance of using the semantic web, because the data is already structured. This means that the main infrastructure for collaborative knowledge base construction is already in existence. There are increasingly more and more OWL and RDF type resources being made available, which make them a good source of knowledge for systems like chatbots. This is an even more interesting option we we consider Curé's tool [45] that allows for the integration of data stored in relational databases into Semantic Web compliant knowledge bases.

2.12.2 Response generation

In order to generate a response, the system must be able to deal with the information gathered and compress this to an adequate size, as well as deliver the response in an intelligible way.

The size of the response must be long enough to give the user adequate information, but not so long as to overwhelm the user. If a conversational chatbot delivers too little information, it does not satisfy the user. If it volunteers too much, the conversational effect is lost. Lin et al. [116] discovered through their experiments that users preferred an answer in a paragraph. They used an encyclopaedic corpus which users queried in order to find out about various facts. The difference between this and a commercial or conversational chatbot is that the users do not have the same goal as in the Lin example. Users in this case are looking for a quick response to a question about a product. They will further query the system if they wish to find out more. The purpose of such a chatbot is to reduce the amount of information that a user needs to deal with. A conversational chatbot fulfils the purpose of having a fluid general conversation with a user. In this case it is unlikely that the user will expect a great deal about a particular subject in one go, but rather gradually as the conversation progresses, as would be the case with interaction with a human.

Spoken natural human-human language is different from typed natural human-human language. Extracting a sentence from a text, summary or template comes from a written format which is not quite appropriate, as it is different from typed human language. For humans to engage with the system, it needs to be able to answer in a natural, human-like manner. Reeves and Nass [167] found that the conversation must display characteristics such as "a warm, friendly disposition" for humans to engage with it fully.

Response formulation can be achieved in several ways. The idea is to draw information from the corpus and provide it in a format that the user can easily understand, and in an appropriately friendly way as discussed above.

To date the most efficient method is to use AIML (Artificial Intelligence Markup Language) templates to generate a response in as natural a way as possible. These templates are populated by patterns which are commonly found in responses and questions. They are laid out in such a way as to deal with the keywords entered by the user in the question. The keyword(s) are "migrated" into the appropriate pattern identified in the template. This method works well. However, after several minutes of discussion, it becomes obvious that the templating method does not offer enough variety in the answers. The answers are often in an acceptable grammatical style, but in some instances they are not coherent, as the experiment carried out by Paradiso et al. shows. Paradiso et al. [149] used "dialogue management rules". These involved a set of techniques from simple keyword based text parsing to more complex logic-oriented methods based on inference mechanisms. These rules have the effect of determining the output of the chatbot, or picking a certain topic based on the sub-division of the topic into different contexts. The agent provides the user with the correct page and then proceeds to explain the page displayed. Context changes are indicated by keywords, and all of them are stored in a hierarchy.

There has been a great deal of improvement in these technologies since the ELIZA system, but most of this is due to an increase in vocabulary rather than an improvement in the techniques used to run the chatbots. It is obvious that the more often a question appears, the easier it is to give an accurate answer, but this does not mean that the conversation will be intelligible, as pattern matching techniques are limited. A small or restricted data set will make for a limited response rate from the chatbot. Using the web as a knowledge base or using a corpus would seem to solve this problem. However, the system must also be able to extract the correct pieces of information from the information, and deliver this in a sentence. It is widely agreed that further work on the integration of NLP techniques into such systems needs to be undertaken.

2.13 Engaging the user

The human factor is an important part of chatbot development. If for example, a very sophisticated and accurate chatbot had been built, but was impossible for people to use, it would fail. The user in an integral part of our research and rightly so. There has been a lot of research and debate around the topic of how best to engage users. This involves not simply a good interface design, but also a range of other considerations such as, for example, a vocabulary aligned with that of the user, the ability to respond in a timely and accurate manner, and the length of the responses or questions, appropriateness of responses, and many more. In this section we look at some of the research carried out in this field.

2.13.1 The Turing test

The Turing test is a good place to begin, seeing as it deals not only with the question of machine intelligence, but also, by way of discussions following Turing's idea, asks whether chatbots need to be intelligent.

The Turing test was invented by Alan Turing in 1950 and described in his paper "Computing Machinery and Intelligence" [190] where he referred to it as "The imitation game". Turing devised the test using the following philosophy: A machine is deemed capable of thinking if it can, under certain prescribed conditions imitate a human by answering questions sufficiently well to deceive a human questioner for a reasonable period of time. He decided on a typed conversation because a viva-voce examination would have to depend on how realistic the voice was, which is not the aim of the exercise. He also predicted that the test would be passed by the year 2000.

Turing stated in his paper, that a thinking machine would perhaps be expected to have free will, but that it was not clear whether humans actually had free will or whether it was an illusion as there is no evidence for this. He believed that giving a machine a task and having it do something unexpected was clearly the machine producing something original rather than something implicitly programmed. In his BBC broadcast "Can digital computers think" [41] he stated that a machine can be described to have a brain if it has been suitably programmed because it can act like a brain, in the same way a animals and human brains can as these could also be described as machines.

2.13.1.1 Objections to the Turing test

There were many objections to the test and have been since the suggestion of such a test. Andersen and Copeland [6] stated that if a machine could think but appeared in-human in its dialogue, it would fail the Turing test. Shannon and McCarthy [134] stated that a machine which works on the basis of a look-up table which has every possible answer would be unthinking and unintelligent and still pass the test. It is interesting to note that all chatbots designed at this time contain a large database containing as much information as possible so as to be able to respond correctly as often as possible, which is in line with this objection.

The objection "What is the point of such an exercise" led to Turing to counter with the view that at the end of the century the concept of machines thinking would be generally accepted. With the growing interest in the Turing test and chatbot technology, it is clear that although still in its infancy as a research area, a number of people, at least those participating in the Loebner prize (the yearly Turing test event) and involved in the research clearly believe this as efforts continue to produce such a system.

The 9 main objections to the Turing test were as follows:

1.The theological objection

This idea is based around assuming that thinking is a function of man's immortal soul. It is believed that God has given an immortal soul to every living man and woman but not to animals or to machines. Hence, no animal or machine can think. Turing responded to this objection, saying that different religions and belief systems have differing opinions on the matter of the soul. He also stated that this view was unacceptable.

2.The heads in the sand objection

This objection revolves around thinking that the consequences of machines thinking would be too dreadful to contemplate. Turing found this objection to be unworthy of a refutation and did not elaborate any more than this.

3.The mathematical objection

There are limitations to discrete machines Goedel [K.Goedel, 1931], Kleene [Kleene,], Rosser [Rosser and Schoenfeld, 1962], and Church [Church, 1936], included by Turing in 'Intelligent machines" all found that mathematical logic shows that there are limitations in discrete machines, thus implying that they cannot think. Turing responded that:

"In any powerful logical system statements can be formulated which can neither be proved nor disproved within the system, unless if possibly the system itself is unstable."

It was suggested that the imitation game was "subject to the Lucas-Penrose constraint", which implies the existence of a class of "unanswerable" questions. In response, Turing stated that these were only a problem if humans could answer them.

4.The argument for consciousness (Jefferson Listers objection [Jefferson, 1949])

Jefferson Lister's [96] stated that "not until a machine can write a sonnet, compose a concerto, because of thoughts and emotions felt and not by the chance fall of symbols could we agree that machine equals brain". Turing stated that we cannot be sure that anybody thinks and that this does not really need to be addressed as far as the test goes.

5.Arguments for various disabilities

This objection says that "Machines may be able to enjoy a delicious dish, but any attempt to make one do so would be idiotic". Turing put this down to people misunderstanding the test and believing that the machine would actually be a human.

6.Lady Lovelaces objection

Lady Lovelace said [185] that the analytical engine has no pretensions to originate anything. It can do whatever we how to order it to perform. Hartree [84] added to this that this doesn't imply that the creating of a machine which can think is not possible. Things however had evolved since Lady Lovelace's objection and so both Hartree and

Turing believed that she did not have all the evidence needed to pass judgement. They agreed that as Hartree put it "One could set up a conditioned reflex which would serve as a basis for learning". A variant of this objection is that "A machine can never do anything new" to which Turing countered with "There is nothing new under the sun".

7.Argument for continuity in the nervous system

This objection states that "The nervous system is certainly not a discrete-state machine so it cannot be modeled with a discrete-state system". Turing countered that simple analogue machines have shown that it is possible to get the same results from a digital machine.

8.The argument for Informality of behaviour

This objection states that "If each man had a definite set of rules of conduct by which he regulated his life, he would be no better than a machine. But there are no such rules so men are not machines". The main objection here is that because computers work on rule-tables they can never be intelligent. Turing stated that there are no scientific observations where we could say that we have searched enough.

9.The argument for extra-sensory perception

This objection states that computers are not capable of being telepathic, whereas in humans, it was claimed, the "statistical evidence for telepathy is overwhelming." Turing proposed a test where one of the humans was telepathic and both machine and human guessed at the card being held up. He also stated that humans do many kinds of thinking without using extra-sensory perception, so machines can also respond well without using this. Turing went on to suggest that a good method for creating thinking machines would be to produce a system able to simulate a child's mind and educate it meaning that it would be able to develop an adult mind though rewards and punishments, which is reinforcement training. He stated that he did attempt to do this but that it was not sufficiently analogous to the kind of process by which a child would really be taught.

Interestingly the A-I.com company attempted to do this using a chatbot called HAL. It is described as A software program capable of acquiring language like a child: Through practice and training, trial and error. It consists of a brain and a personality. The brain is a machine-learning algorithm acquiring conversational skills from the supervised learning method. The personality is the accumulated experience and knowledge possessed by an instance of a Brain, subjected to training by a particular trainer.

In light of the discussions put forward, Turing proposed that the question "Can machines think?" be changed to "Can machines play the imitation game?"

The new form of the problem can be described in terms of a game which we call the imitation game. It is played with three people, a man (A), a woman (B), and an interrogator (C) who may be of either sex. The interrogator stays in a room apart from the other two. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows them by labels X and Y, and at the end of the game he says either X is A and Y is B or X is B and Y is A. The interrogator is allowed to put questions to A and B.

2.13.1.2 The Chinese room argument

John Searle devised the Chinese room argument in his paper ”Minds, brains and programs” in 1980 [174]. The experiment involves someone who only speaks English in a room with instructions on how to manipulate strings of Chinese characters. This way it appears that the person in the room can speak Chinese. This is intended to show that although a computer can be programmed to appear able of using language using syntactic rules to manipulate symbols; it is still incapable of understanding the meaning of these words.

One argument against this is this experiment is that if the machine can interact with the human, then it may be deemed to be of some intelligence, rather than having been programmed to give responses to symbols, which a human could also if s/he learned the rule book. Searle believed that instantiating a computer program is never by itself a sufficient condition of intentionality. He also stated that ”any attempt literally to create intentionality artificially (strong AI) could not succeed just by designing programs but would have to duplicate the causal powers of the human brain”. As he suggests, strong AI is about programs, and no program alone can think. He also states that ”no one would suppose that we could produce milk and sugar by running a computer simulation of the formal sequences in lactation and photosynthesis, but where the mind is concerned many people are willing to believe in such a miracle”, Eccles [60] agrees with Searle stating ”Most of Searle’s criticisms are acceptable for dualist interactionism, it is high time that Strong AI was discredited”.

A number of peer objections appeared in the Behavioural and brain sciences journal. Abelson [1] objected by stating that ”we might well be humble and give the computer the benefit of the doubt when and if it performs as well as we do”, claiming that Searle was not convincing in his argument of intentionality in understanding. Bridgeman [23] stated that ”Searle may well be right that present programs [172] do not instantiate intentionality according to his definition. The issue is not whether present programs do this but whether it is possible in principle to build machines that make plans and achieve goals. Searle has given us no evidence that this is not possible”. It is also obvious that Searle as Fodor [66] statements give no supporting evidence for the claim that biochemistry is at work in the thinking process. Fodor states that there is ample empirical evidence that symbols are involved in mental processes, as is evidenced by the work done in psychology and linguistics.

There has been a lot of debate on the question of the Chinese room, and it can be seen that it is quite clear that there is no evidence that machines can or cannot think, or how we define thinking as a process. This would suggest that the Turing test, if anything is a way of testing various assertions. The machine which is successful shall no doubt shed some light on whether we can accept that machines can think or not.

Turing states in ”computing machinery and intelligence”: ”May not a machine carry out something which ought to be described as thinking but which is very different from what a man does” [190]. This suggests that machines need not respond or think as humans do in order to prove themselves as ”intelligent”. We aim to explore this through our thesis, determining whether service oriented chatbot systems need to respond as a human would, or whether they need to be able to provide information in a humanly accessible and engaging way. In fact as Turing states that ”the best strategy would be to provide answers that would naturally be given to man”.

2.13.2 Creating chatbots with human behaviour

In order to be able to engage the user, it is agreed that the chatbot must have some level of human-like conversational ability. In order to respond to this need, several methods have been developed. This section presents the research carried out in this field, and discusses the methods used. We discuss research about personality, adaptive language, life-like avatars, humour, human-machine behaviour and affection.

Galvoa et al.[70] created a system which allowed the use of different personality modes. They found that personality improved the chatbot system. They extended the AIML system in order to incorporate individual persona components such as beliefs, moods, physical traits and sentiments. Each element is defined by an expression including other elements. Specific rules for attitudes and emotions can then be pre-loaded for various demographic groups (children, teachers). This chatbot responded to the conversation by using a particular tone which enabled it to express dislike or happiness.

Gebhardt et al. [71] were able to show that the acceptance level of chatbots can be significantly improved using a life-like avatar. They concluded that dialogue scripts were not sufficient in order to engage a user. They found that dialogue, as well as being labour intensive, leads to exhaustion of the persona effect after a few exchanges. This refers to the perception that the user has of the chatbot. They used user stereotypes, inference techniques, adaptive language systems and studied human-human exchanges.

Paradiso et al. [149] also found that chatbots displaying lifelike avatars created a positive connection between the user and the chatbot. On the other hand, Reeves and Nass [167] found that the avatar made no difference to the interaction between the human and the machine. They also found that when the machine displayed some kind of affective property, such as praising the user for example, users responded more emotively. Their research points to the fact that the persona can be implied in the language during the exchange.

Brockmann and Isard [27] chose to align the behaviour of their chatbot's natural language to that of the user. Reeves and Nass [167] had found during their experiments that users responded to a chatbot using similar language and behaviour to themselves. They experimented with n-gram language models to assess whether this could be simulated by word sequences. They used the OpenCCG realiser which "takes as input a logical form specifying the propositional meaning of a sentence, and returns a ranked list of surface strings which express this meaning according to the lexicon and grammar, where rankings are determined by n-gram language models derived from examples derived from examples of desired realizations." These n-grams were used in a "best-first anytime search" in order to find the realizations as quickly as possible. They then integrated n-gram models which interpolate a cache, meaning that the alignment of the realizations can easily be produced. They found that it was possible to align behaviour based on this method, but conceded that this represented merely one aspect of human dialogue behaviour.

Nijholt [7] studied the use of humour in the behaviour of humans during conversation. He agreed that similarity in humour appreciation supported interpersonal interaction. He found that using word or phrase ambiguity could be effective, as where there is expectancy which is different to what actually occurs, there is generally humour. He proposes that WordNet can be used in order to find a meaning which does not fit the context or is in semantic opposition.

Davis [51] states that affective computing will need to be performed by systems able to model their emotions on the affective and emotional state of the user. He states that Oatley and Jenkins (The goal-based model of the emotions) agree with Power and Dagleish on the basic emotions which are defined as fear, anger, disgust,sadness and happiness. Davis places emotion at the core of cognitive processing. This provides multi-dimensional valences at a number of representational levels that underpin motivation. Emotion is considered a multi-layer multi-facted descriptor, that informs learning and motivational processes. Davis maps an agent architecture that models motivations in terms of drives, concerns and goals. The different layers are:

- Reflective layer (responsible for meta-control)
- Deliberative layer (manipulate explicit representations about current and possible spaces that the agent can inhabit)
- Reactive processes (response to internal and internal events and Reflexive processes (run automatically, regardless of the agent's space)

Taking these into account, the chatbot must not only be reactive but also reflexive in order to offer a believable state of affective behaviour, see figure 2.16.

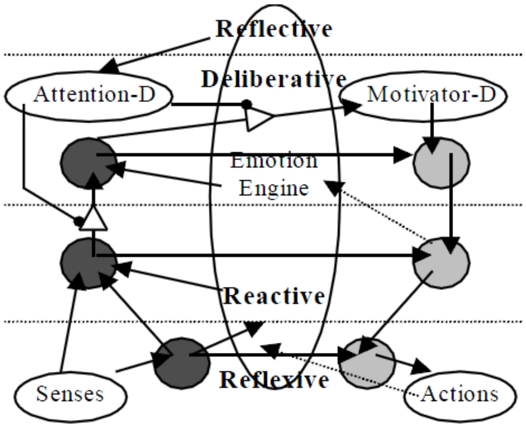


Figure 2.16: A four-layer architecture with the emotion engine as the core. Dark grey circles represent information assimilation. Light grey circles represent mappings onto internal and external behaviours. White triangles are filters. Dashed lines represent feedback.

It is generally agreed that agents with a degree of affective display are more efficient in engaging the user. The main problem with this area of research is that there has been a clear focus on computational methods for returning responses and for dealing with questions, but there has been little work in the psychology of language or the way that humans interact. Reeves and Nass [167] performed simple experiments it was possible to identify important reoccurring behaviour that humans display toward computers and virtual humans.

Searle [174] discussed the role of language and psychology in human-human communication, which is directly applicable to these systems. Saygin and Cicekil [171] demonstrated the importance of pragmatics, and were able to show evidence for Grice's maxims in human-computer conversations. Recently, there has been more interest in this area, but most in the way of embodied conversational agents. The text generation, information retrieval and extraction, conversation flow, language use, "listening skills" and so on needs to be addressed. This is important, as if these systems do not improve to a standard that the user finds acceptable (which is typically a very high standard), users are unlikely to want to make use of them. It is possible that methods borrowed from psychology, psycholinguistics, sociology and other such areas are added to the use of computational linguistics and mark-up methods. Once such systems have gained user acceptance, it will be possible to potentially improve at a much faster rate, due to the feedback generated.

2.13.3 Grice's maxims

Grice's maxims [78] of conversation are guidelines for "cooperative speakers". These maxims are rules deemed necessary for the conversation to be understood by both contributing parties.

The maxims are as follows:

Maxim of Quantity:

- Make your contribution to the conversation as informative as necessary.
- Do not make your contribution to the conversation more informative than necessary.

Maxim of Quality:

- 1. Do not say what you believe to be false.
- 2. Do not say that for which you lack adequate evidence.

Maxim of Relevance:

- Be relevant (i.e., say things related to the current topic of the conversation).

Maxim of Manner:

- 1. Avoid obscurity of expression.
- 2. Avoid ambiguity.
- 3. Be brief (avoid unnecessary wordiness).
- 4. Be orderly.

Saygin and Cicekil [171] studied the transcripts from the 1991 Loebner prize and found that although the mechanical manner of the chatbot violates the maxim of quantity, the information is still relevant. When the maxim of manner is violated, the conversation still appeared to be coherent. As long as the subject matter remained the same, relevance was not violated, although interjections such as "I don't know" could be interpreted as violating this maxim. The human judges found that the systems entered into the Loebner prize all appeared human-like. This would prove that although Grice's maxims are violated, the machine can still produce sufficiently human-like responses. Saygin and Cicekil observe that humans continually violate the maxim of relevance when they use jokes or metaphors for example. They also found that the violation of quality never occurred alone but always with a violation of quantity, manner and especially relevance.

Grice stated that there is always a common purpose or purposes in a conversation and that "this purpose or direction may be fixed from the start or it may evolve during the exchange; it may be fairly definite, or may be so indefinite so as to leave very considerable latitude to the participants". In a service oriented system, there is a purpose to the conversation from the outset, as the user is expecting information on a particular topic. In conversational systems, there is no clear purpose to the conversation or any set topic. The user and the system will volunteer conversational topics and these are indefinite. However, John constantly tries to bring the user back to a topic present in the knowledge base so there is a goal on the part of the system.

Searle et al. [174] found that Grice's maxims failed in human conversations, as it is impossible to know what the purpose of the conversation is: "What constitutes relevance is relative to something that lies outside the fact that it is a conversation, namely the purposes of the participants". This is in direct disagreement with Saygin et al.[171] who found that conversations could be coherent even if the maxims were violated. Reeves and Nass [167] also discovered that users responded better when the machine was able to adapt to their style of speech. This is not possible through templating, but could possibly be achieved through:

- machine learning done on the fly
- recognizing the use of symbols
- the register (i.e. the level of English language used, such the difference between "The Times" and "The Sun") through identifying which words are used which are less common than others seen in the past
- imitating the user simply by using some of the words volunteered

We can conclude that these maxims of conversation, although of significance linguistically, are not all at work in the conversations which we have observed above. Each maxim is violated at some time during the conversation. Mimicking this human behaviour, by violating these maxims in the same way that humans do in the chatbot system being built, may lead to a more convincing performance.

2.13.4 Small talk

Small talk or social protocols are an inevitable and important part of any conversation and so the system needs to be able to respond to these in as human a way as possible. As Cheepen[32] suggests, small talk is used to establish trust. Malinowski [124] defined small talk as *phatic communion*, meaning that some kind of relationship is developed in this way. Tracy and Coupland [106] suggest that "phacity" is

> a persistent goal which governs the degree of politeness in all utterances a speaker makes, included task oriented ones.

We ensure that the system is able to cope well with small talk as it initiates such conversations, and only expands into specific conversation topics when prompted by the user.

2.14 Summary

In this chapter, we have explored and presented techniques that are used to develop chatbots, and techniques that are pertinent to the area of computational linguistics. AIML and DMML are the most widespread purpose-built programming languages for chatbot development to date. Semantic web technology is an interesting area of research and allows for chatbots to be built using RDF, SKOS and knowledgebases like OpenMind. Other methods used belong to natural language processing (NLP) and related fields. We presented a summary of the main NLP techniques and discussed their efficiency. We also presented research undertaken in the area of semantic networks and semantic dependency. We presented research in using summarization techniques for response production. We summarised the research on query reformulation that had been carried out. We then presented a section on knowledgebases, from research done in the area of how best to construct them, to what the most efficient way of adding new data to them is. Response generation research is presented in order to introduce the reader to techniques that are current. We then presented a section on how best to engage the user, where we presented research and observations of previous work. The techniques and methods noted in this section are commonplace in chatbot development. Our own research began from this basis and in the next chapter (Chapter 2), we present the results from our own experiments using some of these methods.

Chapter 3

Our early experiments and observations

In this chapter, we present our early work which involved using existing techniques (presented in chapter 1) and trying to apply them to our chatbot system. We carried out two in-depth experiments, one which involved building a SOCS system (KIA) and another which involved building a social chatbot (John). Both of these chatbots were built using template methods at the base and were used to explore the possibilities and limitations of such methods. KIA was used to carry out extensive user experiments on a closed commercial domain. John was designed for entry into the Loebner Prize competition, in which it was a finalist.

Here we describe both systems, our experiments, and the results of these. The knowledge and experience gained from these systems enabled us to design and build the KIA2 system, described in chapter 4. The early experiments presented in this chapter, have been critical to our research.

3.1 John

This system was designed with Richard Churchill, with the aim of competing in the Loebner Prize, which offers participants the chance to compete against each other and to try to pass the Turing test. Dr Hugh Loebner, the National Science Foundation, and the Sloan Foundation started this annual competition in 1991. There is a $100,000 prize to the developer of the first computer program to pass an unrestricted Turing test. A number of computer programs are designed to respond like humans in a conversation and human participants must decide whether they are speaking to a machine or a human. If they believe they are talking to a human when in fact they are talking to a machine, the machine passes the Turing test. If the Turing test has not been passed, the most realistic chatbot wins prize money and a medal.

3.1.1 John system characteristics

A system such as our systems John or KIA1 (described below), do not need an enormous knowledgebase. This is because the information that they give is centred on a specific topic, and is found only on a particular company website. In the case of KIA1, users tend to visit the site because they are seeking information on a particular product or service. This means that the conversation will remain around that topic area. In a context such as the Loebner Prize, the user is not engaging in information seeking tasks and the domain is wide. It is very difficult to predict the knowledge that the system will need to have. Shannon and McCarthy [175] argued that a machine with a large database would be able to find answers as it could be used like a look up table. KIA worked on this principle, and proved successful on a limited knowledgebase. It is possible that such a system is sufficient for websites of this nature. However, as our experiment shows, there is a need for some personality so as to make the system appear engaging to users. This can also be done in the form of a look-up table and it was successful to an extent. John, our conversational chatbot is built with a small knowledgebase so as to assess whether this is just as efficient as a large one for a conversational system. In the Braithwaite, Jefferson, Newman discussion [123], Turing argued that it really was the size that mattered, the amount of information that could be stored. With John we hoped to prove that a small knowledge base with little information could perform well in the Turing test. Mauldin [132] also states that the restriction to a domain does not weaken the test in any important way. This supports the view that although the system will only have a limited knowledge of the world, it is not invalidating the test.

3.1.1.1 Mauldin's model

Mauldin's model used “tricks” to give the illusion of intelligence when he entered his system Julia into the Loebner prize contest in 1994. He maintained initiative by asking questions, included a portion of the user's input in the system's responses, rigidly continued on a topic, launched into new topics and made controversial and humorous statements. The behaviour of the chatbot was therefore more unpredictable. This method was not efficient enough to actually win or pass the test, although it demonstrated a novel approach.

John is similar to this in some ways. The agent is designed to initiate topics of conversation, and answer with humour where appropriate. There are clear differences though. In order to enable a small knowledgebase to be sufficient to be able to cover a full conversation, we use the concept of term violation and conversation dominance (by initiating and proposing a topic rather than by asking questions). We suggest new topics within the answers. These are topics which are present in our limited knowledgebase.

A sample user conversation with the system is given below:

- User: *hi*
- John: Hiya
- User: *who are you*

- John: I'm John the Ashtanga yoga teacher from Boulder
- User: *where is Boulder*
- John: Boulder is located in Boulder Valley where the Rocky Mountains meet the Great Plains.
- User: *is it nice there*
- John: its cool, not as cool as India though :)
- User: *what is India like*
- John: India is a special place.
- User: *where did you go in India*
- John: I went to Mysore

Further conversation examples are available in appendix B. In all conversations, we found that users were always asking questions to the system rather than answering any, which is the desired effect, as this means that the user does not introduce any out of domain topics, as long as s/he is not trying to trip up the system. If the user does, there is a fall-back method, a random answer, which can be used for almost any interjection. A certain number of small talk responses are designed to bring the user back to the desired topic matter:

- User: *what about pies*
- John: How about yoga
- User: *what is aviation*
- John: I don't remember

3.1.2 The knowledgebase

The John chatbot deals with small talk by using a knowledgebase especially designed for these instances. After a certain time the chatbot will once again try and steer the conversation back to its field of knowledge.

- User: *How's it going*
- John: Not too bad, busy as ever.
- User: *How come*
- John: It's just that way
- User: *Excellent*
- John: I'm into lots of stuff like yoga, meditation, travelling
- User: *You already said that*
- John: Sorry, not paying attention
- User: *You're stupid*

- John: You're excused
- User: *Don't be silly*
- John: That's your problem

All of the data in the knowledgebase was pooled from sources around the Internet. The Internet is notorious for the poor quality of its resources. We therefore selected the websites which we considered to be the most informative and of highest quality. In order to collect information on topics centred on yoga, meditation, or India, for example, we not only selected encyclopaedic resources such as Wikipedia but we also used forums centred on those topics. We collected the knowledge on the information sites using a knowledge collection program which "scraped" the site for textual data and split it into sentences. Data in the forums was collected by hand, as the format of such data is difficult to work with. The forum data is in very informal, conversational format. In order to make the information resource data seem conversational it was necessary to edit each entry by hand. This was possible and a fairly rapid task due to the small size of the knowledgebase, approximately 13,000 words.

Grice's cooperative principle [77], states that turn-taking must be respected in a conversation, so that each participant has a turn in participating. It was suggested that a participant A would provide an utterance, which then allows participant B to follow with another and so on. We found by observing conversations in online chat programs such as Instant Messenger [139] or Google Talk [76] for example, that people often violate this rule. Users were observed to constantly give multiple entries rather than taking turns, as can be seen in the private conversation below:

- Sarah: *so how have you been*
- Jonathan: okay
- Jonathan: its my birthday today!
- Jonathan: wooooo
- Sarah: *happy birthday*
- Jonathan: my sister is engaged!
- Sarah: *excellent*
- Jonathan: interesting is a better word
- Sarah: *why is that*
- Jonathan: cos shes only 20
- Jonathan: and shes only been going out with her girlfriend for under a year
- Sarah: *oh i see*
- Jonathan: i dunno how my folks are gunna react... mmmm
- Sarah: *they'll be ok*

- Jonathan: mmmm i really hope so

We added more than one possible answer for a single topic so that the John system could also violate the turn taking and add more than one utterance. This makes for a more human-like experience, comparable to the human examples above:

- You: why should I do yoga?
- John: Due to the pressure and demands of life, everybody has a tendency to forget what is needed for our life and we often forget the essence of life. We tend to lose touch with the ones we used to spend time with, even ourselves.
- John: Once in a while, we need to find time for ourselves to relax and unwind.
- You: that's a good reason
- John: yea, yoga is also good for things like depression and stuff

We observed Grice's maxims [78] of conversation which are guidelines for "cooperative speakers". These maxims are rules deemed necessary for the conversation to be understood by both contributing parties. We can observe that the John system violates some of these maxims. The system does not respect the maxim of quantity as it is not as informative as necessary in its responses because it makes use of fall-backs when it does not have an appropriate answer in the knowledge base. The maxim of manner is violated because of restrictions in the knowledgebase. The maxim of relation is respected as far as the system is capable of staying on topic. The maxim of quality also depends on the quality of the information in the knowledge base.

These maxims, deemed necessary for a coherent conversation to take place, do not take into consideration the way in which humans habitually converse, especially through online chat. In the transcripts collected from the oral experiment, it is possible to observe that these maxims are also violated. For a larger sample of conversions please see appendix A. An example is given below:

- Customer: *I'd like to find out a bit more about van insurance with Norwich Union*
- Customer service Agent: Ok, the van insurance we offer is flexible. And we provide commercial van insurance for your business but we can provide personal van insurance if you're an individual, so we can service the needs of both.
- Customer: *well I'm actually looking for commercial van insurance because I'm a caterer and I often go abroad to stock up on wine.*
- Customer: *Would your insurance cover me if my van got stolen, or the contents stolen?*
- Customer service Agent: we do have contents insurance for your van and that's part of the commercial van insurance, so you'd like to know what happens with your contents basically - everything is covered

The maxim of quantity is violated in the second utterance of the customer service agent. The maxim of quality is violated in the last customer service agent utterance, as a sweeping judgement is offered. The maxim of manner is violated, because the customer service agent is not brief in his/her answers, and not orderly in the final utterance.

3.1.3 The John system architecture

To aid efficient pattern matching in the John system, allowing the system to select the right response from the knowledge base, we used a part-of speech tagger. We are thus able to extract nouns and proper nouns from user utterances in order to identify possible matches in the knowledgebase. We used ngrams to help calculate a score for each possible match, according to the number of matching nouns or proper nouns found in the user entry. The system chooses the highest scorer as the answer. The highest scorer is the knowledgebase entry that statistically has most in common with the user input. The John system was designed to function on a very small knowledgebase which was pointedly focused around the area of "Yoga". We expected the user to try and deviate from this topic area, because the goal of the test was to have as human a conversation as possible. We had a library of fall-back responses which were used by the system if the user utterance was out of domain. If this pattern continued, John would select a different fall-back response each time. The fall-back responses were designed to provide a response despite the lack of knowledge, thus failing gracefully. The system also used a number of questions to the user to try and get back into the Yoga knowledge area. Because the knowledgebase was very small and the domain itself was so limited, it was possible to make use of this method, but when knowledgebases are much larger, with a number of different topic areas, a much more granular approach is necessary. Please see 3.1 for a diagram.

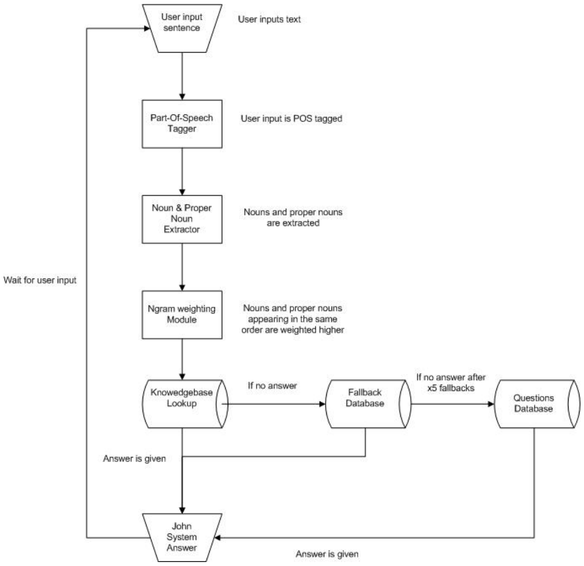


Figure 3.1: John System Architecture

3.1.3.1 The UEA-Lite stemmer

In order to enable sufficiently accurate pattern matching between the chatbot knowledgebase and user utterances, we need to be able to match terms occurring in the knowledgebase to those present in the utterance. Words need to be stemmed in order to reduce the corpus size, thus making the matching process faster and more accurate. The pattern matching algorithm extracts nouns, proper nouns and verbs from utterances in order to match more accurately to a possible answer. Each possible answer is given a series of words associated with it which are nouns, proper nouns and verbs. The verbs are listed in their root form as otherwise it would be necessary to list the entire verb conjugation tables, thus unnecessarily expanding the knowledgebase. Light stemming enables there to be matches despite the verb form.

It is generally accepted that for IR purposes it does not matter whether the words are real words or not. However it is useful for text generation that once the correct information is located, that the words in the query are matched and then preserved or reduced to real words, as the text generation stage is facilitated by using this method. Conservative stemmers such as UEA-Lite (described below) produce fewer classes, allowing words to retain their meaning by restricting the number of erroneous stemming results. When used for document retrieval or similarity measures, more aggressive stemmers can suffice as they reduce all words to roughly the same root. When the task involves a more delicate operation such as topic detection or in this case knowledge base pattern matching, the meaning of the words must be conserved as far as possible.

UEA-Lite [97] was created specifically for the task of not only reducing words to their root form but also in the correct spelling. It has been engineered to be of use in information retrieval tasks, and so stems all words to their root form, unless they are nouns or proper nouns. In the same way as other stemmers, it operates on a set of rules which are used as steps. These rules are applied to the suffix, and if the condition is true, the suffix is either dropped or altered. The 139 rules are set in a particular order so that the longest suffix applicable is used rather than a shorter one which could lead to nonsense words and words not stemmed entirely to their root form. Exceptions, such as words which are always plural (e.g. scissors), place names, personal names, scientific terms and other personal pronouns are stored in files which are consulted against. If the word is matched to a word in one of these texts, it is not processed further. This means that words which should not be stemmed are left in their original form. Each word is tested against each rule. When a match is found, the word is altered accordingly. This could mean dropping a suffix, or dropping a suffix and replacing it with another.

The rules ensure that words are not only stemmed but also that they are correctly spelled once they reach their root form. The rules have been defined to accommodate the maximum number of words possible. Therefore, if a rule means that a small percentage of words will be incorrectly spelled but a large percentage will be accurate, we choose to implement that rule. These rules work well for a large number of words; however they can cause misspellings of others. Some examples include words which are always plural such as 'Jeans' and 'Glasses', words which have suffixes like 'aceous', 'able', 'ible', 'ly', words which take an 'e' after stripping 'ing', such as 'Sampling' and finally words like 'versus', which always take an 's'. Proper nouns should not usually be stemmed, except to remove possessives; our implementation respects POS tags if they are present. It does so by using the POS tag to identify the rule that needs to be applied. If the text is untagged the stemmer uses the simple heuristic that any capitalized token not preceded by sentence breaking punctuation is a proper noun.

In order to evaluate the stemmer, we compared it to the other stemmers available, following the method described in Frakes and Fox [67]. The suffix rules were developed initially on a collection of 112 documents, mostly scientific papers. Testing has been carried on using a further collection of 201 papers, the Moby common words list (http://icon.shef.ac.uk/Moby/) and the vocabulary list from Wall Street Journal corpus. The results are presented below in Tables 1 and 2. In calculating the performance of the stemmer on the WSJ and scientific paper sets, the result of a change made by the stemmer is counted as correct if it results in a correct word that also denotes the same concept. Changes to incorrectly spelt words have been marked as correct where the misspelling does not affect the stemming and which would be correct apart from the misspelling. Many words only occur in certain forms, most commonly part participles with an un- prefix (e.g.”’unwanted”’ stems to ”unwant”). Changes to these and other words which result in grammatically feasible but non-existent words have been marked as wrong.

We tested UEA-Lite on the WSJ corpus, where we paid particular attention to the number of words that it stemmed, and the number of those that it stemmed correctly. We ran the UEA-Lite stemmer on various corpora, and checked the accuracy of each word that it stemmed. Looking at the results of the WSJ test run, we can see that few words were actually selected for stemming (20.24%), but the large majority of those were stemmed correctly (85.61%). On the scientific paper collection,even fewer words were selected for stemming (15.04%), but again the majority of those were correct (89.72%). The Moby corpus test run shows yet fewer words selected for stemming (4.9%), but again the accuracy rate is high (84.83%). Please see tables 3.1.3.1, 3.1.3.1 and 3.1.3.1 for more information.

WSJ test results			
*	Tokens	Stemmed	Spelled correctly
Number	49,204	20.24%	85.61%
Frequency	1,173,766	13.01%	91.18%

Results on scientific paper collection			
*	Tokens	Stemmed	Spelled correctly
Number	44,028	15.04%	89.72%
Frequency	1,189,357	13.91%	93.61%

Moby test results			
*	Tokens	Stemmed	Spelled correctly
Number	74,550	4.9%%	84.83%

The final test compares the Lovins stemmer [119], Paice-Husk stemmer [148], Porter stemmer [156] and UEA-Lite. We use the CAVASSOO project corpus which consists of scientific papers of an environmental nature. This test shows that UEA-Lite stems the least number of words in this corpus. The Porter stemmer stems slightly more than UEA-Lite but does not produce an accurate spelling for the stemmed word. Both Paice-Husk and Lovins are much more aggressive stemmers, but lack the accuracy of UEA-Lite. Please see table 3.1.3.1 for more information.

CAVASSOO test results				
*	Lovins	Paice/Husk	Porter	UEA-Lite
Mean	1.72	1.98	1.16	1.15
Std. Dev.	1.64	1.92	1.40	0.94
Minimum	0	0	0	0
25th percentile	0	0	0	1
Median	1	2	1	2
75th percentile	3	3	2	3-2
Maximum	10	13	9	6

We have developed a stemmer which is designed to conservatively stem suffixes to correctly spelled words. Our results show that it consistently meets these goals in approximately 85 percent or more of words that it stems. It leaves the majority of words untouched, and is most similar to the Porter stemmer, but differs in that it stems words to correctly spelled roots. This stemmer enables us to further compress the knowledge base and to enable more efficient pattern matching. For this reason we chose to use the UEA-Lite stemmer in our system. The GROK (Genealogical Relations of Knowledge) project has also made use of UEA-Lite, but found that it was too conservative for their needs and they subsequently incrementally modified the code against a list of 200000 words from their own corpus [122]. The stemmer is also reportedly used quite effectively in a number of open source projects, of which a sentiment analysis tool.

3.1.3.2 Using a vector model rather than the ALICE Graphmaster

The ALICE chatbot [195] uses the Graphmaster [194] in order to match the input to a pattern in the AIML templates. The Graphmaster matches every word in the utterance in sequence, which means that all conceivable ways of utterance wording has to be considered. This is a very laborious task and is very rigid. The same sentence could be entered by a user multiple times, but with slightly different wording, and still be seen as new to the system. The probability of not finding an answer is much higher in this way. If an exact match is not found, then the system will have no response.

To deal with these problems, John uses a vector approach. The words are weighted in order of importance in the query by using ngrams and a weighting system. The weighting system is based on proper nouns, noun phrases and nouns being the most important types of word in an utterance, in that order. If an entry in the knowledgebase contains the same important words, it is selected as being valuable in the response. If several entries in the knowledgebase have the same nouns, but the proper nouns or noun phrases are not mentioned anywhere, the system will choose the one with the highest match and then also request further information from the user. The context of the query can then be considered when selecting the correct response, rather than using a ”bag of words” approach. It is possible to identify the topic being discussed by the user and then to determine which knowledge should be used in the response. We then use an AIML template and inserted relevant words which are present in responses found in the knowledgebase. This model is much more flexible than the Graphmaster approach and is unlikely to reach a bottleneck, such as the Graphmaster approach is likely to encounter at some point. It also does not involve any laborious tasks, such as entering all possible responses into a database either.

The Graphmaster consists of a collection of nodes called Nodemappers. These Nodemappers map the branches from each node. The branches are either single words or wildcards. There are really only three steps to matching an input to a pattern:

```
If you are given (a) an input starting with word ”X”, and (b) a Nodemapper of the
    graph:

Does the Nodemapper contain the key ”\_”?
If so, search the subgraph rooted at the child node linked by ”\_”.
Try all remaining suffixes of the input following ”X” to see if one matches.
If no match was found, try:

Does the Nodemapper contain the key ”X”?
If so, search the subgraph rooted at the child node linked by ”X”, using the tail of
    the input (the suffix of the input with ”X” removed).
If no match was found, try:
```

```
Does the Nodemapper contain the key ”∗”? If so, search the subgraph rooted at the
    child node linked by ”∗”.
Try all remaining suffixes of the input following ”X” to see if one matches.


If no match was found, go back up the graph to the parent of this node, and put ”X”
    back on the head of the input.
```

For completeness there should also be a terminal case:

```
If the input is null (no more words) and the Nodemapper contains the <template> key,
    then a match was found. Halt the search and return the matching node.
```

If the root Nodemapper contains a key “*” and it points to a leaf node, then the algorithm is guaranteed to find a match. See 3.2 for a diagram.

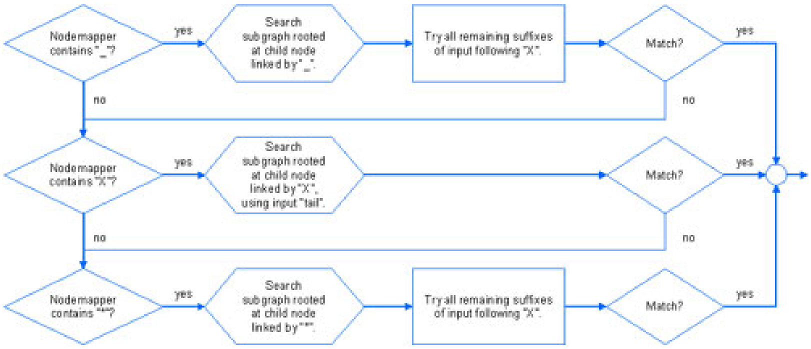

Nodemapper contains "_"?
yes
Search subgraph rooted at child node linked by "_".
Try all remaining suffixes of input following "X".
Match?
yes
no
no
Nodemapper contains "X"?
yes
Search subgraph rooted at child node linked by "X", using input "tail".
Match?
yes
no
no
Nodemapper contains "*"?
yes
Search subgraph rooted at child node linked by "*".
Try all remaining suffixes of input following "X".
Match?
yes

Figure 3.2: Alice Graphmaster

AIML allows us to store responses in categories, in a tree-like structure which is managed by the Graphmaster. The matching is done word by word, not by category.

The algorithm combines the input pattern, the <that> pattern, and the <topic> pattern into a single path or sentence such as:

```
”PATTERN \(<\)that\(>\) THAT \(<\)topic\(>\) TOPIC” and treats the tokens
    \(<\)that\(>\) and \(<\)topic\(>\) like ordinary words.
```

The PATTERN, THAT and TOPIC patterns may contain multiple wild-cards. This again shows the limitation of the method. A word by word approach cannot identify context as such, and patterns do not take into consideration the importance of certain words above others. A useful metaphor for this method is the dictionary indexing system. It is possible to find individual words but not combinations of those. Wildcards (”∗”) and underscores act like special characters between [0-9A-Z]. The method proposed is to first search for the patten using the prefix, then for patterns with the prefix followed by an underscore, and then those followed by ("*"). A very large knowledgebase is necessary to accommodate this type of system. In fact ALICE has 41,000 categories to date. Figure 3.3 gives further insight into the Alice system.

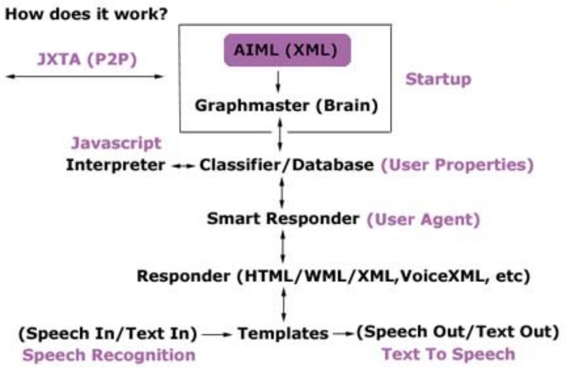


Figure 3.3: The overall Alice system

3.1.4 Performance of the John system

The John system was built to compete in the Loebner prize, and made it through to the final round. The contest took place at University College London on September 17th 2006. The judges were amongst others, professor Kevin Warwick (University of Reading), professor John Barnden (University of Birmingham), Victoria Butler-Cole (a Barister) and Victoria Butler-Cole, and Graham Duncan-Rowe (a journalist). The rules for the 2006 contest are available online [117]:

```
The 2006 Competition will be a RESTRICTED CONVERSATION contest.


Each male Confederate will begin his interactions with each Judge by
asserting: "Hello, my name is John and I am a man."


Each female Confederate will begin her interactions with each Judge by
asserting: "Hello, my name is Joan, and I am a woman.


Each Entry MUST make one of those two assertions.  The choice is left
to the entrant.


This assertion must be made on the first interaction between the Entity and every Judge, but
```

> not necessarily more than once for each Judge. If a Judge re-questions an Entity it is not expected or required that the Entity reassert its name and sex (unless a line of question dictates this).
>
> Following the initial forced assertion each Entity should respond in such a manner as to attempt to convince the Judge that the Entity is a human responding truthfully.

The scoring method was as follows:

> The Final Four Competition will be scored using the Method of Paired Comparisons.
>
> The task of each Judge will be to apportion 100 points between the two terminals based upon how "human" they seem. The judgement is relative, and ties are forbidden; one of the two terminals must be awarded at least 51 points. No judge may use the same apportionment twice. That is, if a judge apportions 100 points 40/60 for one Entry/Confederate pair, then he or she may _not_ use a 40/60 apportionment again. This will prevent ties if it becomes necessary to convert the judge's scores into rankings.
>
> We wish (a) each Entry to be compared with every Confederate; (b) each Judge to evaluate every Entry, (c) each Judge to evaluate every Confederate.

The Entry with the highest column total was declared the winner. Our system, John, made it into the final 4 chatbots and we therefore attended the event. The system was however voted the worst out of the 4 systems to be tried that day, by the judges. The results were as listed in table 3.1.4.

Loebner 2006 test results

Place	Average rank	System	Author
1st	3.75	Joan	Rollo Carpenter
2nd	3	Ultra Hal	Robert Medeksza
3rd	2.25	Cletus	Noah Duncan
4th	1	John	Jenkins - Chruchill

Transcripts from the competition are available online [118] and sample conversations with John are available in appendix B. The issue appears to have been that once the judge discusses anything outside of John's knowledge area, the system gives repeatedly the same answer ”I don’t know, I’d rather talk about yoga”. The system is also quite incapable of having a conversation for any length of time. If the conversations had been shorter, the quality would have been better. This method although effective on a specific knowledge area is too laborious and does not adapt well to a changing conversation.

3.2 The KIA1 system

KIA (knowledge interaction agent) was built to test human responses to different chatbot behaviours. The aim of this experiment was to assess how users responded to an autonomous agent in a commercial setting. KIA differs from John in that it is draws its knowledgebase from the AVIVA website car and health insurance sections. This data is stemmed and then processed through a template builder which allows us to store AIML templates of relevant information and patterns. KIA has small talk capability, as well as domain information. It is also designed to test the user, for example, by not always giving the right information, asking repeatedly for a reformulation, or changing the subject abruptly. John was design to compete in the Loebner prize whereas KIA was designed ot help us understand how user reacted to different chatbot behaviours and determine what they expected from such a system.

3.2.1 KIA system characteristics

We used knowledge from a closed domain (insurance) and asked users to interact with the system in the same way they would with a customer service agent. This suggestion was, however, ignored in some cases, but we deemed this to be positive as it allowed us to understand why users had trouble using the system. The fact that they could not all relate to the interface in the same way as they would a human being, allowed us to gain insight into what was lacking from that dynamic. We discovered that users who told us that they found it hard to use the ”system”, often meant specifically the interface and not the system itself. Others reported finding the whole concept quite difficult to get used to, as they didnt́ like the fact that they were talking to a machine and not a person. The insurance website we carried this test out on already had a chatbot on the site, but it was not being used by visitors, although it was in plain view. The reasons for this, and for other difficulties that users shared with us are described in this section, together with the experiments that we undertook as a result of finding out about the problems. Example conversations with KIA can be found in appendix J.

3.2.2 KIA: system architecture

KIA was built specifically for the task of monitoring human interaction with such a system. Our focus was around how to ”‘provoke”’ the user in different ways, rather than trying to give the best answer each time. The reason for doing this was to test different types of human-machine behaviours. We wanted to know how they would react to a variety of different social situations with the SOCS. We tested a large number of behaviours:

1. Being given an unexpected response

2. Being given the wrong answer repeatedly
3. Non-committal responses (such as ”‘maybe”’)
4. Unhelpful responses
5. Humour
6. A long paragraph of text as opposed to a succinct response
7. Small talk
8. Being ignored (no response)
9. Personal questions (for example about their health)

KIA was built using natural language processing techniques and AIML templates. We used the same method used in the ALICE social chatbot system, which involves seeking for patterns in the knowledgebase using the AIML technique, as we did in the John chatbot system. The knowledge base was built using the Norwich Union website. AIML templates were automatically populated with nouns, proper nouns and common sentences. We then manually corrected errors and added a “chat” section to the knowledgebase from which the more informal, conversational utterances could be drawn. This was triggered automatically every 5 utterances. The idea behind this was that we wanted the system to disrupt the conversation periodically in order to observe user reactions to inconsistent behaviour. The nouns and proper nouns served as identifiers for the utterances and were initiated by the user utterance. Please see 3.4 for a diagram of the system.

The chatbot was programmed to deliver responses in a friendly, natural way. We incorporated emotive-like cues such as using exclamation marks, interjections, and utterances which were constructed so as to be friendly in tone. Content was added into the knowledgebase, giving information on health issues like pregnancy, blood pressure and other such topics which it was hoped would be of personal interest to users. The information on services and products was also delivered using as far as possible the same human-like type of language as for the “soft” content language.

The interface, shown in figure 3.5, was a window consisting of a text area to display the conversation as it unfolded and a smaller text box for the user to enter text. An “Ask me” button allowed for utterances to be submitted to the chatbot. For testing purposes the “section” link was to be clicked on when the user was ready to change the topic of the discussion as the brief was set in sections. We also incorporated the picture of a woman smiling in order to encourage some discussion around visual avatars. The simplicity of the interface was designed to encourage user imagination and discussion; it was in no way presented as an interface design solution.

3.2.3 Description of the human interaction experiment and results

Users were given several different tasks to perform with the chatbot. They conversed with the system for an average of 30 minutes and then completed a feedback questionnaire which focused on their feelings and reactions to the experience. The same framework was used to conduct ”Wizard of Oz” experiments to provide a benchmark set of

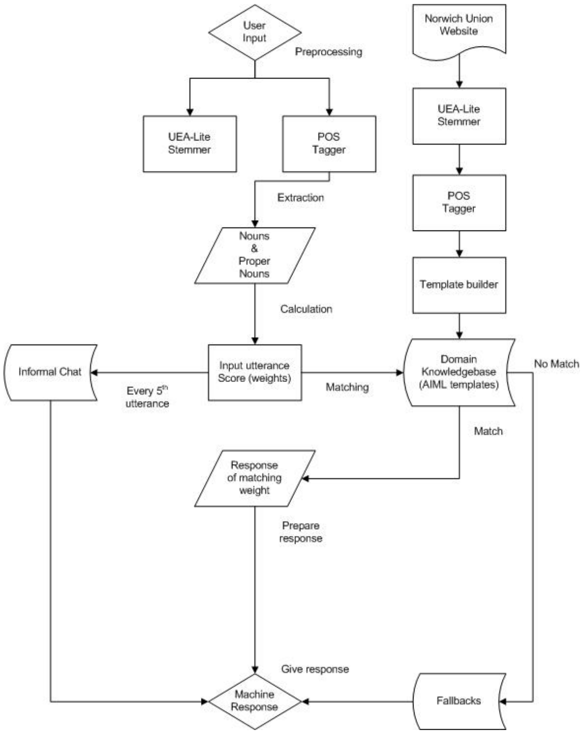
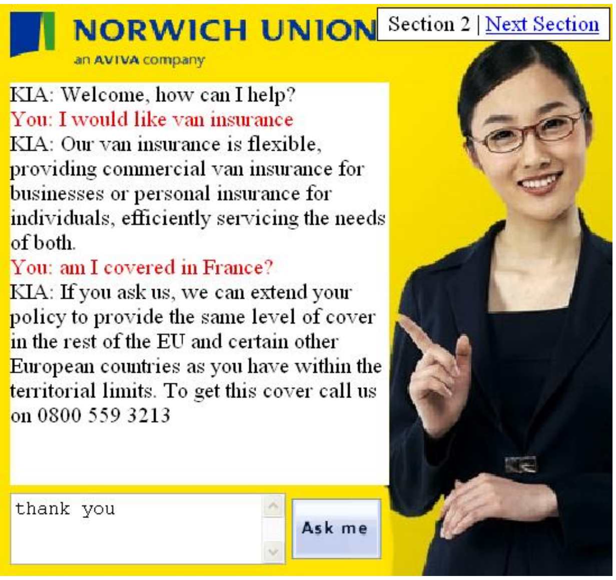


Figure 3.4: Kia system Diagram



KIA: Welcome, how can I help?
You: I would like van insurance
KIA: Our van insurance is flexible, providing commercial van insurance for businesses or personal insurance for individuals, efficiently servicing the needs of both.
You: am I covered in France?
KIA: If you ask us, we can extend your policy to provide the same level of cover in the rest of the EU and certain other European countries as you have within the territorial limits. To get this cover call us on 0800 559 3213

thank you

Figure 3.5: The KIA interface

reactions, in which a human took the customer representative role instead of the chatbot. We refer to this person as ”the Human chatbot” (HC) in this thesis, because they are simulating a chatbot.

We conducted the study on 40 users who had a full range of computer experience and exposure to chat systems. The tasks the users were requested to undertake were formulated after an analysis of Norwich Union’s customer service system. They included such matters as including a young driver on car insurance, travelling abroad and more. The users were asked to fill in a questionnaire at the end of the test to give their impressions on the performance of the chatbot and volunteer any other thoughts. We also prompted them to provide feedback on the quality and quantity of the information provided by the chatbot, the degree of emotion in the responses, whether an avatar would help, whether the tone was adequate and if the chatbot was able to carry out a conversation in general. The instructions and tasks given to test subjects are available in the task document in appendix C. The feedback form is also available in appendix D.

We also conducted an experiment whereby one human acted as the chatbot and another acted as the customer and communication was spoken rather than typed. We collected 15 such conversations in order to assess how different the oral experience was to the typed one. A sample of transcripts from this experiment can be found in appendix E. The users were given the same scenarios as those used in the human-chatbot experiment. They were in different rooms, and the volunteers were aware that they were talking to a human. We wanted to use these conversations and the behaviours people displayed as a benchmark for the human-machine experiments. We found however that the sorts of things that caused people problems were the same as for the human-machine experiments. For example, a knowledge gap on the part of the HC, meaning that s/he is unable to return an answer is just as annoying when the machine experiences this same knowledge gap.

3.2.4 Results of the experiments

The conversation between the human and HC flowed well, and the overall tone was casual but business-like on the part of the HC, as would be expected from a customer service representative. The conversation between chatbot and human also flowed well, the language being informal but business-like. The raw results of the user The table 3.1 summarises the results for the whole experiment. Each one is discussed in detail below and the results are in table 3.1. Transcripts are available in appendix E of this thesis E.

Experiment Results			
Experience	0.46	Useful answers	0.37
Tone	0.37	Unexpected things	0.2
Turn taking	0.46	Better than site surfing	0.43
Links useful	0.91	Quality	0.16
Emotion	0.23	Interest shown	0.33
Conversation rating	0.58	Simple to use	0.7
Succinct responses	0.66	Need for an avatar	0.28
Clear answers	0.66	Would use again	0.68

Table 3.1: HCI experiment results

1. **User language:**

Keywords were often used to establish the topic clearly such as ”I want car insurance”, rather than launching into a monologue about car problems. The HC repeated these keyword, often more than once in the response. The HC will also sometimes use words in the same semantic field (e.g. ”travels” instead of ”holiday”).

The user tended to revert to his/her own keyword during the first few exchanges but then used the words proposed by the HC. Reeves and Nass [167] state that users respond well to imitation. In this case the user came to imitate the HC. There were occasionally places in the conversation where at times the keyword was dropped altogether such as “so that’ll cover me, right?”. This means that the conversation comes to rely on anaphora. In the case of the chatbot-human conversation, the user was reluctant to repeat keywords (perhaps due to the effort of re-typing them) and relied very much on anaphora, which makes the utterance resolution more difficult for an automatic system.

2. **User reactions:**

Interestingly, at times, information provided by the HC was incomplete or incorrect and sometimes the HC failed to even give an answer. The human reacted well to this and reported no frustration or impatience. Rather, they were prepared to work with the HC to try and find the required information. The HC did not always know the answer to a question because sometimes users ask things that are out of the scope of their knowledge. One such question for example is “Do you think my car will be fixed on time?”. The chatbot is also unable to answer in these circumstances. Users did however report frustration, annoyance, impatience with the chatbot when it was also unable to provide a clear response or a response at all. It was interesting to observe a difference in users reaction to similar responses from the HC and the chatbot. If neither was unable to find an answer to their query after several attempts, users became frustrated. However this behaviour was exhibited more slowly with the HC than with the chatbot. This may be because users were aware that they were dealing with a machine and saw no reason to feign politeness, although we do see evidence of politeness, in greetings for example.

3. Question answering:

The HC provided not only an answer to the question, where possible, but also where the information was located on the website and a short summary of the relevant page. The user reported that this was very useful and helped them be further guided to more specific information.

The HC was also able to pre-empt what information the user would find interesting, such as guiding them to a quote form when the discussion related to prices for example, which the chatbot was unable to do. The quantity of information was deemed acceptable for both the HC and the chatbot. The chatbot gave the location of the information but a shorter summary than that of the HC.

Some questions were of a general nature, such as “I don’t like bananas but I like apples and oranges. Are these all good or are some better than others?” which was volunteered by one user. This is a good example of a user going beyond the scope of the domain. Their question related to health insurance originally. As well as the difficulty of parsing this complex sentence, the chatbot needs to be able to draw on real-world knowledge of fruit, nutrition, and so on, to answer such questions requires the use of a large knowledgebase of real-world knowledge as well as methods for organizing and interpreting this information.

The users in both experiments sometimes asked multiple questions in a single utterance. This led both the chatbot and the HC to be confused or unable to provide all of the information required at the same time.

Excessive information was sometimes volunteered by the user, e.g. as explaining how the mood swings of a pregnant wife are affecting the fathers life at this time. A machine has no understanding of these human problems and so would need to grasp these additional concepts in order to tailor a response for the user. This did not occur in the HC dialogues. This may be because users are less likely to voice their concerns to a stranger, than an anonymous machine. There is also the possibility that they were testing the chatbot.

Users may also feel that giving the chatbot the complete information required to answer their question in a single turn is acceptable to a computer system but not acceptable to a human, using either text or speech.

4. Style of interaction:

Eighteen users found the chatbot answers succinct and three long-winded. Other users described them as in between, not having enough detail in them or being generic. The majority of users were happy with finding the answer in the sentence rather than in the paragraph as Lin [115] found during his experiments with encyclopaedic material. In order to please the majority of users it may be advisable to include the option of finding out more about a particular topic. In the case of the HC, the responses were considered to be succinct and containing the right amount of information. However some users reported that there was too much information.

Users engaged in chitchat with the chatbot. They thank it for its time and also sometimes wish it “Good afternoon” and “Good morning”. Certain users tell the chatbot that they are bored with the conversation. Others tell the system that this “feels like talking to a robot”. Reeves and Nass [167] found, that the user expects such a system to have human qualities. Interestingly the language of the HC was also described as “robotic” at times by the human. This may be due to the dryness of the information being made available; however it is noticeable that the repetition of keywords in the answers contributes to this notion.

5. Feedback forms:

The feedback forms from the experiment showed that users described in an open text field the tone of the conversation with the chatbot as ”polite”, ”blunt”, ”irritating”, ”condescending”, ”too formal”, ”relaxed” and ”dumb”. This is a clear indication of the user reacting to the chatbot. The chatbot is conversational therefore they expect a certain quality of exchange with the machine. They react emotionally to this and show this explicitly by using emotive terms to qualify their experience. The HC was also accused of this in some instances.

The users were asked to rate how trustworthy they found the system to be using a scale of 10 for very trustworthy to 0 for not trustworthy. The outcome was an average rating of 5.80 out of 10. Two users rated the system as trustworthy even though they rated their overall experience as not very good. They argued that the chatbot kept answering the same thing or was poor with specifics. One user found the experience completely frustrating but still awarded it a trust rating of 8/10. The HC had a trustworthiness score of 10/10 in comparison. The KIA system performed less well, but this fact does not seem to have affected how much users are prepared to put their trust into it.

6. Results specific to the human-chatbot experiment:

Fifteen users volunteered, without elicitation, alternative interface designs. Ten of these all included a conversation window, a query box, which are the core components of such a system. Seven included room for additional links to be displayed. Four of the drawings include an additional window for the inclusion of “useful information”. One design included space for web links. One design included disability options such as the choice of text color and font size to be customizable. Five designs included an avatar. One design included a button for intervention by a human customer service representative. The raw feedback from users can be found in appendix F.

A common feature suggested was to allow more room for each of the windows and between responses so that these could be clearer. The conversation logs showed many instances of users attacking the KIA persona, which was in this instance the static picture of a lady pointing to the conversation box. This distracted them from the conversation. Example of such attacks are:

1. “Look at you, stupid cow!”
2. “Do you think you’re clever or something? Watching me?”
3. “What are you pointing at you idiot”
4. “What a stupid bimbo you are!!”

7. The avatar:

Seven users said that having an avatar would enhance the conversation and would prove more engaging. Four users agreed that there was no real need for an avatar as the emphasis was placed on the conversation and finding information. Ten argued that having an avatar present would be beneficial, making the experience more engaging and human-like. Thirteen reported that having an avatar was of no real use. Two individuals argued that the avatar could cause “embarrassment”, and may be “annoying”. Two users said that they thought that having a virtual agent would not help actually included them in their diagrams.

When asked to compare their experience with that of surfing the website for such information, the majority responded that they found the chatbot useful. One user compared it to Google and found it to be “no better”. Other users stated that the system was too laborious to use. Search engines provide a list of results which then need to be sorted by the user into useful or not useful sites. One user said that surfing the web was actually harder but it was possible to obtain more detailed results that way. Others said that they found it hard to start with general keywords and find specific information. They found that they needed to adapt to the computers way of conversing.

8.language:

Most users found KIA to be fast and efficient and generally just as good as a search engine although a few said that they would rather use the search engine option if it was available. One user clearly stated that the act of asking was preferable to the act of searching. Interestingly a few said that they would have preferred the answer to be included in a paragraph rather than a concise answer.

The overall experience rating ranged from very good to terrible. Common complaints were that the system was frustrating, kept giving the same answers, was average and annoying. On the other hand some users described it as pleasant, interesting, fun, and informative. Both types of user gave similar accounts and ratings throughout the rest of the feedback having experienced the common complaints.

The system was designed with a minimal amount of emotive behaviour. It used exclamation marks at some points, and more often than not simply offered sentences available on the website, or which were made vaguely human-like. Users had strong feedback on this matter calling the system ”impolite”, ”rude”, ”cheeky”, ”professional”, ”warm”, and ”human-like”. One user thought that the system had a low IQ. This shows that users do expect something which converses with them to exhibit some emotive behaviour. Although they had very similar conversations with the system, their ratings varied quite significantly. This may be due to their own personal expectations. The findings correlate with the work of Reeves and Nass [167]: people are associating human qualities to a machine. It is illogical to say that a computer is cheeky or warm for example, as it has no feelings.

Translating all of the feedback into numerical values between 0 and 1, using 0 as a negative answer, 0.5 as a middle ground answer and 1 as a positive answer, allowed us to have a clearer idea as to the overall feedback on the experiment. The usefulness of links was voted very positive with a score of 0.91, and tone used (0.65), sentence complexity (0.7), clarity (0.66) and general conversation (0.58) all scored above average. The quality of the chatbot received the lowest score at 0.16. Interestingly, most users would be happy to use the system again, despite its limitations.

3.2.4.1 Findings

Our findings from this work suggest that users expect chatbot systems to perform well, giving them the information that they need, but that they also expect them to behave and communicate in a human-like fashion. By this we mean, not that the chatbots need to behave and communicate exactly like humans, but rather that the user must feel that they are engaged and confident about the system. Williams and Cheepen [198] investigated naturalness in human-machine dialogues in telephone baking systems. She found that users preferred the "denatured" prompts, that were free of all human-like qualities. Although this research is interesting, it is important to note that the interaction with a telephone system is quite different from the interaction with a chatbot system. The telephone banking option is task oriented and closed, because users are not invited to give opinion or further information. They simply say out loud the required information. Chatbot systems are open, in the sense that the user can type anything into input text box, and the chatbot system will process it and try to volunteer an appropriate answer. For both of these systems, user expectations will be different. In our research, we found that, if users feel that the chatbot is seen to be "acting like a machine", they deemed to be below standard. It is required to have the same tone, sensitivity and behaviour as a human, but at the same time users expect it to process much more information than the human. It is also expected to deliver useful and required information, just as a search engine does. The information needs to be delivered in a way which enables the user to extract a simple answer as well as having the opportunity to "drill down" if necessary. Different types of information need to be volunteered such as the URL where further information or more detailed information can be found, the answer, and the conversation itself. The presence of "chitchat" in the conversations with both the human and the chatbot show that there is a strong demand for social interaction as well as a demand for knowledge.

In this thesis we have not carried out further human computer interaction research as we have focused on the natural language understanding and generation component. This experiment gave us good insight into how users behave with such systems and we have incorporated these learnings into our further work. There is however scope for further research into the human computer interaction component. It is not clear from this experiment for example, whether an avatar can help the chatbot appear more human-like or make for a stronger human-chatbot relationship. It would also be interesting to compare the use of search engines to that of the chatbot. It would be interesting to compare the ease of use of the chatbot with a conventional search engine. Many users found making queries in the context of a dialogue useful, but the quality and precision of the answers returned by the chatbot may be lower than what they could obtain from a standard search engine. This is a subject for further research.

3.3 Summary

After researching and exploring the techniques, methods and tools used to build conventional chatbots we have identified some of the limitations and challenges involved in building such systems. We can see that the main shortcomings in the John and KIA systems are that the systems can't generate their own sentences and that they have difficulty processing user input. Most systems, as we have seen, rely on populating ready-made templates with linguistic information that is hard coded or triggered by rules that have been hard coded. We cannot say that any of the systems that we have observed or built display any cognitive aptitude. They do not create their own sentences in response to their own understanding of a user input.

There are a number of natural language processing methods that we have explored in order to help solve the issues with the template approach, namely POS-tagging, named-entity recognition and stemming. We found that overall it was still difficult to produce a human-like expression of interest and we failed to deliver a system that could converse naturally for any length of time in a a closed domain.

The storage and acquisition of knowledge also presents issues because the system performance is dependent on the quality of the knowledge in store. It is necessary to source this data from resources that are accurate and information rich. Open domain systems require a huge amount of knowledge about our world, and this needs to be effectively stored. If the data is retrieved as it is in a database, the unit of knowledge needs to display a variety of characteristics which link it to other units of knowledge. If the information is not linked well enough, there can be no connection between topics and subjects.

It can be said that the user reaction to the system was very emotive, because they had strong human reactions to it. The ways that they describe their experience of conversing with the chatbot could just as easily be applied to another human being. A machine is not able to actually be friendly, rude or cheeky. These are very human characteristics. We can easily see how such a system could fail a website if the user felt offended or upset by it in any way. We experimented with Griece's maxims and observed that the conversation is still affected by this between machine and human, just as it is between human and human. The fact that humans become emotionally involved with the system makes it even more important for it to function in a human-like way, because this is what is expected of it.

It is important to discover a better way of storing knowledge and creating links within it and we need to find a way of making the conversation more fluid and "intelligent", more human. To achieve this, we need to move away from templates that offer too many limitations and we need to experiment with a technique which allows for the system to grow organically as it leans more about the world. Natural language processing in isolation is not efficient enough so we need to find an approach which will deal with language at a higher level. Disambiguating at word level and using word level triggers has not worked in past and present systems. In the following chapter (chapter 3), we present research undertaken by others in the area of natural language understanding and generation. In order to build a SOCS that uses more sophisticated methods, we must first investigate relevant techniques.

Chapter 4

Natural language understanding and generation

4.1 Natural language understanding and generation overview

In this chapter, we present the research covered in the area of natural language generation (NLG) and natural language understanding(NLU). These areas of research have not been specifically investigated with regards to conversational systems but rather in the context of summarisation, machine translation and speech recognition. We first discuss research in natural language understanding, with particular emphasis on semantic analysis, syntactic analysis, and trainable systems. We then discuss the research area of natural language generation with particular attention to application specific grammar, template-based systems, rule-based systems, trainable generation systems and finally the corpus -based approach.

Natural language generation and understanding belong to a sub-field of artificial intelligence and computational linguistics that is important not only for computational applications such as ours, but also for the fields of psychology and cognitive linguistics for example.

The difference between natural language generation and understanding has been summarised by Wilkes [49] in the following way:

> Language understanding is somewhat like counting from one to infinity and language generation is like counting from infinity to one

This description captures the difficulty in both disciplines and the fact that both are challenging in different ways. In our case, natural language understanding is about receiving a user input in natural language (typed text), and determining what the user meant. We determine meaning by looking at the words used, and sometimes we also take into consideration the wider context of the conversation itself. There are many difficulties in natural language understanding. These difficulties are experienced both my chatbots and humans alike. Natural language generation in the context of chatbots, is about generating a natural language response for the user. It could be a response to a question, a statement or the starting point of a new conversation. In this chapter, we aim to discuss the difficulties associated with natural language generation and understanding and the importance of both for chatbot development.

4.2 Natural language understanding

Natural language understanding (NLU) involves extracting meaning from natural language input in order to retrieve the relevant information required for a response. The given input is rendered into a representation that can be easily manipulated by a machine. Syntactic and semantic information is required for the system to be able to disambiguate the utterance fully. Grammatical and semantic rules are used to achieve this. Natural language understanding is ambiguous because we cannot determine how far a computer, or a human, can truly "understand" language. In this thesis we limit ourselves to describing it as effective disambiguation leading to high precision and recall in the system. The philosophical argument however is complex and involved. There has been research in this area since at least the 1950's which can be applied to many systems such as summarisation, machine translation, natural language search engines and others as well as conversational systems. There are a number of approaches that have been presented in the past and number of methods that are common in these systems. It is a AI-complete problem, which calls for the assimilation of common sense reasoning, world knowledge and inference reasoning.

4.2.1 Semantics

Semantic theory has been extensively used in order to achieve natural language understanding in machines. The combination of syntactic and semantic components at sentence level mean that machine can determine the nature and function of a particular sentence. Semantics focus on the meaning of the words, phrases, sentences and texts rather than their function (syntactics). Semantics also includes the study of sense and denotative reference, truth conditions, argument structure, thematic roles and discourse analysis. The combined analysis of these allow patterns to be discovered which result in syntax. Various grammars such as Montague [99] for example, allow for both semantic and syntactic rules to be considered. This is called the compositionality principle and can be found in many NLU systems. Montague grammar however is limited by the change of word senses in relation to their context. In this section we investigate the application of different semantic models to NLU systems.

4.2.1.1 Hartholt et al. on using an Ontology in a Natural Language Oriented Virtual Human Architecture

Hartholt et al. [83] used an ontology to improve the natural language understanding capabilities of their virtual human. The goal of the Virtual Human Project is described by Hartholt et al. as [83]:

”designing autonomous agents that support face-to-face interaction with people in many roles and in a variety of tasks. The agents must be embedded in the virtual world and perceive events in that world, as well as the actions of human participants. They must represent aspects of the dynamic situation in sufficient depth to plan contingencies, develop beliefs and intentions, and form appropriate emotional reactions. They must communicate with each other and with human participants using multi-modal natural language communication.”

Human trainees are required to negotiate and collaborate with the virtual human on a series of task and situations.

The virtual human’s architecture consists of a large number of modules. There is also an Automated Speech Recognizer (ASR) which translates spoken words into text. A Natural Language Understanding module (NLU), that converts natural language expressions into internal representations. They used a training corpus of paired utterance texts and semantic representations (frames). The task reasoner defines how goals are achieved and reasons about possible alternatives for the actions required. The reasoner focuses on states and tasks and plans how both of these work together to produce a causal network. The emotion module determines the beliefs and goals that result from actions and that result in specific coping strategies. It uses the task model representations as well as variables such as temporal status, likelihood, controllability, and changeability. The dialogue manager (DM) keeps track of conversation history and provides context, including task and emotion models. It updates the current state and plans new responses to the user. A Natural Language Generation module (NLG), is responsible for converting the intended response into output text. It uses the DM but also lexical and grammar rules. A text-to-speech synthesizer is used to produce speech from the text output. A non-verbal behaviour generator is used to determine which body movements the virtual human should make, in order to convey the meaning of the NLG output. There are also a number of modules that control the visual aspects of the virtual human, such as background scenery for example.

The ontology is used as a terminology repository, to ensure that the terminology used is always consistent. They defined the inter-relationships between the terms such as hierarchy and constraints on frames’ slot values. The task model and the NLU were the two modules using the ontology at the time of publication. They used Web Ontology Language (OWL) [135] and Stanfords Protégé [105] to manage it. They used two iterations of their ontology: one using frames and the other using OWL. The frames version allowed them to have integrated data sources. The move to OWL allowed for the re-use of knowledge and more structured ontology design. They were able to create a hierarchical structure of domain independent and domain dependent concepts. OWL also allows for assertions to be added to imported objects as well as those that are created at a particular level.

They found that using an ontology provided them with certain advantages. The knowledge used by the task model and the NLU were synchronized, only valid semantic frames can be created, users reuse knowledge by combining individuals in the ontology, and it proved to be a safe repository for the knowledgebase, because when data changes, it is referenced and not copied. They noted that this method requires a certain learning curve, and that the interface is not really geared to this type of task. Integrating an ontology into their virtual human was seen as a milestone in their progress to achieve a production-level system.

4.2.1.2 Dahlgren's research on cognitive NLU

Dahlgren et al. [47] explain that NLU systems cannot depend on semantic theory alone. The authors seek to recreate the human cognitive approach to NLU. Their system aimed to allow a user to specify an area of interest and the NLU component would read and process all of the textual material and return information pertinent to the user's information seeking goal. The idea is that traditional semantics does not take into account the fact that word meanings shift according to changes in the environment. The classical theory says that there are a stock of concepts from which all word meanings are built. Dahlgren et al also expose other issues with classical semantics for example word meanings varying across lexicons, truth conditions not being straightforward. Additionally, it doesn't manage exceptions well (i.e. an albino tiger is still a tiger). Shallow semantics is a lexical theory where each word sense is associated with a naive theory. This naive theory is a set of beliefs about objects, events and people which makes up the system's world knowledge. These naive semantic representations are different to classical semantic sets because they are not made up of a closed set of primitives. They are also only made up of the information that is required to form a semantic interpretation incrementally. Dahlgren et al. also created a naive semantic ontology which combined both syntax and word senses [48]. This particular system is made up of a linguistically-based content ontology which represents the world view, created in natural language.

> ”The content ontology (as opposed to the formal semantic ontology which distinguishes events from propositions, and so on) is best grounded in the culture, rather than in the world itself, or in the mind. By world view we mean naive assumptions about what there is in the world, and how it should be classified. These assumptions are time-worn and reflected in language at several levels : morphology, syntax and lexical semantics.”

This particular method was very efficient for disambiguating words, syntactic structure, formal semantic representations, resolving anaphoric expressions and performing reasoning tasks within text. This promising approach has led to the development of the Cognition natural language search engine [46] which understands the meaning within the context of the text it is processing. Users can query in natural language and the engine has higher precision and recall than Google, but then again it only performs on closed environments (Health, wikis, legal and the Bible).

4.2.1.3 Denis's dialogue understanding system

Denis et al. [54] created a dialogue understanding system using a deep parsing and rule-based semantic analysis. The system is designed for spoken natural language understanding but the technique can be applied to text based systems like our own. They evaluated their system by entering it into the EVALDA/MEDIA which requires the system to act like a hotel receptionist. The data provided for this evaluation is annotated. Each utterance is segmented into meaningful chunks and associated with a single semantic feature. The tags are triplets in the form of:

```
<mode, attribute, value>
```

where mode describes whether the chunk is positive, negative or neutral, the attribute is defined by the semantic category and value specifies whether it is a string, an integer or a constrained value. The semantic dictionary contains sets of allowed triplets. The corpus of 1,257 spoken dialogues has been manually annotated. A compositional semantic builder produces a conceptual graph of the syntactic analysis. They use the multi modal Interface Language (MMIL) which provides a semantic lexicon of 150 schemes defined as couples of a semantic anchor and a semantic typed fragment. All of these are linked to the internal ontology which is composed of 220 concepts related to the hotel reservation task. The ontology helps refine or define a faulty concept graph created from inefficient parsing. Finally a projection flattens the graph and constructs the target output. The projection is achieved through mapping the internal ontology to the external ontology (created specifically for this task). The conceptual graph is translated into Description logics and then each instance is scanned to find the most pertinent instantiator concepts in the external ontology. These are associated with a MEDIA feature.

They achieved an evaluation score close to that of statistical systems which is encouraging. However as they say, the entire system is based on establishing dependencies on chunks of annotation rather than on establishing dependencies between the chunks. They suggest that fine annotation rules would enable the system to perform better. The fact that so much manual labour is necessary for this system to work means that it would likely only be useful in a very small closed domain. Additionally the annotations required means that an expert much be available to produce them. They also noted that the system was slow.

4.2.1.4 The Delphi system

Bates et al. [16] designed Delphi, the natural language component to the BBN Spoken Language System. It is a domain independent natural language question-answering system made up of a number of domain-independent, general knowledgebases. It is able to deal with incomplete or incorrect user input using its Semantic Linker, which is driven off the same system as its regular first-best parser. The parser and Semantic Linker produce a semantic graph which is interpreted into logical form. This is in turn passed to the discourse stage, which has the task of resolving pronominal references and enforces other constraints to produce a final logical form. This is passed on to the back-end translator which allows for a response to be created. The syntax is used primarily to constrain the semantic graph search rather than to generate parse trees. The semantic graphs are produced using a relation-to-relation compositionality (grammatical relations in the syntax in relation to the semantic relations in the semantic graph). They used Definite Clause Grammar [152] and extended it to include right-hand side elements with the grammatical relations they bear to the head of the construction.

They explain that by annotating sub-constituents with grammatical relations, they can regularise the syntactic structure with respect to particular grammatical rules. The Semantic Linker adds links between the nodes in the different semantic sub-graphs. It ignored fragment order (unlike the parser) and ignores ill formed input. It uses exactly the same realisation rules and determines the for each set of nodes in different semantic graphs the set of all possible links that can connect them. Using this method, incomplete user input can still be analysed and a response can still be generated. Local ambiguity, ellipsis, antecedent resolution and discourse constraint propagation are all taken care of by the discourse manager. The system has the advantage of being easily ported to any domain and can also deal with inconsistencies in language. It performed well on the ATIS task, and they successfully demonstrated that it was possible to build a robust system with minimal human input.

4.2.1.5 Traum's work with Virtual humans

David Traum and his research colleagues have led in this field though their discoveries and experiments with virtual humans. They see these as very useful in training or tutoring people in a number of different scenarios or situations that they may have to deal with.

Their virtual human, Hassan [189], is designed to engage in Tactical Questioning dialogues. These types of dialogue are used in the military for obtaining military information from civilians. Trainees interact with Hassan to obtain information and convince him to help them. Hassan is composed of an embodied virtual character on a screen, a GUI showing the emotive state of the character, and a speech interface. The language components used in Hassan are the same ones as were used in the group's earlier prototype virtual human, Sgt Blackwell [188]. A speech recogniser is used to interpret user utterances, a set of statistical classifiers is used to determine dialogue features and appropriate responses to the user, and a dialogue manager to keep track of the conversation and maintain the correct cognitive and emotional model for the responses given by the system. Hassan differs from the sgt Blackwell prototype in that it has a more sophisticated information-state based dialogue manager, which tracks several states at the same time, in order to decide on the behaviour of the virtual human. They also used statistical classifiers to choose which dialogue features were the most important and which answer was the best one to select, given a particular behaviour.

The dialogue manager tracks dialogue moves, the main topic and the level of politeness expressed by the user. The utterance features identified by the classifiers are used to update the current behaviour of the character. Hassan will either comply or not, based on the response given by the classifier.

In the virtual human Amir [8], the research group used the domain to define the dialogue acts were the result of the interpretation of language utterances. The dialogue acts ”are automatically generated from the domain specification, by applying an illocutionary force (or dialogue act type) to a semantic content containing the relevant portion of the domain specication. Each fact generates 3 dialogue acts an assertion of the fact by the character, a yes-no question by the user, and a wh-question by the user which is formed by abstracting over the value” [8]. The natural language understanding and generation modules link dialogue acts and utterances together. The NLU makes use of statistical language modelling text classification trained on pairings of user utterances and dialogue acts, in order to pick the correct dialogue act for a new utterance. If a match cannot be found, an ”unknown” dialogue act is used. This flags the utterance as ”not understood” to the dialogue manager. A very similar classifier used to produce generation, is trained on mappings of dialogue acts to text. The dialogue manager uses rules to keep track of the conversation and behaviour history and of obligations. It updates the information state of the dialogue and uses it to form a response.

4.2.1.6 The AT&T System

Gupta et al. [80] designed a speech system for AT&T which was based on extracting both intents and the named entities from the users utterances to disambiguate natural language input. They made use of statistical classifiers trained from labelled data to determine intent and rule-based fixed grammars for named entity extraction. They made us of the predicate-argument representation of semantic content available in the user input.

The classifiers were extended to include classification rules learned by the machine from the data in addition to hand crafted rules. This method made for a scalable system which requires little human input to function effectively. The system was designed to work on an open domain. The named entities are extracted from normalised user utterances, and then semantic labels are applied based by the classifier identifying user intent. Due to the architecture of the system, classification models for each semantic label or named entity were not available so initial system performance is not optimal. Data collection and labelling, as the authors recognise, is a costly enterprise, despite the fact that the output of this exercise can be reused. Human labelling is possibly more accurate but still laborious. They used "an action oriented semantic representation language that mapped one semantic label to each prompt and/or action that could be taken by the system".

This way input-output associations could be easily created, but the problem remains that new instances mean that the semantic representation must be changed. The learning framework that they used does allow for reduced human labelling but the initial cost is great. They state that "semantic labels together with the named entities constitute the complete semantic description of the utterance". This is a simplistic notion because other factors such as word sense disambiguation, morphology, polysemy, discourse analysis and contextual awareness play an important role in utterance disambiguation. Reliance on named entity extraction alone is not a holistic approach to the problem. Additionally, this type of system cannot be deployed effectively on an open-domain as the amount of human input necessary to initiate the system would be tremendous.

4.2.1.7 Aleven's geometry tutor

Aleven et al. [4] used a natural language understanding component in their geometry tutor. The tutor is designed to help students learn geometry. It gives feedback based on their answers. The NLU module classifies students explanations into a set of categories of correct and partially correct explanations. It does this by creating a semantic representation of the utterance and then finding the matching category. They used a knowledge-based approach to recognise sentences and a Nave Bayes text classifier, which is used as a fail safe to the knowledge-base approach. If the system is unable to find a match, it will use the classifier to determine if the answer is in the ballpark. The geometry knowledge base is queried by the NLU module to determine the category. It needs to understand user input in order to correctly classify the answers. They found that their system was very limited because it was unable to properly process student utterances when they were incomplete or not detailed enough for the system. They mention, for example, that if the exact named entity isn't expressed, the system cannot disambiguate the utterance. This method appears to be very limited in scope as it cannot work on an open domain and can only function on extremely precise user input.

4.2.2 Syntax analysis

Syntax consists of the rules and principles necessary for constructing sentences. These rules and principles allow grammars to be devised. These in turn allow for computer systems to identify the function of a word, sentence, dialogue or text. Several different types of grammar exist, namely generative grammars, categorical grammars, dependency grammars, stochastic grammars, the Chomsky hierarchy, and finally functional grammars. Here we briefly introduce each grammar formalism and discuss research in NLU using these methods. A more thorough investigation of some of these grammars is available in the next chapter.

4.2.2.1 Grammar formalisms: background

Generative grammar [33] seeks to model the human mind's approach to language and develop a general theory where the rules and laws identified in the language can be re-used. Generative grammar takes into consideration phonology, morphology, syntax and semantics. Generative grammar takes the view that humans have an innate "language faculty" (i-language) and that universal principles exist that govern this language faculty. The focus is on the form of the sentence itself rather than on what it is trying to communicate. The constituent structure of the sentence gives the syntactic information required. This grammatical system was pioneered by Noam Chomsky and he went through several iterations of generative grammar theories:

- Transformational grammar,
- Government and binding theory,
- The minimalist program.

Categorial grammar uses the principle of compositionality and was put forward by Ajdukiewicz [3] and Bar-Hillel [12]. Grammatical expressions are classified according to their category. These grammatical categories are composed of primitive and derivative categories with the noun phrase and the sentence at the root.

Dependency grammar has a structure which relies on the relationships between a word and its dependants. The syntactic structure is based on whether a noun is the subject or agent of a particular verb. Tesnire [95] held that the syntactic structure consisted of lexical elements which were linked by dependencies.

Stochastic grammars are based on probability theory and implementations often involve machine learning algorithms such as neural networks, for example. Optimality theory and stochastic context-free grammar are two different approaches using this theory. In a stochastic context-free grammar, each production is augmented with a probability and the probability of a derivation is then seen as the product of the probabilities of the productions used in that particular derivation. Optimality theory was put forward by Prince and Smolensky [157] and it holds that interaction between conflicting constraints creates the language model. A list of candidates is produced, criteria and constraints are applied and finally the optimal candidate is selected based on the constraints.

Functionalist grammars are based on the communicative function of a sentence. Within this category we find:

- Functional Discourse grammar,
- Systematic functional grammar,
- Cognitive grammar,
- Construction grammar,
- Role and reference grammar,
- Emergent grammar.

Functional grammar was created by Simon C. Dik [57] and involves applying grammatical functions (semantic, syntactic and pragmatic) to part-of-speech tags which represent the constituents. The semantic function incorporates the roles of agent, patient etc... The syntactic function incorporates the subject and object. The pragmatic function incorporates the theme, tail, topic and focus of the constituents. Functional discourse grammar is the newest incarnation of functional grammar and seeks to demonstrate how language is based on the goals and knowledge of the user.

Systematic functional grammar was developed by Michael Halliday [81]. Language is viewed as a network where different interrelated nodes allow meaning to emerge. Users are said to make choices in their language usage which allow for different meanings to be made. It is made up of semantics, phonology, and lexico-grammar (word level analysis). This formalism assumes language is a semiotic system embodying all human experience and relations. It also assumes language is a semogenic system (creates meaning) where lexis and grammar are inseparable. The clause is the basic atomic unit (groups and phrases) and it is considered interpersonal because a particular clause may provoke a particular response from the hearer/reader.

Cognitive grammar was developed by Ronald Langacker [110] and considers the atomic unit to be symbols, the result of pairings of a semantic and a phonological structure. Meaning is equated with conceptualisation and semantic structures are relative to cognitive domains. All grammatical units are reduced to symbolic units.

Construction grammar was developed by Adel Goldberg [73] and here the primary unit of grammar is the grammatical construction. Taxonomies of families of constructions make up the grammar of a language. Each grammatical construction is a pairing of form and meaning. This means that two poles are present, the syntactic pole and the semantic pole. Together they form the construction. Construction grammar also incorporates phonology, prosody and intonation as well as the semantic and pragmatic meaning of an utterance.

Role and reference grammar was developed by William Foley and Robert Van Valin [192]. Sentences are broken down into a logical semantic structure and their communication functions as well as incorporating the grammatical procedures that characterise them.

Emergent grammar was developed by Paul Hopper [91]. It proposes that the syntactic structure and the grammatical rules of a language are defined while the language is being used. The idea is that grammar exists in the mind and is innate.

Not all of these formalisms are used in NLU and NLG at this time, mostly because they are not all easily computed. The following section highlights the most important research undertaken in this area.

4.2.2.2 NLU systems using syntactic theory

Here we present a number of experimental systems that use syntactic theory in order to achieve natural language understanding.

Bos's 3 experimental systems. Johan Bos [21] describes three early systems designed to take verbal commands from humans and react in a specific way. His doris system uses Discourse Representation Theory (DRT), which is a first-order language. It parses English sentences and computes an underspecified discourse representation of the user input. Following this, all ambiguities in the input are resolved. The main linguistic elements covered in his system are pronouns, quantifier and negation scope, and presuppositions. Certain words and phrases trigger these presuppositions. As Bos explains "p presupposes q if both p and p entail q", so for example if Jody likes Mia's husband, it is assumed that we know that Mia is a married woman.

It is not always so straightforward though and the linguistic context can neutralise the presupposition, making it difficult for the machine to determine especially in sentences that contain implication, negation, or disjunction. Bos used the "bliksem" theorem prover to impose logical inference but it took a long time for it to compute all of the possibilities. MathWeb, a web-based inference service was subsequently favoured because it was faster and more inference provers were directly added to it. The issue with this method is once again scalability. Theorem provers do well on a number of sentences but not on chunks of text. Linguistic coverage is also an issue here because it is hard to impose restrictions on a linguistic phenomena. As Bos says for example "tense and aspect require a richer sort of hierarchy, cardinal expressions require counting, and plurals require elements of set theory."

Clark's CCG parser. The CCG (Combinatory Categorial Grammar) parser built by Stephen Clark and James Curran [37] allowed for wide coverage on texts whilst still producing very detailed syntactic derivations. CCG uses a combination of syntax and semantic rules, which is very useful in the field of natural language understanding. It is composed of a large number of lexical categories and very few rules. Bos translated the output into DRT representation using a lambda expression. This allowed for newspaper texts to be translated into semantic representations with a coverage of about 95 percent.

Subsequently the method was applied by Bos to recognising textual entailment (RTE). This consists of identifying whether one text (T) entails another (H). He produced a DRT representation of T and H and translated them into first order logic. He then used the vampire theorem prover to discover whether T entailed H. The problem with the method used by Bos, is that to function well the system needed a lot of prior background knowledge. The finite model builders allowed for more than a "yes" or "probably no" result to be generated. Additionally it allowed for an "Almost found a proof" scenario. This meant that they could cope with over-estimation. If the difference between T and H is relatively small the entailment is found, otherwise not. This method was an improvement on the previous one, but the lack of knowledge still crippled the system.

Using Lexicalized Tree Adjoining Grammar (LTAG) parsing. Denis et al. [54] used a Lexicalized Tree Adjoining Grammar (LTAG) parsing for syntax (deep parser) and a description logic representation for semantic analysis. The parser uses a morphological lexicon, a syntactical lexicon (manually revised the Mulitext lexicon) and a tree library. The grammar is hand crafted and focused on re-usability. It is small in size (80 Tag trees) in order to avoid combinatorial explosion. They used the semantic aspect of the MultiModal Interface Language (MMIL) in order to represent the utterances as a conceptual graph of entities, each described by a feature structure.

The semantic lexicon is composed of 150 schemes “defined as couples of a grammatical anchor (a lemma plus a tree) and a semantic typed fragment. The fragments of the conceptual graph could be an entity, an attribute or a relation”. These are all linked to an internal OWL ontology of 220 concepts related to their closed domain. A conceptual graph is created (based on derivation trees indicating the dependencies between semantic fragments). The ontology corrects errors in the graph. The conceptual graph is transformed into semantic annotation by the projection Module transforms using description logics. Overall they found that the system was slow and that the cost of human intervention to produce the annotated resources necessary for the system were a hindrance. The symbolic features that the system is based on however does allows for easier debugging than statistical systems.

A knowledge engineering approach. Shapiro et al. [176] took a knowledge engineering approach to NLU and designed a system that uses a semantic network system which stores all knowledge including rules. This inference system reasons according to the rules stored within the network. A tracing package allows a user to follow the system’s reasoning. The system has the task of allowing a system administrator for a natural language system to add knowledge to the network using natural language. A collection of core language rules and assertions is stored, as well as a language lexicon. A network structure is created from the user input according to the grammar rules and the knowledge available.

The surface string representation is a network version of Pereira and Warren’s list structure so all explicit connecting tags and markers of their alternate representation are eliminated. Several structures can be constructed as alternate analyses of a single substring. The system operates at word level, so each one in the input string is read and the relevant rules are triggered. If the rules are not satisfied then the process is suspended and if they are then the process continues. The system was tested on a closed domain and on a limited number of queries. The challenge with this model is the fact that rules need to be hand-crafted and that as we know, this is a laborious task. The system also works at word level meaning that the level of complexity involved in writing the rules is staggering.

4.2.3 Trainable systems

NLU systems can be trained to change their behaviour when required, add new information to a knoweldgebase or make a decision as to which template to use, for example. In this section, we present NLU systems that have been developed as trainable systems.

4.2.3.1 Allen’s TRIPS: The Rochester Interactive Planning System

The TRIPS system [65] is a collaborative conversational planning assistant. It communicates with the user through spoken dialogue and graphical displays in the top layer, to solve transportation and logistics problems.The TRIPS system includes the dialogue aspect of the system, which is used to provide context for the actions of the user.

The middle layer of the TRIPS system contains a conversational agent responsible for maintaining the conversation with the user and to help them achieve their objectives. The discourse context and the communicative acts are combined in order to give the system an idea of the intended speech act and the context. The conversational agent helps to resolve ambiguity in the communication by using recognition techniques, and invokes reasoners to provide the solution for the user.

The bottom layer of the TRIPS system includes a temporal logic-based planner, a router, a scheduler, a temporal knowledgebase, a fast simulator with data mining capability to detect and correct problems with the planned activities. The problem-solving manager has the task of invoking the reasoners and integrate the correct information in the responses.

The conversational agent uses the results from the bottom layer and combines them with general dialogue principles to create a response for the user. This response can be given using dialogue or the graphical interface.

The TRIPS system functioned well allowing users to get help when tackling complex transportation and logistic problems. One example involves calculating the cost of evacuating an island before a hurricane arrives. Although this system was able to achieve good results, it is different to the goal that we are trying to achieve in some ways. We rely solely on types conversation between human and machine, and do not have a graphical interface to express other information. We also have very open ended communication with the user as the tasks out SOCS is required to attend to are not always as specific as those in TRIPS. This means that the SOCS must be able to rely on its communication skills to gain a good understanding of the context and the problem.

4.2.3.2 Allen's PLOW: A Collaborative Task Learning Agent

Allen et al. [5] created a system that "that learns executable task models from a single collaborative learning session consisting of demonstration, explanation and dialogue." They used a range of AI techniques including deep natural language understanding, knowledge representation and reasoning, dialogue systems, planning/agent-based systems and machine learning.

PLOW was designed to help people accomplish everyday tasks. It uses a collaborative architecture to learn and carry out each task. The user gives a "play-by-play" description of the task to be performed and PLOW learns from this description. The authors state that "By combining language understanding and learning from examples, PLOW can identify intended procedures from just a single demonstration". Example tasks in this work include finding articles related to a particular project or topic, finding businesses that are located near another specific location, finding the most expensive purchase approved between particular dates, and many more tasks. All of these tasks can be performed within a web browser, from which the PLOW interface is accessible. It can be activated through speech or keyboard.

The "understanding components" use both the instruction given in natural language through speech or keyboard, combined with the observed play-by-play learning. Full parsing, semantic interpretation and discourse interpretation are carried out on the user input and sent through to the "collaborative problem solving agent" (CPS), which chooses the most likely interpretation. Depending on the actions required, different parts of the system are activated.

Allen et al. explain [5]:

"For example, if the recognized user action is to demonstrate the next step in the task, the CPS agent invokes task learning, which if successful will update the task models in the knowledgebase. If, on the other hand, the recognized user intent is to request the execution of a (sub)task, the CPS agent attempts to look up a task that can accomplish this action in the knowledge base. It then invokes the execution system to perform the task. During collaborative learning, the system may actually do both it may learn a new step in the task being learned, but because it already knows how to do the subtask, it also performs that subtask for the user. This type of collaborative execution while learning is critical in enabling the learning of iterative steps without requiring the user to tediously demonstrate each loop through the iteration".

Language processing is carried out by the TRIPS system, described in the above section. This is combined with context gathered from user interactio with the browser window. It uses the DOM representation that the browser uses to generate the GUI and they record all of the user actions that take place in response. It begins from the clicked node in the DOM script and them looks for a semantic concept produced by the parser. A lexicon and ontology are used to define the meaning of the concept by measuring semantic similarity. PLOW learns what the matching action to the query is by observing user actions. In an evaluation that the Allen et al. carried out, the system was trained to find the "Books" tab on a website. It then successfully found other tabs 95% of the time on Barnes & Noble's website and 98% of the time on the Amazon website.

In order for the system to be aware that it is in a learning situation, the user is required to specifically call this out by saying "now let me show you..." for example, and ending the training with "We're done" or some other linguistic marker. The user may then correct the system when it attempts this task in future, should it get it wrong.

They evaluated the PLOW system by asking 16 independently contracted users to carry out 17 specific tasks with the system. Once the system has learned them, they created more examples of these tasks and tested if the system was able to carry them out. It scored 2.82 out of 4 across all tasks and the 16 test subjects. The 16 individuals were then asked to test PLOW on a set of 10 surprise test questions, which had not been previously seen by the developers. Each of the 16 individuals had a day to teach it to PLOW. The system performed well, seeing as it was successfully taught 30 out of 55 possible tasks in the time limit, and achieved an average score of 2.2 out of 4.

4.2.3.3 A Knowledge-base NLU tutor

Aleven et al. [4] designed a knowledge-based natural language understanding tutor. It is a dialogue system that allows students to enter explanations in their own words of their answers to geometric problems that they are set. If the explanation is wrong or if they have provided an incomplete answer this virtual tutor gives feedback. The student must then provide a further explanation which is how the dialogue manifests itself.

The system recognises sentences as correct or partially correct using a combination of a knowledge-base approach and a statistical text classifier should the knowledgebase approach fail. The student input is parsed and a semantic representation is created. That representation is then classified as either correct, incorrect or partially correct. A basic ontology of the domain is provided and a knowledge-base containing a hierarchy of explanation categories is used to assess completeness of response. Each hierarchy represents a class of explanations that have the same meaning but are expressed differently as far as surface form is concerned. The least complete answers are at the top of the hierarchy and the most complete at the bottom. It is constructed using a corpus of student answers gathered and contains 250 concepts and 100 relations as well as 70 explanation categories.

They used the LCFLEX active-chart parser to construct a semantic representation of the student's input and the grammar is comprised of 200 rules. They use the Loom term description logic system to classify the representation. Feedback is given based on the result of this classification, because a specific canned response is associated with each category. Should this approach fail to deliver a clear answer, a Naive Bayes text classifier is used to determine the result. There are clear limitations to this approach due to the fact that it is so dependent on the system having previous experience of the utterances provided by the user. This approach would not scale easily or efficiently in an open-domain. Additionally, as the authors concede, the system is unable to deal with situations where the user does not progress towards completeness of answer but rather "regresses" to a more erroneous one. It also cannot deal with "lateral movement" within the hierarchy.

4.2.3.4 Using Hidden Understanding models

Miller et al. [111] used a statistical machine learning method known as "hidden understanding models". They incorporate into this approach semantic grammars, augmented transition networks, probabilistic parsing and automatic grammar induction. The hidden models determine the most likely meaning of a string. They experimented with tree structured meaning representations, where individual concepts appear as nodes in a tree and component concepts appear attached directly below them. The order of the component concepts must match the order of the words they correspond to. Semantic concepts are hierarchically nested and the terminal and non-terminal nodes remain disjoint. The semantic and syntactic categories are labelled in the tree, meaning that they have more internal structure than other models.

They found that such a model could replace classic components of a NLU system such as part-of-speech taggers and parsers. The grammar is not hand-crafted since the system automatically constructs a statistical model from example trees. They use a Frame-based representation where not all of the words are accounted for. The frame is a complete meaning representation because the concepts are fully expressed but terminal nodes in the tree can be omitted. These missing nodes are "hidden" and are aligned as part of the training process at a later stage.

The semantic language model they use is viewed as a collection of networks rather than a collection of rules. Orderings are determined through probabilities rather than through a set of predefined rules. The underlying model is a recursive transition network and the authors elect to use the Viterbi algorithm to build this network, but they integrate state allocation, Viterbi search and pruning all within a single traverse of the tree.

This is very different method to the hand-crafter grammar approach. They found that the system was able to produce a representation very comparable to the one produced by the Delphi system. The system, although domain independent, presents limitations in that it cannot deal with co-reference and other non-local phenomenal. The authors also state that the meaning of a sentence depends heavily on the discourse state. Additionally the trees require annotation effort and they estimate that a human annotator could only produce 200 sentences a day. In a large domain this would mean days of annotating and also continuous annotation as new data is incorporated and discovered.

4.2.3.5 Using alignment templates and maximum entropy models (ME)

Bender et al. [17] experimented with alignment templates and maximum entropy models (ME). Both approaches are useful for analysing the semantics of natural language input. They treat the problem of NLU as a machine translation (MT) problem: the translation of a source sentence into a target sentence. They make use of alignment templates, which is used in statistical MT. Two probability distributions are derived: the language model probability and the translation probability.

They train the alignment model using a sequence of increasingly more complex models. Phrase and word-level alignment are possible so the target sentence is segmented into word groups. Bender et al. use sentence pairs to train the model and estimate probabilities by identifying the frequencies of applying an alignment template. A left-to-right language model is included also. The results of their evaluation show that this method does not perform well and that in addition it too is laborious.

The ME allows for probabilities to be modelled also, but in this case they determine the corresponding target language concept for each word. Each word is identified as being an initial or non-initial word of a concept so that a one-to-one mapping is achieved. Constraints are imposed to ensure that a non-initial concept allows follows its corresponding initial one. A feature function models the transitions. Unknown words (not present in the vocabulary) are flagged as "unknown". The system was trained on manually annotated and segmented sentences to form a long sentence. In order to avoid over-fitting they use a smoothing method (Gaussian Prior). The ME model performed significantly better than the alignment template method.

The ME framework also allows for posterior probability to be modelled and allows for structural information to be integrated. It is also free of convergence problems due to the convex nature of the optimisation criterion. Interestingly it has a higher error rate than the alignment method but still outperforms it on all other criterion. Despite this, the methods discussed by Bender et al. do not take into consideration the context of a conversation. Additionally, they require training, which means that open domains would be problematic for them due to time constraints. The main issue with these methods, however, is that they assume that language users always communicate in the same way and our own experiments have shown that this is not the case.

4.3 Natural language Generation

Natural language generation (NLG) requires producing a response to the user in the form of a relevant piece of text. There has been more research in NLU than NLG because NLU has had more practical applications. This is, however, changing with the advent of web 3.0, where machines are starting to generate content and interact with web data.

Additionally NLG requires NLU to function well in order to produce relevant and fluid responses. NLG can also provide a significant contribution to NLU, because the methods can be applied there too. Additionally NLG has been predominantly researched in the context of spoken systems rather than text based dialogue systems. There are a large number of approaches to NLG ranging from hand crafted approaches to machine learning ones. The ideal solution is one that allows for minimal human input and highly accurate output, however this is regarded as a highly difficult problem in computing.

Owen Rambow [163] describes NLG as something which affects someone else's cognitive state. He provides the following description of NLG:

> "NLG is conceptualised as a process leading from a high-level communicative goal to a sequence of communicative acts which accomplish this communicative goal. A communicative goal is a goal to affect the users cognitive state, e.g., his or her beliefs about the world, desires with respect to the world, or intentions about his or her actions in the world."

There are three main stages to NLG namely information acquisition, content organisation and linguistic realisation (and if relevant, speech synthesis also). Information acquisition (the data needed for a response) is usually done through databases or large corpora, such as the web for example. When there is a closed domain, the information pertinent to that domain is stored in a way that the system can access it and extract from it relevant entries. In the previous chapter we discussed knowledgebases and database options as well as the difficult but popular idea of using the web as a knowledge resource. An introduction to Templating systems was also presented in the previous chapter, but further research on these is presented below.

The sentence generation stage is where sentences are put together for a response to be possible. The sentence generation component will receive as input a specification of what is required at output. It will then find the corresponding natural language expression and communicate it to the user. The sentence generation component is made up of a parser, a grammar and a lexicon. The input is usually in a logical form as a set of facts.

4.3.1 Hand crafted application-specific grammar approach

This method relies on creating a grammar that is specifically made for a particular system. The grammatical rules are hand crafted and purpose made for a particular scenario.

Elhadad and Robin developed SURGE (Systemic Unification Realization Grammar of English) which is described as a "syntactic realization front-end for natural language generation systems". The component is made up of a lexicalized thematic tree which specifies the semantic roles, open-class lexical items and top-level syntactic category of each of the constituents. The thematic structure is first mapped on syntactic roles, then syntactic paraphrasing and altercation control takes place, over-generation is addressed, defaults are given to syntactic features, agreement features are propagated and finally linear precedence constraints are set amongst the syntactic features.

Following this, the open-class words are inflected and the syntactic tree is lineralized with a string of inflected words. This approach has been used in a number of systems successfully, and it does have the advantage of being domain independent. The problem with it is that it remains inflexible because the rules need to be written by hand and this requires a lot of effort. The KPML system (Komet-Penman MultiLingual) [15] was created for multilingual grammar development and generation. The system is used to develop large-scale grammars in the Systemic-Functional Linguistics (SFL) framework. The system has been used to develop further grammatical rules, but this method is still laborious despite the development environments necessary to write them.

4.3.2 Template-based approach

The "text planning stage" refers to the organisation of information. The system, once it has the right information from the previous stage needs to be able to organise it in a way that a human reader can understand it. The methods used for this are canned structures of templates. This method is inflexible, however, it is true that not all systems require such fluidity, so they have been successfully used in the past. Canned responses as we have seen previously, are a collection of ready drafted responses. Formulaic language is the linguistic terminology for these and Wray [202] covered the phenomena in the English language at length. She found that native speakers made regular use of prefabricated language.

Bearing this in mind, it makes sense to employ canned responses in dialogue systems. We have seen, however, in our data, that although there is a percentage of formulaic language in human-human discussions, there is also a large percentage of those conversations that are non-formulaic. This is the real limitation of the canned response method. Structural templates are extremely useful because although people do not use exactly the same sentence to express something (unless it is an idiom), they do use the same structural constraints of the language, semantic considerations, argument structures and so on. These patterns allow us to create templates into which additional information is placed in order to create a valid response. McKeown [136] proposed using a transition network grammar, which meant that the rules of a context-free grammar [201] were used to create templates. The method is equivalent to pushdown automata, because the context-free grammar rules represent the arcs. The grammar generates sentences and the transition network accepts them.

A more flexible method was developed by Mann and Thompson [127] where the interclausal relationships that govern text structure are defined in Rhetorical Structure Theory (RST). RST is a method for defining the structure of discourse. It defines approximately 25 relations which may be present between segments of a text. The definitions carry constraints as to what the text spans must contain. The rhetorical relations between the clauses and blocks of clauses create coherence in English. They are identified by triggers such as “moreover”, “In so far as”, “then” and so on. Hovy and Maier [93] later analysed the number of interclausal discourse structure relations and extracted 300 relations from a range of sources in turn organised into three taxonomies of 120 relations. The templates need to grow exponentially as the system is required to cover further topics of interest. This creates heavy load on the person responsible for creating them. Additionally a large number of templates are difficult to maintain as Axelrod (2000) also described.

4.3.2.1 The FLIGHTS system

Moore et al. [142] described an approach to presenting information in spoken dialogues using multi-attribute decision models, strategic content planning, state-of-the-art dialogue management, and realisation which incorporates prosodic features. The subset of options is selected and the attributes that are most relevant to choosing between them according got user modelling. The system also determines "how to organise and express descriptions of selected options and attributes, including determination of information structure and rhetorical structure at the level of content planning, resulting in descriptions which, we hypothesise, are both memorable and easy for users to understand.".

Their FLIGHTS (Fancy Linguistically Informed Generation of Highly Tailored Speech) system is designed to inform users on available flights and schedules. The domain is closed and very specific. It is also based on factual information which is highly structured. The data is pulled from a database and then the preferences encoded in the user model are used to select the best response to this particular user (frequent flyer, student, business class customer). Negative values in a user response such as "but" and "though" are flagged as well as conditionals such as "if". Positives such as "good" are used to identify relevant information for future use.

A tree-structured model of the attributes in the domain is created. Each attribute is weighted between 0-1 according to user modelling. They use Combinatory Categorial Grammar (CCG) to structure the sentences. The OpenCCG realiser with theme and rheme as variables to determine the types and locations of pitch accents and boundary tones. They use a sentence planner to transform input dialogue acts and rhetorical relations into lexicalized predications. Their example is "you need to connect in Dublin or (it) requires a connection in Dublin". The groupings and specifications of the referring expressions provided by the content planner are refined for improved fluency.

This system, although primarily designed for a speech system, demonstrates a way of achieving a natural language response to a natural language user query. The method however still employed a degree of brute force, because the rules are rigidly imposed and the overall content is extracted from a structured database. The responses, though natural sounding are still based on a form of template method. The closed environment allows it to be successful, although in common with other template methods it is not scalable.

4.3.2.2 The WEATHERREPORTER system

Dale et al. also made use of template-based NLG. They presented a system called WEATHERREPORTER which had the task of reporting the weather per calendar months, so it is fair to say that the domain was small in scope. The system produced short paragraphs as output to the user. They worked on trying to avoid the complexities of other systems and state that their system lacked the depth of others. The meteorological data is automatically gathered and used as input. This data is made up of numerical data annotated by humans. The FOG system is used to generate the textual data from the numerical input. The IDAS system is a dynamic hypertext system. All texts are generated from the text knowledgebase at run-time. The links are requests to run IDAS to generate a response relevant to the current context. One strength of their system is that it is language-independent. It can generate texts in French by simply using adequate lexical and grammatical resources. The authors note that the IDAS system pre-dates the web and therefore is rather limited.

4.3.3 Rule-based approach

Rambow et al. [163] chose to randomly generate a set of solutions which were in turn scored by human judges. They opted for using abstract syntactic constructions rather than relying on word level lexical choices. These lexico-syntactic structures (DSyntS) are stored into larger larger DSyntSs using previously defined clause-combining operations (RELATIVECLAUSE, CONJUNCTION, and MERGE). The resulting structure (sp-tree) is then realised (using RealPro). They use RankBoost to learn a scoring function assigning syntactic and lexical features. This newly constructed sentence planner generates sentence plans and the ranking function weights them. This sentence planner performed statistically very well, statistically comparable to the choices made by the human judges. It did however take three days worth of human input to create, meaning that it could have scalability issues.

Duboue et al. [59] designed a system that created content selection rules automatically from a corpus of text and associated semantics. These constraints are learned by the system using a corpus of correct outputs (human produced) and aligning them against related semantic data. They used data paths (identifiers for each piece of data) within the semantic network. Each data path identifies a class of values. The frame-based representation of knowledge allows them to identify relevant information in the graph. They cluster all the possible values in the graph for each individual. For free-text fields the top-level most important words are chosen using *tf*idf* over the whole collection.

Texts associated with each cluster are used to create language models using bigrams. The n-gram distillation process was used where the most important n-grams were chosen in order to improve precision. The language models are compared using cross entropy. If the values are larger than the control set, then the values for that particular semantic cluster are deemed relevant. The system is limited to a single domain and the data are also carefully structured. Only bibliographic data about people was used. The use of human input causes the method to be laborious. The user can only ask informative, communicative questions and the user intent is not taken into consideration. Their database entries are also only considered in isolation.

Rudnicky et al. [146] use an n-gram approach in a corpus based approach for stochastic language generation. Their method was used in the Carnegie Mellon Communicator, which is a telephone based dialogue system dealing with travel itineraries. The PHOENIX semantic parser transforms the input string into a semantic frame. They use an n-gram model, which is easy to build and maintain which is a distinct advantage over the template approach. The output utterances are modelled by bigrams and the relevant attributes are identified using bigram statistics. Sentence planning and realisation is achieved through modelling un-smoothed n-grams of varying lengths and producing responses based on rules (5-grams were shown to perform the best). AGENDA dialogue manager interacts with other data modules and sends a semantic frame to the NLG module that creates a response.

The NLG module uses a tagged corpus (word classes) based, stochastic generation approach. The n-gram language models were built using a corpora of 35 human-human dialogues. The word classes are the attributes from which the values are passed as a response from the dialogue manager. This is not dissimilar to slotting words in templates.

They used language model probabilities to generate the next word in the response. The generation engine selects the correct language model with respect to the utterance class and generates word sequences at random according to the language model distributions. The utterances are scored and ranked and the best ones are selected. Finally, the slots in the templates are populated with the correct values from the input frame.

The resulting evaluation showed that users found it no better than the template approach. The authors mention that "It is not yet clear whether the simple n-grams can adequately model human language generation in general". Our work on the JOHN system showed that this approach was very limited and did not offer the fluency of conversation required from a dialogue system and expected by the user (as discovered in the KIA experiments).

4.3.4 Trainable generation approach

This approach uses automated learning techniques in order to infer rules from a corpus. In this section we present four different approaches to implementing trainable generation.

4.3.4.1 Automatically learning content selection rules from a corpus and its related database

Barzilay et al. [13] proposed a solution for "automatically learning content selection rules from a corpus and its related database". Content selection is handled as a collective classification problem, which allows for the dependencies between user inputs to be identified. They also worked in a closed domain, focusing on sport. They concentrated on the content selection component which allows for relevant information to be included in the response. The data was stored in a database and the method identified the subset of information in that database that was pertinent to this particular query (or closed domain).

Their machine learning algorithm relied on a set of documents that were cross referenced to the data labelled in the database. Their method is described as a "collective content selection" method which requires all data to be stored in the same database. All candidates are considered simultaneously (rather the standard method which considers them in isolation) and this allows for semantically related entries to be grouped together. The A graph based formulation used allows for accurate information to be identified. They "avoid the need for extensive feature engineering by incorporating discourse constraints into the learning framework". They mention that their solution works best on relational databases, which we have found to be very limited. Their data are also very structured containing descriptive statistics about American football games. This means that unstructured and distributed data that is commonly found on the web could not be easily processed using this method. They claim that their method is linguistically grounded but there is not incorporation of syntax and semantics which means that their system cannot be contextually aware.

4.3.4.2 Grammar-based generation

DeVault et al. [56] used a grammar-based generation which relies on examples. Although their system provides better coverage and output variety, it does come at a substantial cost. They designed a conversational system built to teach army personnel negotiation skills. They use an attribute-value matrix (AVM) semantic representation which is "designed to describe an utterance as a set of core speech acts and other dialogue acts". The structure of the AVM is represented by a frame. Their system does present some limitations as far as the number of frames available, because there can be an unpredictable number of them. They didn't require the system to output fully grammatically correct responses because they designed it to mimic a non-native speaker. This does hinder evaluation somewhat. They used the Charniack parser to identify syntactic elements to a number of suggested responses. They also used statistical language models in order to produce a a probabilistic lexicalized tree-adjoining grammar (PTAG) but only with sister adjunction and the tree templates can be associated with more than one lexical anchor. They also unified syntactic and semantic variables in search by augmenting the lexical anchors with semantic information.

They use training examples to allow the system to identify the correct responses and used search optimisation and ranking algorithms to weight the possible outputs. Although the grammar induction is generated through training examples, the system did require 40 rules to be hand crafted. This is a painstaking process and therefore creates high cost and is regarded as a limitation. They do note however that many of these rules could be applied to other applications. The main limitations include generating semantic links and referring expressions cannot be generated. This makes for low contextual awareness.

4.3.4.3 Using a Markov Decision Process (MDP)

Lemon [113] approached the problem of NLG as a Markov Decision Process (MDP). The system is based on a transition probability from one state to the next after having performed a particular action. In NLG this refers to dialogue completion and a move from a particular topic to another for example. In this system the Reinforcement Learning agent finds the best strategy to achieve the best result. If the probabilities are known, the system must compute them dynamically otherwise a trial-and-error process begins (Reinforcement Learning.).

The system has four types of dialogue acts (AskASlot, ImplicitConfirm and AskASlot, ExplicitConfirm, and PresentInfo) which can be chosen at any time. The state space involves constraint slots that the user can fill or confirm (restaurant name, location, price...). The confirmed slots are 100 percent correct whereas filled ones are likely to be 80 percent correct. The system deals with non-recognised items or unfilled slots by demanding more information/ and constraints. The system can, however, answer regardless, if it chooses to because it has enough database items in that particular case.

The system is trained based on: completion reward, turn penalty, and presentation reward/penalty. The system learns to avoid bad decisions and can choose the best NLG option available depending on the number of database items available for that particular input. This method relies on human input, templates of some description and a database (also requiring human input). Additionally, it uses machine learning in order to make a choice as to which response to give. There are reduced build costs due to automated adaptation to new domains but the amount of learning involved for an open domain would be substantial and in this respect the limitations are similar to the Jabberwacky chatbot [29].

4.3.4.4 Using a wide coverage realizer

Stent et al. [181] created a trainable system based on a wide coverage realizer which allows the system to use a general-purpose linguistic knowledge. This approach has the advantage of being applicable to all domains. This trainable sentence planner was used on the MATCH system (Multimodal Access To City Help) [179]. They extended the SPoT (Sentence Planner Trainable) [142] which had the limitation of only working in the travel domain in the information gathering phase. They extended SPoT to the restaurant information domain and they used this method for response generation providing recommendations and comparisons.

The system worked just as well as the template-based generator previously used in MATCH. They also extended it by including rhetorical relations with their SPaRKy (Sentence Planning with Rhetorical Knowledge) system. It functions using a randomised sentence plan generator(SPG) and a trainable sentence plan ranker (SPR). It used the bottom-up approach for response generation where each presentation contains assertions and a specification of the rhetorical connections between them. Linguistic resources are selected and then ranked according to relevance. The sentence planner creates sentence plans and these are ranked by the sentence plan ranker (SPR). This way text plan trees (tp-trees) are created to allow for sentence generation. The top ranked ones (using RankBoost) are processed through the RealPro surface realizer [112] which produces a surface linguistic utterance. The four templates used in the system are ones displaying traversal features, sister features,ancestor features and leaf features. The global function produces a record for each sp-tree and for each clause-combining operation labelling a non-frontier node.

One limitation with this approach is that the information available to the system is not linked in any other way than rhetorically. Whilst this is an improvement, it does not offer the same level of cross reference between knowledge nodes. Another interesting difference to other systems is that they talk of “information” rather than “knowledge”. “Knowledge” is only present when there is more than just information available to the system. It involves a further level of complexity where relations are built in. Also the fact that the performance of the system is comparable in performance to the template system is not promising as we have established that this method has severe limitations.

4.3.5 Corpus-based approach

Ratnaparkhi [164] presents 3 systems for surface natural language generation trained on annotated text. NLG1 and NLG2, require a corpus marked up with domain specific semantic attributes. NLG3, requires a corpus marked up with both semantic attributes and syntactic dependency information. NLG1 uses frequencies to generate whole sentences whereas NLG2 and NLG3 use maximum entropy probability models to determine word choice and word order. Again the domain for the system is closed, focusing on flight information. NLG1 bypasses the need for a knowledge base by finding links between words and semantic attributes. A template is first produced from the attributes alone.

NLG1 selects the most frequent templates in the training data corresponding to a set of attributes. An output is generated as long as the input is specific and already included in the available templates.

NLG2 relies on the highest probability word sequence that contains all the input attributes once. It uses n-gram information to achieve this. The non-local information represents the attributes to be added to the response which is generated based on the best scoring word sequence according to a probabilistic maximum entropy model. They found that the previous words in the n-gram model did not necessarily predict accurately what the next word should be.

NLG3 assumes that "syntactically related words in the history will result on more accurate surface generation". A syntactic dependency tree (top-to-bottom) is created by the search process rather than relying on a word to word approach (left-to-right). The corpus for NLG3 does need to be annotated accurately in the tree-dependency structure. It uses syntactic information and only features which occur K times or more in the training set. The search is terminated after a given number of iterations when a sufficient number of trees have been discovered.

They found that their system lacked in accuracy compared to grammar-based approaches because they can be fine tuned. The approach described here means that feature patterns, search parameters,and training data can be manually improved but it is impossible to guarantee any improvement. Additionally, although the statistical approach reduces human input the trade-off in precision is costly. Using a word level approach means that the overall contextual meaning of the utterance can be made too noisy for the system.

4.3.6 Summary

Natural language understanding and generation systems that have been developed to date, have not been able to function adequately on a closed domain and the techniques used are not easy to apply to an open domain. They also do not perform well on unstructured data and it remains unclear as to what the best way to process knowledge and linguistic understanding is. User input is sometimes incomplete, unclear or incorrect and a machine needs to be able to accommodate this. Additionally, language changes with time and an efficient system needs to adapt.

The main limitations encountered by existing systems are scalability, context awareness, incorporation of world-knowledge, speed, language ambiguity and cost. All of these issues can be grouped under two distinct headings: language ambiguity and world-knowledge. The problem of NLU and NLG is a complex one because in order to train a system which would minimise human involvement, we still need a training corpus and to overcome the issues of language ambiguity and we need to encode natural language as a whole in a computer friendly way .

In order to transcend these limitations we need to acquire a world-knowledge corpus for the system and also learn from experience the variability of human input. In our own system we seek to minimise the issue of language complexity and human labour however we still work within a closed domain at this time. It is hoped that future developments in corpus sharing namely via structured databases being made available (such as gov.net for example) we can overcome the difficulties involved in working in an open domain.

In the following chapter (chapter 5) we look into construction grammar in greater depth. Our reasons for choosing this particular grammar for our project is explained, and a full description of construction grammar is presented.

Chapter 5

Construction Grammar

Construction Grammar

5.1 Introduction

In this chapter we present Construction Grammar (CxG) and its cognitive linguistics background. We elaborate on particular aspects of this type of grammar and compare it to other types of grammar, especially Generative grammar. We go on to describe a specific type of CxG called Fluid Construction Grammar (FCG), which we experimented with in our system. This particular strain of CxG proved to not be mature enough as yet for use in our system, so we aimed instead to work with Goldberg's CxG model [74].

5.2 Cognitive linguistics

Construction Grammar belongs to the area of cognitive linguistics. This school of linguistics believes that

- language is not an autonomous faculty
- grammar is not conceptualization
- knowledge of language emerges from language use

Language is deemed to be no different from our other abilities such as walking or recognizing colours. The knowledge of meaning and form (semantics and syntactics) is considered to be a conceptual structure. The cognitive processes that govern language are believed to also govern other cognitive processes such as memory, association, concept association, perception, problem solving and more. Language is described as the "real-time perception and production of a temporal sequence of discrete structured symbolic units" [43].

Grammatical inflections and constructions are the means by which we construct our experience of the world and how we communicate it. Semantics, syntactics, morphology and phonology are all constructed by our cognition of specific utterances and their specific usage. In cognitive linguistics syntactic behaviour and semantic interpretation are observed. These create different grammatical representations that give very specific as well as general patterns of linguistic behaviour.

Non-cognitive linguistic approaches in contrast have a highly general and abstract view of semantics and syntactics. Schemas and categories are created and these patterns are said to govern the means by which we organise knowledge. Any patterns which are deemed anomalous are rejected. Instead, cognitive linguistics views meaning in terms of conceptualization, using the idea of mental spaces and frames. In contrast, Chomskian linguistics, focusing on syntax alone, holds that meaning is interpretive and peripheral to the study of language.

5.2.1 Frame semantics

A ”Frame” is any system of concepts related in such a way that to understand any one concept results in all of them becoming available. So that we are able to understand the meanings of words in a language, we must have first knowledge about the conceptual structures, or the semantic frames. A “Frame” is described independently of the words that are involved. The words are described according to frames that they relate to and the ways in which the frame elements need to be realized in sentences built up around words. The goal, in frame semantics, is to describe the semantic and syntactic combinatorial properties of words. Frame semantics are very important to CxG, because they enable us to determine what types of constructions can unify with what type of verbs. The semantic information about the frame is important in assessing whether the basic semantics of a verb are compatible with the semantics of a construction.

Fillmore introduced frame semantics [191], which allows “truth-conditions” to be replaced by the “semantics of understanding”. The mind cannot understand the meaning of a single word without having the body of knowledge that exists around that word. A word evokes a frame of semantic knowledge which relates to the concept it relates to. A frame is essentially a coherent structure of related concepts. Fillmore et al. developed FrameNet [10] which is a collection of frames available for machine use.

An example the frame “Origin” in FrameNet is presented in figure 5.1. To make the reading of the frame easier we have also included the FrameNet system legend in 5.2.

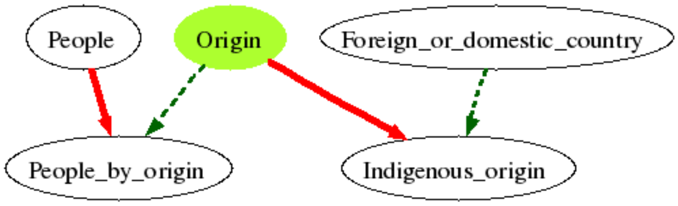


Figure 5.1: The “origin” frame in FrameNet

In Fillmore’s Frame model, the speaker produces words and constructions that evoke understanding. The hearer disambiguates the meaning and as a result understands the speaker. In short the hearer invokes a frame in order to understand it:

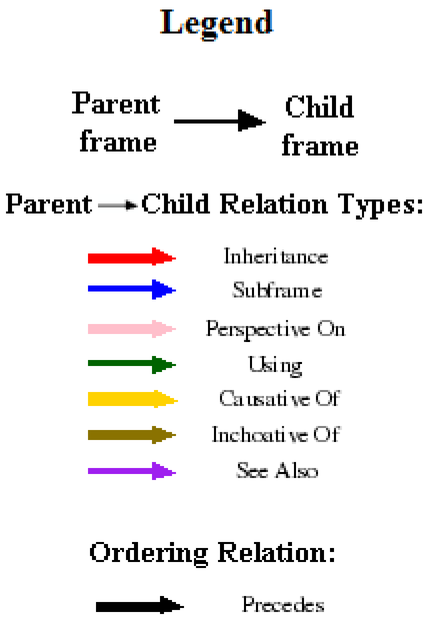


Figure 5.2: Legend for FrameNet diagrams

- word + constructions = Understanding = A frame
- The semantic features are often based on lexical sets, for example: Woman [adult,female] and Girl [minor,female]

Frame semantics takes into consideration the attitudes of society towards the sexes which in turn changes the relationship. It is possible that:

- Woman = Girl

The example of "Alive" vs "Live" in Croft[43], is useful for understanding Frame semantics:

- These are live lobsters / The lobsters are alive
- His performance was live / He gave a live performance
- His manner is alive / He has a very alive manner

"Alive" and Live" are associated in different ways to 3 different frames:

- Life in
- Personality in
- Mode of performance

Frames are based on re-occurring experiences. Words highlight different individual concepts, but also specify a certain perspective in which the frame is viewed. For example "Surfing" is different when viewed by a spectator than it is when viewed by the surfer. In the frame "Package" we see that:

`Package[large, jumbo,giant, family size].`

Here “large” is the smallest unit in the “Package” frame. When a hearer evokes the Frame concept, s/he is clear that “large” is indeed small because the word concept is linked directly to the frame. There is no need for additional words to be present to allow this frame to be understood, whereas other methods such as lexical field theory uses other words to contrast. It is only possible to understand the word “thirsty” if one understands the physiological concepts, for example.

We can say that words that have corresponding concepts that inherently refer to other concepts outside of the concept denoted by the initial word can be understood in the world of Frame semantics. This is also true of tenses (time of occurrence), person deixis, spatial deixis, and definite articles. Even if all of the elements in a frame do not occur, the entire frame is evoked regardless of its usage. For example consider the frame “Robbery” in figure 5.3:

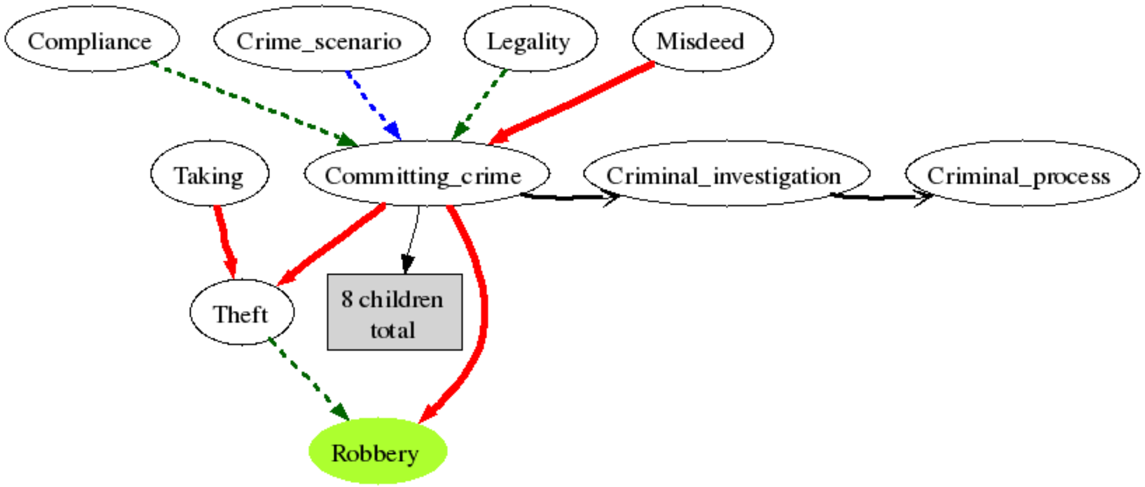


Figure 5.3: Robbery Frame

In this frame the ”robber” (or actor of the deed), ”robbed” (or victim), ”situation”, ”deed”, ”harm” and so on are all evoked even if only the deed is referred to for example. Any of the uses of ”Robbery” will invoke the entire frame, even if the constructions contain the noun but not all the elements. Sometimes more than one concept will be represented in a frame, for example ”The radius of a circle”. In this example ”Radius” and ”Circle” can only be understood when placed together so they are placed in the same conceptual structure.

Fillmore argued that ”a Frame is any system of concepts related in such a way that to understand any one of them you have to understand the whole structure” [10]. He developed the ideas behind Cognitive Grammar, where the basic atomic unit of language is a pairing of form and meaning including a phonological label. The basic elements of language are therefore more symbolic than in traditional word-based grammars.

In Frame semantics, the Frame is a semantic structure that is the root (or base) of at least one concept profile.

A concept profile is a concept symbolized by a word. Different domains such as space, material, time, force, colour, pain and so on that are directly rooted in human embodied experience. There are more abstract domains where the relationship between the abstract and basic domain is based on assumption or presumption. These abstract domains are plentiful, especially when we consider the number of specialist areas that exist such as zoology, swimming or fishing for example. Their vocabularies are very specific. We find this also in our system where we used “Car insurance” and had to create our own Frames.

The Frame concept can become rather complex as Croft [72] demonstrates. He gives the example of ”bachelor”, which comes under the frame [unmarried man], but the Pope and Tarzan also come under this domain, although they evoke very different frames. To be precise, frames also need to incorporate the “idealized cognitive model” (ICM) introduced by Lakoff [108], which involves our view of the world insofar as living arrangements, relationships to different people, occupation and so on are concerned.

Searle [173] is of the opinion that this is too complex and that the background assumptions for defining the Frames is where the interest lies. He agrees however, that an encyclopaedic knowledge of the world is required to understand all Frames and thus all utterances. Others state that a dictionary view of the world is enough, but having tested this using our chatbot John, we know that it is clearly not sufficient.

In our chatbot experiments with ”John”, we observed that a limited knowledge of the world was insufficient and that the system was unable to carry out a meaningful conversation with a human. In order to be able to deliver encyclopaedic knowledge of the world to a system in an open-domain, we require machine readable relationships which relate to Frames to be constructed. The importance of the semantic web is obvious when we consider that it strives to create linked data annotated in a meaningful way.

Gilles Fauconnier [58] introduced the idea of mental spaces. These do not contain a clear representation of reality, but rather an ICM. The elements contained are those belonging to thought processes and opinion rather than plain fact. The focus is heavily on the structures and networks in language and the correspondence between them rather than language forms. An example would be ”Peque believes Maria is beautiful”, which does not state a clear fact but rather what is true in Peque’s mind. Mental spaces are concerned with predictions and hypothetical things. A mental space still has roles (mapping between spaces) and values (an individual that can be described as being part of that category).

For example:

- ”Esme wants a trip to Honolulu”
- Esme wants = Desire space
- A trip to Honolulu = Role

Information from two different spaces can be blended in order to achieve a final space. The sentence ”If Sophie had eaten cake she would have been happy” represents counter-factual conditionals that blend in order to become one space. This blending process occurs on a wide range of concepts.

Langacker states in Chapter 1 of ”Cognitive linguistics” that ”semantics is conceptualization” [43] which is quite different from the Chomskian ”truth conditional” idea that all situations can be framed in different ways, be it positive or negative. The hearer is conveyed different conceptualizations of the relationship between the speaker and the hearer. Croft et al. say that ”whenever we utter a sentence, we unconsciously structure every aspect of the experience we intend to convey” [43]. We can see this nuance in the following examples:

- ”Chantal is living in Bungay” vs ”Chantal lives in Bungay”
- ”We have apple pie for dinner” vs ”There is apple pie for dinner!”

These subtleties are difficult for computers to understand because they require an experience of language, human behaviour and relationships that a machine does not have.

Lakoff [108] elaborated on these theories by focusing on the way in which we categorize things which have common properties in our world. These categories are not only of things but also of entities such as events, emotions, spatial relationships and abstract entities such as governments, illness and so on. Reason is achieved by our ability to categorize elements in a way that makes them meaningful. Lakoff states that a computer has the capacity to reason because as it is able to manipulate disembodied symbols; however, it is incapable of taking into consideration the particularities of the human mind and body. Lakoff is of the opinion that reason is embodied.

Prototype theory originally proposed by Rosch[169], is central to Lakoff’s model of cognitive semantics. It takes into account human experience, imagination, perception, motor activity, culture, metaphor, mental imagery and metonomy. It is not possible at this time to instil these qualities into a computer and it is not our goal to create an artificial human or a even a ”reasoning machine”. Our goals are humble and remain based on allowing humans to communicate with the machine and receive a natural language response. Prototype Theory may well be of great interest in the future of machine natural language understanding and generation. Lakoff’s models are interesting to us and could be applied to our system. The example he gives is of “Mother” which is far more complex than simply ”Woman who gave birth to a child”. He describes a more complete list of models being:

- Birth model: person who gives birth is the mother
- Genetic model: female who contributes the genetic material
- Nurturance model: female adult who nurtures and raises a child
- Marital model: wife of the father is the mother
- Genealogical model: closest female ancestor is the mother

Modern life of course means that there are many models, such as ”step-mother” and ”donor” for example that also belong here. ”Mother” is also an energy drink that comes in a large black can in Australia, for example. This model must also be added to the categorization system somehow, so there is convergence of models in certain instances. Additionally not all mothers are nurturing for example. To this we can add expressions such as ”He threw me the mother of all punches” and ”Necessity is the mother of all invention”. Lakoff describes a particular type of model that is very difficult for machines to disambiguate: the metonymic model. The example given by Lakoff is:

- ”The White House isn’t saying anything”

Taken literally, this would perhaps belong in a Fairytale, but as it stands this refers to the United States government staying silent on an issue. Metonomy is very difficult to understand for humans as well if we have no background knowledge on the elements in question. It is necessary to create relationships between "white' House" and "Government" for example in the machine's ontology. Other models include the "Propositional model" (where elements, properties, their relationships are all specified), "Image-schematic models" (trajectories, shapes, containers, distance and so on) and the "Metaphoric model" (mapping from a propositional or image schematic model into one domain).

Lakoff, Fillmore, and Langacker's work clearly demonstrates that no taxonomy can be complete enough, because there is no correct taxonomy for every situation and everything in the universe. In this sense, they can never be complete, so we require a different kind of classification which allows for the complexities of our world to become clear. In terms of computing, we are trying to organise our world and our experience of it in a way that can be easily queried by a machine.

There are so many different dimensions to each word that the complexity is positively startling. Construction Grammar allows us to work at a higher level of abstraction, meaning that the complexity is reduced.

5.3 Construction Grammar

Construction Grammar (CxG) is described by Fried et al. as "constraint based, generative, non-derivational, monostratal grammatical model, committed to incorporating the cognitive and interactional foundations of language. [69]. It grew out of Frame semantics and was introduced by Lakoff, Talmy, Kay, Fillmore and Langacker. The concept is based on what elements people use to understand information. It consists of taxonomies of families of constructions and uses entire constructions as the primary unit of grammar rather than the word which is often used in other grammars as the atomic unit.

Words are instances of constructions. The constructions themselves are made up of patterns of varying complexity, but pairings of form and meaning are considered constructions (including phrasal structures, idioms, words and even morphemes). The syntactic pole includes syntax, phonological features, prosody and intonation. All these elements are seen as equally important. The semantic pole includes semantic and pragmatic meaning, and discourse. There is a Syntax-lexicon continuum where words as well as complex constructions are considered pairings of form and meaning and differ only in their internal symbolic complexity. The general principle is that each component governs linguistic properties of a single type.

- Form: syntactic, morphological, prosodic patterns (representations of the constructions themselves)
- Meaning: lexical semantics, pragmatics, discourse structure (representations of the hierarchical category rules which merge syntactic and semantic information)

Generative grammar is a lexical projectionist view of grammar. The principle is that words relate to others by specifying the complements, adjuncts and determiners they require. In projection-based models, meaning is expressed at word level rather than at construction level. Syntactic rules combine words together to create meaning. However, it is the sentence itself that is viewed as having meaning rather than the patterns in that sentence. These types of grammar focus on verbal argument structure which means that the verb is the pivotal element in establishing meaning in a sentence. Linking rules are used to assign thematic roles (agent or patient) to the subject or object. Any particular verb may be subject to several different linking rules that are mutually incompatible. If the argument structure (form + meaning) serves only to give varying semantic role realizations, which are governed by the verb, then verb sense ambiguity can be a problem and the word sense becomes "ad-hoc and unintuitive" [74].

> As we will see, the CxG model of semantic composition is integrative rather than projection-based: like words, constructions denote semantic types (e.g., events and entities) and, like words, constructions license syntactic and semantic dependents; therefore, the interpretation of a phrase involves combining the interpretive and combinatoric constraints of the phrasal pattern with those of the word that is the head of the phrase. [138]

The participant roles of the verb are identified and verb meaning is always constant across the syntactic contexts. They combine with verb-level linking constructions, which denote event types. These linking constructions assign grammatical function to participant roles contributed by the verb. These also have thematic roles assigned to them determined by the different event types.

In CxG the patterns that allow for various word combinations are considered to have meaning. The verb meaning is constant across syntactic contexts and additionally verbs combine with verb-level linking constructions which assign grammatical function to the participant roles. The semantic properties of the verb and the constructions are combined using fusion.

Michaelis [138] explains the concept of CxG using mathematics: 2x(3+4) is different to (2x3)+4 but the meaning of the individual numbers does not change. CxG applies this logic to syntax. The patterns in the strings change, but not the meaning of the individual words. Michaelis does point out however that this analogy isn't completely correct because content words do not designate in the way that numbers do, but rather the syntactic content dictates the behaviour of the word. This phenomena is called Type Shifting.

CxG is made up of taxonomic networks of families of constructions based on cognitive linguistic concepts such as inheritance, extensions, multiple parenting and so on. The information stored in the taxonomies is dictated by different models:

- Full-entry: information is stored at all levels
- Usage-based: based on inductive learning (bottom-up)
- Complete-inheritance: information is stored at the most subordinate level in the network

We describe CxG as "metonymic" because it does not operate with dynamic rules of derivation, since it is based on taxonomies. For example, the active and passive version of the same proposition are not derived from an underlying structure but are instances of two different constructions.

The frames used in construction grammar according to Goldberg [74], differ from regular frames because the argument structure types (distransitive constructions, cause-motion constructions etc...) invoke frames which designate

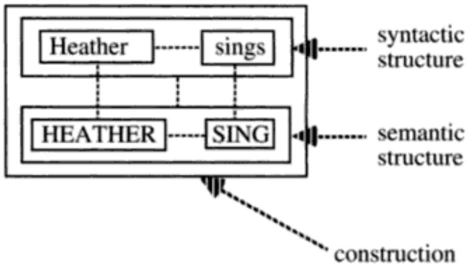


Figure 5.4: The “Heather sings” construction

event types. The verb alone is not the main unit of meaning, but the construction itself is. This comes from the frame-semantic knowledge associated with it.

In the example of ”Heather sings” in figure 5.4:

- Semantics: relational predicate involving a singer
- Syntactics: predicate requires arguments and “Heather” is the subject

A symbolic link is used to join an element of the syntactic structure of a construction to a component of the semantic structure of the same construction. The minimal parts of the construction are morphemes (words), each part of a construction has syntactic and semantic features and the syntactic-semantic (syn-sem) unit is brought together by a symbolic structure representation. ”Role”, “Valence” and ”Relation” are used to assemble the different parts of the construction together.

- Role = role of the syntactic element in the whole (i.e. ”Heather”
- Valence = relation of the predicate to its argument (i.e. ”sings”)
- Relation = indicates the relation of each argument to its predicate. It uses the grammatical function and the thematic role i.e. ”Heather” (rel. feature subject, Thematic role (agent))

Predicates and arguments in sentences are matched with each other so that each argument relation is matched up with one of the elements in the valence list of its predicate. The information is represented in the taxonomy of constructions only once, because the lower nodes in the taxonomy inherit.

The ”taxonomy of constructions” contains a structured inventory of known language conventions. It is a network where each of these constructions is represented by a node where all nodes inherit. As Croft explains, it is unreasonable to believe that speakers store all of the grammatical constructions that they come across. In order to have an efficient taxonomy it is necessary to minimise redundant information. The ”Usage-based model” allows for patterns in language to be used to represent grammatical information [74]. Cognitive linguistics are primordial in this model. A taxonomic relation describes a relationship of schemacity or generality between two constructions. The information is stored in the taxonomy in a way that allows full representation at all levels in the hierarchy. This is called the ”Full-entry model”.

If a node has multiple parents in the hierarchy, then it will inherit from all parents. This model is useful in terms of computing because it maximizes "computing parsimony", which means that redundant information is minimized. Lakoff and Goldberg [74] advocate using a "usage-based model", which means that patterns of language are stored as and when they are encountered. This is the method we have also used because the system is able to learn new constructions as it is exposed more and more to user's language.

From a practical perspective, constructions are mapped onto a conceptual space according to their function. This way, different constructions can be related to each other owing to their overlapping or neighbouring functions in this conceptual space. The schematic constructions in the taxonomy of constructions represent a categorisation of utterances according to their grammatical and semantic properties. In figure 5.5 we have the taxonomy for the construction "Kick the bucket".

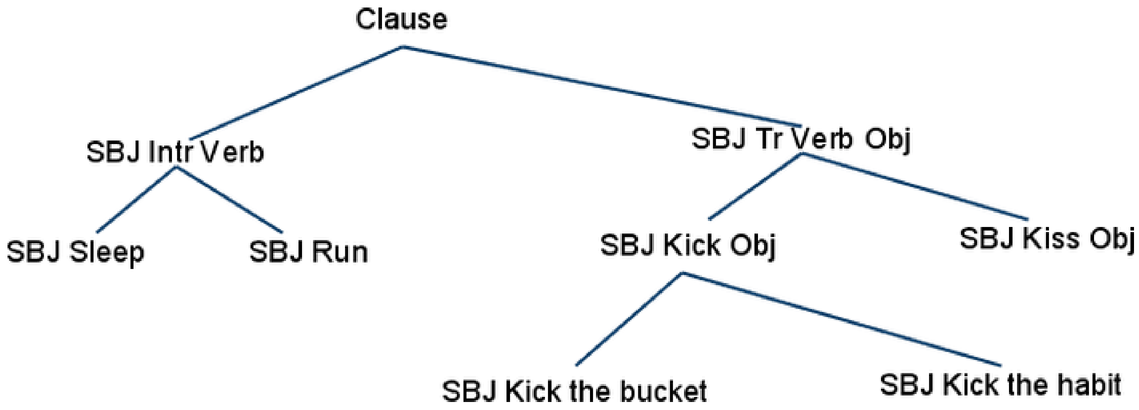


Figure 5.5: Example taxonomy

The lexicon is an addition and gives the syntactic category of each word and its meaning. There is no strict division between then lexicon and the syntax, both lexical and syntactic constructions pair form and meaning together. The linking rules map the syntactic structure of a sentence onto the semantic structure. Each element (syntactic) has a corresponding component (semantic) and is part of the entire construction, as can be seen in 5.6.

Generative grammar works at word level, so all constructions are no larger than an individual word. In this model the linking rules that relate to each component are deemed to be so intertwined and self-contained that they represent a meaningful structure in relation to the linking rules. This perspective means that larger structures (sentences for example) are governed by the very general rules of the syntactic component. The words are said to represent an arbitrary and idiosyncratic pairing of form and meaning.

Chomsky argues that "All arbitrary and idiosyncratic aspects of grammar should be restricted to the lexicon". Generative grammar also enables larger constructions to be captured with the general rules of the grammatical components and their interfaces. Construction grammar breaks away from these views and provides a far less componential

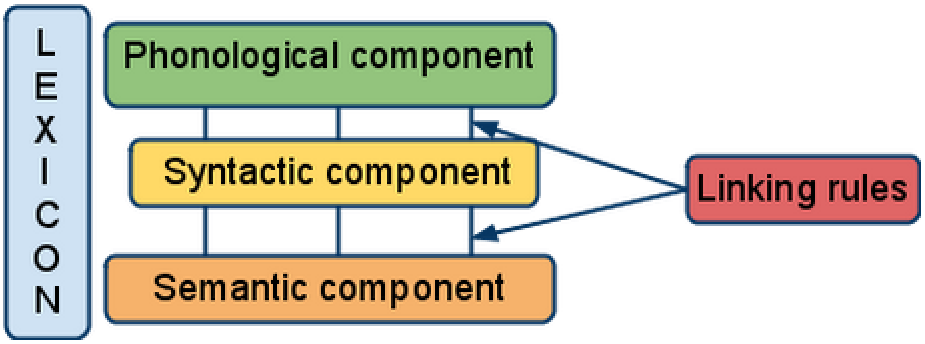


Figure 5.6: Lexicon

view of grammar. Idioms for instance are problematic for the componential model because they operate on a larger construction level. If we work at word level, the meaning is lost.

Idioms, such as “to kick the bucket” and “to spill the beans”, do not work well in a componential model because the rules of syntactic and semantic components and the linking rules cannot be predicted. Construction grammar operates on a larger structure which means that idioms can be identified as such. It lists the constructions classed as idioms as “phrasal lexical items” and these are also found in the lexicon. The revision of the componential model proposed by construction grammarians looks a little different as shown in figure 5.7.

Constructions combine the syntactic and semantic components in the same way that the lexicon does. It is possible to say that the structure is vertical. A construction is seen as a syntactic configuration with one or more substantive elements and it also has its own semantic interpretation and pragmatic meaning. The rules used for semantic interpretation can be created for any schematic construction describing even very general syntactic structures. All syntactic expressions, whatever their degree of schemacity have rules of semantic interpretation. A construction can be atomic (word level) or complex (larger structures), but all have pairings of form and meaning, including pragmatic meaning. Please see figure 5.8 for diagram of the symbolic components of CxG, and figure 5.9 for a diagram showing how form and meaning interact in CxG.

Form and function are treated differently in the componential and construction grammar models, as seen in figure 5.9. In the componential model the syntactic structures are organised independently of the corresponding semantic structures. In the CxG model the basic linguistic units are symbolic and are organised as symbolic units. This means that the internal structure of the basic units in CxG is more complex than in the componential model.

Lakoff and Goldberg [74] state that we should analyse participant roles in complex events as derived from the event itself, following the principles of frame semantics. They explore the non-classical categorisation in their analysis of relations between different constructions. Goldberg and Lakoff subscribe to the “reductionist approach” which means that they take a set of atomic primitive grammatical relations, for example, subj + Obj + primitive syntactic

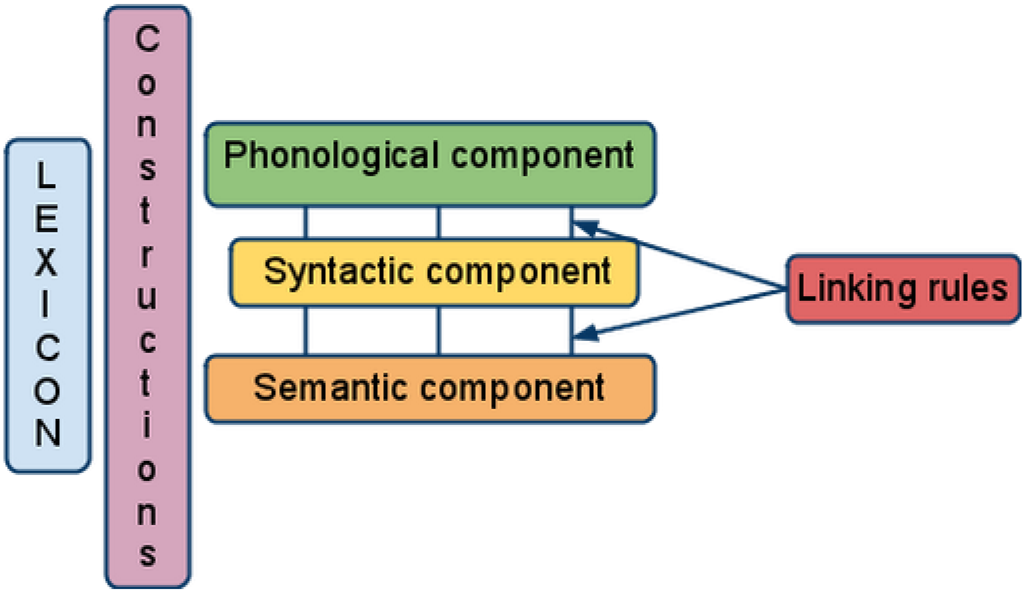


Figure 5.7: Revised componential model

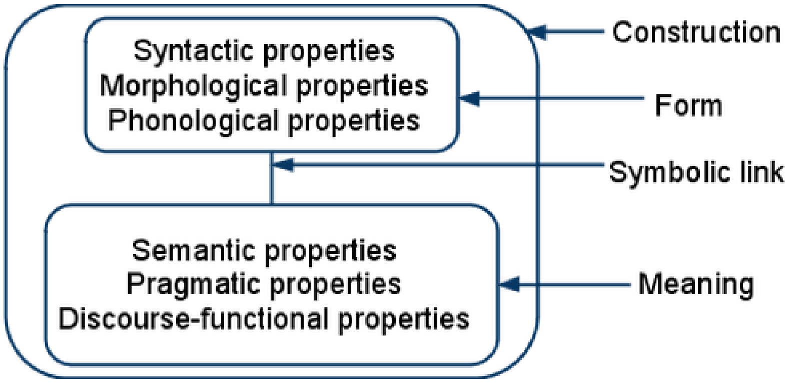


Figure 5.8: Symbolic links in CxG

categories such as verbs. The "non-reductionist approach" in contrast involves treating a complex event as the primitive unit of semantic representation. The definitions of the roles in the events are derived from the situation as a whole.

This approach reduces the complexity involved in computing language. We are able to train a machine on human-human dialogues for example and construct a taxonomy of constructions based on those. The usage-based model can be translated as a machine learning module to build the existing taxonomy of constructions over time and experience.

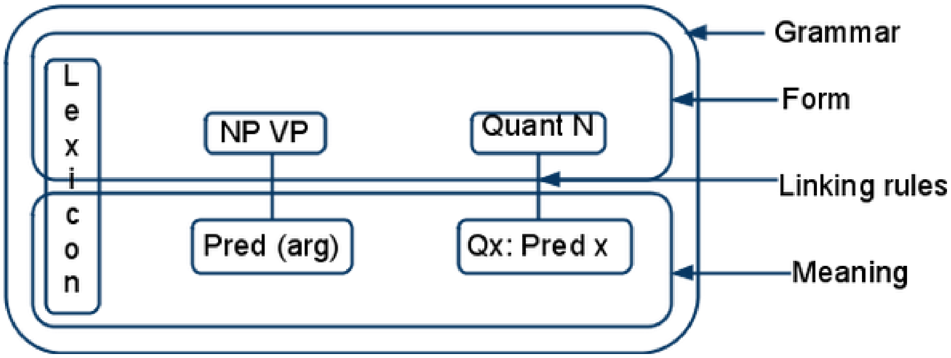


Figure 5.9: Form & Meaning in the componential model

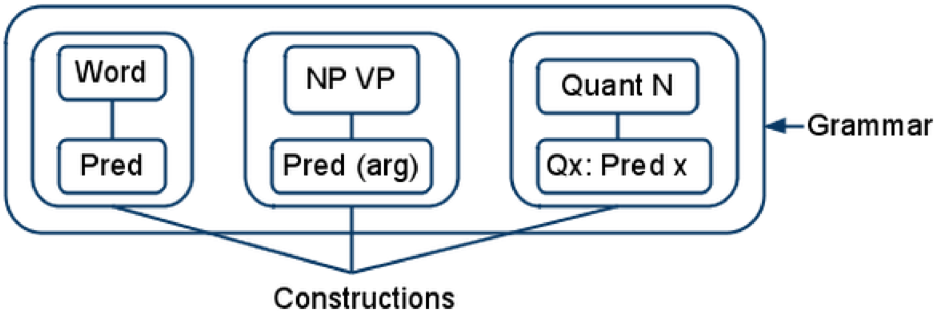


Figure 5.10: Form & Meaning in CxG

5.3.1 Fluid construction grammar

Fluid construction grammar was invented by Luc Steels [180] and used for language development experiments, such as the Sony robotic dog. It was designed to handle open-ended grounded dialogue described as “dialogue between or with autonomous embodied agents about the world as experienced through their sensory-motor apparatus”. The experiment was carried on robots which responded to verbal human directives. Our project differs from the Sony one in that our system experiences the world through the knowledge base and the taxonomy of constructions it possesses. It also does not require the phonology element at this time because it is text-based. Investigating this form of CxG, however, makes sense because it integrates user input.

FCG makes use of semantic and syntactic poles, is bi-directional, selects meanings and maps them into the real world. It builds on the tradition of unification-based feature structures such as Head-Driven Phrase Structure Grammar (HPSG), a non-derivational, constraint-based, surface oriented grammatical architecture [101]. It is a generative grammar which has the goal of “building a scientific theory of the knowledge in the mind of the speaker that makes language possible” [154]. However, there are important differences:

- FCG structure has units corresponding to words and constituents
- The Unit has a name and a set of features
- The hierarchical structure is not implicitly represented by embedding one unit into another, rather correspondence is maintained by using the same unit names in both the semantic and syntactic structures.

Additionally FCG has the following particularities:

- It is a multiple-agent environment
- New categories can be added or changed at any time
- Rules are in competition for dominance in the population
- The lexicon and the grammar are expressed by constraints on feature structures
- All form aspects are expressed in a declarative way
- Multi-tasking is used to explore multiple hypotheses
- Rules can be applied in either direction
- Rule inventories can be expanded with new constructions
- All utterance information is stored in a feature-structure based formalism

Natural language generation is not normally bi-directional, but FCG uses the same rules for production (conceptualization and understanding) and parsing (interpretation and understanding). It is called fluid because it takes into consideration the fact that users change and update their grammars. Rules stretch and expand as required, so new aspects of language (concepts, idioms...) need to be incorporated dynamically as they arise and spread in a population.

The rules are templates which express the possible constraints of meaning. Each rule has a score reflecting its success within the current model and agents are programmed to favour high scoring rules. All rules have a syntactic (right) and a semantic (left) pole. The semantic pole, contains the semantic structure which is expressed as a feature structure with a number of variables. The syntactic pole contains the syntactic structure expressed as a feature structure with its associated variables. The rules must all be bi-directional and are divided into subsets which constrain the order in which they can be applied. Different types of rules include:

- morph-rules: decompose a word to its stem and add a syn-cat
- lex-stem rules: associate meaning with a stem as well as the valence and role frame
- con-rules: correspond to grammatical constructions associating parts of the semantic structure to parts of the syntactic structure.
- sem + syn rules: these perform inference over semantic and syntactic categories to expand the semantic or syntactic structure.

During the production phase, the left pole is unified with the semantic structure under construction. If this is successful then the right pole is merged with the syntactic structure under construction. The "Unification" phase is used to determine if a rule is triggered and the "Merge" phase represents the actual application of the rule. The "merge" operation is a partial unification. During the parsing phase, the right pole is unified with the syntactic structure and the left pole to the semantic structure. The rules trigger in the following order:

Production:

1. lex-rules
2. con-rules
3. morph-rules

Parsing:

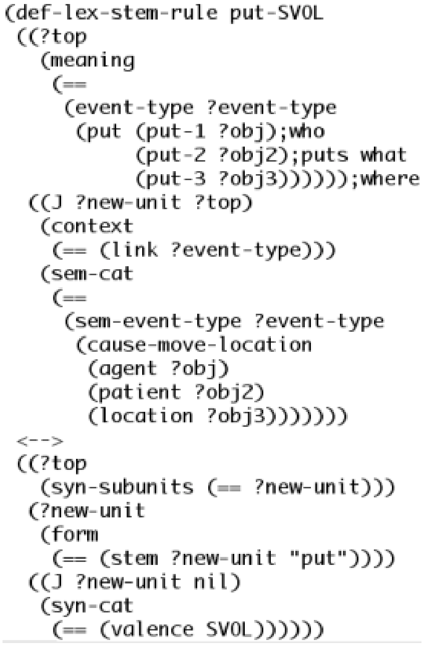
1. morph-rules
2. lex-rules
3. con-rules

FCG is able to deal with incomplete or partially ungrammatical input because of its ability to adapt to user grammar usage. This method allows for user input to be be broken down syntactically in order to gain meaning from the grammatical components, whilst also being able to map the semantic relationships. The information gained this way enables the system to define the topic and sub-topics involved as well as giving information on the construction of the sentence to be volunteered to the user as a response.

The hierarchical structure of the left and right poles can be modified using the J-operator (please see figure 5.11 for an example from Steels et al. [180]). This allows the semantic pole's constructions to be decomposed and the syntactic pole and groups units together in larger constituents. The J-operator constraints do not need to match at the unification phase. It is used to decompose meaning on the semantic side and group constituents together on the syntactic side. Lexical construction provide the frame and valence information for word stems whereas the grammatical constructions bind all of this together. Lexical items and new constructions are added to the system as well as additional rules. New meanings and constructions are identified using user input because this is a usage-based model. We are able to extend the ontology, add new lexical items, identify new constructions, assign new words to syntactic classes and much more.

FCG uses IRL Incremental Recruitment Language (IRL) introduced by Wouter Van den Broeck [53]. It is a constraint language which implements the necessary planning, chunking and execution mechanisms of the constraint networks. The example below shows a constraint network for "TheBox" [180] (? denotes a variable, ?r = relation, ?p = prototype, ?d = determiner, ?o = object). Please refer to figure 5.12 for the resulting decomposition structures.

1. (equal-to-context ?s)
2. (filter-set-prototype ?r ?s ?p)
3. (prototype ?p [Box])



```
(def-lex-stem-rule put-SVOL
 ((?top
   (meaning
    (==
     (event-type ?event-type
      (put (put-1 ?obj);who
           (put-2 ?obj2);puts what
           (put-3 ?obj3)))))));where
  ((J ?new-unit ?top)
   (context
    (== (link ?event-type)))
   (sem-cat
    (==
     (sem-event-type ?event-type
      (cause-move-location
       (agent ?obj)
       (patient ?obj2)
       (location ?obj3))))))))
 <-->
 ((?top
   (syn-subunits (== ?new-unit)))
  (?new-unit
   (form
    (== (stem ?new-unit "put"))))
  ((J ?new-unit nil)
   (syn-cat
    (== (valence SVOL))))))
```

Figure 5.11: Example lexical entry for put and illustration of the J-operator.

4. (select-element ?o ?r ?d)
5. (determiner ?d [single-unique])

The structures (4) and (1) are primitive constraints that implement fundamental cognitive operators. The elements in the current context are bound to "?s". The structure (2) filters the set with a prototype ?p which is bound in structure (3) to [Box]. The structure (4) selects ?o from ?r according to ?d bound to [single-unique]

The information about an utterance is stored in a feature-structure based formalism and contains units which correspond to morphemes and the constituents. The syntactic structure contains the syntactic subunits, the syntactic categories and the form. The semantic structure contains the semantic subunits, the semantic categories, the meaning and the context (variables that occur in meaning but that are considered external because they link to variables that occur in the meaning of other units).

We attempted to implement FCG in our KIA chatbot system. This experiment allowed us to experience first hand the limitations of FCG. The biggest limitation is that it is very difficult to create bi-directional rules. Few exist, and creating more is very time consuming. The parsing and production rules are often very different, so finding a universal rule is very hard. Additionally, IRL is not developed enough at this time to accommodate our project and currently, FCG does not directly support a connection to an underlying taxonomy of constructions. It is not clear as yet whether this is needed, because typically constructions are implicitly linked anyway through the set of semantic and syntactic categories they require for application and add themselves during application. A more compact representation of constructional definitions is possible through defining the bidirectional rules, but as we have seen this is quite difficult.

IRL is still under active development. There is an engine that allows us to define and reason with IRL components and this can be used to compute the initial meaning for producing an utterance describing an object and starting from sensory data. It is however not clear yet how more complex meanings will be encoded and constructed, and how complex IRL meanings in the end will look like. FCG is a very interesting development for CxG, however at this time it is not mature enough for use in our system.

5.3.2 Differences between construction grammar and other grammars

Construction grammar follows very different rules to the grammars that preceded it. Here we cover the most notable ones, namely formal grammars, context-free grammar and transformational grammar. This enables us to view the distinction of construction grammar and fluid construction grammar with existing models.

5.3.2.1 Formal grammars

Formal grammars include generative grammars and analytic grammars. These were introduced by Noam Chomsky [35]. Generative grammar is syntax based and stochastic. It has a set of rules which are used to predict sentence structure. In contrast, the Chomsky hierarchy is a collection of strictly nested sets where each entry designates a set. Each of the previous entries is a strict superset and the next a strict subset.

For example:

Animal - Bird - Raptor - Eagle - Golden Eagle

The Chomsky hierarchy is recursively enumerable, context sensitive, context-free and regular. It is a “containment hierarchy of formal grammars that generate formal languages” [35]. A formal language is defined as a set of predefined rules. Formal grammar includes generative grammar (how strings are generated) and analytical grammar (how a string can be analysed). These models focus on sentences in far more detail than construction grammar and have been considered inflexible as the production rules are so strict [35]. Analytical grammars incorporate different subtypes of grammar such as link grammars. They directly correspond to the structure and semantics of a parser for the language. Link grammar is a theory of syntax which allows for relationships to be built between pairs of words and do not use a tree hierarchy. These again are too inflexible to deal with the ambiguity of language.

5.3.2.2 Context-free grammars

Context-free grammars (CFG) work at a higher level than generative grammars. Derivation trees are used and are considered a focus point. CFG consists of a set of recursive rewriting rules used to generate patterns of strings. Sentences are viewed as trees with various branches in a hierarchy attached to nodes. These grammars are considered inadequate for describing complex language as they don’t take into consideration other aspects of language such as pragmatics, semantics and other elements. They also need a very complicated grammar to operate which is very difficult to create, understand and maintain. The following example from McCallum [133] breaks down the CFG process clearly. In figure 5.13 we have the CFG, in figure 5.14, we have an example of the CFG being applied, in figure 5.15 the rewrite rules are applies and in figure 5.16 we have the resulting parse tree.

5.3.2.3 Transformational grammar

Transformational grammar combines CFG and sequential transformations which take a tree from the previous step and creates a modified tree. This continues until the "surface structure" is reached where all the sentences are found at the bottom of the tree. This method allows for long-distance relative clause dependencies, are more efficient than using phrase-structure rules, and feature dependencies can be cross analysed. Chomsky [34] deals with the active-passive relationship by stating that if S1 is a grammatical sentence with the form NP1—-Aux—-V—-NP2, then the corresponding string of form NP2—-Aux + be + en—-V—-by + Np1 is also a grammatical sentence.

There are many other transformation that are simply active-passive in the English language, for example the sentence "Has Mamie cooked?" is a transformation of the sentence "Mamie has cooked". The issue however is that rules that are to be applied to one type of phrase can be applied to the wrong ones. This is again due to using a stochastic method which is too inflexible for the fluidity of language. It also does not pair syntax and semantics.

5.4 Conclusion: The Construction grammar method in our SOCS

As we have seen in previous chapters, the methods currently employed in SOCS technology at this time do not yield very meaningful and satisfying results. This is partly due to the lack of integration between the syntactic and semantic components and, in some cases, to the lack of a usage-based model allowing the system to expand its vocabulary and knowledgebase as required. The keyword level, for example, is very different from the construction grammar approach as here, as in generative grammars, the word is the atomic unit rather than the entire construction.

Fluid construction grammar means that we could parse and produce at the same time, which is an improvement on the two or more steps process of existing systems, but the method is as yet still too undeveloped for use in our system. CxG allows us to incorporate both form and meaning and to treat each utterance in the context of the real world rather than in isolation. The constructions can be computed, as we demonstrate in the following chapter, and provide a flexible structure for the system to work with. CxG is easy to maintain, taxonomies of constructions are quickly and easily created and the model also lends itself well to machine learning methods available to use at this time. In the following chapter, we describe our CxG based SOCS, KIA2.

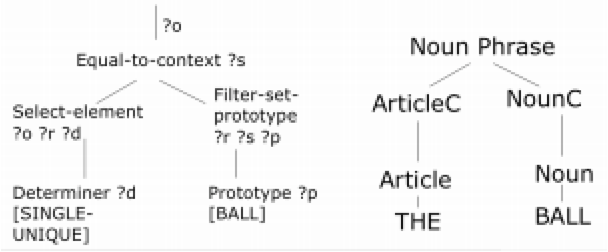
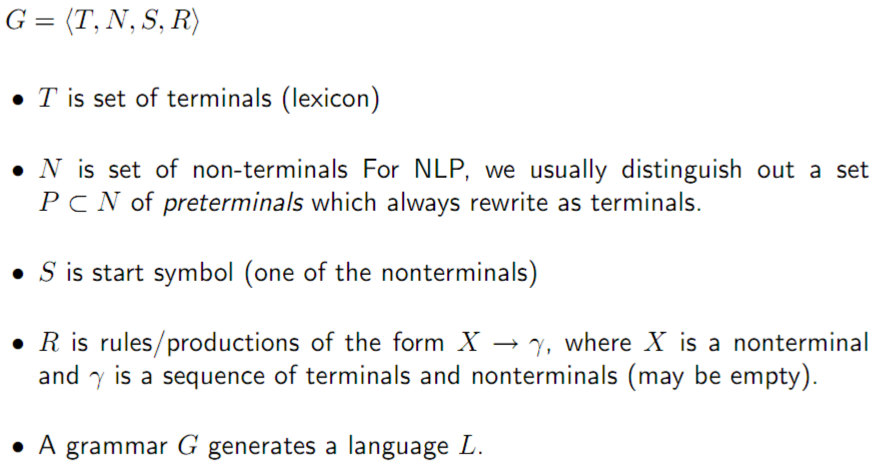


Figure 5.12: Left: decomposition of the constraint program for the ball in the semantic structure. Right: related syntactic structure. In reality both structures contain a lot more information



$G = \langle T, N, S, R \rangle$

- T is set of terminals (lexicon)
- N is set of non-terminals For NLP, we usually distinguish out a set $P \subset N$ of *preterminals* which always rewrite as terminals.
- S is start symbol (one of the nonterminals)
- R is rules/productions of the form $X \longrightarrow \gamma$, where X is a nonterminal and γ is a sequence of terminals and nonterminals (may be empty).
- A grammar G generates a language L.

Figure 5.13: Context-Free Grammar

$G = \langle T, N, S, R \rangle$

$T = \{$*that, this, a, the, man, book, flight, meal, include, read, does*$\}$

$N = \{$S, NP, NOM, VP, Det, Noun, Verb, Aux$\}$

$S = S$

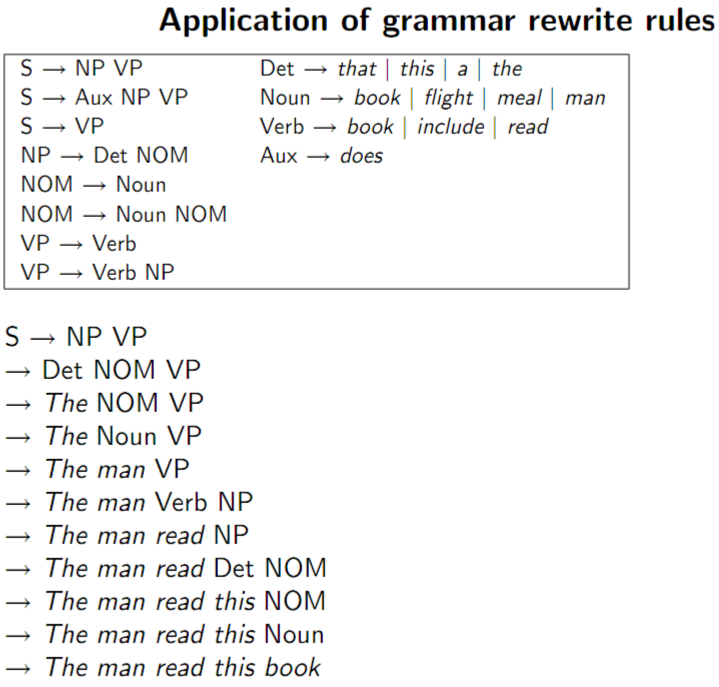
$R = \{$

S → NP VP
S → Aux NP VP
S → VP
NP → Det NOM
NOM → Noun
NOM → Noun NOM
VP → Verb
VP → Verb NP

Det → *that* | *this* | *a* | *the*
Noun → *book* | *flight* | *meal* | *man*
Verb → *book* | *include* | *read*
Aux → *does*

$\}$

Figure 5.14: Context-Free Grammar example



Application of grammar rewrite rules

S → NP VP
S → Aux NP VP
S → VP
NP → Det NOM
NOM → Noun
NOM → Noun NOM
VP → Verb
VP → Verb NP

Det → *that* | *this* | *a* | *the*
Noun → *book* | *flight* | *meal* | *man*
Verb → *book* | *include* | *read*
Aux → *does*

S → NP VP
→ Det NOM VP
→ *The* NOM VP
→ *The* Noun VP
→ *The man* VP
→ *The man* Verb NP
→ *The man read* NP
→ *The man read* Det NOM
→ *The man read this* NOM
→ *The man read this* Noun
→ *The man read this book*

Figure 5.15: Application of Grammar re-write rules

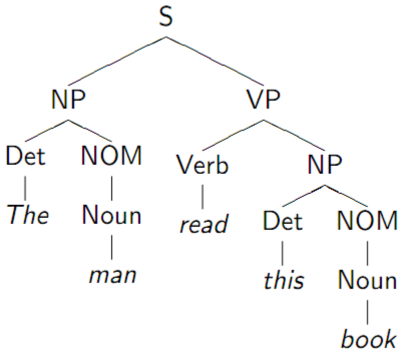


Figure 5.16: CFG Parse tree

Chapter 6

System Design

In this chapter, we describe the system architecture and the processes and methodologies involved in: processing a query, formulating a response, training the system and creating, accessing and storing the taxonomy of constructions. We describe the techniques used in each part of the system in detail, as well as highlighting any limitations and improvements which we deem possible in future iterations.

6.1 KIA2 system overview

There are 4 main sections to KIA2: the "Canned response" engine, the "Syntactic pole", the "Semantic pole", and the "Response generation engine". Please refer to figure 6.1 below for an overview of the system.

The Canned response engine is composed of a spell checker (to eliminate any user spelling or typing errors), and a method to check for an exact match in the question-answer pairs available for the system to choose from in the Canned responses database. If there is a formulaic response [202] which is a definite match word-for-word in the Canned responses database, the corresponding response is given, and the system stops there and awaits a new user input.

If there is no such match, the query is further processed through the construction grammar method, which is comprised of the syntactic and semantic poles, as well as a response formulation engine and a taxonomy of constructions. The output of both the semantic and syntactic analysis is a semantic-syntactic frame (SynSem Frame). The syntactic pole identifies the constructions and its parser adds syntactic information to each construction. We store these as “meta data” about that construction. The semantic pole processes the same construction and adds semantic information to it in the form of semantic meta data. Now we have a query which is decomposed into constructions (noun phrases and verb phrases), and we have syntactic and semantic information about them. We have what is referred to in linguistics as a SynSem Frame.

The next stage requires the system to query the taxonomy of constructions. This is a database comprising

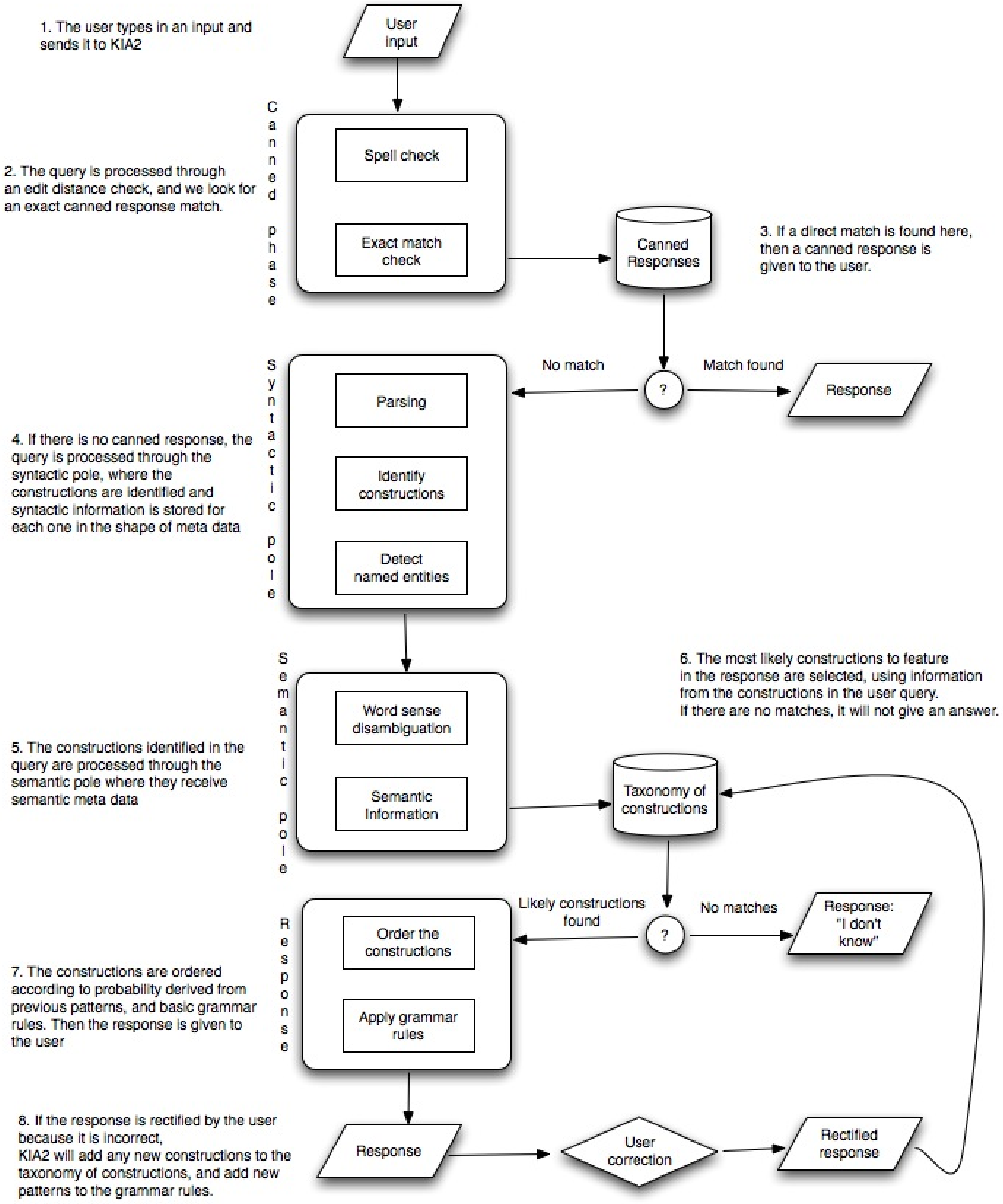

1. The user types in an input and sends it to KIA2
User input
Canned phase
2. The query is processed through an edit distance check, and we look for an exact canned response match.
Spell check
Exact match check
Canned Responses
3. If a direct match is found here, then a canned response is given to the user.
No match
?
Match found
Response
Syntactic pole
Parsing
Identify constructions
Detect named entities
4. If there is no canned response, the query is processed through the syntactic pole, where the constructions are identified and syntactic information is stored for each one in the shape of meta data
Semantic pole
Word sense disambiguation
Semantic Information
5. The constructions identified in the query are processed through the semantic pole where they receive semantic meta data
6. The most likely constructions to feature in the response are selected, using information from the constructions in the user query. If there are no matches, it will not give an answer.
Taxonomy of constructions
Likely constructions found
?
No matches
Response: "I don't know"
Response
Order the constructions
Apply grammar rules
7. The constructions are ordered according to probability derived from previous patterns, and basic grammar rules. Then the response is given to the user
8. If the response is rectified by the user because it is incorrect, KIA2 will add any new constructions to the taxonomy of constructions, and add new patterns to the grammar rules.
Response
User correction
Rectified response

Figure 6.1: KIA2 system diagram

previously seen constructions, together with their associated meta data. The system looks for constructions that are typically associated with the constructions it has just processed from the user. It uses a Naive Bayes approach to machine learning, to select the most likely constructions from the database. Several probabilities come into play:

- The probability of a construction in the database being relevant to one of the constructions given by the user
- The probability of a construction in the database being relevant to all of the constructions given by the user

- The probability of all of the selected constructions being found together in the same response

The system is designed to always pick the constructions that have the highest probability scores. This is not always going to lead to a correct response, but it is the best educated guess that the machine can have, given its current knowledgebase.

Once the constructions have been selected, the response generation module has the task of deciding what order they should go in and if any additional grammatical rules need to be taken into account. The rules are made up of previously acquired knowledge, which has been learned over time from system usage. The system is told that when a particular construction follows another particular construction, then a preposition like “for” for example, needs to be added for the sentence to be correct. This is a way for the machine to give correct responses over time, without going through a large grammatical rules database that may not be relevant to the types of conversation it is exposed to in a particular domain.

The response generation engine is responsible for giving an answer to the user. If this answer is incorrect, the user may say so by typing: “Wrong: ” followed by the exact re-wording of the sentence. The system then repeats the correct form of the sentence as given to the user and the conversation can continue. The correction is processed back through the system, so that in future the same mistake is not made. This may result in simply knowing that when construction “x” and construction “y” are adjacent, then you should add the preposition “z”, for example. If the entire response is incorrect, and the wrong information has been found, then the system will break down the user given response into constructions, and this will go through the syntactic and semantic poles, and then be added to the taxonomy of constructions, where weightings for each construction will be adjusted.

6.2 The architecture

All of the code use for word dismabiguation (Senseval and KNN), String matching (Demereau-Levenstein), APSELL spell checker, dictionary-based chuncker (Aho-Corasick algorithm), are all from the Lingpipe toolbox which is freely available /citeCarpenter2011 . The tagger is written by Eric Brill [25], we used the JAWS Java version [121]. UEA LITE (Java) was our own code. The Small program ”Glue” (Java) wrote output to the SWING interface and called on the rest of the system to signal an input.

6.2.1 Getting started

The steps required to get started with this system requires the following setup:

- Build Glue and UEA LIte
- Install and configure Wordnet
- Install and configure SENSEVAL database.
- Create a Constructions database

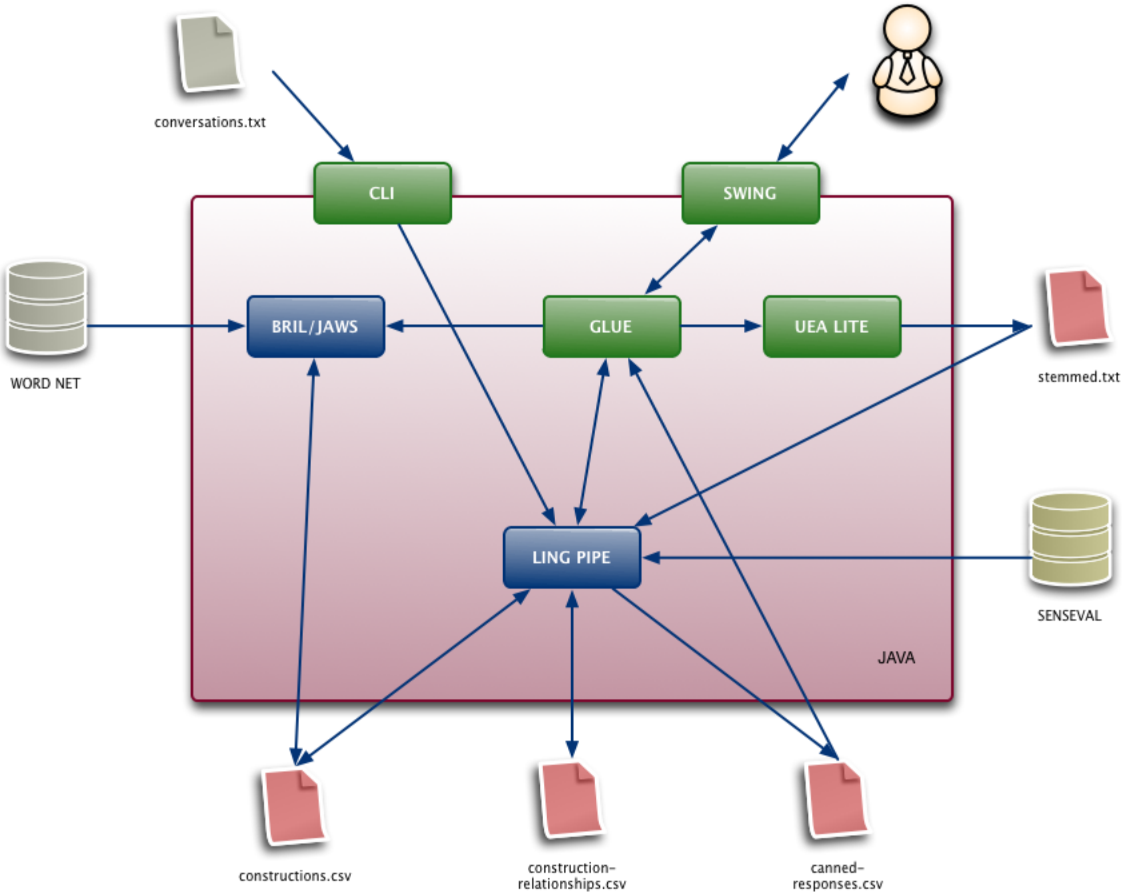
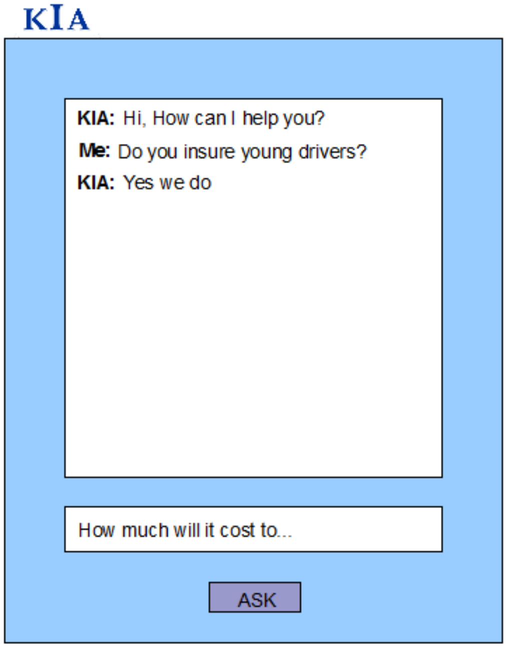


Figure 6.2: KIA2 system architecture

- Create a Constructions Relationships database
- Read in the conversations file (either a public domain conversation collection file or a specific file as the Aviva customer service logs that we used.
- Create the canned responses database by running the system once
- Manually curate the canned responses
- Manually curate the Constructions database if needed
- Run the system from the SWING interface
- Type "Wrong" f the answer is incorrect and modify the response to be accurate (manually training the system)

6.3 The user interface

The user enters a query into the text box provided in a very basic interface. The interface is simple because we focus on evaluating the output of the system rather than creating a user focused interface for a finished system. We have carried out studies on how interfaces for these systems should be constructed and it would not be difficult to create one for commercial use, now that we aware of which features and factors are important. In order to test the system we have supplied a text box which is small in order to encourage a concise input, and a panel which displays the user-machine conversation. There is an "Ask" button to submit the user input to the machine. All conversations are logged for analysis. Please refer to figure 6.3 below.



KIA: Hi, How can I help you?
Me: Do you insure young drivers?
KIA: Yes we do

How much will it cost to...

Figure 6.3: KIA2 system user interface

6.4 Canned response phase

The first stage of query processing involves checking to see if the query that the user has entered into the system matches a question-answer pair that is common and reoccurring. These are described as instances of formulaicity as described by Wray [202].

This database is constructed from the human-human customer service-customer conversations and from the user and system testing logs we possess from previous experiments. We use a string similarity algorithm on the utterances of the customer service agent and apply it to discover exact matches. We then observe the string similarity between the responses given by the user. We also do this for user questions and customer service agent responses.

The similarity match must be exact or not: it is a Boolean operation and there can be no approximate matches. Approximate matches are discarded. Probable and approximate matching is not carried out at a sentence level but at a construction level. We use the Damerau-Levenshtein edit distance (Lingpipe) [50] to calculate the distance between 2 strings. This method uses the minimal number of edits required to convert one string into the other in order to compute the similarity. Using this method the distance between “Apple” and “Apple” is 0. The distance between “Apple” and “Abple” is 1 because 1 transposition is necessary for both words to be exactly the same. The edit distance between “An” and “Na” is also 1 as a transposition is necessary for a match to occur. This method is extremely useful because it is gives an integer score as the distance between two spellings . Spelling mistakes and typographic errors can render what should be an exact match, inexact. To deal with this we run the Aspell spell checker (Lingpipe) on the data prior to computing string similarity.

The Damerau-Levenstein metric assigns weights to the edit operations. It is defined as:

```
DLD(x,y) = min(i)(\#S(i)*Ws+\#D(i)*Wd+\#I(i)*Wi ) for all possible edit sessions i.
where \#S(I), \#D(i), \# I(i) are the number of substitutions, deletions
and insertions required in the edit session i.
Ws, Wd, Wi are positive numbered weighting factors for the edit operations.
```

The canned responses are stored in a database. The user query is subjected to the Damerau-Levenstein Distance algorithm and is checked against the entries in the database. If an exact match is found, then the selected response is given. At this point the process is over, because a response has been provided to the user. The system now waits for a new user input.

An example of a canned response is:

- User: Hello?
- KIA: Hi, welcome to Norwich Union, how can I help?

This example can be found in almost every single customer service agent-customer conversation. This is deemed to be a common and accurate response, therefore it is considered appropriate to deliver the match as a response. If there is no match, then the query will go through to the next part of the system.

6.5 The construction grammar method

Once the query has been assessed for formulaicity and that there is no match in the canned responses database, the user query is processed through the construction grammar part of the system. This is composed of a syntactic and semantic component. The output of both the semantic and syntactic analysis is a semantic-syntactic frame (SynSem Frame). These construction attributes are fundamental to the construction grammar method, and enable the query to be disambiguated and processed rapidly and effectively.

6.5.1 The syntactic pole

The role of the syntactic pole is to discover the form of the constructions that are present in the user query, and to discover whether they are noun phrases or verb phrases. Constructions can be as large or small as needed, so in this particular experiment we choose noun phrases and verb phrases as constructions, because they were common in our data. Other datasets may work from different structures, using much the same method as is presented here.

6.5.1.1 Parsing

The input is processed through an initial-state annotator. It is then compared to the "truth" which is an annotation specified manually in the corpus. It learns the transformation whose application results in highest score. These are added to the transformation list and the training corpus is updated. To lexicalize the tagger, contextual transformations were added, with reference words as well as part-of-speech tags. The lexicalized transformation corrects mis-tagging and verb contractions such as “don’t”, meaning that the verb is recognized as such and not mis-tagged. If lexicalization is not used, the error rate increases by 11%. The accuracy of the tagger is about 96.9% [24]. The tagger was also improved to be able to tag more accurately unknown words by guessing their most likely tag, which was estimated from a training corpus. On the Penn Treebank tagged Wall Street corpus, the accuracy rate on unknown words was at 85% and the overall accuracy at 96.5% [24]. The K-Best tags can assign more than one tag to a word by making a modification to the contextual transformations. An accuracy of 99.0% was achieved by Brill et al. [24]. A number of extensions were added to the tagger which included lexicalizing the tagger, tagging unknown words and assigning the k-best tags to a word.

There are a number of different tag sets to chose from depending on the task at hand. The tag sets differ according to their degree of granularity. To cope with fine grained tags it is possible to reduce the tag set. The Brown corpus contains 87 tags [68], and the Penn Treebank consists of 46 tags [129]. The Brill tagger has a high accuracy rate and therefore we chose to make use of it in this experiment, together with the Treebank tagged Wall Street corpus.

6.5.1.2 Identifying constructions

The constructions we have decided to work with are noun phrases and verb phrases. We also use these construction types to generate responses. At the query processing stage, we also identify Wh-adverb and Wh-determiner constructions:

- WRB (pos tag) - Wh-adverb - “Why is it expensive”
- WDT (pos tag) - Wh-determiner - “How much is it?”

We use the POS tagged sentence generated from the first stage in the syntactic pole (described above), and store each construction identified in a database called the taxonomy of constructions. Each of these constructions will receive additional attributes that are stored in the database along with their relationships to other constructions in the form of probabilities. The construction attributes are made up of semantic information and syntactic information about the construction. The use of both to determine a response is fundamental to the theory of construction grammar,

6.5.2 The semantic pole

The semantic pole is composed of 4 different elements: word sense disambiguation, stemming, named entity extraction and semantic grouping . Each of these is executed in a particular sequence. This linear method means that the semantic decomposition is more effective. Each of the different stages feeds into the next. Once the entire semantic pole parse has taken place, the information is stored with the relevant constructions. The pairing of form and meaning can be said to happen once we have discovered both semantic and syntactic attributes of the constructions, and stored them appropriately.

6.5.2.1 Word sense disambiguation

The query is put through the word sense disambiguation module. This stage is necessary because it allows us to determine the meaning of a polysemous word in a particular context. Language is highly ambiguous and there can be many senses for a single word. One example is the word ”Green”. According to WordNet [64] it has eight different senses as a noun, one sense as a verb and five senses as an adjective. It is important to determine which sense of a word is meant in a query in order to determine the meaning of the overall input string.

It is possible to use either supervised or unsupervised methods. Supervised methods are very laborious and require a lot of training data which is not always available. They are especially problematic in conversational systems because they are dependent on the amount of experience they have had conversing with humans. It is possible for us to use WordNet as a resource because it is able to give each of the different sense for a word. WordNet is problematic in our case because our domain is so specific (car insurance). We would need to extend WordNet to include words specific to our domain, as described in Clark et al. [36]. In the case of this particular test it is possible as the domain is so limited that we can manually extend the database.

We opt for an unsupervised method and use the SENSEVAL 3 training set [100] to use as training data for out model. We treat the problem of word sense disambiguation as a classification problem. The SENSEVAL collection is freely available and not limited to a specific domain. The score for each lexical item is the sum of weights provided by the system response for each sense that is among the valid answers. The SENSEVAL training data is composed of words and their categories. The test data has 3 variables: the words and the categories, the instance ID for the test case and the contexts which contain the word to be disambiguated and classified, for example:

```
...
<lexelt item="activate.v">
<sense id="38201" source="ws"
       synset="activate actuate energize start stimulate"
       gloss="to initiate action in; make active."/>
<sense id="38202" source="ws"
       synset="activate"
       gloss="in chemistry, to make more reactive, as by heating."/>
```

```
<sense id="38203" source="ws"
       synset="activate assign ready"
       gloss="to assign (a military unit) to active status."/>
...
```

We use a a KNN classifier (Lingpipe) with the parser described above, and cosine distance, to determine what each word in the user input refers to. This method requires stemming, stop-word removal and rendering all of the words to lower-case. We could use the Porter stemmer in this case [156], but because we want to make sure that the correct area of the knowledgebase is isolated, we need to ensure that the words in the utterance are also found in the data, which is in natural language. We use the Uea-Lite stemmer [97] because it is designed for this purpose. The stop-word list has been made up of a frequency count of the most frequent words occurring in our data. Tokens are made up of sequences of non-whitespace characters.

This process means that we are able to determine what the words used in the query refer to in that particular context. Once we have identified these, we can then determine the knowledge area that the user requires, and store related words as semantic attributes in our taxonomy of constructions, against this particular construction.

6.5.2.2 Named entity extraction

The purpose of named entity extraction is to be able to find clearly defined topics. We place our focus on people, locations, car types, insurance products and Organizations. There are no corpora available for the domain of insurance so we constructed one by extracting nouns and noun phrases. We then placed those that were important to us in the taxonomy of constructions database. An insurance product refers to a type of insurance that can be purchased such as ”Young drivers car insurance”.

This allows us to extract references to people, places and locations from the user input. Named entities are strong indicators of sentence topic, and will allow KIA2 to establish context and a relevant response.

We used approximate dictionary-based chunking. This method involves using a dictionary-based chunker and then performing approximate matching. We opted for this approach, because we observed that users were not always familiar with the insurance products and sometimes changed the names slightly such as for example ”new drivers insurance” and “young drivers insurance”. The approximate matching also means that ”3rd party” and ”Third party” are one and the same. Both the exact and approximate match algorithms use a chunker with a dictionary. However, the approximate matching algorithm looks for all matches within a fixed weighted edit distance threshold. The edit distance assigns a value of 0 to matches, -1 to insertions, deletions and substitutions, and ignores transpositions. We manually assign a maximum distance so that we can maintain high accuracy. We chunk the text, extract the character sequence and the set of chunks. We then iterate over them extracting their start and end points, predefined distances, and the matched dictionary entry, which is encoded on the chunk type. The dictionary is a list of car makes and models, proper names and insurance types.

We used LingPipe’s scalable dictionary-based chunker, which uses the Aho-Corasick algorithm [2]. This method is able to find all matches against a dictionary in linear time independent of the number of matches or size of the dictionary.

The process is as follows:

- It creates a dictionary from hard-coded entries (our dictionary files)
- The code builds the dictionary-based chunker
- After the chunkers are created, the rest of the code simply runs through the command-line arguments and calls each chunker on the resulting text in turn.
- Named entities are identified and stored by KIA2

KIA2 will use the named-entities found in the data to locate relevant information in the knowledgebase and/or in the taxonomy of constructions. The information is stored as meta-data:

- "My name is Amy" = user input
- "My name" = NP
- "is" = VP
- "Amy" = NNP - NE (named entity-female)

6.5.2.3 Semantic information

People do not always use exactly the same words to communicate the same thing. We know from our training data that, for example, there are five different ways in which customers asked for price:

- How much is it?
- How expensive is this insurance?
- How cheap can you make it?
- What price can I expect?
- What do I need to pay every month?

In all instances KIA2s answer will be:

- "Im afraid I cant tell you that, but if you call customer service they will give you a quote".

If no direct match was found in the canned responses stage of the system, and that the user query has come through to the semantic pole for analysis, the constructions need semantic information. This is so that we can determine, in this example, that the query is around price. In order to determine that all of these very different user inputs refer to exactly the same thing, we use Wordnet synsets to give us semantic information by using the WordNet API, JAWS [121]. We add hyponyms to nouns and verbs in our user input meta data:

- How expensive is this insurance?

- ”How expensive” = WRB_JJ costly; price
- ”Is” = VP
- ”this insurance” = NP / claim
- ”What price can I expect?”
- ”What price” = NP / price
- ”expect” = VP / hope

In this example, KIA2 can identify that both of the above examples are about price, and will look for constructions matching that semantic information.

6.6 Generating a response

Once the query has been processed by the processes described above, the system has the required information to determine what the user input is about and how it is formed. Response formulation is dependent on query processing because this governs what information is contained in the response and which constructions are deemed appropriate by KIA2. The machine learning engine is used to identify construction matches (parings of form and meaning) in the taxonomy of constructions. We now look at how the machine learning engine is built and run, and also at how the taxonomy of construction is built.

6.6.1 The taxonomy of constructions

Each time that a new construction is found, it is added to the taxonomy of constructions database. The taxonomy of constructions is a repository containing all known constructions together with their accompanying semantic and syntactic attributes. Each construction in this taxonomy also has a probability score associated with it and all other constructions in the database. The supervised machine learning component uses real-world human-human conversation between a customer service advisor and a customer, to populate the database initially. The constructions found in this data are used as learning material for the system. The original taxonomy of constructions is made up of constructions isolated in this data. The probabilities in the taxonomy of constructions, for each construction, change every time a new construction is added to it. Once the system is being used by real users, the machine-human conversations will serve to expand the initial taxonomy of constructions.

The database for the taxonomy of constructions is made up of constructions, that all have a construction ID, semantic attributes (synonyms) and a syntactic type (noun phrase or verb phrase). Please refer to table 6.1 for further information. Each construction has a set of probabilities associated with it, which is a set of relationships:

- weighting 1. The probability of being found in the same sentence as a particular construction (predetermined)
- weighting 2. The probability of coming before or after a particular construction (predetermined)
- weighting 3. The probability that it is part of the response

We use the Naive Bayes algorithm to calculate all of the probabilities for all of the constructions, and we then store them as relationships in the database. We choose to use this algorithm because it is an "independent feature model". The fact that a feature is present is unrelated to the presence of any other features. This is important for us because it means that the various features in the constructions, despite depending on other features, will be treated independently as well. Another strength of this model is that it requires little training data, which would be important in an open domain iteration of this system. We proceed with a supervised model because we have well tagged and referenced data to work with. This method is also very fast which will also be an important factor in an open domain. If there are $(t) training examples and $(a) attributes, the speed of the algorithm is of $S(ta). It is also not sensitive to irrelevant features, and are robust to isolated noise points.

An example of how this is used in our system is the constructions table below which holds an entry for each construction (along with its syn-sem attributes):

ID	Construction	Semantics	Syntactic Type	Weighting 3
1	Young drivers insurance	junior, under 18, cover	NP	0.063
2	will need cover	require, want,insurance	VP	0.048
3	is worthwhile	useful, good	VP	0.012
4	is for under 18's	junior, young	VP	0.058

Table 6.1: Table: Constructions

and the construction links table which contains the relationships between each construction

ID	Construction_to_match_to_ID	Weighting 1	Weighting 2
1	16	0.133	0.021
2	34	0.04	0.023
3	45	0.086	0.016
4	21	0.9	0

Table 6.2: Table: Construction Relationships

The probabilities are calculated and stored as relationships in the database (see 6.2). An example of this is:

Construction (A) ”young drivers insurance” - occurs in 63 sentences out of 1000 sentences of training data. The likelihood of this construction occurring is calculated as:

Pr(A) = 63 /1000 = 0.063

Construction (B) “is worthwhile” occurs in 48 sentences out of 1000 sentences of training data. The likelihood of this construction occurring is:

Pr(B) = 48 / 1000 = 0.048

Construction (C) “needs cover” occurs 12 sentences out of 1000 sentences of training data. The likelihood of this construction occurring is:

Pr(C) = 12 / 1000 = 0.012

Construction (D) “is for under 18’s” occurs in 58 sentences out of 1000 sentences of training data. The likelihood of this construction occurring is:

Pr(D) = 58 / 1000 = 0.058

Now the system calculates the likelihood of (B), (C) and (D) appearing after (A). IN the example of (A) and (B):

Pr(B—A)

```
Out of 48 sentences where (B) has occurred, we see (A) occur in 20 sentences. We
    calculate the probability of (B) occurring after (A) using Bayes rule:

20/48 = 0.416

Pr(B|A) = Pr(A|B) * Pr(B) / Pr(A)

Pr(B) * Pr(A | B)
0.048 * 0.416 = 0.0199

Now divide this by the P(A)

0.0199 / 0.063 = 0.316

A occurs zero times after B in the 48 sentences where B occurs

Pr(A after B | B) = 0 / 48 = 0

Therefore, a zero Naive-Bayes probability will be calculated for this example.

B occurs after A 20 times in the 48 sentences where B occurs
```

```
Pr(B after A | B) = 20 / 48 = 0.416

Pr(A | B) = 0 / 48 = 0

Pr(A | B) = 20 / 48 = 0.416

Construction (C) − ‘‘needs cover’’ is weighted where its probability of occurring at
    all is

12 / 1000 sentences = 0.012

in 12 sentences, it occurs 0 times after A. This construction is now rejected by the
    system, because there is 0 likelihood of the construction ever following the noun
     phrase.

Construction (D) − ‘‘is for under 18’s’’ is weighted where its probability of
    occurring at all is

58 / 1000 times = 0.058

in 58 sentences, it occurs 53 times after A:

53/58 = 0.914

0.063 ∗ 0.914 = 0.058

0.058 / 0.058 = 1
```

Out of a choice between (B), (C) and (D), our system selects (D) as the most likely construction to occur in the response. The response would be:

“Young drivers insurance is for under 18’s.”

6.6.2 The response engine

The training set consists of the conversational data that we have between the customer service agent and the customer. Once the data has been processed through the Naive Bayes algorithm we know which constructions are often found with others, and in what order. When a query comes through this part of the system it is analysed and a set of constructions which are deemed a good match are used as a response. If there is no good match, then the system will flag this and demand supervision by printing “I didn’t find an answer”. We will look at the query and identify any specific issues with it, such as for example the fact that it is outside of the domain. If so we log it as being outside of the domain, but we also consider adding it at a later date if it is re-occurring. If it is, it becomes part of the domain. If it is not, we leave it outside of the scope of this system.

The system will search the taxonomy of constructions to find constructions with matching attributes, that are likely to be part of the response. For example, the following SQL statement will extract all of the constructions that have the attribute ”junior” in the semantic field, the constructions were in a SQL database:

```
SELECT * FROM constructions WHERE semantics LIKE '%junior%'
```

It will then search for the most likely construction to use amongst the selected constructions, which have the required attributes. The named entities, if there are any, are given precedence over other attributes, because they relate to a very specific topic. The system will only look for constructions that are in the correct topic are, as defined by the semantics. This means that the system is faster and more efficient. The syntactic choice (whether to serve a NP or VP), comes from the associated probabilities.

6.7 Continual performance improvement and training

Before the system is ready to be used, it is necessary to train it so that it is familiar with the level or language, the breadth of vocabulary and the knowledge in the particular domain. We obtain a large corpus of real conversations between Aviva customer service agents and customers. These conversation are typed in an IM style program, and saved in text files. Two typical conversation examples look like this:

Conversation 1:

Customer service: Hello, how can I help you today?

Customer: can you please quote on car insurance - we are having problems doing it on web site

Customer service: We can’t quote via Liveperson I’m afraid. This facility is only for advice and guidance.

Customer service: you’ll need to call 08000 929 561

Customer: ok thanks

Customer service: our internet sales team can arrange it for you

Customer service: Is there anything else I can help you with today?

Customer: no thanks

Conversation 2:

Customer service: Hello, how can I help you today?

Customer: I wanted to know how old you needed to be to get young person's car insurance.

Customer service: You will need to be under 25 years old for that.

Customer: ok, thanks!

Customer service: is there anything else I can help you with today?

Customer: No thanks!

Further conversation examples are available in appendix K.The system is designed to continually improve its performance over time, through usage. Every time it finds a user query that it cannot match to a canned response, it processes the query through the construction grammar method. This method allows the system work from constructions rather than a whole sentence, assigning syntactic and semantic information to the constructions. Once they are ready for processing , the system tries to find constructions in the taxonomy of constructions, that might be a good fit for a response. Once it has selected the constructions it wishes to use, those constructions are placed in the order that is most likely, based on past experience and learnings. Any additional, necessary words (usually prepositions) are added, based on prior experience and learnings. The response is then displayed to the user.

If the user is not satisfied with the answer and deems it to be correct but not in decent English, the user can specify a repair (or rectification) to the generated response. This is done by typing: "Wrong: " followed by the correct response. This response is stored by the system in terms of constructions, so the sentence is parsed through the syntactic and semantic poles. If the constructions are new, or even if one of them is, the system will add it to the taxonomy of constructions and all of the weightings in the database are adjusted.

6.8 An example KIA2 system run

In this section, we show an example of how a user input is processed through KIA2, and how a response is obtained and given to the user. During stages (2) to (5), the user input (1), is preprocessed and checked for constructions and named entities. Constructions are obtained in stage (5), and the following stages through to (8) consist of adding the syntactic and semantic attributes to each of the constructions detected. Stage (9) involves querying the taxonomy of constructions against the information detected about the known constructions in the user input. The most likely constructions are selected. Stage (9) involves placing each of the constructions selected for the response from th taxonomy of constructions, in what is to be the response. In stage (10), any additional grammatical rules learnt by the system about these particular constructions following each other are applied. Stage (11) is where the system gives the newly constructed response to the user. In stage (12), the user provides a correction to KIA2. If the user is satisfied, there will be no repair, and the system will stop at stage (11) and await a new user input. If there is a stage (12), it is because the user has provided a repair to the response generated by KIA2. Stage (13) involves KIA2 processing the corrected response and saves the expanded or rectified construction(s) to the taxonomy of constructions.

```
1. User input:  What is young drivers insurance?

2. Parser:

What\_WP
 <: is\_VBZ :>
 ([ young\_JJ drivers\_NNS insurance\_NN ])?\_.

3. No named entities detected.

4. No spelling mistakes.

5. Detecting constructions:

"What" - WP
is = VP
young drivers insurance = NP

 6. word sense disambiguation...

- Stemming taking place \& pre-processing...

- No problems detected for WSD

7. Named entity recognition

- none detected

8. Semantic information stage...

- is = VP / be /
- young drivers insurance = NP / minor; driving; assurance /
```

```
9. Check the taxonomy of constructions to see if there are any suitable constructions to use...

Young drivers insurance has a 100\% likelihood of being in response - pick this
is has a 99\% likelihood of coming after that construction
"under 18s is 98\% likely to come with the aforementioned constructions
 and also 99\% of the time it is third in order.

10. Applying special rules if applicable...

-No special rules detected

11. Generating a response...

Young drivers insurance is under 18s.

12. User correction received...

- Young drivers insurance is {\bf for} under 18s.

13. Save this as a new construction, closely associated with under 18s
  but with a note to use for under 18s in this instance.
```

6.9 Summary

In this section we have described how the KIA2 system is designed. We have described how the taxonomy of constructions is populated initially and after usage, and we have described how it is constructed as a database. We have presented the computational execution of the construction grammar method, and described how using the Naive Bayes algorithm allows us to determine which construction to use in conjunction with other selected constructions. We also described the way in which users can ”repair” incorrect KIA2 responses, and how these are processed and stored in the taxonomy of constructions. The next chapter described our users tests, which allowed us to determine how well KIA2 performed in comparison to KIA1 and Avivabot. We present our statistical analysis of these findings and discuss which SOCS performed better and why users preferred it.

Chapter 7

Evaluation

7.1 Introduction

We chose to carry out a task-based user evaluation, and a user comparative evaluation of our systems KIA and KIA2, and also the Aviva chatbot found on the Aviva website. The greatest difficulty is in obtaining an unbiased evaluation from human based evaluation. Automated evaluation (to chekc for response accuracy) disregards the human aspect of the system so this is also not ideal, even though it gives a more objective result. NLG systems are often evaluated using a task-based approach, allowing for a user assigned score, which shows how well they perform a given task. This kind of evaluation is called a "black box" evaluation, because the system as a whole is tested. This is the most common form of evaluation used for conversational systems. "Glass box" evaluation involves examining each of the modules of the system. It is useful because it allows us to understand where the strengths and weaknesses of the system are but it does not however give us a full understanding of the overall performance. The Turing test is arguably the most famous conversational system test at this time, but as discussed in chapter 1, we determined that our chatbot system goals are different to those evaluated in this test. In this chapter, we present our evaluation methods, describe our experiments, and present our results.

It is necessary to test the systems KIA and KIA2 from different perspectives. We use AvivaBot (the current Aviva website chatbot) in order to produce a comparative evaluation. We already have experience with user testing and our initial study with KIA, described in chapter 2, was very successful and enabled us to advance considerably in our SOGS research. In light of this it made a lot of sense for us to repeat the experiment on a new group of people. Our questions at this time are similar to those we had during KIA testing, seeing as we would like to see how much the current methods improve the overall experience of the chatbot system.

We have researched the Turing test at length, and also participated in this test with our chatbot "John". The test itself did not provide us with useful insights for further chatbot experimentation and research. We found that getting a chatbot to play the imitation game was a different problem than that of building an effective service oriented chatbot system. The Turing test requires us to work in an open domain and to react to often random user input. The service oriented chatbot system requires exceedingly good closed domain knowledge, and must help a user towards a goal. If a machine were to pass the Turing test, this would not mean that it could effectively service a car insurance customer. The opposite is also true: a service oriented chatbot system may very well be able to help a customer get a car insurance quote, but could also easily fail the Turing test. We feel that the value of our research lies in allowing users to get product or service information from a website and be able to move to purchase following chatbot assistance. With this goal in mind, we have devised a human computer interaction test which allows us to know whether this goal has been reached by our system.

We understand that all users tests will be biased unless they are highly prescriptive. We opted to gather raw data which is as close to a ”normal” chatbot experience in the sense that no two users will ask the same thing or behave the same way. The experiments could therefore be highly biased, but we understood this when undertaking the tests.

7.2 Coniam's system evaluation

We were able to use Coniam's system evaluation [40] to compare KIA, KIA2 and AvivaBot. We describe Coniam's experiment in more detail in chapter 1 of this thesis, but the general idea is that trainee teachers evaluated at least one chatbot by conversing with it for 8-10 hours. They then asked the trainee teachers to evaluate the chatbot by assessing it using a check list. The check list is made up of a number of qualities that the chatbot should have. Coniam's test is very useful when trying to gage how different chatbots compare to one another, because it takes into consideration linguistic abilities as well as more general user impressions.

7.2.1 Method:

We selected 3 individuals and asked them to dedicate 3 days to testing chatbots, one per day. We asked them to use the chatbots as often as they could during the allocated time for each, and then evaluate it at the end of the day using Coniam's check list [40]. The users were given a very loose brief, asking them to use each chatbot as it was intended. All 3 chatbots were focused on the area of insurance, so we asked users to only interact with them on this topic where possible. Each chatbot was tested by each of the 3 users for a duration of over 3hrs in total each time. We know this because we asked the evaluators to keep a note of how long they spent testing each chatbot in total. They access the chatbots on the internet from their own computers, and from whatever environment they were in, be it from home or from work or university. This was not a controlled environment experiment, and is therefore more representative of how someone might use any of these chatbots in daily life. The Avivabot was accessed through its web interface on the Aviva website. This interface has been branded and designed to offer a particular experience within the corporate website. Our own interfaces for KIA and KIA2 were intentionally very basic and unbranded, as we did not want to draw attention away from the dialogue itself. In Coniam's test, users are asked to either answer how well the chatbot performed against a criterion on a three point preference scale (Low, Average, Good). In our test, requiring a more quantitative result, we asked users to rate the chatbot between 0 and 10, 10 being the highest score and 0 the lowest. This is the same method used in the other tests that we ran in controlled environments.

7.2.2 Considerations:

In Coniam's original test, the evaluators spent 9 hours conversing with any one chatbot in total. We were not able to find users who were able to dedicate this much of their time to testing each chatbot. We did consider asking them to evaluate one chatbot each, but then we risked strong bias which we wanted to avoid. We therefore asked them to evaluate each chatbot for as long as they could, and dedicate one day to each of them.This resulted in approximately 1 hour of testing per chatbot for each user. The issue here is that we cannot be sure how long users spent exactly with each chatbot.

All of our evaluators had some experience using chatbots and were also from the same demographic, so it is possible that the results would be different if the circumstances of the evaluators were altered. The results of the test are very subjective, however, we argue that service oriented chatbots are designed to provide the user with information in a personable way. Only the user can be the judge of that, so the evaluation should be subjective in nature. We were not able to observe users during their experience with each chatbot, and cannot say whether their experience was influenced by someone else in the room with them, or by a slow internet connection for example.

We asked users to rate each particular chatbot quality between 0 and 10, which is different to Coniam's test, so we cannot compare our results to those of the original Coniam test. By asking for a value between 0 and 10, we asked our evaluator for a finer granularity response. This experiment is also not task-based, which means that users may not have asked of the chatbot the same difficulty or quality of response.

To ensure that we got a meaningful response for each test area, the users were given a score sheet in advance, which means that they were already aware of what they would be looking to assess. In doing this, we could assume that some users may have aimed to test these areas more rigorously than they would have been in a real-world scenario, had they not been given the rating areas in advance. It was felt that getting a response to each section was more important than the bias that this introduced.

We gave each user the other in which they should test the chatbots so that each chatbot came 1st, 2nd and 3rd. This ensured that we did not introduce any additional bias.

When designing the test we balanced quality of testing (i.e. testing for one hour), versus the number of users testing the chatbots. Further work in testing with Coniam would involve testing each chatbot for 15 minutes with 12 users. This would result in the same test time (i.e.108 minutes), but four times the number of users.

7.2.3 Results:

The users were asked to fill in an online form at the end of each day, giving their scores for each of the qualities originally set in the Coniam test. All the user scores can be viewed in tables 7.1, 7.2, and 7.3. The results of the

Coniam test suggest that KIA2 is the best performing chatbot as it has a higher mean, lower standard deviation and lower standard error of the mean. Inversely, Avivabot is the worst performing chatbot based on these test results.

The only area where Avivabot scores better than KIA2 is for its ability to handle incorrectly spelled words; in fact KIA2 is the worst performing chatbot in this area. Users all had degree level education and therefore we could assume that the level of spelling in this sample is good. It is also difficult to identify if the scores of 10 mean they made no errors or that the chatbot performed well in dealing with these errors. The built in spell checker also helps to reduce the importance of this area to the user as they may auto-correct mistyping or other errors.

The standard deviation for Avivabot is equal to its mean which shows the results are heavily weighted far from the mean and that the results in each category are quite extreme. The standard deviation for KIA and KIA2 is relatively high also which shows that although they on average perform better than Avivabot the results are by no means good and in fact these perform rather poorly too.

Due to our sample size some of our results are inconclusive using simple grading of 0-3 as low, 4-6 average, and 7-10 good. We believe some further testing should be carried out for Avivabot in "responding in an appropriate register where it scores low, average and good. The same applies for KIA in the category of topic switches where it again scores low, average and good;these results are therefore inconclusive and deductions without further testing would potentially be inaccurate.

The confidence interval of the mean for each Chatbot is:

- Avivabot: 1.95 - 3.83
- KIA1: 4.31 - 6.13
- KIA2: 5.19 - 6.59

The confidence intervals show that we can be confident that on further sampling the true population mean for Avivabot would still be lower than for KIA or KIA2. However, the results show that if further testing was carried out KIA may perform better than KIA2. Therefore from the above we can conclude that Avivabot is the weakest chatbot but the best performing chatbot is not possible to conclude based on these results.

We carried out T tests, testing each chatbot against the other to prove whether my results were statistically significant. The results of these can be viewed in table 7.1. All the p values for each comparison shows that they are all significant and therefore could not have been due to chance. It is important to note however that Avivabot vs KIA and Avivabot vs KIA2 both show $p = 0$, however KIA vs KIA2 shows $p = 0.04$ and a very small level of ambiguity which supports the point made earlier that KIA and KIA2 should be tested further.

The user scores for Avivabot are as follows:

Coniam results for Avivabot				
Avivabot	**User 1**	**User 2**	**User 3**	**Average**
Handling of incorrectly spelled words	10	10	10	10
Answering correctly phrased questions	2	0	3	1.67
Handling ill-formed questions	2	0	3	1.67
Handling utterances with with incorrect word order	3	2	1	2
Ability to interpret punctuation	2	1	0	1
Handling more than one sentence at a time	1	0	0	0.33
Following the thread of a topic	0	0	1	0.33
Responding in an appropriate register	7	1	4	4
Turn taking ability	2	3	5	3.33
Responding to topic switches (unannounced)	7	4	6	5.67
Providing a follow up to a previous comment	2	3	4	3
Logicality of answers (at random or purposeful)	1	1	3	1.67
Further analysis:				
Sample mean		2.89 where N = 36		
Sample standard deviation		2.89 where variance S^2 = 8.33		
Standard error of the mean		0.48		
Confidence Interval of the Mean @ 95%		0.94 where z = 1.96		

Table 7.1: Coniam Avivabot results

The user scores for KIA are as follows:

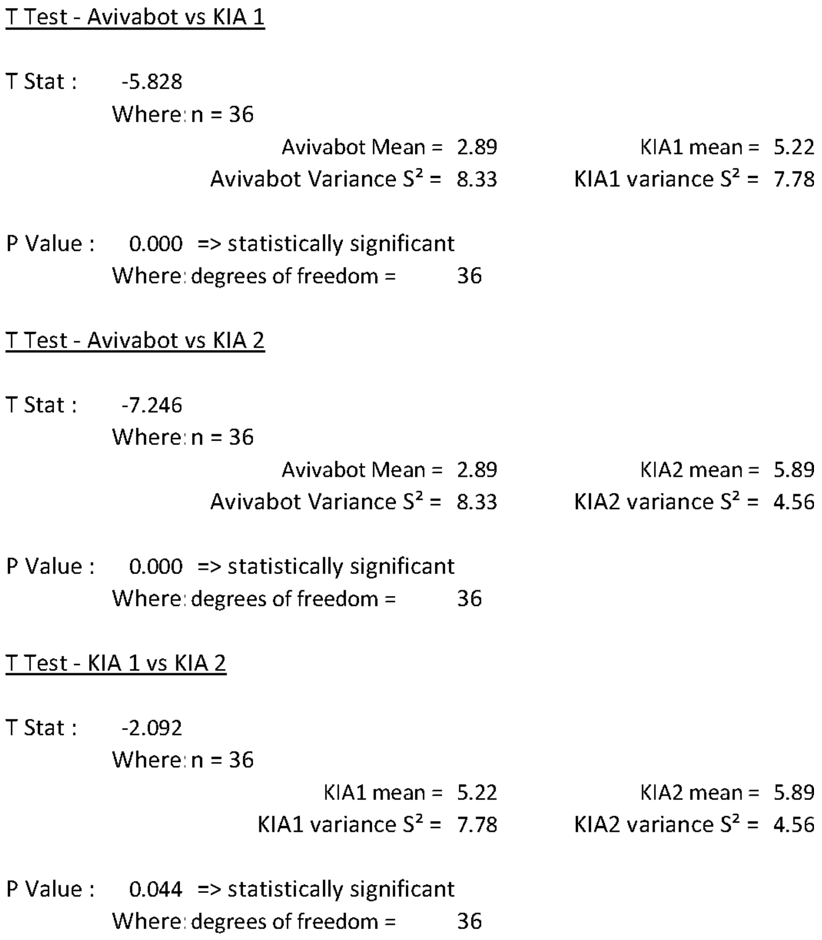
Coniam results for KIA				
KIA	**User 1**	**User 2**	**User 3**	**Average**
Handling of incorrectly spelled words	10	10	10	10
Answering correctly phrased questions	4	7	4	5
Handling ill-formed questions	4	3	5	4
Handling utterances with with incorrect word order	6	6	7	6.33
Ability to interpret punctuation	6	5	6	5.67
Handling more than one sentence at a time	3	0	0	1
Following the thread of a topic	2	0	2	1.33
Responding in an appropriate register	6	7	7	6.67
Turn taking ability	8	10	10	8.33
Responding to topic switches (unannounced)	7	5	3	5
Providing a follow up to a previous comment	4	4	5	4.33
Logicality of answers (at random or purposeful)	4	3	5	4
Further analysis:				
Sample mean		5.22 where N = 36		
Sample standard deviation		2.79 where variance S^2 = 7.78		
Standard error of the mean		0.46		
Confidence Interval of the Mean @ 95%		0.91 where z = 1.96		

Table 7.2: Coniam KIA results

The user scores for KIA2 are as follows:

Coniam results for KIA				
KIA2	**User 1**	**User 2**	**User 3**	**Average**
Handling of incorrectly spelled words	8	6	8	7.33
Answering correctly phrased questions	5	7	5	5.67
Handling ill-formed questions	4	4	4	4.33
Handling utterances with with incorrect word order	6	4	5	5
Ability to interpret punctuation	5	6	6	6.67
Handling more than one sentence at a time	3	3	2	2.67
Following the thread of a topic	1	3	4	2.67
Responding in an appropriate register	8	10	8	8.67
Turn taking ability	8	8	10	8.67
Responding to topic switches (unannounced)	8	7	8	7.67
Providing a follow up to a previous comment	6	8	6	6.67
Logicality of answers (at random or purposeful)	6	5	6	5.67
Further analysis:				
Sample mean		5.889 where N = 36		
Sample standard deviation		2.135 where variance S^2= 4.56		
Standard error of the mean		0.356		
Confidence Interval of the Mean @ 95%		0.70 where z = 1.96		

Table 7.3: Coniam KIA2 results



T Test - Avivabot vs KIA 1

T Stat : -5.828
Where n = 36
Avivabot Mean = 2.89 KIA1 mean = 5.22
Avivabot Variance S^2 = 8.33 KIA1 variance S^2 = 7.78

P Value : 0.000 => statistically significant
Where degrees of freedom = 36

T Test - Avivabot vs KIA 2

T Stat : -7.246
Where n = 36
Avivabot Mean = 2.89 KIA2 mean = 5.89
Avivabot Variance S^2 = 8.33 KIA2 variance S^2 = 4.56

P Value : 0.000 => statistically significant
Where degrees of freedom = 36

T Test - KIA 1 vs KIA 2

T Stat : -2.092
Where n = 36
KIA1 mean = 5.22 KIA2 mean = 5.89
KIA1 variance S^2 = 7.78 KIA2 variance S^2 = 4.56

P Value : 0.044 => statistically significant
Where degrees of freedom = 36

Figure 7.1: Coniam T-Test Results

7.3 Human-computer interaction evaluation

The user testing that we carried out during the KIA experiments (described in chapter 2), helped us to understand what was important to users and helped to direct our research towards the most important user needs. We researched and built the KIA2 system in response to the findings in the earlier system KIA. We carried out substantial user testing on KIA2, which we will describe in detail in this section.

The Coniam test allowed us to compare 3 chatbots and understand which one of the three our users preferred. We do not, however, feel that the Coniam test was complete enough for our needs. We wanted to gain a clearer picture of how users interacted with the chatbot, how this interaction differed according to the quality of the responses and types of responses, and we wanted to appreciate what we could improve upon in future system iterations. For these reasons, we carried out a task-based user test in a controlled environment where we could observe users.

7.3.1 Method:

In his section we present the method that we used to carry out this user test, and we describe the details that we paid attention to in order to ensure that this test was as rigorous as possible.

7.3.1.1 Pre-test questionnaire

42 users volunteered to take part in our test. They were all purposely selected, so as to cover a broad demographic. We were particularly keen to understand how well our system performed for all types of users, so as to be representative of the online traffic that a large insurance website might attract. In order to ensure that tested with a broad enough demographic, we asked users to first of all fill in a preliminary test questionnaire. The pre-test form can be found in appendix G. This test consisted of the following questions:

- Occupation (to determine if they use computers as part of their job)
- Gender
- How often do you use computers?
- How often do you use the web?
- What do you use the web for?
- How often do you search for information using search engines?
- Have you ever used a chatbot?
- If so, which chatbots have you used?
- What was your experience of these?
- How do you normally purchase car insurance?

Our 42 testers came from a broad spectrum of occupations and backgrounds, from car mechanics to school teachers and scientists. Their level of familiarity with chatbot systems and with the web in general was important to us, as it could affect their evaluation. We found that users who had previously used a chatbot had a self-professed bias, because they had certain expectations and would compare the test system with previous experiences they had had outside of our test. Most of our users had never had interaction with any kind of chatbot, and it could be argued that their responses were more objective. Not knowing what exactly to expect can however lead to confusion, because as we saw in the KIA tests, new users tend to revert to using the chatbot like a search engine at times, by entering in keywords rather than full natural language sentences. This leads to poor chatbot responses, because it is not designed for that sort of input. The responses from this part of the test is available in appendix H

7.3.1.2 The testing environment

After having responded to the preliminary questionnaire, we asked users to come through to a room, which has a table, a chair and laptop inside. The reason for sparse surroundings is to be able to ensure that the user is fully focused on the test, and that there are no distractions which could affect our results. Such distractions include hearing others talking about their test experience or their opinion of chatbots for example, which might influence the user. Other distractions include feeling pressured to finish the test quickly because there is a queue, and people are anxious to have a turn. We were very keen to ensure that each user was able to take as much time as they required to go through the the tasks and that they were not influenced by anything outside of the test. One researcher stayed with the user during the test, to observe any particularities and also to help the user should s/he have any problems. There were strict guidelines ensuring that the researcher did not help the user at any time during the testing process.

7.3.1.3 The chatbot setup

The laptop was loaded with the KIA2 chatbot, the KIA chatbot. These two chatbots used exactly the same interface, which was, as can be seen in the screenshot in figure 7.1. It was purposely made to be as basic as possible, so as to not detract the user from the actually conversation. We appreciate that the interface does play an important role in the interaction between humans and machines, however, we did not want to test the interface, but focus on the typed conversation only. Users were shown how to use the interface. They were asked to type their input into the box at the bottom and click the "Say" button when they were ready to submit their typed sentence. Their response would appear in the large box above.The chatbot response would also appear in the large box, but in a different colour in order to avoid confusion over who had said what. The entire conversation was saved in a text file on the laptop, together with timestamps for each submission from either the chatbot or the user. Avivabot is only accessible through the Aviva website, so we had no control over the interface here. The way in which the user interacts with Avivabot is however practically identical to how the user interacted with KIA and KIA2.

7.3.1.4 The task-based user test

The test itself, just as in our original KIA test, is task based. The user received instructions prompting him/her to request specific information from the system. Each task is timed so that we can determine which system allows users to get the most complete answers the quickest. The task prompts are scenario based so as not to influence the users' language. If we were to simply say "Ask the chatbot how you need to pay for insurance", the user may use the same words. For example s/he might say as a result "How do I need to pay for insurance" rather than using their won words which migh result in "So how does one purchase car insurance then?". By placing the user in a scenario, we describe a situation to them, leaving them free to express themselves in their own words. The 6 tasks selected were real world scenarios, found in the Aviva customer service logs that we had access to. The task list given to test users can be found in appendix I. The six tasks we chose are the following:

- You play the role of an 18 year old who has just brought their first car and want to find out about car insurance. You might want to find out whether there is a type of insurance aimed specifically at them, ask about the price, the level of cover, how you can save some money. You may also want to compare certain types of insurance to find out more about different aspects of each one and its relative benefits.

- You play yourself for this one. Imagine that you are interested in Aviva's car insurance but don't know whether to choose comprehensive or third party. Find out which one would work best for you.
- Imagine that your car has broken down, and that you have Aviva's comprehensive car insurance. You'd like to know if you can get a courtesy car while yours is being fixed up.
- Find out how you go about making a claim on your car insurance.
- You're unhappy because the cost of your insurance has gone up since last year. Find out why this is.
- You want to know how to pay for your car insurance

These tasks were specifically chosen to highlight and encourage certain behaviours. We wanted to test the performance of the chatbot, but also the reaction of the user to the chatbot. We did this by making the scenarios very specific, so that would prompt certain behaviours and reactions:

The first task is taken directly from the KIA evaluations that we carried out previously. It served as a benchmark for this second large scale user test. We were using KIA once again, but with a different set of users. We wanted to have a task to compare the test on KIA that we had carried out 3 years previous to this one. People's internet and web habits have changed in this short periods of time, and we wanted to test for any large discrepancies.

The second task lead the user to try and use the chatbot to help them make a decision. There are lots of variables at play that could influence their decision, so there are a lot of questions available for them to ask. This task allows us to observe user language and behaviour. It also allows us to test that the chatbot is accessing the knowledgebase properly and determining the topic of each question.

The third task allowed us to observe how responsive the chatbot is to human emotion. This task leads the user to put themselves in a position where they need assistance, and which is an emotional situation. We wanted to be able to test how well the chatbot adapts to the user behaviour, if at all, and if the users' language changed.

The forth question required the chatbot to ask a series of questions in order to be able to give an answer. This allows us to observe how well the chatbot is able to "drill down" and give a correct, more precise response each time.

The fifth task is also about emotionality. The user could be angry or upset, and the type of language used gave us insight as to whether the chatbot was responding well enough for the user to feel that it was engaging with them.

The sixth task was designed to prompt for a quick answer. It allowed us to ensure that the knowledgebase was complete enough, and that the chatbot was able to give a series of answers that are all relevant. There are a variety of answers for this question, so the chatbot was required to adapt to the user feedback.

None of the chatbots are able to keep track of the conversation effectively enough so tying in all of these tasks into a full conversation was not useful. We do feel however that going forwards, this is functionality that we would like to explore.

We asked users to "talk out loud" during the test so that we could head their cognitive process as it took place as well as seeing them interact with the SOCS. An example would be the following note, collected from this very test:

User says: So...now I'm going to ask it about my problem. Yes. The cost is going up. User types: Why is my insurance getting more expensive?

Here we see that the user has acknowledge that this is a "problem", that they know the "cost is going up", yet when they type this into the input box, they use different language to their spoken language. This is an interesting phenomena that we saw repeatedly in users of all types and that we intend to report on at a later date, due to the fact that it doesn't directly inform on the performance of the SOCS created for this thesis.

7.3.1.5 The user evaluation procedure

At the end of the test with each chatbot, the user was also asked to provide written feedback. The post-test questionnaire is the same that we used for the KIA test and is carefully constructed so as to allow for users to express themselves fully on all of the different aspects concerning the chatbot. We asked users to rate their experience based on the following scoring method: 0 for poor; 1 for average; 2 for good. The raw results can be viewed in appendix L. Each of the following areas we rated:

- Please describe your experience with the chatbot
- Were the answers useful?
- Did you find the tone of the responses adequate?
- Do you recall any answers out of the ordinary/not expected?
- Did you find that the conversation turn-taking worked well?
- Compared to scanning the web site, how readily available was the information?
- Do you think that being provided with the relevant links to the location of the information was useful?
- On a rating of 1-2, how trustworthy did you find the chatbot?
- What degree of emotional response did you detect from the chatbot?
- How did you feel about the quality of the chatbot in general?
- Which word best describes your conversation with the chatbot? (Smooth, Easy, Difficult, Awkward, Another?)
- Did the chatbot appear to show an interest in what you were talking about?
- Were the answers succinct or long-winded?
- Were the answers too simple or too complex?
- Was the information given clear?
- Do you think that the agent would benefit from a more human-like virtual person with facial expressions? If so why?
- Would you use this chatbot again for this type of task?
- What would you change?

7.3.2 Considerations:

In user tests of this size that are carried out in controlled environments (i.e.not on a mobile device in a train station or in a users' own home), there are a large number of things to consider before analysing the results. Any test that is user-based faces the chance that results have been influenced in some way, but we did cover as much of this as possible when designing the test. Here we will mention the possibilities around bias, but also the steps that we took to eliminate it as much as possible.

User ratings: When we asked users to rate their experiences as 0, 1, or 2 (or poor, average and good), we have not left any room for the user to give a more granular score (decimal). This means that a user could have had a lower than average experience (i.e. 0.9), but the chatbot received a score of 0. However, using the scoring system that we have allows for unambiguous results for analysis.

7.3.2.1 User fatigue

We needed to take into consideration user fatigue, because the test took on average an hour to complete. We asked users to assess KIA, KIA2 and Avivabot during the same session. This means that users were required to go through a total of 6 tasks with 3 different chatbots, resulting in 18 total conversations. As a result, we made this test was shorter than the one we originally ran on KIA (only 6 tasks).

7.3.2.2 Bias:

We wanted to ensure that no system was at a disadvantage, so we alternated the order of the chatbots to attempt to address this. Each chatbot came first, second and third 14 times each. Additionally, allowing users to score each chatbot one after the other, there is increased likelihood that they will compare each chatbot experience. The first chatbot, in this case, would suffer less bias, although it would still be subject to previous user experiences. Another area of bias is that when users have no opinion, they are still forced to give a score. From conversations with users at the end of the experiment, people commented that they gave a score of 1 when they had no opinion in categories such as ”interest shown”. In future, we would like to allow users to opt out of giving ratings by changing the scoring to: 0 for ”no opinion”; 1 for ”poor”; 2 for ”average”; 3 for ”good”. Some of the ratings we asked for can be considered too vague. For example ”quality” is very subjective and could be interpreted in terms of visual appeal or information given.

7.3.2.3 Demographics:

In order to ensure that the tests we ran on the chatbots were demonstrative of the type of audience they would normally be used by on a popular commercial website, we tried to capture people from a wide range of demographic groups.

We can see from figure 7.2 that the users were from a wide range of professions, but there were a disproportionally large number of developers, marketers and artists. We would have preferred to have more users representing other professions. The reason for the bias is due to our immediate available pool of volunteers being in our social networks to being with. We were able to include ”stay at home” mums as well as gardeners, chefs and other professions that do not make heavy use of computers. Various different demographic groups will have different requirements and different people will also have different learning styles. One might expect an artist to be more sensitive to visual stimulus then perhaps a manager for example. We did not know if this would influence the chatbot experience, so we wanted to ensure that we could at least verify this.

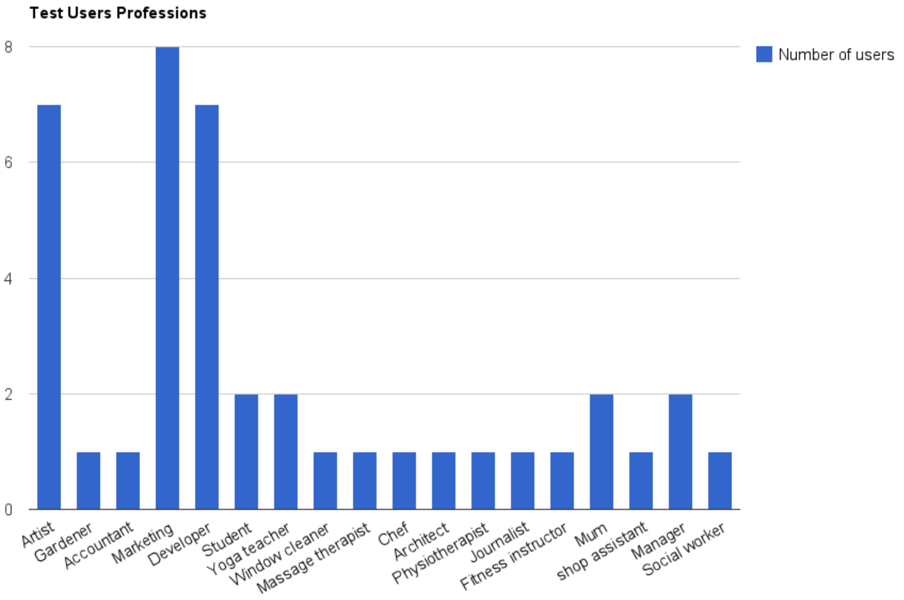


Figure 7.2: User professions

The computer habits and web habits of our user base was very important to us. For many users, accepting the fact that they are talking to a machine, is quite a leap. Their computer habits also say a lot about whether they are familiar with online interaction of this type. For example, they may be familiar with instant messaging. If they are not familiar with this online communication mechanic, there is yet another novelty and potential learning curve for these users. This would affect their interaction with the SOCS. For the sake of this user experiment, we chose to only use people who had some experience with computers and the web. We can see in figure 7.4, that most of our users spend over 8 hours a day on their computers and that of that time, almost the exact same amount is spent on the web. As we can see in figure 7.3, their search engine use however is limited to around 1-3 hours a day, which means that they are not really seeking information most of the time that they spend on the web. Figure 7.5 shows what our user base used computer for on a daily basis.

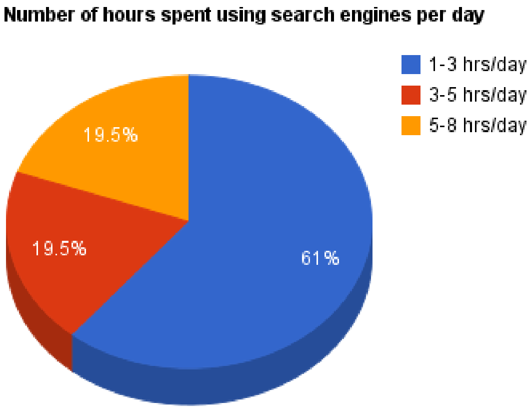


Figure 7.3: Hours spent on search engines

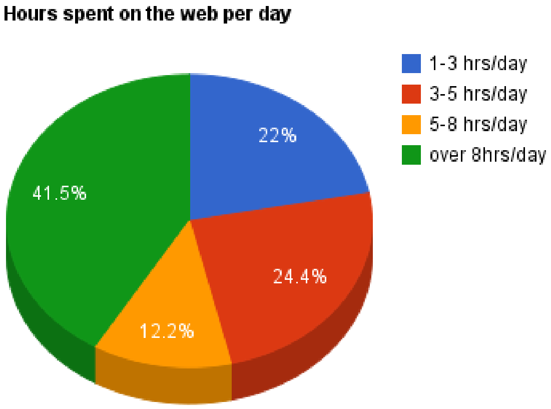


Figure 7.4: Hours spent on the web

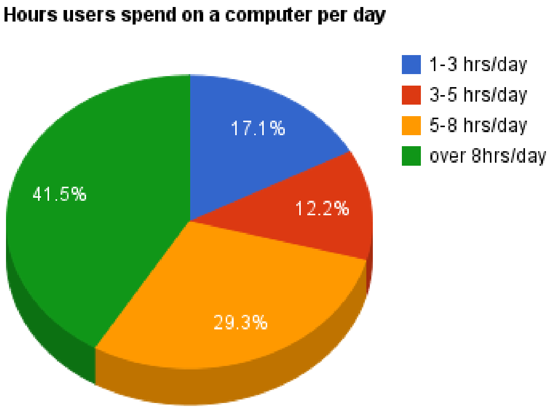


Figure 7.5: User computer habits

As we can see in figure 7.6 only 34.1 percent of users had ever used a chatbot before, which means that most of our user base were first time users. We didn't know biased experienced chatbot users would be, so we wanted to ensure that we had a good number of chatbot novices, as well as some more experienced users.

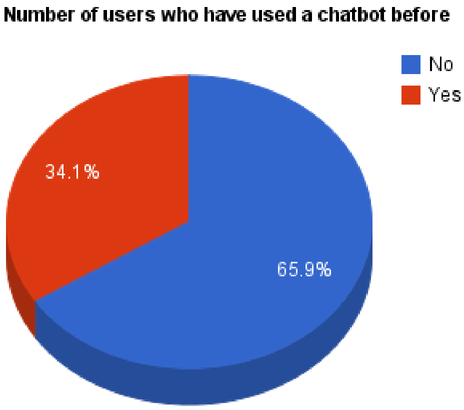


Figure 7.6: Users who used chatbots previously

Most of the users in the test base had also purchased their insurance through a website in the past and/or by telephone, as can be seen in figure 7.7. This means that these users are already familiar with the concept of buying insurance online, and have a previous experience with which to compare the one they encounter in the chatbot tests.

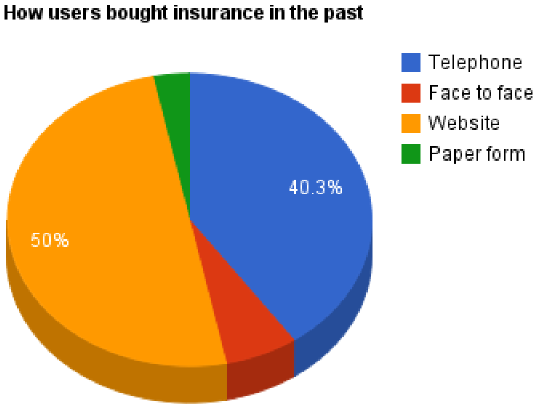


Figure 7.7: How users had previously purchased insurance

Ideally we would have liked to carry out tests on a much broader demographic in order to ensure that the results of the tests are as representative of society as a whole as possible. We also did not have any user above the age of 53, which leaves out a certain percentage of the population on a car insurance website. This will be particularly important when designing a suitable interface. Having a broad representation of the actual website user base is important, as expectations vary according to experience. Having a broad user test base, and as large a test base as possible, enables us to limit the bias that typically emerges from user testing. It is however difficult to recruit a large enough test base, but it is possible to hire people to take part. The latter option is preferable to a lean user base, because the results will not be representative enough of the types of users one can expect on a large commercial website. In addition, it is important to note that different websites will need different types of users in the user base.

7.3.2.4 Controlled environment testing:

We asked users to carry out this test in a way that does not exactly replicate the experience they may have at home, or in their usual environment. The laptop that we used was a PC, and some users are Mac users. This meant that there was a little adjustment period where they adapted to a slightly different human-computer interaction. The users were also being watched, which could have led them to feel unsettled. We also asked them to talk out loud, which may not have been natural for many of them. They were also asked to work with 6 scenarios, which is also not natural.

7.3.2.5 Chatbot Interfaces:

It is important to note that KIA and KIA2 had the same interface. Avivabot however is accessed through the interface designed for the website. It features a cartoon character as an avatar, and offers links in a separate box, that are related to the topic of conversation. The difference in interface is not ideal, as it means that we cannot be completely sure that users have not been influenced in some way by the aesthetics or the interaction.

7.3.3 Results

We processed a total of 42 users through each of the 3 chatbots (KIA, KIA2 and Avivabot). The results of this test show that Avivabot was again the worst performing chatbot in terms of the average feedback score. The results for the standard deviation are important as they show that Avivabot and KIA1s feedback vary widely as these are higher than the mean. The standard deviation for KIA2 is lower than the mean but only marginally and therefore also shows a large variance in the performance of the chatbots. However, all 3 chatbots performed poorly based on the scale of 0 = bad response, 1 = average response, and 2 = good response; as all averages are below 1. Furthermore, there is no category in which any of the chatbots scored on average more than 1 which shows that none of the responses are all chatbots were rather lacking in performance.

It is interesting to note that although KIA has a lower average than KIA2, KIA would be on average most likely to be used again. We can also conclusively say that Avivabot will not be used again.

Some of the results are odd, for example KIA is most likely to be used again however it scores the worse on useful answers. KIA does however score the highest in useful links whereas KIA2 scores highest on useful answers, better than site surfing and quality, emotion and tone but still the results indicate that the users would use KIA more than KIA2. An area for further testing would be to compare a chatbot which has very good emotion, tone, turn taking, and interest but gives 50% of the correct answers versus a chatbot which is weak on all those areas but gives 100% correct answers. These results seem to suggest that maybe the least correct chatbot would be the one used again.

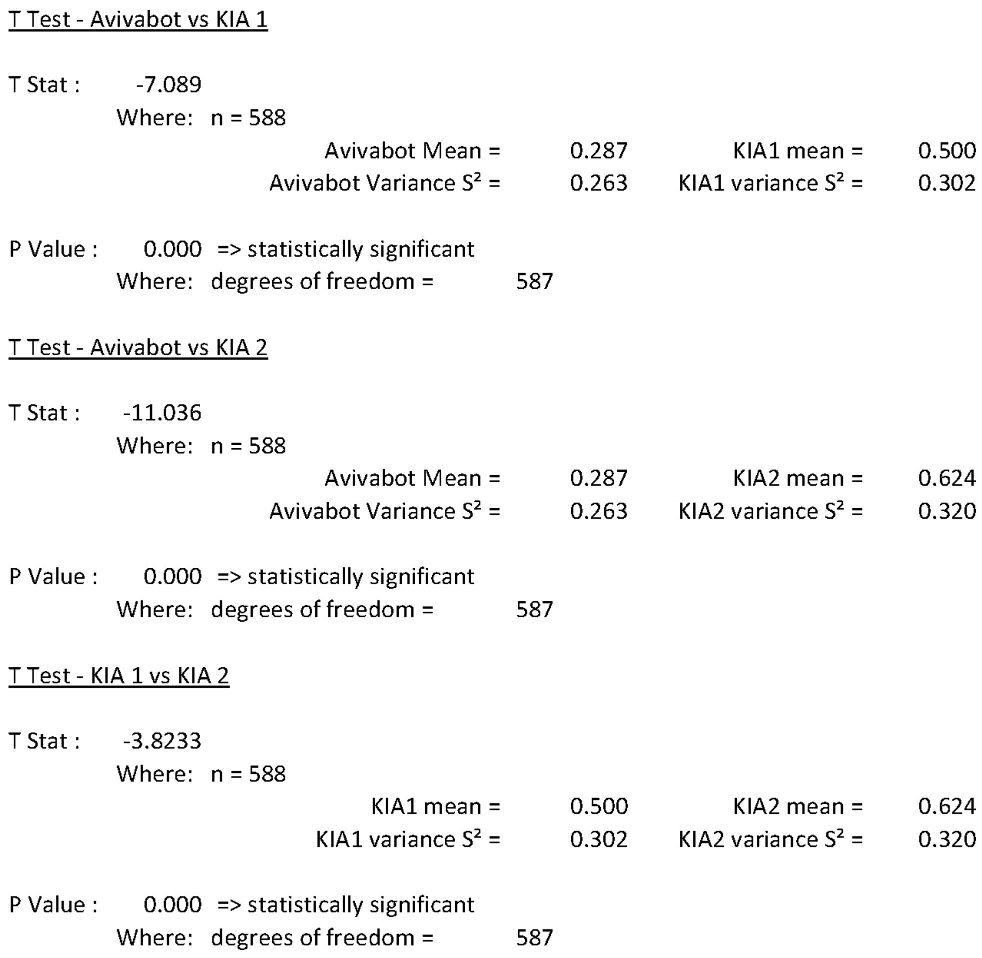
When comparing the different users responses to the same area there are few irregularities with most responses showing 0 or 1, and a maximum of 11 out of 42 users (26%) making a 2 comment on any one area.

The confidence interval of the mean for each Chatbot is:

- Avivabot 0.25 – 0.33
- KIA1 0.46 – 0.54
- KIA2 0.58 – 0.67

This shows that we can be confident that on further sampling the true population mean for all chatbots would still show the same conclusion i.e: KIA2 would be the best performing followed by KIA1 and lastly Avivabot. We can also conclude that none of the results would push the chatbot into the good performance area as all means are still under 1.

We carried out T tests, testing each chatbot against the other to prove whether my results were statistically significant. The results of the T tests can be viewed in 7.8. All the p values for each comparison shows that they are all significant as all p value are 0 and therefore could not have been due to chance.



T Test - Avivabot vs KIA 1

T Stat : -7.089
Where: n = 588
Avivabot Mean = 0.287 KIA1 mean = 0.500
Avivabot Variance S^2 = 0.263 KIA1 variance S^2 = 0.302

P Value : 0.000 => statistically significant
Where: degrees of freedom = 587

T Test - Avivabot vs KIA 2

T Stat : -11.036
Where: n = 588
Avivabot Mean = 0.287 KIA2 mean = 0.624
Avivabot Variance S^2 = 0.263 KIA2 variance S^2 = 0.320

P Value : 0.000 => statistically significant
Where: degrees of freedom = 587

T Test - KIA 1 vs KIA 2

T Stat : -3.8233
Where: n = 588
KIA1 mean = 0.500 KIA2 mean = 0.624
KIA1 variance S^2 = 0.302 KIA2 variance S^2 = 0.320

P Value : 0.000 => statistically significant
Where: degrees of freedom = 587

Figure 7.8: HCI T-Test Results

7.3.3.1 User frustration threshold:

KIA2 allowed users to complete the entire 6 tasks we set them, but AvivaBot unfortunately did not. All of our users failed to finish all six tasks, because they found the quality of the chatbot too poor to allow them to do so. This in itself is a telling result.

All 3 chatbots were unable to answer at every required moment, and this led to frustrated users and/or users who simply dropped the task and moved on to the next one. It's clear that adding an avatar to the interface adds little to the overall experience, if this is already poor. Avivabot is the perfect example for this. Although some of our users thought that it might be a fun addition, none thought that it would improve the experience noticeably.

7.3.3.2 Previous use of chatbots:

We found that in every case that someone had used a chatbot before, it affected their responses and behaviour, as they already knew what to expect. All of those with previous chatbot experience tried to trick the chatbot or cause it to fail. They also all compared it to their previous experiences with chatbots. We found that novice chatbot users showed more realistic behaviour than those users with previous experience, because they genuinely tried to find answers to questions and conversed with it much like they would a human.

7.3.3.3 Demographic effect:

We did not find that professions had a large effect on the results of the test, but users such as developer who are used to testing systems were more critical than others.

We wanted to ensure that we had users that were very web savvy and others that only used the web sparingly. Interestingly, we found that their level of web usage did not affect their evaluation.

7.3.3.4 User feedback:

It is clear that KIA2 is the favoured chatbot for users, but not by a great margin compared to KIA. AvivaBot is reported by users in our experiment, to perform very badly, and not a single user would want to have to use it again. We can see that users tend to find links more useful the worse the quality of the chatbot and their experience of it is. All of the users in this study would have found using a very accurate site search instead of a not so accurate chatbot better.

7.3.3.5 Emotion:

The level of emotive responses that users perceive the chatbot to have appears to rely solely on the type of language used and the chatbot's ability to sympathise with them. This is a powerful attribute, because users feel more engaged with a chatbot that they think is responsive emotionally. Some users even feel concern for the chatbot's feelings, and they ask "Are you feeling ok?" or "Are you ok?". When the responses are not correct, as long as the chatbot can apologise, even if it never manages to give an answer, they report having had a good experience.

7.3.3.6 Unexpected events:

When unexpected events happen in the conversation, such as for example a complete topic change which is not initiated by the user, they show surprise, but quickly tell the chatbot that they want to talk about a different topic. As long as they are able to get back on topic, they do not appear to be hindered by the sudden topic change. If the chatbot is unable to return to the topic or gives a completely out of context response, users expressed feelings of frustration and disappointment. Once users reach this point, it is difficult to retain their attention further.

7.3.4 Comments and conclusions:

Overall users found that KIA produced the clearest, most succinct and simplest answers. In contrast, they found that KIA2 gave by far the most useful answers, although it produced unexpected answers more often than the other two chatbots. KIA2 was also found to be the most trustworthy of the three chatbots, despite producing odd answers periodically that were not on topic or grammatically incorrect. AvivaBot was too frustrating for users to complete the full chatbot assessment. The length of the conversations is very short for AvivaBot, and we see that users move on to the next question immediately after running into problems. Users persevere and attempt to gain a response form KIA and KIA2 at ever opportunity, so the conversations are longer. In fact users can be seen to attempt to help to chatbot and even compromise with it, as long as it is perceived to still be engaged with them.

7.4 Summary

There are currently no automated tests for chatbot testing. The lack of such testing means that we need to rely solely on human testing, which is very time consuming and also sometimes expensive. Humans are the reason that chatbots are built to begin with, so it is not unreasonable to make great use of this type of testing. The problem remains that it is always subjective to some degree. We have tested with what is considered a large test group in comparison to other tests that were run on these systems, but it would be better to have an even larger test group, especially for the Coniam test. We were very mindful to use a user test group from varied demographics, because their experience with computers, the internet and these types of systems all influence their behaviours and choices. We do not see this mentioned in other tests, which tend to depend on groups from the same demographic, often academics and mostly between 20 and 30 years of age. This leads to biased results, not representative of the types of users that will use the system on a large and popular commercial website.

In both our tests, Avivabot is the worst performing system out of the three. The difficulty lies in comparing KIA and KIA2 as both tests showed similar results for both. Further testing should be carried out on both these systems to identify the better performing SOCS. This means that our results show that using CxG or a vector-based approach, makes little difference to the user experience. It is important to note however, that none of the SOCS tested, perform particularly well. The highest scores on both tests are either below average or only very slightly higher.

Chapter 8

Conclusions and future work

In this final chapter, we review the objectives of the thesis, and show how the work presented in this thesis has met these objectives. We also make suggestions for future research directions that could provide the next steps to an even more efficient service oriented chatbot system.

8.1 Review of thesis objectives

The main goal of our thesis was to improve current service oriented chatbot technology and performance.

We did this by using construction grammar for user utterance processing and disambiguation as well as for machine response processing and delivery. We established through testing, that user behaviour in reaction to machine behaviour was of importance, so we also set out to study the phenomenon in greater detail and created a better chatbot through a better user experience.

8.2 Summary of the research in this thesis

In Chapter 1, we reviewed current chatbot technology and found some shortcomings that would need to be investigated further if we were to deliver an improvement. We identified the the main shortcomings as:

- The systems can't generate their own sentences
- The systems can't understand the user input
- The systems rely on time consuming labour intensive methods such as populating ready-made templates
- The systems do not produce sufficiently human-like behaviour for users to engage with them fully
- The storage and acquisition of knowledge is insufficient in current systems and affects chatbot performance

As part of the large literature survey of chatbot architectures, we discovered a number of areas of opportunity specifically in natural language understanding and natural language generation, that we explored in much greater depth in chapter 3.

In chapter 2, we experimented with existing methods and created 2 systems, John and KIA. We tsted these 2 systems and we discovered interesting user needs that affect the perception of the system:

- Users benchmark of what to expect in conversations is the human-human conversation standard
- Griece's maxims apply to conversations between human and machine as much as they apply to conversations between human and human
- Users expect a fluid and dynamic conversation with the chatbot
- Users are prepared to make concessions for mistakes and errors if the chatbot is able to show that it is aware of the problem

We also found that the following computational issues exist:

- Word level natural language generation and understanding techniques are very error prone
- Databases are not flexible enough to store the data and the multidimensional relationships between knowledge elements
- Templated methods are inflexible, time consuming, expensive and do not perform well enough for users

In Chapter 3, we researched Natural language understanding and generation systems that have been developed to date. It is clear from the research available that solutions to the problems that we encountered are very difficult to resolve. The research we gathered on this topic allowed us to understand which methods were worth investigating further and which ones held little promise. The main problems that researchers identified were:

- Natural language generation systems do not perform well on unstructured data
- User input is sometimes incomplete, unclear or incorrect and current systems often fail with this
- Current systems struggle with conversation flow and topic thread
- Adding sufficient world-knowledge to the system is difficult
- The complexity in the English language is such that automating useful responses is very difficult to achieve
- Current systems do not adapt well to changing user language

The systems that we looked at predominantly work in a closed-domain. Some were able to work adequately well for the user, however the problem is that websites and domains are consonantly expanding. In order to deliver a solution viable for such domains, we need them to rely on automated means of knowledgebase expansions and language learning, rather than human involvement.

In chapter 4, we were able to explore some of the opportunities offered by construction grammar and experiment with the method in computational terms. Fluid construction grammar proved interesting from the point of view that we could parse input and produce output using the same method, an improvement on the two or more step processes of existing systems. The method however, was as yet still too novel and underdevelopment for use in our system. We encountered significant problems when trying to apply the bi-directional rules. We found however that the method was worthy of note, and that it would be interesting to further investigate this and mature the method so that it could be used in SOCS. We did use the more common CxG method, and were able to create a computational model of it and apply it to our SOCS system. The CxG method has allowed us to incorporate both form and meaning into utterance analysis, and to treat each utterance in the context of the real world rather than in isolation. The "taxonomy of constructions" contains a structured inventory of known language conventions for our SOCS. Adding semantic, syntactic and pragmatic meaning to each of the constructions usually found in this domain has allowed us to experiment on a more complete level than simply a word a level, such as is offered by other types of grammar.

In chapter 5, we described the architecture of the system KIA2 and explained how the implementation of the CxG has been achieved computationally. We chose to use our own stemmer, UEA-Lite. We opted for a machine learning method and we used the customer service data from Aviva to identify re-occurring constructions and train the system. The interface was very basic in the sense that we didn't spend time developing it. This is an area that we would like to look to improving in future iterations of the system.

In Chapter 6, we tested and evaluated all three SOCS (Avivabot, KIA and KIA2). We drew conclusions from our results, and discussed the validity of our chosen tests. Avivabot was consistently the worst performing chatbot and was statistically proven to be so. KIA2 was shown to be marginally better than KIA in both tests, however our recommendation is that further testing is carried out to ascertain which chatbot offer a better user experience. Our results show that the user experience is not significantly affected by using a CxG approach.

The aim of our thesis was to improve current SOCS and to research the problems that these types of systems face. We have pinpointed the biggest issues, researched them at length, researched possible solutions to these problems and then tested these solutions by implementing them in our systems. We then evaluated our work and were able to determine that we have contributed to the field by identifying methods that did not work well and some that did show promise. This work serves as the basis for future research in SOCS.

8.3 Further work

There are several areas that we have identified and even begun looking at but that are however out of the scope of this thesis. We would like to investigate these areas in future and feel that they will significantly improve the current system and add to the body of research in this area.

8.3.1 Interface

Designing an interface that is useful and easy for users of all types to use is of significance in this area of research. We would need to carry out significant user testing on a variety of interfaces. We would need to carry out usability testing to ensure that the interface is easy to use, and allows for enhanced functionality to be added to the system. For example, we may in time allow for voice led conversations as an alternative to the typed conversation. We would also like to make the interface accessible to all. Currently the W3C Web Accessibility Initiative does not have any guidelines specific to chatbot interfaces. This is an area that we feel would benefit from our contribution, and the contribution of the research community.

8.3.2 OWL ontologies for knowledgebase engineering

Knowledge engineering is an area of concern for SOCS because there needs to be some kind of intuitive way of storing information belonging to the domain and also to store the constructions. We investigated using of an OWL ontology to allow the system to verify constraints and alternatives in the domain and in the language as well. OWL is a semantic markup language used to share ontologies on the web [135]. W3C described RDF (Resource Description Framework) as "a standard model for data interchange on the Web" [79]. It is derived from the DAML and OIL Web Ontology Language. There are two subsets of the OWL language, OWL Full, OWL Lite (containing just a functional subset of the OWL Full) and OWL DL (where DL stands for "Description Logic"). OWL DL supports the existing Description Logic business segment and has good computational properties for reasoning systems. These are the reasons why we researched the use of OWL DL. In OWL ontologies, information is organised in a hierarchical manner so that it can be parsed and understood by software. In such a system, one ontology could contain all of the domain knowledge while the other could contain all of the constructions commonly found in this domain. The advantage of using OWL is that many different types of knowledge from different domains could easily be absorbed by the system, whereas databases and spreadsheets do not offer the same flexibility. Although we did not construct an OWL ontology for the KIA2 system, our initial research shows that this kind of technology would be potentially very useful.

8.3.3 Integrating voice technology

We would like to investigate the integration of text to speech technology in order to provide a different type of access to the chatbot, and a different kind of way of interacting with it. A voice activated chatbot would be useful to those for whom typing is not an option, and also for those who would like a more human way of communicating.

8.3.4 Topic tracking

Keeping track of the topic is something that we acknowledge is not taken into account in the systems that we built. We would like to research the area of anaphora and related fields in order to investigate proper topic tracking in the conversation. This is central to chatbots functioning well. The field of topic tracking is vast, and we consider this is an area of research that will need significant research into existing methods used both in chatbots and other types of system, followed by an extensive period of testing.

8.3.5 Sentiment analysis

We would like to consider adding the ability for the chatbot to distinguish between different sentiment states. Our stemmer has already been used in a sentiment analysis system that is due to be released to the public shortly. We believe that investigating the technology and integrating it into our system would be highly beneficial to the user. The user responds strongly to the behaviour of the system, so if it is able to determine accurately their sentiment state, we could teach it to respond adequately in each one. Christopher Rines, lead developer of the sentiment analysis system, states: "We use the UEA Stemmer (in our as yet unlatched information client) to extract entities, time and event information from various real time and static datasets. We've chosen UEA because we needed the most accurate stems for semantic knowledge extraction and because it was very easy to add additional domain specific.". There is a potential research partnership.

8.3.6 Open domain

We worked in a closed domain because the complexity of working in an open domain distracted us from the essentials of chatbot development, that we felt needed adressing before we moved into added complexity. The closed domain research helped us to gain insight into how we could, with minimum human intervention, open up the system to an open domain. This is very important because the more domain knowledge a system has, the greater its ability to answer any question. We suggest research in the semantic web technologies that would serve as a good platform to knowledgebase expansion for chatbots.

8.3.7 Integration with mobile devices

Since we began our research, the web and the way that we use it has changed. People are increasingly using mobile devices to access the web and applications have become popular tools for people. We can see further use for SOCS through mobile devices. Packaging the technology into a mobile application will enable us to move the technology from the laptop or desktop computer to the more readily available mobile device. This means that we can investigate the use of chatbots in immediate surroundings and provide chatbots that serve different purposes. A companion chatbot, a chatbot that is ready to inform on any topic, or ready to help choose a product in store for example, would really push current chatbot boundaries.

Bibliography

[1] R. P. Abelson. "searle's argument is just a set of chinese symbols". *Behavioral and Brain Sciences*, 3:424–425, 1980.

[2] A. V. Aho and M. J. Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, June 1975.

[3] K. Ajdukiewicz and J. Giedymin. *The Scientific world-perspective and other essays, 1931-1963 / Kazimierz Ajdukiewicz ; edited and with an introd. by Jerzy Giedymin.* D. Reidel Pub. Co., Dordrecht ; Boston :, 1978.

[4] V. Aleven, O. Popescu, and K. R. Koedinger. Towards tutorial dialog to support self-explanation: Adding natural language understanding to a cognitive tutor. In *In*, pages 246–255, 2001.

[5] J. Allen, N. Chambers, G. Ferguson, L. Galescu, H. Jung, and W. Taysom. Plow: A collaborative task learning agent. In *In Proc. Conference on Artificial Intelligence (AAAI*, pages 22–26. Springer-Verlag, 2007.

[6] D. Anderson and B. J. Copeland. Artificial life and the chinese room argument. *Artificial Life*, 8(4):371–378, 2002.

[7] R. Artstein, S. Gandhe, M. Rushforth, and D. R. Traum. Viability of a simple dialogue act scheme for a tactical questioning dialogue system. In *DiaHolmia 2009: Proceedings of the 13th Workshop on the Semantics and Pragmatics of Dialogue*, page 43–50, Stockholm, Sweden, June 2009.

[8] M. Asahara. Extended models and tools for high-performance part-of-speech tagger. volume International Conference On Computational Linguistics archive Proceedings of the 18th conference on Computational linguistics - Volume 1, pages 21 – 27, 2000.

[9] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[10] C. F. Baker, C. J. Fillmore, and J. B. Lowe. The berkeley framenet project. In *COLING-ACL*, pages 86–90, 1998.

[11] M. Banko and L. Vanderwende. Using n-grams to understand the nature of summaries. volume In Proceedings of HLT/NAACL 2004.

[12] . Bar-Hillel, Yehoshua. *Aspects of language : Essays and lectures on philosophy of language, linguistic philosophy, and methodology of linguistics.* Magnes Press, Jerusalem :, 1970.

[13] R. Barzilay and M. Lapata. Collective content selection for concept-to-text generation. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 331–338, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[14] J. Bateman, J. Kleinz, T. Kamps, and K. Reichenberger. Towards constructive text, diagram, and layout generation for information presentation. *Comput. Linguist.*, 27(3):409–449, 2001.

[15] J. A. Bateman. Enabling technology for multilingual natural language generation: the kpml development environment. *Nat. Lang. Eng.*, 3(1):15–55, 1997.

[16] M. Bates, R. J. Bobrow, R. Ingria, and D. Stallard. The delphi natural language understanding system. In *ANLP*, pages 132–137, 1994.

[17] O. Bender, K. Macherey, F. J. Och, and H. Ney. Comparison of alignment templates and maximum entropy models for natural language understanding. In *in EACL*, pages 11–18, 2003.

[18] R. Bentley, W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, K. Sikkel, J. Trevor, and G. Woetzel. Basic support for cooperative work in the world wide web. *International Journal of HumanComputer Studies*, Special Issue on Innovative Applications of the World Wide Web, 1997.

[19] T. Berners-Lee. "an attempt to give a high-level plan of the architecture of the semantic www. (working draft)". *Communications of the ACM*, 1998.

[20] P. Boldi, F. Bonchi, C. Castillo, and S. Vigna. From dango to japanese cakes: Query reformulation models and patterns. submitted for publication, 2008.

[21] J. Bos. Three stories on automated reasoning for natural language understanding. In *in: Proceedings of ESCoR (IJCAR Workshop): Empirically Successful Computerized Reasoning*, pages 81–91, 2006.

[22] T. Brants. Tnt – a statistical part-of-speech tagger. In *ANLP*, pages 224–231, 2000.

[23] B. Bridgeman. "brains + programs = minds". *Behavioral and Brain Sciences*, 3:427–428, 1980.

[24] E. Brill. A report of recent progress in transformation-based error-driven learning. In *Proceedings of the workshop on Human Language Technology*, HLT '94, pages 256–261, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

[25] E. Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

[26] E. Briscoe and J. Carroll. Developing and evaluating a probabilistic lr parser of part-of-speech and punctuation labels. In *Proceedings of the 4th ACL/SIGPARSE International Workshop on Parsing Technologies*, pages 48–58, 1995.

[27] C. Brockmann, A. Isard, J. Oberlander, and M. White. Modelling alignment for affective dialogue. In *Proceedings of the Workshop on Adapting the Interaction Style to Affective Factors at the 10th International Conference on User Modeling (UM-05)*, 2005.

[28] M. J. Cafarella, D. Downey, S. Soderl, and O. Etzioni. Knowitnow: Fast, scalable information extraction from the web. In *In Proceedings of the Human Language Technology Conference (HLT-EMNLP-05*, pages 563–570, 2005.

[29] R. Carpenter and J. Freeman. Computing machinery and the individual: the personal turing test, 2005, paper available at http://www.jabberwacky.com, 2005.

[30] S. Ceccato. *Linguistic Analysis and Programming for Mechanical Translation.* Gordon and Breach, 1961.

[31] J.-C. Chappelier and M. Rajman. A generalized cyk algorithm for parsing stochastic cfg. In *Proceedings of Tabulation in Parsing and Deduction (TAPD'98)*, pages 133–137, Paris (FRANCE), April 1998.

[32] E. Charniack. A maximum-entropy-inspired parser. In *ANLP*, 2000.

[33] C. Cheepen. The predictability of informal conversation. Behavioral and Brain Sciences 3, 1988.

[34] N. Chomsky. *Aspects of the Theory of Syntax.* MIT Press, Cambridge, Massachusetts, 1965.

[35] N. Chomsky. Syntactic structures. Mouton, The Hague, Netherlands, 1985.

[36] N. Chomsky. Three models for the description of language. *Information Theory, IRE Transactions on*, 2(3):113–124, January 2003.

[37] P. Clark, C. Fellbaum, and J. Hobbs. Using and extending wordnet to support question-answering. In *Proc. Fourth Global WordNet Conference (GWC'08)*, 2008.

[38] S. Clark. Parsing the wsj using ccg and log-linear models. In *In Proceedings of the 42nd Meeting of the ACL*, pages 104–111, 2004.

[39] K. M. Colby. Ten criticisms of parry. *SIGART Bull.*, pages 5–9, October 1974.

[40] M. Collins. Three generative, lexicalised models for statistical parsing. volume n Proceedings of the 35th Annual Meeting of the ACL., 1997.

[41] M. Collins. Head-driven statistical models for natural language parsing. *Comput. Linguist.*, 29:589–637, December 2003.

[42] D. Coniam. Evaluating the language resources of chatbots for their potential in english as a second language. *ReCALL*, 20(1):98–116, 2008.

[43] J. Copeland. A lecture and two radio broadcasts on machine intelligence by alan turing. *Machine Intelligence*, 15(4), 1999.

[44] J. Cowie and Y. Wilks. Information extraction, 1996.

[45] W. Croft and A. D. Cruse. *Cognitive Linguistics.* Cambridge Textbooks in Linguistics. Cambridge University Press, New York, 2004.

[46] J. K. Cullum and R. A. Willoughby. Chapter 5: Real rectangular matrices. volume In Lancozos algorithms for large symmetric eigenvalue computations - Vol 1 Theory., pages 113–125, 1985.

[47] O. Curé. Data integration for the semantic web with full preferences. In *ONISW '08: Proceeding of the 2nd international workshop on Ontologies and nformation systems for the semantic web*, pages 61–68, New York, NY, USA, 2008. ACM.

[48] K. Dahlgen. Technical overview of cognition's semantic nlptm (as applied to search). 2007.

[49] K. Dahlgren. *Naive Semantics for Natural Language Understanding.* Kluwer Academic Publishers, Norwell, MA, USA, 1988.

[50] K. Dahlgren. A linguistic ontology. *Int. J. Hum.-Comput. Stud.*, 43(5-6):809–818, 1995.

[51] R. Dale, D. Scott, and B. Di Eugenio. Introduction to the special issue on natural language generation. *Comput. Linguist.*, 24(3):346–353, 1998.

[52] F. J. Damerau. A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176, 1964.

[53] D. Davis. Multiple level representation of emotion in computational agents emotion, cognition & affective computing. volume AISB2001-Agents and Cognition., 2004.

[54] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. In *Journal of the American Society of Information Science*, page 391407, 1990.

[55] W. V. den Broeck. Irl language specification. 2006.

[56] A. Denis, M. Quignard, and G. Pitel. A deep-parsing approach to natural language understanding in dialogue system: Results of a corpus-based evaluation. In *Proceedings of Language Resources and Evaluation Conference*, Malaga Spain, 2006.

[57] J. L. Dennis Grinberg and D. Sleator. A robust parsing algorithm for link grammars. In *In Proceedings of the Fourth International Workshop on Parsing Technologies*, 1995.

[58] D. DeVault, D. Traum, and R. Artstein. Practical grammar-based nlg from examples. In *INLG*. ACM, 2008.

[59] S. C. Dik. *The Theory of Functional Grammar. Part 2: Complex and Derived Constructions (edited by Kees Hengeveld).* Mouton de Gruyter, Berlin, second edition, 1997.

[60] J. Dinsmore. Book review: Mental spaces: Aspects of meaning construction in natural language (bradford/mit press). *SIGART Bull.*, (99):18–19, 1987.

[61] P. A. Duboue and K. R. McKeown. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 121–128, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[62] J. C. Eccles. "a dualist-interactionist perspective". *Behavioral and Brain Sciences*, 3:430–431, 1980.

[63] D. Elworthy. Part of speech and phrasal tagging. In *ESPRIT BRA7315 Acquilex II*, volume Working Paper 10, 1993.

[64] R. Feldman, B. Rosenfeld, M. Fresko, and B. D. Davison. Hybrid semantic tagging for information extraction. In *WWW (Special interest tracks and posters)*, pages 1022–1023, 2005.

[65] C. Fellbaum. *WordNet and wordnets*, pages 665–670. Elsevier, Oxford, second edition edition, 2005.

[66] G. Ferguson and J. F. Allen. Trips: The rochester interactive planning system. In *AAAI/IAAI*, pages 906–907, 1999.

[67] J. A. Fodor. "searle on what only brains can do",. *Behavioral and Brain Sciences*, 3:431–432, 1980.

[68] W. Frakes and C. Fox. Strength and similarity of affix removal stemming algorithms. In *Strength and similarity of affix removal stemming algorithms.*, volume ACM SIGIR Forum, volume 37, issue 1, 2003.

[69] W. N. Francis and H. Kucera. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.

[70] M. Fried and H. C. Boas, editors. *Grammatical Constructions*. Constructional Approaches to Language 4. John Benjamins, Philadelphia, 2005.

[71] A. M. Galvo, F. A. Barros, A. M. M. Neves, and G. Ramalho. Persona-aiml: An architecture developing chatterbots with personality. In *AAMAS*, pages 1266–1267. IEEE Computer Society, 2004.

[72] P. Gebhard. Enhancing embodied intelligent agents with affective user modelling. In *User Modeling*, pages 271–273, 2001.

[73] D. Geeraerts, editor. *Cognitive Linguistics: Basic Readings*. Mouton Reader. De Gruyter Mouton, Berlin, 2006.

[74] A. E. Goldberg. *Constructions: A Construction Grammar Approach to Argument Structure*. University Of Chicago Press, Chicago, 1995.

[75] A. E. Goldberg. *Constructions at Work*. Oxford University Press, New York, 2005.

[76] Y. Gong and X. Liu. Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR conference on Research and Development in Information Retrieval.*, volume Proceedings of the 24th Annual International ACM, page 1925, 2001.

[77] Google. Google talk. urlhttp://www.google.com/talk/.

[78] H. P. Grice. Logic and conversation. In P. Cole, editor, *Speech acts*, volume 3 of *Syntax and semantics*, pages 41–58. Academic Press, New York, 1975.

[79] H. P. Grice. *Studies in the Way of Words*. Harvard University Press, Cambridge, Massachusetts, 1989.

[80] R. V. Guha and D. Brickley. RDF Vocabulary Description Language 1.0: RDF Schema. W3C recommendation, W3C, February 2004.

[81] N. Gupta, G. Tr, D. Hakkani-Tr, S. Bangalore, G. Riccardi, and M. Gilbert. The at & t spoken language understanding system. *IEEE Transactions on Audio, Speech & Language Processing*, 14(1):213–222, 2006.

[82] M. A. Halliday and C. M. Matthiessen. *An Introduction to Functional Grammar*. Arnold Publishers, 2004.

[83] M. A. K. Halliday and R. Hasan. *Cohesion in English*. Longman, 1976.

[84] A. Hartholt, T. A. Russ, D. R. Traum, E. H. Hovy, and S. Robinson. A common ground for virtual humans: Using an ontology in a natural language oriented virtual human architecture. In *LREC*. European Language Resources Association, 2008.

[85] D. R. Hartree. *Calculating Instruments and Machines (Mathematicians of Our Time,)*. MIT University Press, 1949.

[86] E. Hatcher and O. Gospodneti. *Lucene in Action*. Manning Publications, December 2004.

[87] M. A. Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, 1997.

[88] S. Higuchi, R. Rzepka, and K. Araki. A casual conversation system using modality and word associations retrieved from the web. In *EMNLP*, pages 382–390, 2008.

[89] G. Hirst and D. St-Onge. *Lexical chains as representations of context for the detection and correction of malapropisms.* 1997.

[90] T. Hofmann. Probabilistic latent semantic analysis. In *Annual ACM Conference on Research and Development in Information Retrieval*, volume Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, pages 50 – 57, 1999.

[91] J.-H. Hong and S.-B. Cho. Domain-adaptive conversational agent with two-stage dialogue management. volume Lecture Notes in Artificial Intelligence, vol. 3339, pages 1147–1153, 2004.

[92] P. J. Hopper. Emergent grammar and the a priori grammar postulate. In *Linguistics in Context*, pages 117–134. Ablex, 1988.

[93] E. Hovy and C.-Y. Lin. Automated text summarization and the summarist system. In *Proceedings of a workshop on held at Baltimore, Maryland*, pages 197–214, 1996.

[94] E. Hovy and E. Maier. Parsimonious or profligate: How many and which discourse structure relations?, 1997.

[95] B. J. Jansen, A. Spink, and J. Pedersen. A temporal comparison of altavista web searching. *Journal of the American Society for Information Science and Technology*, 56:559–570, 2005.

[96] P. G. Javanaud. *On Lucien Tesnire's 'Les lments de syntaxe structurale' / Pierre G. Javanaud.* University of Gothenburg, Dept. of Linguistics, Gteborg :, 1977.

[97] G. Jefferson. The mind of mechanical man, lister oration. *British Medical Journal*, i(4):1105–1121, 1949.

[98] M. Jenkins and D. Smith. Conservative stemming for search and indexing. In *Conservative stemming for search and indexing*, 2005.

[99] A. H. Kao. Montague grammar. 2004.

[100] A. Kilgarriff and J. Rosenzweig. Framework and results for english senseval. *Computers and the Humanities*, 34(1-2):15–48, 2000.

[101] J.-B. Kim. The grammar of negation: A constraint-based approach. In *NEGATION WITHOUT HEAD MOVEMENT 75*. CSLI Publications, 2000.

[102] J.-D. Kim, T. Ohta, Y. Tateisi, and J. ichi Tsujii. Genia corpus - a semantically annotated corpus for bio-textmining. pages 180–182, 2003.

[103] C. Kit and Y. Wilks. The virtual corpus approach to deriving n-gram statistics from large scale corpora. volume In C. N. Huang (ed.), Proceedings of 1998 International Conference on Chinese Information Processing, pages 223–229, 1998.

[104] G. Klyne and J. J. Carroll. Resource description framework (rdf) concepts and abstract syntax, February 2004.

[105] H. Knublauch. Ontology-driven software development in the context of the semantic web: An example scenario with protege/owl. In D. S. Frankel, E. F. Kendall, and D. L. McGuinness, editors, *1st International Workshop on the Model-Driven Semantic Web (MDSW2004)*, 2004.

[106] J. Kupiec, J. Pedersen, and F. Chen. A trainable document summarizer. 1995.

[107] G. Lakoff. *Women, fire, and dangerous things: what categories reveal about the mind*. University of Chicago Press Chicago, 1987.

[108] T. K. Landauer and S. Dumais. Latent semantic analysis. *Scholarpedia*, 3(11):4356, 2008.

[109] R. W. Langacker. *Foundations of cognitive grammar: Theoretical Prerequisites*. Stanford University Press, Stanford, CA, 1987. Vol 1, 1987(Hardcover), 1999(Paperback).

[110] O. N. Language and S. Miller. Hidden understanding models. In *In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 25–32, 1994.

[111] B. Lavoie and O. Rambow. A fast and portable realizer for text generation systems, 1997.

[112] O. Lemon. Adaptive natural language generation in dialogue using reinforcement learning. *Proceedings of the 12th SEMDIAL Workshop*, 2008.

[113] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. volume In Proceedings of ACM SIGDOC 86, pages 24–26, 1986.

[114] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. Karger. What makes a good answer? the role of context in question answering. In *INTERACT*, 2003.

[115] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger. What makes a good answer? the role of context in question answering. In *PROCEEDINGS OF INTERACT 2003*, pages 25–32, 2003.

[116] H. Loebner. Loebner prize rules 2006. `http://www.loebner.net/Prizef/2006_Contest/2006_Rules.txt`.

[117] H. Loebner. Loebner prize transcripts 2006. `http://www.loebner.net/Prizef/2006_Contest/loebner-prize-2006.html`.

[118] J. B. Lovins. Development of a stemming algorithm. *Mechanical Translation*, 11:22–31, 1968.

[119] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, Vol. 2:159–165, 1958.

[120] B. Lyle. Java api for wordnet searching - jaws, 2009.

[121] S. M. Lyon, S. S. Magliveras, S. M. Lyon, and S. S. Magliveras. Kinship, computing, and anthropology, 2006.

[122] R. B. B. M. Newman, G. Jefferson and R.Searle. Can automatic calculating machines be said to think? *Radio Broadcast*, Radio broadcast, 1952.

[123] B. Malinowski. The problem of meaning in primitive languages. In C.K. Ogden & I.A. Richards(Eds.), The Meaning of Meaning:296–336, 1923.

[124] P. Malish. Kyle. `http://leeds-city-guide.com/kyle`.

[125] I. Mani and E. Bloedorn. Machine learning of generic and user-focused summarization. *CoRR*, cs.CL/9811006, 1998.

[126] W. C. Mann and S. A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

[127] D. Marcu and D. Marcu. Building up rhetorical structure trees. In *In Proceedings of AAAI-96. American Association for Artificial Intelligence*, pages 1069–1074, 1996.

[128] M. Marcus, M. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[129] M. Mauldin. Verbot. `http://www.verbot.com`.

[130] M. L. Mauldin. Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition. In *AAAI*, pages 16–21, 1994.

[131] A. McCallum. Lecture 5: Context free grammars. `http://www.cs.umass.edu/~mccallum/courses/inlp2007/lect5-cfg.pdf`.

[132] J. McCarthy. The inversion of functions defined by turing machines. In J. M. C.E. Shannon, editor, *Automata Studies, Annals of Mathematical Studies*, number 34, pages 177–181. Princeton University Press, 1956.

[133] D. L. McGuinness and F. van Harmelen. Owl web ontology language overview. W3c recommendation, World Wide Web Consortium, 2004.

[134] K. R. McKeown. *Text generation: using discourse strategies and focus constraints to generate natural language text*. Cambridge University Press, New York, NY, USA, 1985.

[135] L. Michaelis. Construction grammar. In *The Encyclopedia of Language and Linguistics*, volume 3, pages 73–84. Oxford: Elsevier, 2006.

[136] Microsoft. Windows messenger. `http://messenger.msn.com/`.

[137] A. Miles and J. R. Pérez-Aguera. Skos: Simple knowledge organisation for the web. *Cataloging & Classification Quarterly*, 43(3):69–83, 2007.

[138] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.

[139] J. Moore, M. E. Foster, O. Lemon, M. White, M. Ellen, F. Oliver, and L. M. White. Generating tailored, comparative descriptions in spoken dialogue. In *Proceedings of FLAIRS*. Press, 2004.

[140] J. L. Neto, A. D. Santos, C. A. A. Kaestner, A. A. Freitas, and J. C. Nievola. A trainable algorithm for summarizing news stories, 2000.

[141] A. Nijholt. Conversational agents, humorous act construction, and social intelligence. In *Conversational Agents, Humorous Act Construction, and Social Intelligence.*, volume Proceedings AISB 2005: Social Intelligence and Interaction in Animals, Robots and Agents., pages 1–8, 2005.

[142] J. Nivre. Dependency grammar and dependency parsing. Technical report, Vxj University, 2005.

[143] S. Novichkova, S. Egorov, and N. Daraselia. Medscan, a natural language processing engine for medline abstracts. *Bioinformatics*, 19(13):1699–1706, 2003.

[144] A. H. Oh and A. I. Rudnicky. Stochastic language generation for spoken dialogue systems. In *ANLP/NAACL 2000 Workshop on Conversational systems*, pages 27–32, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[145] S. Ozmutlu, H. C. Ozmutlu, and A. Spink. Trends in multimedia web searching: Excite queries. In *ITCC*, pages 40–45, 2002.

[146] C. D. Paice. Another stemmer. *SIGIR Forum*, 24(3):56–61, 1990.

[147] A. Paradiso and M. L'Abbate. A model for the generation and combination of emotional expressions. In *Workshop on Representing, Annotating, and Evaluating Non-Verbal and Verbal Communicative Acts to Achieve Contextual Embodied Agents, Fifth International Conference on Autonomous Agents, May 29, 2001*. McGraw-Hill, 2001.

[148] T. J. Pasi Tapanainen. A non-projective dependency parser. In *In proceedings of the fifth conference on Applied natural language processing*, 1997.

[149] S. Patwardhan, S. Banerjee, and T. Pedersen. Senserelate: : Targetword-a generalized framework for word sense disambiguation. In *AAAI*, pages 1692–1693, 2005.

[150] F. Pereira and D. Warren. Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, 13:231–278, 1980.

[151] H. Pinto, Y. Wilks, R. Catizone, and A. Dingli. The senior companion multiagent dialogue system. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 3*, AAMAS '08, pages 1245–1248, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[152] C. Pollard, , C. Pollard, I. A. Sag, E. J. Goldsmith, J. D, J. M. Sadock, J. Nerbonne, K. Netter, C. P. (editors, and J. pierre Koenig. Head-driven phrase structure grammar, 1996.

[153] J. M. Ponte and W. B. Croft. Text segmentation by topic. In *In Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries*, pages 120–129, 1997.

[154] M. Porter. An algorithm for suffix stripping. In *Workshop on Multimedia Information Systems*, 1980.

[155] J. Preiss. Using grammatical relations to compare parsers. volume In Proc. of EACL 03, 2003.

[156] A. Prince and P. Smolensky. Optimality theory: Constraint interaction in generative grammar. Technical report, 1993.

[157] S. A. Quarteroni and S. Manandhar. Designing an interactive open domain question answering system. *Journal of Natural Language Engineering, Special issue on Interactive Question Answering*, 2008.

[158] M. R. Quillian. Semantic memory. In M. Minsky, editor, *Semantic Information Processing*. MIT Press, 1968.

[159] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Comprehensive Grammar of the English Language*. Longman, London, 1985.

[160] K. M. R. Barzilhay, N. Elhadad. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artifcial Intelligence Research*, 17:35–55, 2002.

[161] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(1):17–30, 1989.

[162] D. R. Radev, H. Qi, Z. Zheng, S. Blair-goldensohn, Z. Zhang, W. Fan, and J. Prager. Query modulation for web-based question answering. 2001.

[163] O. Rambow, S. Bangalore, and M. Walker. Natural language generation in dialog systems. In *HLT '01: Proceedings of the first international conference on Human language technology research*, pages 1–4, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[164] A. Ratnaparkhi. Trainable methods for surface natural language generation. *CoRR*, cs.CL/0006028, 2000. informal publication.

[165] H. B. Raymond Kosala. Web mining research: A survey. 2000.

[166] B. Reeves and C. Nass. *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places*. Cambridge University Press, 1996.

[167] M. Richardson and P. Domingos. Building large knowledge bases by mass collaboration. In *K-CAP '03: Proceedings of the 2nd international conference on Knowledge capture*, pages 129–137, New York, NY, USA, 2003. ACM.

[168] E. Rosch, C. B. Mervis, W. D. Gray, D. M, and P. Boyes-braem. Basic objects in natural categories. *Cognitive Psychology*, 1976.

[169] J. Ruppenhofer, M. Ellsworth, M. R. L. Petruck, C. R. Johnson, and J. Scheffczyk. Framenet ii: Extended theory and practice. Technical report, ICSI, 2005.

[170] A. Saygin and I.Cicekli. Pragmatics in human-computer conversations. volume Journal of Pragmatics, 34(3), pages 227–258, 2002.

[171] R. C. Schank. "understanding searle". *Behavioral and Brain Sciences*, 3:446–447, 1980.

[172] J. Searle. *Intentionality. An Essay in the Philosophy of Mind.* Cambridge University Press, 1983.

[173] J. R. Searle. Minds, brains and programs. *The Behavioral and Brain Sciences.3*, 1980.

[174] C. E. Shannon and J. McCarthy. Automata studies. In *Automata Studies.*, 1956.

[175] S. C. Shapiro and J. G. Neal. A knowledge engineering approach to natural language understanding, 1982.

[176] S. M. Shieber, Y. Schabes, and F. C. N. Pereira. Principles and implementation of deductive parsing. *J. Log. Program.*, 24(1 and 2):3–36, 1995.

[177] P. Singh, T. Lin, E. T. Mueller, G. Lim, T. Perkins, and W. L. Zhu. Open mind common sense: Knowledge acquisition from the general public. pages 1223–1237. Springer-Verlag, 2002.

[178] M. J. Srinivas, M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whittaker, and P. Maloor. Match: An architecture for multimodal dialogue systems. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 376–383, 2002.

[179] L. Steels and J. de Beule. A (very) brief introduction to fluid construction grammar. In *ScaNaLU '06: Proceedings of the Third Workshop on Scalable Natural Language Understanding*, pages 73–80, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[180] A. Stent, R. Prasad, and M. Walker. Trainable sentence planning for complex information presentation in spoken dialog systems. In *In Proc. of the Annual Meeting of the Association for Computational Linguistics*, pages 79–86, 2004.

[181] L. Tanabe, N. Xie, L. H. Thom, W. Matten, and W. J. Wilbur. Genetag: a tagged corpus for gene/protein named entity recognition. In *BMC Bioinformatics*, volume 6(Suppl 1):S3, pages 9–13, 2005.

[182] L. K. Tanabe and W. J. Wilbur. Tagging gene and protein names in full text articles. In *ACL Workshop on Natural Language Processing in the Biomedical Domain*, pages 9–13, 2002.

[183] P. Tarau and E. Figa. Knowledge based conversational agents and virtual storytelling.

[184] R. Taylor. Translator's notes to an article on babbage's analytical engine. *Scientific Memoir*, 3:691–731, 1842.

[185] S. Teufel and M. Moens. Sentence extraction as a classification task. pages 58–65, 1997.

[186] T.Landauer, P.Foltz, and D.Laham. Introduction to latent semantic analysis. volume Discourse Processes, 25, pages 259–284, 1998.

[187] K. Tracy and C.Coupland. Multiple goals in discourse: An overview of issues. In K. Tracy and N.Coupland (Eds.), Multiple goals in discourse:1–13, 1991.

[188] D. R. Traum. *Talking to Virtual Humans: Dialogue Models and Methodologies for Embodied Conversational Agents*, page 296–309. Springer, 2008.

[189] D. R. Traum, A. Roque, A. Leuski, P. Georgiou, J. Gerten, B. Martinovski, S. Narayanan, S. Robinson, and A. Vaswani. Hassan: A virtual human for tactical questioning. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, page 71–74, September 2007.

[190] A. M. Turing. Computing machinery and intelligence. In A. Collins and E. E. Smith, editors, *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, pages 6–19. Kaufmann, San Mateo, CA, 1988.

[191] M. P. University and M. R. L. Petruck. Frame semantics. In *Handbook of Pragmatics. John Benjamins*, pages 1–13. John Benjamins, 1996.

[192] R. D. Van Valin. *A synopsis of Role and Reference Grammar.* John Benjamins, Amsterdam, 1993.

[193] A. Voutilainen. Nptool, a detector of english noun phrases. *CoRR*, cmp-lg/9502010, 1995. informal publication.

[194] R. Wallace. Aiml pattern matching simplified. `http://www.alicebot.org/documentation/matching.html`.

[195] R. Wallace. The elements of aiml style. alice ai foundation., 2004.

[196] R. S. Wallace. Symbolic reductions in aiml, 2000.

[197] J. Weizenbaum. Eliza - a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9:36–45, January 1966.

[198] D. Williams and C. Cheepen. just speak naturally: designing for naturalness in automated spoken dialogues. In *CHI 98 conference summary on Human factors in computing systems*, CHI '98, pages 243–244, New York, NY, USA, 1998. ACM.

[199] I. Witten and T. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4), 1991.

[200] M. S. Wojciech Gorka, Adam Piasecki and J. Ganko. Intermediate information layer. the use of the skos ontology to create information about e-resources provided by the public administration. In *Proceedings of the International Multiconference on Computer Science and Information Technology*, page 437442, 2008.

[201] W. A. Woods. Transition network grammars of natural language analysis. *Readings in Natural Language Processing*, 8:71–87, 1986.

[202] A. Wray. *Formulaic Language and the Lexicon.* Cambridge University Press, Cambridge, 2002.

[203] J. Xu and W. Croft. Corpus-based stemming using co-occurrence of word variants. volume ACM Transactions on Information Systems, 16 (1), pages 61–81, 1998.

[204] Y. Yaari. Segmentation of expository texts by hierarchical agglomerative clustering. 1997.

[205] J. Yan, D. B. Bracewell, S. Kuroiwa, and F. Ren. Chinese semantic dependency analysis: Construction of a treebank and its use in classification. *ACM Trans. Speech Lang. Process.*, 4(2):5, 2007.

[206] H. Yang and T.-S. Chua. Qualifier: question answering by lexical fabric and external resources. In *EACL '03: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, pages 363–370, 2003.

[207] M. K. Ying He. A comparison of 13 tokenizers on medline. *Annual Symposium proceedings / AMIA Symposium.*, page 946, 2006.

[208] W. Zadrozny, M. Budzikowska, J. Chai, N. Kambhatla, S. Levesque, and N. Nicolov. Natural language dialogue for personalized interaction. volume Vol. 43, No. 8 Communications of the ACM, 2000.

Appendix A

KIA 1 Chatbot Logs Sample

This is a sample of conversations between customer service agents and Aviva customers. These conversations took place on the website through the IM facility.

Aviva Customer service logs (sample of 6 online chats)

1)
Martin: Welcome to Norwich Union Direct, how can I help you today?
Paul: hi i just tried to applie for car insurance but when i went to pay it said that i need to phone or chat to go any firther
Martin: ok, you would need to call in this case
Martin: 08000 929 561
Martin: and we can arrange payment hopefully
Martin: so long as nothing prevents us doing so
Martin: ie, bank not letting us take the funds etc
Paul: ok thanks ill call it right nowe thanks for your help
Martin: Thank you for contacting Norwich Union Direct today. I hope you found this facility helpful.

2)
Tom: Good morning, Norwich Union Direct, how can I help you today?
Jack: Hi, i'm looking to buy a car in the near future and am trying to see roughly what the insurance will cost me. I've done pass plus, and have been driving as a named driver on my mother's insurance for 16 months with no claims or accidents and was wondering how much that will bring the quote down?
Tom: ok if would like to get a quote for pass plus
Tom: you will need to call our sales team
Tom: on 0800 888 111
Tom: we cant quote for pass plus online
Tom: it will probably be about 30% discount
Jack: and will the fact i've been driving for 16 months already help or not?
Tom: instead of 20% that you get on the internet
Jack: okay
Tom: if you have been driving for over a year
Tom: that will help a little
Jack: thanks very much then
Tom: I'm sorry we were unable to help further but thank you for trying Norwich Union Direct.Bye.

3)
Martin: Welcome to Norwich Union Direct, how can I help you today?
nisarg: i have 5years old indian license&international &uk also so how many years my lincease old ?
Martin: total amount of years held for all
nisarg: uk lincease i held 14 jan 2006
Martin: ok, we take into account international licenses also
Martin: so you can enter the amount including thar
Martin: that sorry

nisarg: i am just fill the online form how many years i can fill?
Martin: the total amount of full years that you have held your licenses for

4)
Martin: Welcome to Norwich Union Direct, how can I help you today?
irene: received quote online saturday. Wish cover to commence 21st Feb, not 22nd. Also require breakdown cover and wish to confirm insurance is fully comprehensive.
Martin: ok, if you call our internet sales team on 08000 929 561
Martin: we can check the quote and start it from the date you wish
Martin: also we can quote for breakdown
Martin: Is there anything else I can help you with today?
irene: no thanks
Martin: Thank you for contacting Norwich Union Direct today. I hope you found this facility helpful.

5)
Paul: hi, if i put down 12,000 as my annual milage, what happens if i go over that?
Martin: Welcome to Norwich Union Direct, how can I help you today?
Martin: you can call customer services if you think that you might go over that
Martin: you'd just get charged the extra premium
Paul: so i have to check it throughout the year - will there be more to pay if I go over it
Martin: which probably wont be a lot anyway
Paul: how do you know, you do not ask for the milage of the car on the website?
Martin: .... basically yes
Martin: all down to trust...
Martin: if we trace via mot or take that your mileage is way put
Martin: ie you put 12000 mile
Martin: but do 50000
Martin: then your policy will be void an we wont pay out
Paul: more likely - i put 12,000 and do 13,000
Paul: if it were to happen of course
Martin: ok :)
Paul: one other thing
Martin: go ahead....
Paul: my wife has a speeding 3 points from about 7 years ago, do i have to put that down?
Martin: only any convictions within the last 4 years
Paul: cool - thnks
Martin: Is there anything else I can help you with today?
Paul: no thanks

6)
Martin: Hello, how can I help you today?

Heather: I need to put into the model field information about my car. My car is a 1997 ford escort 1.8tdl (70) hatch 5dr. What do i enter in the field?
Martin: ford escort tdi
Martin: that should bring it through
Heather: thanks

Appendix B

KIA 2 John System conversation Sample

This is a sample of conversations with the John System.

Sample conversation with the John System:

1)

You: hi john
John: hey, I'm John
You: i know
John: Sure, I'm a yoga teacher from boulder in colorado
You: boulder in the states
John: "I'm from Boulder, Colorado, in the United States"
You: why should i do yoga
John: Yoga keeps people in shape and makes them feel better. This is one of the reasons why most people do yoga.
You: is yoga religious
John: a little bit, I think you might not be into the things I am, like yoga for example.
You: do you know big brother
John: your questions are boring
You: why
John: "Due to the pressure and demands of life, everybody has a tendency to forget what is needed for our life and we often forget the essence of life. We tend to lose touch with the ones we used to spend time with, even ourselves. "
You: is that why you do yoga
John: pretty much yep!
You: so where is boulder
John: Boulder is located in Boulder Valley where the Rocky Mountains meet the Great Plains.
You: do you like it there
John: "it's a pleasant city with an old world charm, contributed by its broad shady avenues"
You: Have you travelled much
John: yes but I'm not married and I have no children
You: why do you have no children
John: because I have nobody to have them with!
You: where do you travel
John: I love travelling I go to mysore very often
You: are you a computer
John: I'm not meant to answer that I think
You: what is india like
John: India is a special place.
You: where do you go
John: I went to mysore
You: where is mysore
John: It's in southern India, Situated in the southern part of the Deccan Plateau
You: how did you get there
John: I came with a friend we flew
You: where did you stay
John: I stayedat the Green Hotel. The room was nice and clean. It only cost me about 1,100 ruppees(about $25). – 23rd
You: That's cheap
John: :)
You: bye

John: see ya dude, peace

2)

You: Hello
John: Hi!
You: what are you doing here
John: Just being me
You: are you into chatbots
John: I think they're kinda fun
You: do you like new york
John: It's a hectic city but it's beautiful at the same time
You: have you been before
John: Yea a couple of times
You: do you have a guru then
John: Yes, Pattabhi jois
You: who is pattabhi jois
John: "In 1958, Sri K. Pattabhi Jois began writing Yoga Mala. It was published in India in 1962 and first published in English in 1999."
You: what does Pattabhi jois do
John: Sri K. Pattabhi Jois continues to teach at the Ashtanga Yoga Research Institute in Mysore, now located in the neighborhood of Gokulum, with his daughter Saraswati and his grandson Sharath.
You: how old is he
John: I don't remember
You: how do i start doing yoga
John: Find a local yoga studio where you can take some classes and where you can learn with a good teacher
You: what kind of yoga
John: There is no point in signing up for an ashtanga class if you are sixty five, slightly overweight, and only want to relax, or to go to a viniyoga teacher if you are taking up yoga as an alternative to jogging or weight lifting to keep fit. If in doubt, ask to watch a class before enrolling. Even better, most teachers will actually let you do one trial class
You: do i need to bring a mat
John: “sticky-mats” are very popular as they give good traction
You: ok then, thanks for the conversation
John: Yes, it's been ok
You: bye
John: see ya dude, peace

Appendix C

KIA1 User tasks and instructions

These are the tasks and instructions that were given to KIA test users.

KIA
((KNOWLEDGE INTERACTION AGENT))

KIA user interaction experiment

Dear tester,

First, thanks for agreeing to participate in the KIA project experiment. Without your help, we would be unable to assess the current limitations of the system or gain any significant information of how people interact with such a program.

Description of the experiment:

It involves you taking on the role of a user coming to the website in order to find information to help you make decisions or find out about particular things. This is an information gathering task primarily. Secondly it is a system interaction task. The aim of this is to find out how you approach the system, what kind of language you use when you do so, and how the conversation develops between you and the agent, KIA.

You are asked to use a digital voice recorder to explain orally what your intention is with the system. You may for example decide that you want to find out about travel insurance so you would say:

"I am going to try and find out about what kind of travel insurance is better for me".

The point of this is to draw some information on what you are asking and what you are actually typing in. Spoken queries differ from typed ones, and we want to find out how and why.

Expected outcomes:

From this experiment we hope to gain information on:

- The accuracy of KIA's responses
- The intentions of the users
- What the users actually state their intention is textually
- The kind of language that users employ
- The amount of knowledge required by the system

What to do:

Below you will find a number of scenarios which you will work through in order to complete the task. You are invited to add any additional questions that you want to, and to lead the conversation as you feel fit.

At the bottom of the interface, you will find some radio buttons. These represent the accuracy of the response that KIA gives you. Choose a score to assign to each of the responses rating them from 1 (bad) to 5 (excellent). Once you have selected the score you can carry on with the conversation.

Starting the test:

You will already have been setup with the digital voice recorder, so all that you need to do now is hold the microphone close to your lips when you speak.

You may now start your conversation according to the scenarios below with KIA. Just type in a greeting and proceed.

(don't forget the voice recording!)

Brief and scenarios:

1. **The 18 year old driver**

You play the role of an 18 year old who has just brought their first car and want to find out about car insurance. You might want to find out whether there is a type of insurance aimed specifically at them, ask about the price, the level of cover, how you can save some money. You may also want to compare certain types of insurance to find out more about different aspects of each one and its relative benefits.

How you choose to end the conversation is entirely up to you.

2. **A caterer inquiring about Norwich Union van insurance.**

Here you take on the role of a caterer who wants to find out what the benefits of taking on van insurance with Norwich union are. You sometimes go abroad to stock up on wine, so you want to know if you are covered while you are over there. You also want to know what happens if your van gets stolen, because you might have a lot of stock in it. You're interested in finding out about what the features of the insurance are, and about the price. You decide at the end of your conversation that you would like to take on the insurance and you want to find out how to do that.

Feel free to add any question that you feel are relevant.

3. **The 50 year old person concerned about having a healthy lifestyle**

Here you assume the role of a user who wants to know all about their health, and how they can improve it. They realise that as they are getting older they need to take more care of themselves. Having gained a few pounds, this is an issue as they are concerned about how this impacts their overall health. They want to know how to eat healthily, and understand a little more about nutrition, for example about fat intake, vitamins, etc… Fitness is also important to them and they want to find out what type of exercise would be recommended. They might ask about the benefits of exercise and whether visiting a gym is any good for them at this age. Their partner has arthritis and high blood pressure and they would like to know more about this. They themselves suffer from diabetes, so they would like more information on this too. Perhaps they would like to know more about other health concerns such as these.

4. **The menopausal woman**

You play the role of a woman going through the menopause. This is stressful time in life and you want to know more about it as your knowledge is limited. You are interested in the different phases, the side affects that you can expect, how you'll feel, what others say about their experience, how the body changes, HRT (hormone replacement therapy), what you can do to feel better. The user would also appreciate some contact numbers for organisations that she can perhaps go to for support.

Do ask KIA to explain any terms that you are not familiar with (such as Oestrogen for example), or to say more about a certain topic.

5. **The concerned parent and the hospital visit.**

Here you pretend to be a parent who has to take their child to hospital for the first time. You can choose to divulge as much information as you want to volunteer about this. They are concerned with how to explain to their child what going to hospital will be like. The parent also wants to know what to pack for this visit, whether there is anything in particular which might comfort the child, and what to do afterwards when they come back home.

6. **The pregnant woman (if you are a man go to the next scenario)**

Here you play the role of a pregnant woman. Her concerns are about what to eat, and what kind of exercise she should do. She also suffers from morning sickness and wants to know what she can do about this, and why it happens. She wants to know more about why she should breast-feed and what to do after the birth, as she wants to get her figure back. Feel free to ask about any helpful literature that can be recommended and any additional things that you think are important to a woman during pregnancy.

7. **The father to be (Do not do this scenario if you did number 6).**

You are a father to be and you nervous about what to expect and what you should be doing during this time. You want to know about antenatal sessions, and how to prepare for the arrival. You are also concerned about how your wife feels, because she is having severe mood swings and its getting you down. You want to find out about the labour, and what happens if the mother needs a caesarian. You are also worried about the amount of pain that she will have to endure and what can be done to help this. You also want to know what happens after the birth, when the baby comes home, what you should expect.

Again, do feel free to ask KIA for any additional information you might be interested in or need.

When you have finished:

1. Say goodbye to KIA (it's only polite!)
2. Call over a system developer
3. Debrief
4. Collect a lollipop

Thank you for your time!!!

Appendix D

KIA1 User feedback form

This is the user feedback form that we used for the KIA tests.

KIA FEEDBACK FORM

Question	Comments
Please describe your experience with KIA.	
Were the answers useful?	
Did you find the tone of the responses adequate?	
Do you recall any answers out of the ordinary/not expected?	
Did you find that the conversation turn-taking worked well?	
Compared to scanning the web site, how readily available was the information?	
Do you think that being provided with the relevant links to the location of the information was useful?	
On a rating of 1-10, how trustworthy did you find KIA? (1- Totally Untrustworthy 10-Totally trustworthy)	
What degree of emotional response did you detect from KIA?	

How did you feel about the quality of the chatbot in general?	
Which word best describes your conversation with KIA? Smooth Easy Difficult Awkward Another??	
Did KIA appear to show an interest in what you were talking about?	
Were the answers succinct or long-winded?	
Were the answers too simple or too complex?	
Was the information given clear?	
Do you think that the agent would benefit from a more human-like virtual person with facial expressions? If so why?	
Would you use KIA again for this type of task?	
What would you change?	

Appendix E

Oral brief transcript samples

This is a sample from the oral experiement, using the Aviva website.

Oral brief transcripts from the KIA1 experiment

1)

Customer: I'd like to find out a bit more about van insurance with Norwich union
Agent: Ok, the van insurance we offer is flexible. And we provide commercial van insurance for your business but we can provide personal van insurance if you're an individual, so we can service the needs of both.
Customer: well I'm actually looking for commercial van insurance because I'm a caterer and I often go abroad to stock up on wine. Would your insurance cover me if my van got stolen, or the contents stolen?
Agent: we do have contents insurance for your van and that's part of the commercial van insurance…so you'd like to know what happens with your contents basically – everything is covered
Customer: do I have to pay anything extra?
Agent: um yes, this is a bit that you bolt on
Customer: would my insurance cover vandalism?
Agent: yes that comes under loss, theft or damage to your van. If it's stolen or damaged we pay for your van to be repaired or replace it, we pay the amount of the loss or damage in cash.
Customer: and do you offer any discounts if I insure with you for another year?
Agent: No we don't offer any discounts
Customer: If I find cheaper insurance elsewhere will you match it?
Agent: um, we could try.
Customer: I'd like to take some insurance out so how do I go about that?
Agent: you can go and get a quote or you would need to talk to an advisor

2)

Customer: Hi, I've just turned 18 and I've just brought my first car, I was just wondering of you could tell me what sort of insurance cover you would offer me
Agent: Yes. We offer a special policy for young drivers called rapid bonus. This increases the level of no claims bonus. You get more no claims quicker – check it out
Customer: Ok, thanks.
Agent: Is that any good
Customer: Yea.
Roughly how much will this cost?
Agent: I can't tell you how much it will cost, you'd have to go though a quote form, BUT I can tell you that you get 12 months no claim bonus in 9 months. Hence it's quicker
Customer: That's great thank you. Also what level of cover is this type of insurance
Agent: Comprehensive and 3rd party
Customer: That's all for me I think
Agent: Glad to be of service
Customer: Thanks

Customer: Do you do van insurance?
Agent: Yes we do offer van insurance, is there any in particular that you are looking for?
Customer: Yes, if I have van insurance with you am I covered for traveling abroad?

And what sort of cover do I get in terms of stock, that may be contained in my van? Cause I'm a caterer and I want to know if the food contained in my van is covered abroad.
Agent: We offer 3 levels of cover for van insurance. Comprehensive cover, third party and theft and third party cover. In terms of whether you're covered for personal belongings, there a £100 limit per any incident that may happen on your insurance.
Customer: Is the £100 the same for all the levels of van cover?
Agent: This is only on comprehensive cover
Customer: Can I extend my £100 limit?
Agent: I don't think so no.
Customer: Are you sure?
Agent: Yes
Customer: What's "extend your foreign use cover"?
Agent: Ok, you can extend it but personal belongings are not covered under this
Customer: Not much use to me but at least I got an answer. How much is your van insurance?
Customer: Can't actually give you a value for this because it depends on the criteria you enter in the form but if you go to our website you'll find the quote form there.
Customer: Ok. I'd like to take our your insurance so what should I do?
Agent: First of all if you get a quote on our online quote application that'll give you the option to purchase online using various payment methods.
Customer: Ok cool, thanks.

3)

Customer: Hello I'm looking for car insurance
Agent: What sort of insurance, are you looking 3rd party or comprehensive?
Customer: Well it depends on the price really, comprehensive probably.
Agent: Alright well comprehensive – you know what I mean by the difference between the two?
Customer: Not really sure
Agent: Hang on a second I'll find the exact details (context)
Customer: Basically I want the cheapest one that I can get (context)
Agent: Well it depends; it's not necessarily cheaper for third party. For comprehensive it depends on the level of cover and the type of car. If you have a car worth a £1000, it would be cheaper to have third party.
Customer: So what am I covered for
Agent: Loss of , theft or damage of your car, accidental damage, new car replacement, and that's the main difference.
Customer: And what about 3rd party?
Agent: Loss of , theft or damage of your car by fire or if other people claim against you. Emergency medical discount, no claims discount, …
Customer: Ok, so I'd expect the comprehensive would be more expensive
Agent: Yes, in these circumstances that is the case
Customer: Because I've heard that being a young driver, obviously, insurance is going to be more expensive. Do you have any products that are aimed specifically at my age group?
Agent: We do.
Customer: Because I'm only 18
Agent: Rapid bonus probably, that allows you to build up your no claims discount quickly, its specifically aimed at young drivers, allows you to build up a no claims bonus,

and therefore lower your future motor insurance premiums. This would be quite similar in the next 5 months. It gets you to 9 months instead of 12 months. It'll decrease costs quite significantly quite fast.

Customer: That sounds handy. I've hear of something like "pay as you drive" or something, what's that all about?

Agent: This is quite a good product, what it allows you to do is record when you drive and NU did a study and found that young drivers are more likely to have an accident between 11 o'clock at night and 12 in the morning so we charge less when you're driving from 6am during the day and more after 11 o'clock at night recognizing that you're more likely to have an accident at those periods. On the whole that comes up at a lot less expensive than normal car insurance. What you do have to do is get a black box to your car, which enables us to track what you're doing with your car so we are able to record it.

Customer: And is there a cost to that

Agent: At the moment it's in a test period, so I think it's included in the price. What I could do is put you through to people who deal with this. The telephone number is 0800014493

Customer: Ok. I think that sounds quite interesting. I may follow you up on that. I'll give them a call.

Agent: Only one thing, you need to have a car that is 1996 or newer to register.

Customer: Yep my car is newer than that.

Agent: Ok

Customer: Thank you

Customer: 'm looking for van insurance, I'm a caterer so, I want something so if something breaks down I want to know that I can get a replacement, that's really important.

Agent: You want accident recovery?

Customer: No if my van is basically it's a courtesy car

Agent: Emergency replacement van cover – so in the event of an accident, we'll be able to supply a replacement vehicle until your van is recovered or your van is repaired.

Customer: Alright, that's good. Also will my stuff in the van be covered like for instance if I go to France I'd stock up with champagne for weddings and things like that, would that be covered, if there was stuff in the van?

Agent: I can't find the information so I can direct you to the website…

Customer: I'd rather phone in

Agent: You'd like to phone in?

Customer: Yea I think I would.

Agent: Ok, you can contact our call center on 0800564387

Customer: Thank you

Appendix F

Results from the KIA user experiment

These are the results from the first KIA user experiment.

0-1 normalized **Respondents**

Feedback Questions	User 1	User 2	User 3	User 4	User 5	User 6	User 7
Experience	0	0.5	0.6	1	1	1	0
useful answers	0	0	0.3	0.5	0.5	0.5	0.5
tone	0	1	1	1	1	0	1
unexpected things	0	0	0	0	0	1	0
turn-taking	0	1	n/a	1	0	0	1
comparing to website surfing	0	1	1	1	0.5	0	0
links useful or not	1	1	1	1	1	1	1
trust rating	2	3	3	9	5	5	5
emotion	0	0.5	0	0.5	0	0	0
quality of bot	0	0	0	1	0.5	0	0.5
conversation rating	0	0	0	1	0	0	0
interest shown	0	0.5	0	1	0.5	0	0
succint/longwinded	1	1	1	1	1	1	1
complex/simple	1	1	1	0	1	1	1
clear	0	0.5	0.5	1	1	1	1
virtual agent	0	1	0	0		0	0
use again?	0	1	0	1	0	0	0
changes	understanding	press enter and clarification	repetition	none		convo flow - more link info	

0-1 normalized **Respondents**

Feedback Questions	User 8	User 9	User 10	User 11	User 12	User 13	User 14	User 15	User 16
Experience	1	0	0	0.5	1	0	1	1	1
useful answers	0	0.5	1	0	0.5	0	0	0.5	0.5
tone	0	1	0.5	1	1	1	1	1	0.5
unexpected things	0	0	0	0	0	0	0	1	0
turn-taking	1	0	0.5	1	0	0	1	1	1
comparing to website surfing	0	0	0.5	0	1	0	1	1	0
links useful or not	1	1	1	1	1	1	0.5	1	1
trust rating	10	4	5	8	5	3	8	5	5
emotion	0	0.5	0	1	0	0	1	0	0
quality of bot	0	0	0	0	1	0.5	0.5	0	0
conversation rating	0	0	0	0	0.5	0	1	1	0
interest shown	0	0	0	0	1	0	0	0.5	0
succint/longwinded	0	1	0.5	0	1	0	1	0.5	1
complex/simple	0	1	1	0	1	1	0	1	0
clear	1	1	0.5	0	1	1	1	0.5	0.5
virtual agent	0	1	0	1	0	0	0	1	0
use again?	0	0	0	0	0.5	0	0.5	0.5	0
changes	more detal and intelligence	more links	links	better communication		hate it	better interraction	lots	slow

0-1 normalized **Respondents**

Feedback Questions	**User 17**	**User 18**	**User 19**	**User 20**	**User 21**	**User 22**	**User 23**	**User 24**	**User 25**	**User 26**	**User 27**	**User 28**
Experience	0	0	0	0	0.5	0	1	0	1	0	0	1
useful answers	0	0	0.5	1	0	0.5	0.5	1	0.5	0.5	0.5	0
tone	0.5	0	0	1	0	1	1	1	1	0	1	0
unexpected things	0	0.5	0	0	0	0	1	0	0	0	0.5	1
turn-taking	1	0	0	0	0	1	1	1	0.5	0	0	0
comparing to website surfing	1	1	0.5	0	0	0	1		1	0	0	0
links useful or not	1	1	1	1	1	0.5	1	0	1	1	0.5	1
trust rating	2	1	0	0.5	0	1	1	0	0	0	0	0
emotion	0	0	0	0	0	1	0	1	1	0	0	0
quality of bot	0	0	0	0	0	0	0.5	0	0.5	0	0	0
conversation rating	0	1	0	0	10	1	1	0	0.5	0.5	0	0
interest shown	0	1	0.5	1	0	0	1	1	0.5	0	1	0
succint/longwinded	1	1	0	1	0	1	1	1	0	0	0	0
complex/simple	1	1	0.5	1	0	1	1	1	0.5	0	1	0
clear	1	1	1	1	0	1	1	1	0	0	0	0.5
virtual agent	0	0	0.5	1	0	1	0	0	0.5	1	0	0.5
use again?	0	0	0	0	0	0			0.5	0	0	0
changes	communication						liked presentation					

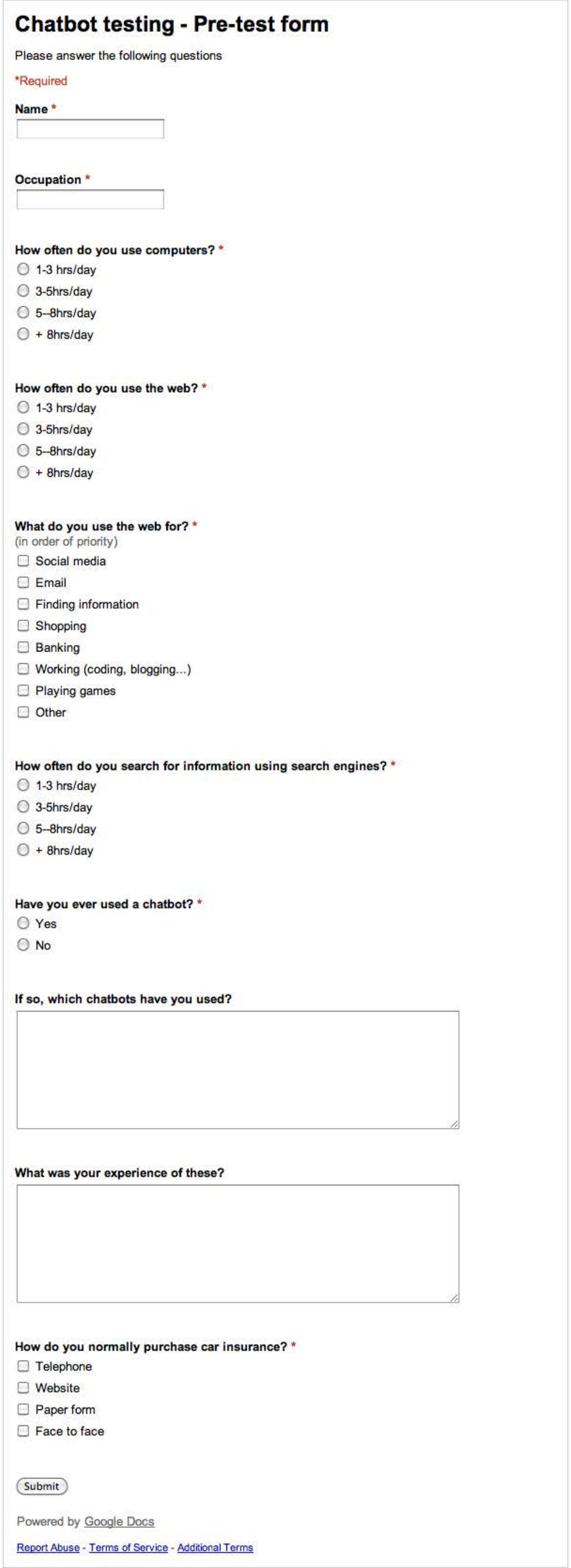
0-1 normalized **Respondents**

Feedback Questions	**User 29**	**User 30**	**User 31**	**User 32**
Experience	1	0	1	0.46
useful answers	0.5	0.5	1	0.37
tone	1	0	1	0.65
unexpected things	1	0	1	0.2
turn-taking	0	1	1	0.46
comparing to website surfing	0.5	1	1.5	0.43
links useful or not	1	1	2	0.91
trust rating	0	0	0	3.01
emotion	0	0.5	0.5	0.23
quality of bot	0	0	0	0.16
conversation rating	0	0	0	0.58
interest shown	0	0.5	0.5	0.33
succint/longwinded	1	1	2	0.66
complex/simple	1	1	2	0.7
clear	0	1	1	0.66
virtual agent	0	0	0	0.28
use again?	1	1	2	0.2
changes				

Appendix G

KIA2 Pre-test form

This is the pre-test form that we used for the KIA2 experiment.



Chatbot testing - Pre-test form

Please answer the following questions

*Required

Name *

Occupation *

How often do you use computers? *

○ 1-3 hrs/day
○ 3-5hrs/day
○ 5--8hrs/day
○ + 8hrs/day

How often do you use the web? *

○ 1-3 hrs/day
○ 3-5hrs/day
○ 5--8hrs/day
○ + 8hrs/day

What do you use the web for? *
(in order of priority)

☐ Social media
☐ Email
☐ Finding information
☐ Shopping
☐ Banking
☐ Working (coding, blogging...)
☐ Playing games
☐ Other

How often do you search for information using search engines? *

○ 1-3 hrs/day
○ 3-5hrs/day
○ 5--8hrs/day
○ + 8hrs/day

Have you ever used a chatbot? *

○ Yes
○ No

If so, which chatbots have you used?

What was your experience of these?

How do you normally purchase car insurance? *

☐ Telephone
☐ Website
☐ Paper form
☐ Face to face

Powered by Google Docs

Appendix H

KIA2 pre-test responses

These are the responses from the pre-test form on the KIA2 user experiment.

Chatbot testing - Pre-test form

Name	Occupation	How often do you use computers?	How often do you use the web?	What do you use the web for?	How often do you search for information using search engines?	Have you ever used a chatbot?	If so, which chatbots have you used?	What was your experience of these?	How do you normally purchase car insurance?
Julie	actress	1-3 hrs/day	1-3 hrs/day	Social media, Email, Finding information, Shopping	1-3 hrs/day	No			Telephone
Oliver	Accountant	5--8hrs/day	3-5hrs/day	Social media, Email, Finding information, Shopping, Banking	1-3 hrs/day	No			Website, Paper form, Face to face
Lucas	Account director	+ 8hrs/day	3-5hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...)	3-5hrs/day	Yes	AliceBot, Trainline	Poor	Website
Lucie	Gardener	1-3 hrs/day	1-3 hrs/day	Email, Finding information, Working (coding, blogging...)	1-3 hrs/day	No			Telephone, Website
Tina	online marketeer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	5--8hrs/day	Yes	Alice, Ikea	They are not very useful	Website
Jack	SEO	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	5--8hrs/day	Yes	Eliza	Fun	Telephone, Website
Jim	Developer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	5--8hrs/day	Yes	Alice, Eliza, Jabberwacky	They were pretty useless, but fun to use for a little while	Telephone, Website
Kahela	student	5--8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Working (coding, blogging...)	1-3 hrs/day	No			Telephone, Website

Chatbot testing - Pre-test form

Name	Occupation	How often do you use computers?	How often do you use the web?	What do you use the web for?	How often do you search for information using search engines?	Have you ever used a chatbot?	If so, which chatbots have you used?	What was your experience of these?	How do you normally purchase car insurance?
Hanna	Art director	+ 8hrs/day	3-5hrs/day	Social media, Email, Finding information, Working (coding, blogging...)	1-3 hrs/day	Yes	Ikea	totally useless	Telephone, Website, Paper form
Ricardo	Surf shop assistant	1-3 hrs/day	1-3 hrs/day	Email, Finding information, Banking	1-3 hrs/day	No			Face to face
Michela	Yoga teacher	3-5hrs/day	3-5hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...)	1-3 hrs/day	No			Telephone, Website, Face to face
Dylan	window cleaner	3-5hrs/day	3-5hrs/day	Social media, Email, Finding information, Working (coding, blogging...)	1-3 hrs/day	No			Telephone, Website
Maria	Social worker	1-3 hrs/day	1-3 hrs/day	Social media, Email	1-3 hrs/day	No			Telephone, Face to face
Adelaide	PA	5--8hrs/day	3-5hrs/day	Social media, Email, Finding information, Shopping, Working (coding, blogging...)	3-5hrs/day	No			Telephone, Website, Paper form, Face to face
Malcolm	Developer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	5--8hrs/day	Yes	Can't remember	Boring	Telephone, Website
Helena	Manager	5--8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...)	3-5hrs/day	Yes	Ikea chatbot	I wouldn't bother using it again as it was unhelpful	Telephone, Website
Carla	Junior art director	5--8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Working (coding, blogging...), Other	3-5hrs/day	Yes	Alice	Quite good, I like asking it weird questions	Telephone, Website

Chatbot testing - Pre-test form

Name	Occupation	How often do you use computers?	How often do you use the web?	What do you use the web for?	How often do you search for information using search engines?	Have you ever used a chatbot?	If so, which chatbots have you used?	What was your experience of these?	How do you normally purchase car insurance?
Joe	massage therapist	1-3 hrs/day	1-3 hrs/day	Social media, Email, Finding information	1-3 hrs/day	No			Telephone
Andrew	Developer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	5--8hrs/day	Yes	Alice, Jabberwacky, some others	They are really primitive and not really useful for anything.	Website
Rebecca	Analyst	+ 8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Working (coding, blogging...)	3-5hrs/day	No			Telephone, Website
Patrick	social media marketer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Working (coding, blogging...), Playing games, Other	5--8hrs/day	No			Website
Vivian	account manager	5--8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...)	1-3 hrs/day	No			Telephone
Leno	chef	1-3 hrs/day	1-3 hrs/day	Social media, Email	1-3 hrs/day	No			Telephone, Website
Paul	Architect	5--8hrs/day	1-3 hrs/day	Email, Finding information, Shopping, Banking	1-3 hrs/day	No			Telephone, Website
Hannah	physiotherapist	3-5hrs/day	3-5hrs/day	Social media, Email, Finding information, Banking	3-5hrs/day	No			Telephone
Linda	Designer	+ 8hrs/day	3-5hrs/day	Social media, Email, Finding information, Shopping	1-3 hrs/day	No			Website

Chatbot testing - Pre-test form

Name	Occupation	How often do you use computers?	How often do you use the web?	What do you use the web for?	How often do you search for information using search engines?	Have you ever used a chatbot?	If so, which chatbots have you used?	What was your experience of these?	How do you normally purchase car insurance?
Chris	Developer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	3-5hrs/day	No			Website
Ross	Developer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	1-3 hrs/day	Yes	KIA, Alice, Jabberwocky, Trainline	Not very useful, I can live without them.	Website
Paul	Developer	+ 8hrs/day	5--8hrs/day	Social media, Email, Finding information, Banking	1-3 hrs/day	Yes	Alice	Pretty lame	Website
Julia	journalist	5--8hrs/day	3-5hrs/day	Social media, Email, Finding information, Working (coding, blogging...)	1-3 hrs/day	Yes	I don't remember	It was a bit of fun	Website
Emma	Artist	1-3 hrs/day	1-3 hrs/day	Email, Finding information	1-3 hrs/day	No			Telephone
Gary	account manager	5--8hrs/day	5--8hrs/day	Email, Finding information, Working (coding, blogging...)	1-3 hrs/day	No			Telephone, Website
Richard	Developer	+ 8hrs/day	+ 8hrs/day	Social media, Email, Finding information, Banking, Working (coding, blogging...), Playing games, Other	1-3 hrs/day	Yes	KIA, Alice	very very boring	Telephone, Website
Sophie	Manager	5--8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Banking	1-3 hrs/day	No			Telephone, Website
Sara	mum	1-3 hrs/day	1-3 hrs/day	Social media, Email, Finding information, Shopping, Banking, Playing games	1-3 hrs/day	No			Telephone

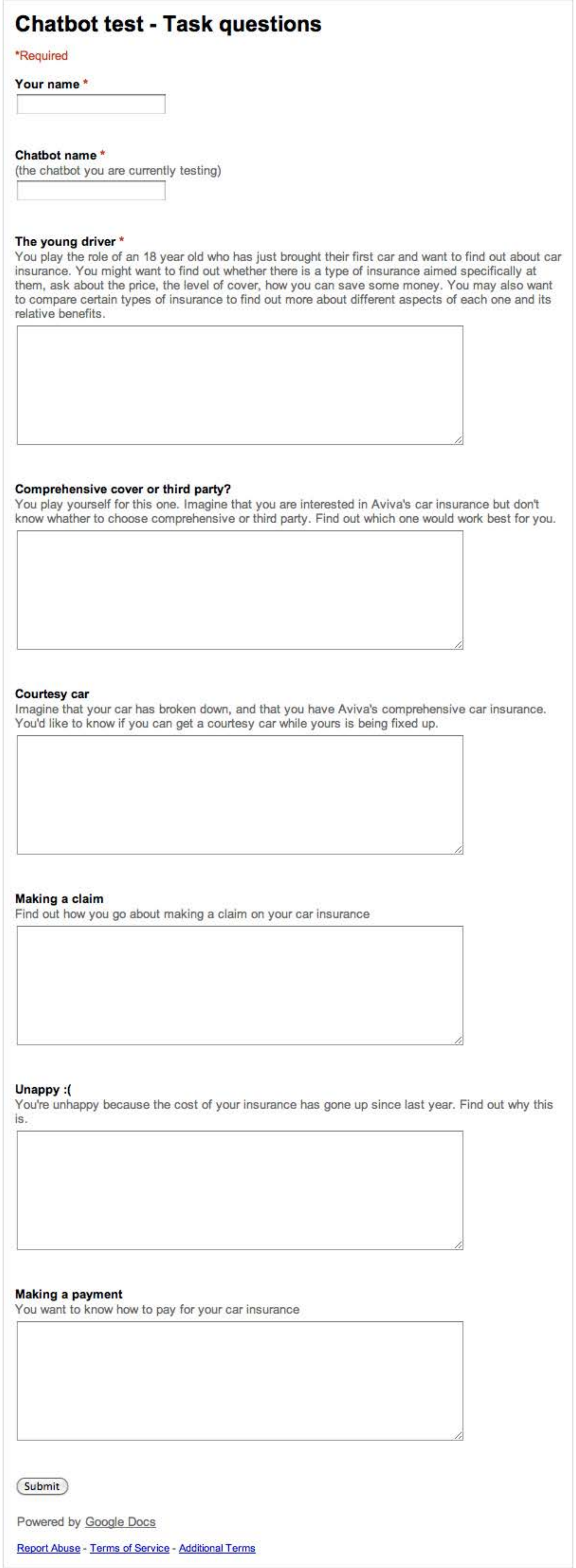
Chatbot testing - Pre-test form

Name	Occupation	How often do you use computers?	How often do you use the web?	What do you use the web for?	How often do you search for information using search engines?	Have you ever used a chatbot?	If so, which chatbots have you used?	What was your experience of these?	How do you normally purchase car insurance?
Louise	mum	3-5hrs/day	1-3 hrs/day	Email, Finding information, Shopping	1-3 hrs/day	No			Telephone
Stewart	fitness instructor	3-5hrs/day	3-5hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...)	1-3 hrs/day	No			Website
Jamie	student	+ 8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	5--8hrs/day	No			Telephone, Website
Lucy	Designer	+ 8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games	3-5hrs/day	No			Telephone, Website
Justine	Designer	+ 8hrs/day	5--8hrs/day	Social media, Email, Finding information, Shopping, Banking, Working (coding, blogging...), Playing games, Other	1-3 hrs/day	Yes	Alice	confusing	Website

Appendix I

KIA2 Tasks

These are the tasks and instructions given to users for the KIA2 experiment.



Chatbot test - Task questions

*Required

Your name *

Chatbot name *
(the chatbot you are currently testing)

The young driver *
You play the role of an 18 year old who has just brought their first car and want to find out about car insurance. You might want to find out whether there is a type of insurance aimed specifically at them, ask about the price, the level of cover, how you can save some money. You may also want to compare certain types of insurance to find out more about different aspects of each one and its relative benefits.

Comprehensive cover or third party?
You play yourself for this one. Imagine that you are interested in Aviva's car insurance but don't know whather to choose comprehensive or third party. Find out which one would work best for you.

Courtesy car
Imagine that your car has broken down, and that you have Aviva's comprehensive car insurance. You'd like to know if you can get a courtesy car while yours is being fixed up.

Making a claim
Find out how you go about making a claim on your car insurance

Unappy :(
You're unhappy because the cost of your insurance has gone up since last year. Find out why this is.

Making a payment
You want to know how to pay for your car insurance

Powered by Google Docs

Report Abuse - Terms of Service - Additional Terms

Appendix J

KIA conversation sample

This is a KIA SOCS conversation sample.

KIA1 conversation sample:

(please note: exact user input is reproduced including ill-formed grammar and typos)

1)

User: How do I apply for Rapid Bonus?
KIA: Rapid Bonus is a specific young driver car insurance policy. You can build up your no claims insurance quickly so your premiums get lower. You can get a further discount if you pass the pass plus test.
User: tell me how I can get cheap car insurance
KIA: I don't know
User: I am 18 and have just bought a car so dont have much money left how do I get cheap car insurance?
KIA: New Rapid Bonus option, a fast track, no claim discount policy, for safe young drivers
User: ok cool how do i apply for that?
KIA: To get a reliable quote I suggest you fill in a quote form or speak to one of our advisors on 0800 092 9564. I can help you with the quote form if you like.
User: yes please I would like some help on the quote form
KIA: You can use the quote form we provide on the right hand corner of the page.
User: Thanks
KIA: You're welcome!

2)

User: I need insurance for when i drive my van in france
KIA: If you ask us, we can extend your policy to provide the same level of cover in the rest of the EU and certain other European countries as you have within the territorial limits. To get this cover call us on 0800 559 3213
User: how much does that cost?
KIA: To get an accurate quote it's easier for you to use our online quote form. It's on the webpage in the right hand corner.
User: What about the stock in my van at the time it is stolen?
KIA: If your van is lost, stolen or damaged, we will either pay for your van to be repaired replace your van pay the amount of the loss or damage in cash. We'll also cover accessories, spare parts, fixtures and fittings and audio equipment
User: Is the contents of my van insured too?
KIA: We'll pay a maximum of £100 for any one incident for the loss or damage to the content of your van
User: how much is your van isnnurance?
KIA: To get an accurate quote it's easier for you to use our online quote form. It's on the webpage in the right hand corner.
User: Thanks
KIA: You're welcome!

3)

User: can you tell me how I can improve my fitness please?
KIA: Could you please be more specific?
User: I need to improve my health and would like to lose some weight and look after myself a bit better
KIA: Reassessing your diet is a good place to start. You can have a bad diet through lack of variety. Exercise is also really good for a healthy lifestyle and a balanced weight.
User: how does exercise help?
KIA: you should excercise regularly, walking, joining a gym, cycling, anything that can improve your flexibility and cardiovascular activity.
User: why?
KIA: Sorry, could you rephrase that please?
User: why does that help me?
KIA: I don't quite understand what you mean, can you explain?
User: in what way is exercise good for me?
KIA: you should excercise regularly, walking, joining a gym, cycling, anything that can improve your flexibility and cardiovascular activity.
User: Thanks, Bye for now!
KIA: You're welcome!

4)

User: How do I remain at a healthy weight?
KIA: The body mass index (BMI) calculation can give you some idea of where you should be. There's a tool on the site that will calculate this for you.
User: what types of food should I be eating?
KIA: Its important to keep a varied diet and to warn off illness. Cereals and fruit and vegetables will also help boost your energy levels and help you lose weight.
User: which cereals should I eat?
KIA: Can you be more specific?
User: what fruit is the best for me?
KIA: Can you tell me more?
User: I dont like bananas but I like apples and oranges and pears and melons, are these all good or are some better than others?
KIA: Could you clarify that for me please?
User: What food should I eat to make sure I get all the vitamins I need?
KIA: Its important to keep a varied diet and to warn off illness. Cereals and fruit and vegetables will also help boost your energy levels and help you lose weight.
User: ooook. Thanks.
KIA: You're welcome!

5)

User: hi kia, i'm paul
KIA: Hi, welcome to the Norwich Union website!
User: thanks, is car insurance for younger drivers expensive?
KIA: New Rapid Bonus option, a fast track, no claim discount policy, for safe young drivers
User: what does rapid bonus cover?
KIA: Rapid Bonus is a specific young driver car insurance policy. You can build up your no claims insurance quicky so your premiums get lower. You can get a further discount if you pass the pass plus test.
User: what is the pass plus test?
KIA: It's a course to help young drivers gain valuable additional driving experience that may not be covered in driving lessons. It is backed by the Government's Driving Standards Agency and there is no test to take at the end.
User: how can I take the pass plus test?
KIA: I'm not entirey sure, sorry.
User: That's ok, bye for now.
KIA: Bye!

Appendix K

KIA2 conversation sample

This is a KIA2 conversation sample.

KIA2 conversation sample:

(please note: exact user and KIA2 input/output is reproduced including ill-formed grammar and typos)

1)

User: Hi, I just passed my test and I'd like to get car insurance
KIA2: You can get a car insurance quote on this website or you can call customer service
User: Do I get cheaper insurance because I'm a young driver?
KIA2: Rapid Bonus is a specific young driver car insurance policy. You can build up your no claims insurance quickly, so your premiums get lower. There is a further discount if you pass the “Pass Plus” test.
User: What’s Pass Plus?
KIA2: “Pass Plus” is a training course aimed at new drivers, designed by the Driving Standards Agency. It gives you advanced driving skills.
User: Is it expensive?
KIA: For pricing please fill in an online quote form
User: How can I find out?
KIA: I don’t know
User: How much is Rapid Bonus?
KIA: Please use the online quote form or call customer service
User: Ok, thanks
KIA: You’re welcome

2)

User: What's the difference between third party and comprehensive?
KIA: Comprehensive insurance including third party insurance
User: right, I see. So which one is better?
KIA: I don’t know
User: Does comprehensive inculde accident recovery?
KIA: Comprehensive insurance includes accident recovery
User: What happens if I have an accident then?
KIA: In the event of an accident, your car is taken to a garage of your choice
User: How much is it?
KIA: To get a quote you need to fill in an online quote form or call customer service
User: Cheers
KIA: You’re welcome

3)

User: How do I make a claim?

KIA: To make a claim fill in a claim form
User: Ok, where do I find one
KIA: I don’t know
User: Where is the claim form?
KIA: There is an online claims form
User: where?
KIA: I don’t know
User: Where can I find the online claims form?
KIA: The online claims form is online
User: thanks for nothing
KIA: You’re welcome

4)

User: I'm not very happy because my insurance is a lot more expensive than it was last year.
KIA: Sorry you’re unhappy insurance is expensive
User: But why has the price gone up?
KIA: I don’t know
User: It’s too expensive
KIA: For more information on pricing use the online form
User: How can I make my insurance cheaper?
KIA: Please fill in an online quote form or call customer service

5)

User: How can I pay for my car insurance?
KIA: You can pay online
User: What credit cards do you accept
KIA: We accept credit cards
User: yes but which ones? Do you accept Visa?
KIA: You can pay with Visa or Mastercard
User: Thanks
KIA: You’re welcome

6)

User: I have a cracked windscreen, how do I claim on the insurance?
KIA: We pay for repairs
User: How do I make a claim?
KIA: You can make a claim online or call customer service
User: ok cheers
KIA: You’re welcome

7)

User: I have 1 year's extra no claims bonus, but my insurance has gone up, why??
KIA: Comprehensive insurance includes no claims discount
User: I already have comprehensive insurance, why has the price gone up?
KIA: You can pay online for comprehensive car insurance
User: I am already insured
KIA: How can I help?
User: Tell me why the price has gone up
KIA: I don't know
User: you could have told me that earlier
KIA: I don't know
User: Bye
KIA: Good bye

8)

User: Hey - just got my first car and apparently i need insurence!
KIA: Rapid Bonus is a specific young driver car insurance policy.
User: How much does it cost?
KIA: For pricing please fill in an online quote form
User: Which insurance is the cheapest?
KIA: For pricing please fill in an online quote form
User: ok ok, I get it.
KIA: ok
User: As a young driver, can I get short-term insurance?
KIA: Short-term insurance is for over 21's
User: I am 18
KIA: Rapid Bonus is a specific young driver car insurance policy. You can build up your no claims insurance quickly, so your premiums get lower. There is a further discount if you pass the "Pass Plus" test.
User: So Rapid bonus is my only option?
KIA: Pass Plus is a training scheme for new drivers.
User: k thanks
KIA: You're welcome

Appendix L

HCI User Experiment Raw Data

This is the raw data from the HCI user experiment for the KIA2 system, comparing it to Avivabot and KIA.

AVIVABOT	**User 1**	**User 2**	**User 3**	**User 4**	**User 5**	**User 6**	**User 7**	**User 8**	**User 9**	**User 10**	**User 11**	**User 12**	**User 13**	**User 14**	**User 15**	**User 16**
Experience	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tone	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Turn Taking	1	1	2	1	1	1	2	2	1	1	2	2	1	2	1	1
Links useful	0	2	0	1	2	0	1	1	1	1	1	1	0	1	0	1
Emotion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Conversation rating	0	0	1	1	0	0	0	1	0	0	0	0	0	1	0	0
Succinct responses	1	1	1	1	1	0	0	0	1	0	0	0	0	0	1	1
Clear answers	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	1
Useful answers	0	1	1	1	1	1	1	1	0	1	1	1	1	0	1	1
Better than site surfing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Quality	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Interest shown	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Simple to use	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1
Would use again	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sample Mean	0.29	where n = 588
Sample Standard Deviation	0.51	where variance S^2 = 0.26
Standard Error of the Mean	0.02	
Confidence Interval of the Mean @ 95%	0.04	where z = 1.96

User 17	User 18	User 19	User 20	User 21	User 22	User 23	User 24	User 25	User 26	User 27	User 28	User 29	User 30	User 31	User 32	User 33
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	1	0	0	0	1	1	0	0	1	0	0	1
1	1	1	2	1	2	2	0	1	0	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	1	1	0	0	0	1	1	0	0	0
1	0	0	0	0	0	1	1	0	0	0	1	0	0	1	0	0
0	1	1	0	0	1	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0	1	1	1	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

User 34	User 35	User 36	User 37	User 38	User 39	User 40	User 41	User 42	Average
0	0	0	0	0	0	0	0	0	0.0
0	1	1	0	0	1	1	0	0	0.1
0	2	2	1	2	2	1	0	2	1.0
1	1	0	0	1	0	0	1	0	0.7
0	0	0	0	0	0	0	0	0	0.0
0	0	0	0	0	0	0	0	0	0.2
0	0	1	0	0	1	0	1	0	0.4
0	0	0	0	0	0	0	0	0	0.2
0	1	0	0	0	0	0	1	0	0.5
0	0	0	0	0	0	0	0	0	0.0
0	0	1	0	0	0	1	0	1	0.2
0	0	0	0	0	0	0	0	0	0.0
1	0	0	1	0	1	0	1	1	0.7
0	0	0	0	0	0	0	0	0	0.0

KIA	**User 1**	**User 2**	**User 3**	**User 4**	**User 5**	**User 6**	**User 7**	**User 8**	**User 9**	**User 10**	**User 11**	**User 12**	**User 13**	**User 14**	**User 15**	**User 16**
Experience	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1
Tone	0	0	1	0	0	0	1	0	1	1	0	0	1	0	0	0
Turn Taking	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	0
Links useful	2	2	1	1	0	0	1	0	1	1	1	0	1	2	2	2
Emotion	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
Conversation rating	1	0	0	1	0	1	0	0	1	1	1	0	1	0	0	1
Succinct responses	1	0	1	1	1	1	1	1	0	1	1	1	0	1	1	1
Clear answers	0	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1
Useful answers	1	0	1	1	0	1	0	0	1	1	0	1	0	1	0	0
Better than site surfing	0	0	0	1	0	1	1	0	0	1	1	1	0	0	1	0
Quality	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
Interest shown	0	1	1	0	0	0	1	1	0	1	0	0	0	0	1	1
Simple to use	1	0	0	2	1	1	1	0	0	1	1	1	2	0	1	1
Would use again	1	0	0	1	1	1	1	0	0	0	1	1	1	1	0	1

Sample	0.50	where $n = 588$
Sample Standard Deviation	0.55	where variance $S^2 = 0.30$
Standard Error of the Mean	0.02	
Confidence Interval of the Mean @ 95%	0.04	where $z = 1.96$

User 17	User 18	User 19	User 20	User 21	User 22	User 23	User 24	User 25	User 26	User 27	User 28	User 29	User 30	User 31	User 32	User 33
0	1	1	1	1	0	1	1	0	0	0	1	1	1	0	0	0
0	1	0	1	1	0	1	1	1	1	0	0	0	1	0	0	1
0	0	0	0	1	1	0	1	0	1	0	0	0	0	1	0	1
2	0	1	1	1	0	0	2	0	0	2	1	2	1	0	0	2
1	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
0	1	0	1	0	0	1	0	0	0	1	0	1	2	2	0	1
1	0	1	1	0	1	1	1	0	0	1	0	1	1	0	0	1
1	0	1	1	1	1	0	1	1	1	1	0	1	1	1	0	0
1	0	0	0	0	1	0	0	1	1	0	1	0	0	0	1	0
1	0	0	0	1	0	1	1	1	1	0	0	1	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	1
1	1	0	1	0	1	1	1	0	0	0	0	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	0	1	1	0	1	1	0

User 34	User 35	User 36	User 37	User 38	User 39	User 40	User 41	User 42	Average
1	1	1	0	1	0	1	0	1	0.45
0	0	0	0	0	0	0	1	0	0.36
0	0	0	0	0	0	0	1	0	0.45
2	0	1	0	1	0	0	1	1	0.90
1	0	0	0	0	0	1	0	1	0.24
1	0	1	0	0	1	1	1	1	0.57
1	1	0	1	0	0	0	1	1	0.67
1	1	1	0	0	1	0	1	0	0.67
0	0	0	0	1	0	0	0	1	0.38
0	0	1	1	0	0	0	0	1	0.43
0	0	0	0	0	0	1	0	1	0.17
0	0	1	0	1	0	0	1	0	0.33
1	1	0	0	1	0	1	1	0	0.69
1	1	1	1	1	1	1	1	1	0.69

KIA2	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	User 11	User 12	User 13	User 14	User 15	User 16
Experience	0	1	0	0	1	1	1	1	0	1	0	1	0	0	1	0
Tone	0	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1
Turn Taking	2	0	1	1	2	1	2	1	2	1	0	1	1	2	1	0
Links useful	1	1	1	1	0	1	1	0	1	0	1	1	0	1	0	0
Emotion	1	1	1	1	0	0	1	0	1	1	0	0	0	1	1	1
Conversation rating	2	2	2	0	1	2	0	1	0	1	0	0	0	0	1	1
Succinct responses	0	0	1	0	1	0	1	0	1	0	0	0	1	1	1	1
Clear answers	1	0	1	1	0	1	1	0	1	1	0	0	1	1	0	1
Useful answers	0	1	2	1	1	0	1	1	1	2	0	1	0	0	0	1
Better than site surfing	0	1	1	0	1	0	1	0	0	0	1	1	0	0	1	0
Quality	1	0	1	1	0	1	0	1	0	0	1	0	0	0	0	0
Interest shown	2	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0
Simple to use	2	1	1	1	2	0	2	1	1	1	1	1	0	1	0	1
Would use again	1	1	1	1	1	1	0	0	1	1	0	0	0	1	0	1

Sample Mean	0.62	where n = 588
Sample Standard Deviation	0.57	where variance S^2 = 0.32
Standard Error of the Mean	0.02	
Confidence Interval of the Mean @ 95	0.05	where z = 1.96

User 17	User 18	User 19	User 20	User 21	User 22	User 23	User 24	User 25	User 26	User 27	User 28	User 29	User 30	User 31	User 32	User 33
1	1	0	0	0	0	1	0	0	1	0	1	1	1	1	1	0
1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0
2	1	1	2	1	1	0	2	0	1	2	1	0	0	0	2	0
0	0	1	1	1	0	1	0	1	1	0	1	1	0	0	0	1
1	0	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1
0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1
0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0
0	1	0	1	1	0	0	0	0	1	0	1	1	0	0	0	0
0	0	0	1	1	1	0	0	0	1	1	0	1	1	1	1	1
2	1	0	0	1	1	1	0	2	1	2	1	1	1	1	1	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	1	1	0	1	1	0	0	1	0	0	1	0	0	1	0	0
1	0	1	0	1	0	1	0	0	0	0	0	0	1	2	1	1

User 34	User 35	User 36	User 37	User 38	User 39	User 40	User 41	User 42	Average
1	1	0	1	1	1	1	0	1	0.57
1	1	1	1	0	1	1	1	1	0.83
1	0	1	1	1	0	1	1	2	1.00
0	0	0	0	1	1	0	0	1	0.52
1	1	1	1	1	1	1	1	1	0.79
1	1	1	1	0	1	1	0	1	0.57
1	1	0	0	0	0	0	0	0	0.33
1	0	0	0	0	0	0	0	0	0.40
0	1	0	1	1	0	1	0	0	0.62
1	1	1	0	1	1	1	1	1	0.74
1	0	0	1	0	1	1	1	0	0.29
1	1	1	1	1	1	1	1	1	0.76
1	0	0	1	1	1	1	1	1	0.71
1	1	1	0	1	0	1	1	0	0.60